Weierstrass Institute for
Applied Analysis and Stochastics

On Solvers for Saddle Point Problems Arising in Finite Element Discretizations of Incompressible Flow Problems

Master Thesis

Freie Universität zu Berlin
Fachbereich Mathematik und Informatik
Institut für Mathematik

Supervisor: Prof. Dr. Volker John

Presented by: Natalia Schönknecht

Berlin, October 5, 2015

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorstehende Masterarbeit mit dem Titel "On solvers for saddle point problems arising in finite element discretizations of incompressible flow problems" selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel erstellt habe.

Die Stellen, die anderem Werken dem Wortlaut oder dem Sinn nach entnommen wurden, habe ich in jedem einzelnen Fall durch die Angabe der Quelle als Entlehnung kenntlich gemacht.

Berlin, den October 5, 2015

Contents

1

Introduction

Navier-Stokes equations are the basic differential equations of the viscous incompressible fluid dynamics. They define one of the fundamental models in fluid mechanics, which describes the movements of a wide class of real fluids. However the solution of boundary problems for Navier-Stokes equations represents a complex challenge of computational fluid dynamics.

Today the leading positions in the numerical solution of the differential equations are taken by the methods based on the use of the variational formulation equivalent to an original differential boundary problem. This class includes different variants of the finite element method (FEM). One of the important advantages of the finite element method for problems of aerohydrodynamics is the efficiency of the construction of finite element approximation on unstructured meshes and the possibility of almost complete automation of all stages of solving the problem - from mesh generating to building a global matrix of the resulting linear system.

The solving of systems of the linear algebraic equations arising from the discretization of nonlinear problems is an important step in the mathematical simulation. A non-self-adjoint differential operator of the original problem raises a number of difficulties in the application of many of the numerical methods. As a rule, to non-self-adjoint operators correspond non-symmetric matrices of linear algebraic systems, what complicates the solving of such systems.

The aim of this project is the study of different solution methods for linear systems of saddle point form, with emphasis on iterative methods for large and sparse problems, arising in finite element discretizations of incompressible flow problems.

In Chapter 2 the mathematical foundations for the incompressible steady-state Navier-Stokes equations are introduced, whereas Chapter 3 presents the corresponding unsteady case.

The subsequent chapter discusses basic algebraic properties of the saddle point matrices, the overview of solution algorithms and detailed presentation of the Schur complement reduction method, general strategies for preconditioning of the saddle point system arising from the Navier-Stokes equations, popular techniques of block preconditioners as the *Least Squares Commutator (LSC) Preconditioner* and the *Semi-Implicit Method for Pressure-Linked Equations (SIMPLE)*, based on the Schur complement approximation, the overview of multigrid methods and the used sparse direct solvers.

The numerical results can be found in Chapter 5 and Chapter 6. Finally a summary of results is presented.

2

Steady-State Navier-Stokes Equations

This chapter introduces the mathematical foundations for the incompressible steady-state Navier-Stokes equations. First, the base equations and boundary conditions used in this work are presented. Subsequently, the derivation of the weak formulation is briefly described. The next sections are used to present the linearization methods, finite element discretization and the resulting for computational implementation matrix-vector form. The presentation of this chapter mostly follows [9] and [3].

2.1 Basic Equations

Consider the steady-state Navier-Stokes equations for the incompressible flow of a Newtonian, viscous fluid, with constant viscosity:

$$\begin{aligned} -\nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \mathbf{f} \qquad \text{in } \Omega, \\ \nabla\cdot\mathbf{u} &= \mathbf{0} \qquad \text{in } \Omega, \end{aligned}$$

where

$\Omega \subset \mathbb{R}^d$	– is a bounded domain with a Lipschitz boundary $\partial\Omega$, $d \in \{2,3\}$,
$\mathbf{u}$	– is the fluid velocity,
p	– is the pressure field,
$\nu > 0$	– is the dimensionless viscosity coefficient,
$\mathbf{f} \in L^2(\Omega)$	– is the source term,
∇	– the gradient and
$\nabla\cdot$	– is the divergent operators.

The first equation represents conservation of momentum while the second one represents the incompressibility condition, also referred to as mass conservation. These equations are second order partial differential equations with respect to space. Thus they have to be equipped with boundary conditions on the boundary $\Gamma = \partial\Omega$ of Ω.

2.2 Boundary Conditions

There are several kinds of boundary conditions which can be prescribed for incompressible flows.

Case 1. *Dirichlet boundary conditions, no-slip boundary conditions, essential boundary conditions.*

Dirichlet boundary conditions describe the velocity field on a part of the boundary or on the whole boundary

$$\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) \qquad \text{in } \Gamma_{\text{diri}} \subseteq \partial\Omega.$$

It models in particular prescribed inflows into Ω and outflows from Ω. The special case

$$\mathbf{u}(\mathbf{x}) = \mathbf{0} \qquad \text{in } \Gamma_{\text{diri}},$$

is called no-slip boundary condition. Let $\mathbf{n}$ be the unit normal vector in $x \in \Gamma_{\text{nosl}} \subset \Gamma_{\text{diri}}$ and $\mathbf{t}_1$, $\mathbf{t}_2$ unit tangential vectors such that $\{\mathbf{n}, \mathbf{t}_1, \mathbf{t}_2\}$ is an orthonormal system of vectors. Then the no-slip boundary condition can be decomposed into three parts:

$$\mathbf{u}(\mathbf{x}) = \mathbf{0} \Longleftrightarrow \mathbf{u}(\mathbf{x}) \cdot \mathbf{n} = 0, \quad \mathbf{u}(\mathbf{x}) \cdot \mathbf{t}_1 = 0, \quad \mathbf{u}(\mathbf{x}) \cdot \mathbf{t}_2 = 0$$

on Γ_{nosl}. The condition $\mathbf{u}(\mathbf{x}) \cdot \mathbf{n} = 0$ states that the fluid does not penetrate the wall. The other conditions describe that the fluid does not slip along the wall. If Dirichlet boundary conditions are prescribed on the whole boundary of Ω, there are two additional issues. First, the pressure is determined only up to an additive constant. An additional condition for fixing the constant is that the integral mean value of the pressure should vanish

$$\int_\Omega p(x)\,\mathrm{d}x = 0.$$

Second, it follows from the divergence-free constraint and the integration by parts that the boundary condition has to satisfy the compatibility condition

$$0 = \int_\Omega \nabla \cdot \mathbf{u}(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_\Gamma (\mathbf{u} \cdot \mathbf{n})(s)\,\mathrm{d}s = \int_\Gamma (\mathbf{g} \cdot \mathbf{n})(s)\,\mathrm{d}s.$$

In the case of the Navier-Stokes equations and their special cases, Dirichlet boundary conditions are so-called essential boundary conditions.

Case 2. *Free slip boundary conditions, slip with friction boundary conditions.*

The free slip boundary condition is applied on boundaries without friction. It has the form

$$\begin{aligned} \mathbf{u} \cdot \mathbf{n} &= g \qquad \text{on } \Gamma_{\text{slip}} \subset \Gamma, \\ \mathbf{n}^T \mathbb{S} \mathbf{t}_1 &= 0 \qquad \text{on } \Gamma_{\text{slip}}, \end{aligned}$$

where $\mathbb{S}(t, x)$ is the Cauchy stress tensor, that represents all internal forces of the flow:

$$\mathbb{S} = -\nu \nabla u + p\mathbb{I}.$$

If $g = 0$ on Γ_{slip}, there is no penetration through the wall.

The slip with linear friction and no penetration boundary condition has the form

$$\begin{aligned} \mathbf{u} \cdot \mathbf{n} &= 0 \qquad \text{on } \Gamma_{\text{slfr}} \subset \Gamma, \\ \mathbf{u} \cdot \mathbf{t}_k + \beta^{-1} \mathbf{n}^T \mathbb{S} \mathbf{t}_k &= 0 \qquad \text{on } \Gamma_{\text{slfr}}, \quad k \in \{1, 2\}. \end{aligned}$$

These boundary conditions state that the fluid does not penetrate the wall and it slips along the wall while loosing energy. The difficulty in the application of this boundary condition consists in the determination of the friction parameter β, which might depend, e.g., on the local flow field and on the roughness of the wall.

Case 3. *Outflow or do-nothing boundary conditions, natural boundary conditions.*

For numerical simulations, the so-called outflow boundary condition or do-nothing boundary condition

$$\mathbb{S}\mathbf{n} = 0 \qquad \text{in } \Gamma_{\text{outfl}} \subset \Gamma$$

is often applied. This boundary condition models the situation that the normal stress, which is equal to the Cauchy stress vector, vanishes on the boundary part Γ_{outfl}. The do-nothing boundary condition is often used if no other outflow boundary condition is available.

From the mathematical point of view, the do-nothing boundary conditions are natural boundary conditions.

To simplify the presentation, only the problems with homogeneous Dirichlet boundary conditions on the whole boundary will be taken into account.

In this thesis we thus consider the strong form of the stationary Navier-Stokes problem

$$-\nu \Delta \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p = \mathbf{f} \qquad \text{in } \Omega, \tag{2.1}$$

$$\nabla \cdot \mathbf{u} = 0 \qquad \text{in } \Omega, \tag{2.2}$$

$$\mathbf{u} = \mathbf{0} \qquad \text{on } \Gamma, \tag{2.3}$$

$$\int_\Omega p(\mathbf{x}) \, \mathrm{d}\mathbf{x} = 0. \tag{2.4}$$

2.3 Weak Formulation

The numerical solution of the Navier-Stokes problem (2.1)-(2.4) with finite element methods is based on its variational or equivalently called weak formulation. For this purpose, the following sub-spaces of the usual Lebesgue function space $L^2(\Omega)$ of square-integrable functions on Ω are used

$$\begin{aligned} L_0^2(\Omega) &= \left\{ q: \quad q \in L^2(\Omega) \quad \text{with} \int_\Omega q(\mathbf{x})\,\mathrm{d}\mathbf{x} = 0 \right\}, \\ H^1(\Omega) &= \left\{ v: \quad v \in L^2(\Omega), \quad \partial_i v \in L^2(\Omega), \quad 1 \le i \le d \right\}, \\ H_0^1(\Omega) &= \left\{ v: \quad v \in H^1(\Omega) \quad \text{with } v = 0 \quad \text{on } \Gamma \right\}, \end{aligned}$$

where the value of v on the boundary is to be understood in the sense of traces.

Define

$$V = (H_0^1(\Omega))^d, d \in \{2,3\} \quad \text{and} \quad Q = L_0^2(\Omega).$$

Both spaces are Hilbert spaces. The inner product in V and the induced norm are given by

$$(\mathbf{u},\mathbf{w})_V = \int_\Omega (\nabla\mathbf{v}, \nabla\mathbf{w})(\mathbf{x})\mathrm{d}\mathbf{x}, \qquad \|\mathbf{v}\|_V = \|\nabla\mathbf{v}\|_{L^2(\Omega)} = (\mathbf{v},\mathbf{v})_V^{1/2}.$$

The inner product and the induced norm in Q are

$$(q,r)_Q = \int_\Omega (qr)(\mathbf{x})\mathrm{d}\mathbf{x}, \qquad \|q\|_Q = \|q\|_{L^2(\Omega)} = (q,q)_Q^{1/2}.$$

To obtain a weak formulation from the strong problem (2.1)-(2.4) one needs the usual steps:

- multiplication of the momentum equation (2.1) with a test function $\mathbf{v} \in V$ and of the continuity equation (2.2) with a test function $q \in Q$,
- integration over Ω and applying integration by parts.

Consequently one has

$$\int_\Omega -\nu\Delta\mathbf{u}\cdot\mathbf{v} + \int_\Omega (\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{v} + \int_\Omega \nabla p\cdot\mathbf{v} = \int_\Omega \mathbf{f}\cdot\mathbf{v} \tag{2.5}$$

$$\int_\Omega q\nabla\cdot\mathbf{u} = 0 \tag{2.6}$$

for all $\mathbf{v} \in V$ and all $q \in Q$. Assume that the functions are sufficiently smooth. Then the continuity requirements on the weak solution $(\mathbf{u}, p)$ can be reduced by "transfering" derivatives onto the test functions $\mathbf{v}$ and q.

Consider the equation (2.5). The first term:

$$\begin{aligned} -\int_\Omega \nu\Delta\mathbf{u}\cdot\mathbf{v} &= -\int_\Omega \nu \sum_{i=1}^d \Delta u_i v_i, \text{ using integration by parts} \\ &= -\int_\Gamma \sum_{i=1}^d \nu\,(\nabla u_i\cdot\mathbf{n})\,v_i + \int_\Omega \nu\sum_{i=1}^d \nabla u_i\nabla v_i \\ &= -\int_\Gamma \nu\,(\nabla\mathbf{u}\cdot\mathbf{v})\cdot\mathbf{n} + \int_\Omega (\nu\nabla\mathbf{u}) : (\nabla\mathbf{v})\,. \end{aligned}$$

Since the test functions are zero at the boundary, the first integral disappears and one gets:

$$-\int_\Omega \nu\Delta\mathbf{u}\cdot\mathbf{v} \;=\; \int_\Omega (\nu\nabla\mathbf{u}) : (\nabla\mathbf{v})\,. \tag{2.7}$$

Here $\nabla\mathbf{u} : \nabla\mathbf{v}$ represents the componentwise scalar product

$$A : B = \sum_{i=1}^{d}\sum_{j=1}^{d} A_{ij}\cdot B_{ij}.$$

The third integral in (2.5) can be transformed as:

$$\begin{aligned}
\int_\Omega \nabla p\cdot\mathbf{v} &= \int_\Omega \sum_{i=1}^{d}\frac{\partial p}{\partial x_i}v_i \\
&= \int_\Omega \sum_{i=1}^{d}\frac{\partial}{\partial x_i}(pv_i) - p\frac{\partial v_i}{\partial x_i} \\
&= \int_\Omega \nabla\cdot(p\mathbf{v}) - p\nabla\cdot\mathbf{v}, \quad \text{using the product rule,} \\
&= \int_\Gamma (p\mathbf{v})\cdot\mathbf{n} - \int_\Omega p\nabla\cdot\mathbf{v}, \quad \text{using the Gaussian theorem.}
\end{aligned}$$

The first boundary integral vanishes and

$$\int_\Omega \nabla p\cdot\mathbf{v} \;=\; -\int_\Omega p\nabla\cdot\mathbf{v}. \tag{2.8}$$

Combining (2.5), (2.7), and (2.8) gives

$$\int_\Omega (\nu\nabla\mathbf{u}) : (\nabla\mathbf{v}) + \int_\Omega (\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{v} - \int_\Omega p\nabla\cdot\mathbf{v} = \int_\Omega \mathbf{f}\cdot\mathbf{v} \quad \forall\mathbf{v}\in V.$$

Then the standard weak formulation is the following:
find $\mathbf{u}\in V$ and $p\in Q$ such that

$$\nu\int_\Omega (\nabla\mathbf{u}) : (\nabla\mathbf{v}) + \int_\Omega (\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{v} - \int_\Omega p\nabla\cdot\mathbf{v} \;=\; \int_\Omega \mathbf{f}\cdot\mathbf{v} \quad \forall\mathbf{v}\in V, \tag{2.9}$$

$$\int_\Omega q\nabla\cdot\mathbf{u} \;=\; 0 \qquad \forall q\in Q. \tag{2.10}$$

The construction of the weak formulation implies that any solution of (2.1)-(2.4) satisfies (2.9)-(2.10).

Introducing two continuous bilinear forms $a : V\times V\to\mathbb{R}$, $b : V\times Q\to\mathbb{R}$ and a trilinear form $c : V\times V\times V\to\mathbb{R}$ the weak formulation of steady-state Navier-Stokes equations with homogeneous Dirichlet boundary conditions can be written in the form: find $(\mathbf{u},p)\in V\times Q$ such that for all $\mathbf{v}\in V$ and $q\in Q$

$$\begin{cases} a(\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) + c(\mathbf{u},\mathbf{u},\mathbf{v}) = \langle\mathbf{f},\mathbf{v}\rangle \\ b(\mathbf{u},q) = 0, \end{cases} \tag{2.11}$$

where

$$\begin{aligned} \langle \mathbf{f}, \mathbf{v} \rangle &= \langle \mathbf{f}, \mathbf{v} \rangle_{V',V} = (\mathbf{f}, \mathbf{v}), \\ a(\mathbf{v}, \mathbf{w}) &= (\nu \nabla \mathbf{v}, \nabla \mathbf{w}), \\ b(\mathbf{v}, p) &= -(\nabla \cdot \mathbf{v}, p), \\ c(\mathbf{z}, \mathbf{v}, \mathbf{w}) &= ((\mathbf{z} \cdot \nabla)\mathbf{v}, \mathbf{w}). \end{aligned} \tag{2.12}$$

Consider the space of weakly divergence-free functions

$$V_{\text{div}} = \{\mathbf{v} \in V : \quad (\nabla \cdot \mathbf{v}, q) = 0, \quad \forall q \in Q\},$$

where the divergence of the functions from V_{div} vanishes in the sense of $L^2(\Omega)$, i.e., it is $(\nabla \cdot \mathbf{v})(\mathbf{x}) = 0$ almost everywhere in Ω. Then, an associated reduced problem can be introduced: find $\mathbf{u} \in V_{\text{div}}$ such that for all $\mathbf{v} \in V_{\text{div}}$

$$a(\mathbf{u}, \mathbf{v}) + c(\mathbf{u}, \mathbf{u}, \mathbf{v}) = \langle \mathbf{f}, \mathbf{v} \rangle. \tag{2.13}$$

In the both variational formulations the term $c(\cdot, \cdot, \cdot)$ is trilinear, that makes the whole problem nonlinear. For this term there are several forms

$$\begin{aligned} c_{\text{conv}}(\mathbf{u}, \mathbf{v}, \mathbf{w}) = ((\mathbf{u} \cdot \nabla)\mathbf{v}, \mathbf{w}) \quad &: \quad \text{convective form,} \\ c_{\text{skew}}(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \frac{1}{2}\left(c_{\text{conv}}(\mathbf{u}, \mathbf{v}, \mathbf{w}) - c_{\text{conv}}(\mathbf{u}, \mathbf{w}, \mathbf{v})\right) \quad &: \quad \text{skew-symmetric form.} \end{aligned}$$

The spaces V and Q defined above satisfy the inf-sup condition

$$\exists \beta > 0 : \quad \inf_{q \in Q, q \neq 0} \sup_{\mathbf{v} \in V, \mathbf{v} \neq \mathbf{0}} \frac{b(\mathbf{v}, q)}{\|\mathbf{v}\|_V \|q\|_Q} \geq \beta. \tag{2.14}$$

This condition, which is also called Ladyzhenskaya-Babuška-Brezzi (LBB) condition, is very important in order to guarantee a unique solution of the problem (2.11).

Theorem 2.1 (**Existence and uniqueness of the Navier-Stokes problem with homogeneous boundary condition).** *Let $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$, be a bounded domain with Lipschitz boundary Γ and $\mathbf{f} \in (H^{-1}(\Omega))^d$. Then there exists at least one weak solution $(\mathbf{u}, p) \in V \times Q$, which satisfies (2.9)-(2.10) or equivalently (2.11). If in addition the condition*

$$\nu^{-2} \|f\|_{V'} \sup_{\mathbf{u}, \mathbf{v}, \mathbf{w} \in V_{div}} \frac{c(\mathbf{u}, \mathbf{v}, \mathbf{w})}{\|\mathbf{u}\|_V \|\mathbf{v}\|_V \|\mathbf{w}\|_V} < 1 \tag{2.15}$$

be fulfilled with $V' = (H^{-1}(\Omega))^d$ the dual space of V_{div}, then this solution is unique.

Proof. The proof can be found in [6], Chapter IV, Theorems 2.1 and 2.2. □

Lemma 2.2 (**Stability of the Navier-Stokes problem with homogeneous boundary condition).** *Let $(\mathbf{u}, p) \in V \times Q$ be any solution of (2.9)-(2.10) or equivalently (2.11), then*

$$\|\nabla \mathbf{u}\|_{L^2(\Omega)} \leq \frac{1}{\nu} \|\mathbf{f}\|_{H^{-1}(\Omega)}, \tag{2.16}$$

$$\|p\|_{L^2(\Omega)} \leq \frac{1}{\beta}\left(2 \|\mathbf{f}\|_{H^{-1}(\Omega)} + \frac{C}{\nu^2} \|\mathbf{f}\|^2_{H^{-1}(\Omega)}\right). \tag{2.17}$$

Proof. The proof can be found in [9], Chapter V, Lemma 5.17. □

2.4 Linearization

As one can see, the Navier-Stokes equations are nonlinear because of the existence of the convective term. The usual approach to solve these equations is a non-linear iteration with a linearized problem being solved at every step. Therefore an "initial guess" $(\mathbf{u}_0, p_0) \in V \times Q$ is given, a sequence of iterates

$$(\mathbf{u}_0, p_0), (\mathbf{u}_1, p_1), (\mathbf{u}_2, p_2) \ldots \in V \times Q$$

is computed, which converges (hopefully) to the solution of the weak formulation. In this work two classical linearization procedures are introduced (see [3], §7.2.2).

2.4.1 The Newton's Linearization Scheme

The iterate $(\mathbf{u}_k, p_k)$ is given. Start by computing the *nonlinear residual* associated with the weak formulation (2.11). This is the pair $R_k(\mathbf{v})$, $r_k(q)$ satisfying

$$\begin{aligned} R_k &= \langle \mathbf{f}, \mathbf{v} \rangle - c(\mathbf{u}_k, \mathbf{u}_k, \mathbf{v}) - a(\mathbf{u}_k, \mathbf{v}) - b(\mathbf{v}, p_k), \\ r_k &= -b(\mathbf{u}_k, q), \end{aligned}$$

for any $\mathbf{v} \in V$ and $q \in Q$. Let

$$\mathbf{u} = \mathbf{u}_k + \delta\mathbf{u}_k$$

and

$$p = p_k + \delta p_k$$

be the solution of (2.11). It is easy to see that the corrections $\delta\mathbf{u}_k \in V$ and $\delta p_k \in Q$ satisfy

$$D(\mathbf{u}_k, \delta\mathbf{u}_k, \mathbf{v}) + a(\delta\mathbf{u}_k, \mathbf{v}) + b(\delta p_k, \mathbf{v}) = R_k(\mathbf{v}), \tag{2.18}$$

$$-b(\delta\mathbf{u}_k, q) = r_k(q) \tag{2.19}$$

for all $\mathbf{v} \in V$ and $q \in Q$, where $D(\mathbf{u}_k, \delta\mathbf{u}_k, \mathbf{v})$ is the difference in the nonlinear terms:

$$D(\mathbf{u}, \delta\mathbf{u}, \mathbf{v}) = c(\delta\mathbf{u}, \delta\mathbf{u}, \mathbf{v}) + c(\delta\mathbf{u}, \mathbf{u}, \mathbf{v}) + c(\mathbf{u}, \delta\mathbf{u}, \mathbf{v}).$$

Expanding $D(\mathbf{u}_k, \delta\mathbf{u}_k, v)$ and dropping the quadratic term $c(\delta\mathbf{u}, \delta\mathbf{u}, \mathbf{v})$ gives the linear problem:
for all $\mathbf{v} \in V$ and $q \in Q$, find $\delta\mathbf{u}_k \in V$ and $\delta p_k \in Q$ satisfying

$$\begin{aligned} c(\delta\mathbf{u}_k, \mathbf{u}_k, \mathbf{v}) + c(\mathbf{u}_k, \delta\mathbf{u}_k, \mathbf{v}) + a(\delta\mathbf{u}_k, \mathbf{v}) - b(\delta p_k, \mathbf{v}) &= R_k(\mathbf{v}), \\ b(\delta\mathbf{u}_k, q) &= r_k(q). \end{aligned} \tag{2.20}$$

The solution of (2.20) is the so-called Newton correction. Updating the previous iterate via

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \delta\mathbf{u}_k, \qquad p_{k+1} = p_k + \delta p_k$$

defines the next iterate in the sequence.

2.4.2 The Picard's Linearization Scheme

In terms of the representation (2.18)-(2.19), the quadratic term $c(\delta\mathbf{u}_k, \delta\mathbf{u}_k, v)$ is dropped along with the linear term $c(\delta\mathbf{u}_k, \mathbf{u}_k, \mathbf{v})$. Thus, one has the following linear problem:
for all $\mathbf{v} \in V$ and $q \in Q$, find $\delta\mathbf{u}_k \in V$ and $\delta p_k \in Q$ satisfying

$$\begin{aligned} c(\mathbf{u}_k, \delta\mathbf{u}_k, \mathbf{v}) + a(\delta\mathbf{u}_k, \mathbf{v}) - b(\mathbf{v}, \delta p_k) &= R_k(\mathbf{v}), \\ b(\delta\mathbf{u}_k, q) &= r_k(q). \end{aligned} \tag{2.21}$$

The solution of (2.21) is the Picard correction. Updating the previous iterate via

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \delta\mathbf{u}_k, \qquad p_{k+1} = p_k + \delta p_k$$

defines the next iterate in the sequence. Substituting

$$\delta\mathbf{u}_k = \mathbf{u}_{k+1} - \mathbf{u}_k, \qquad \delta p_k = p_{k+1} - p_k$$

into (2.21), one obtains an explicit definition for the new iterate:
for all $\mathbf{v} \in V$ and $q \in Q$, find $\mathbf{u}_{k+1} \in V$ and $p_{k+1} \in Q$ such that

$$\begin{aligned} c(\mathbf{u}_k, \mathbf{u}_{k+1}, v) + a(\mathbf{u}_{k+1}, \mathbf{v}) - b(\mathbf{v}, p_{k+1}) &= \langle \mathbf{f}, \mathbf{v} \rangle, \\ b(\mathbf{u}_{k+1}, q) &= 0. \end{aligned} \tag{2.22}$$

The formulation (2.22) is commonly referred to as the Oseen system. One can see that comparing (2.21) with the weak formulation the Picard iteration corresponds to a simple fixed point iteration strategy for solving (2.11), with the convection coefficient evaluated at the current velocity. As a result, the rate of convergence of the Picard iteration is only linear in general.

The main drawback of Newton's method is that the radius of the ball of convergence is typically proportional to the viscosity parameter ν. Thus, as the Reynolds number is increased, better and better initial guesses are needed in order for the Newton iteration to converge. The advantage of the Picard iteration is that, relative to Newton iteration, it has a huge ball of convergence.

2.5 Finite Element Discretization

The principal idea of using finite element methods consists in replacing the infinite-dimensional spaces V and Q by a finite-dimensional velocity space V^h and a finite-dimensional pressure space Q^h and to apply the Galerkin method.

If $V^h \subset V$ and $Q^h \subset Q$, the finite element method is called conforming otherwise it is called non-conforming. In this work only the conforming finite element method is considered. For conforming spaces one can use the same bilinear and trilinear forms as in the continuous case.

The spaces V^h and Q^h have to satisfy a discrete inf-sup condition

$$\inf_{q^h \in Q^h, q^h \neq 0} \sup_{\mathbf{v}^h \in V^h, \mathbf{v}^h \neq 0} \frac{b(\mathbf{v}^h, q^h)}{\|\mathbf{v}^h\|_{V^h} \|q^h\|_{Q^h}} \geq \beta_{is}^h > 0.$$

The finite element formulation. Let V^h be a velocity finite element space and let Q^h be a pressure finite element space. The finite element discretization of the (2.11) is read as follows:
find $(\mathbf{u}^h, p^h) \in V^h \times Q^h$ such that

$$a(\mathbf{u}^h, \mathbf{v}^h) + b(\mathbf{v}^h, p^h) + c(\mathbf{u}^h, \mathbf{u}^h, \mathbf{v}^h) = \left\langle \mathbf{f}, \mathbf{v}^h \right\rangle \qquad \forall \mathbf{v}^h \in V^h \tag{2.23}$$

$$b(\mathbf{u}^h, q^h) = 0, \qquad \forall \mathbf{q}^h \in Q^h. \tag{2.24}$$

This problem is equivalent to:
find $(\mathbf{u}^h, p^h) \in V^h \times Q^h$ such that

$$a(\mathbf{u}^h, \mathbf{v}^h) + b(\mathbf{v}^h, p^h) + c(\mathbf{u}^h, \mathbf{u}^h, \mathbf{v}^h) - b(\mathbf{u}^h, q^h) = \left\langle \mathbf{f}, \mathbf{v}^h \right\rangle, \quad \forall (\mathbf{v}^h, q^h) \in V^h \times Q^h.$$

The space

$$V_{\text{div}}^h = \left\{ \mathbf{v}^h \in V^h : \quad b(\mathbf{v}^h, q^h) = 0 \quad \forall q^h \in Q^h \right\}$$

is the space of discretely divergence-free functions, which is not empty, because of the fulfillment of the discrete inf-sup condition. It follows that the finite element approximation of the velocity can be computed by solving the following problem:
find $\mathbf{u}^h \in V_{\text{div}}^h$ such that

$$a^h(\mathbf{u}^h, \mathbf{v}^h) + c^h(\mathbf{u}^h, \mathbf{u}^h, \mathbf{v}^h) = \left\langle \mathbf{f}, \mathbf{v}^h \right\rangle, \qquad \forall \mathbf{v}^h \in V_{\text{div}}^h. \tag{2.25}$$

Existence and uniqueness of a solution are proved in the similar way as for the continuous equation. In particular, a unique solution can be expected only for small right-hand side or large viscosity. The condition (2.15) is sufficient and it will be assumed.

Lemma 2.3 (Stability of the finite element solution). *Let $V^h \times Q^h$ be inf-sup stable finite element spaces. Then, the finite element solution of the steady-state Navier-Stokes equations with skew-symmetric form of the convective term is stable:*

$$\left\|\nabla \mathbf{u}^h\right\|_{L^2(\Omega)} \leq \frac{1}{\nu} \|\mathbf{f}\|_{H^{-1}(\Omega)}, \tag{2.26}$$

$$\left\|p^h\right\|_{L^2(\Omega)} \leq \frac{1}{\beta_{is}^h} \left(2 \|\mathbf{f}\|_{H^{-1}(\Omega)} + \frac{C}{\nu^2} \|\mathbf{f}\|_{H^{-1}(\Omega)}^2 \right). \tag{2.27}$$

Theorem 2.4 (**Finite element error estimate for the $L^2(\Omega)$ norm of the gradient of the velocity**). *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with polyhedral and Lipschitz-continuous boundary and let $\nu^{-2} \|\mathbf{f}\|_{H^{-1}(\Omega)}$ be sufficiently small such that in particular the Navier-Stokes equations (2.9)-(2.10) possess a unique solution $(\mathbf{u}, p) \in V \times Q$. Assume that this problem is discretized with inf-sup stable finite element spaces $V^h \times Q^h$ and denote by $\mathbf{u}^h \in V^h_{div}$ the velocity solution. Then the following error estimate holds*

$$\begin{aligned} &\left\|\nabla(\mathbf{u} - \mathbf{u}^h)\right\|_{L^2(\Omega)} \leq \\ &\leq C \left(\left(1 + \frac{1}{\nu^2} \|\mathbf{f}\|_{H^{-1}(\Omega)}\right) \left(1 + \frac{1}{\beta^h_{is}}\right) \inf_{\mathbf{v}^h \in V^h} \left\|\nabla(\mathbf{u} - \mathbf{v}^h)\right\|_{L^2(\Omega)} \right. \\ &\qquad \left. + \frac{1}{\nu} \inf_{q^h \in Q^h} \left\|p - q^h\right\|_{L^2(\Omega)} \right) \end{aligned} \tag{2.28}$$

Theorem 2.5 (**Finite element error estimate for the $L^2(\Omega)$ norm of the pressure**). *Let the assumptions of the Theorem 2.4 be fulfilled, then the following error estimate for the pressure holds*

$$\begin{aligned} &\left\|\nabla(p - p^h)\right\|_{L^2(\Omega)} \leq \\ &\leq C\nu \left(1 + \frac{1}{\nu^2} \|\mathbf{f}\|_{H^{-1}(\Omega)}\right)^2 \left(\frac{1}{\beta^h_{is}} + \frac{1}{(\beta^h_{is})^2}\right) \inf_{\mathbf{v}^h \in V^h} \left\|\nabla(\mathbf{u} - \mathbf{v}^h)\right\|_{L^2(\Omega)} \\ &+ C \left(\frac{1}{\beta^h_{is}} \left(1 + \frac{1}{\nu^2} \|\mathbf{f}\|_{H^{-1}(\Omega)}\right) + \left(1 + \frac{1}{\beta^h_{is}}\right) \right) \inf_{q^h \in Q^h} \left\|p - q^h\right\|_{L^2(\Omega)}. \end{aligned} \tag{2.29}$$

The constants depend on the domain Ω.

2.6 Matrix-Vector Form

In order to derive an algebraic system from (2.23)-(2.24), the spaces V^h and Q^h are equipped with a basis. A standard approach for choosing the basis of the vector-valued velocity space is as follows:

$$\begin{aligned} V^h &= span\left\{\phi_i^h\right\}_{i=1}^{3N_v} \\ &= span\left\{ \left\{ \begin{pmatrix} \phi_i^h \\ 0 \\ 0 \end{pmatrix} \right\}_{i=1}^{N_v} \cup \left\{ \begin{pmatrix} 0 \\ \phi_i^h \\ 0 \end{pmatrix} \right\}_{i=1}^{N_v} \cup \left\{ \begin{pmatrix} 0 \\ 0 \\ \phi_i^h \end{pmatrix} \right\}_{i=1}^{N_v} \right\}, \end{aligned}$$

i.e.. each basis function does not vanish in one component only. Here, N_v is the number of unknowns (degrees of freedom, d.o.f.) for one component of the velocity. The pressure space is

$$Q^h = span\left\{\psi_i^h\right\}_{i=1}^{N_p},$$

where N_p is the number of pressure degrees of freedom. Hence one has the unique representation

$$\mathbf{u}^h = \sum_{j=1}^{3N_v} u_j^h \phi_j^h, \qquad p^h = \sum_{j=1}^{N_p} p_j^h \psi_j^h, \tag{2.30}$$

with unknown real coefficients $\vec{u} = (u_j^h)_{j=1}^{3N_v}$ and $\vec{p} = (p_j^h)_{j=1}^{N_p}$. Inserting (2.30) into (2.23)-(2.24) leads to a nonlinear system of algebraic equations.

Linearization of this system using the Newton iteration gives:
find corrections $\delta\mathbf{u} \in V^h$ and $\delta p^h \in Q^h$ (here the subscript k is dropped to avoid notational clutter) satisfying

$$\begin{aligned} c(\delta\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h) + c(\mathbf{u}_h, \delta\mathbf{u}_h, \mathbf{v}_h) + a(\delta\mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, \delta p_h) &= R_k(\mathbf{v}_h), \quad (2.31) \\ -b(\delta\mathbf{u}_h, q_h) &= r_k(q_h), \quad (2.32) \end{aligned}$$

for all $\mathbf{v}_h \in V^h$ and $q_h \in Q^h$. Here, $R_k(\mathbf{v}_h)$ and $r_k(q_h)$ are the nonlinear residuals associated with the discrete formulation (2.23)-(2.24). The corrections $\delta\mathbf{u}$ and δp^h can be also represented with basis functions as:

$$\delta\mathbf{u}_h = \sum_{j=1}^{3N_v} \Delta u_j^h \phi_j^h, \qquad \delta p^h = \sum_{j=1}^{N_p} \Delta p_j^h \psi_j^h. \tag{2.33}$$

To define the corresponding linear algebra problem, substitute (2.30) and (2.33) into (2.31)-(2.32). Then one obtains a system of linear equations

$$\begin{bmatrix} \nu\mathbf{A} + \mathbf{N} + \mathbf{W} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u} \\ \Delta\mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix}. \tag{2.34}$$

The matrix $\mathbf{A}$ is called the vector-Laplacian matrix, and the matrix B is called

the divergence matrix. The entries are given by

$$\mathbf{A} = (\mathbf{a}_{ij})_{i,j=1}^{3N_v}, \qquad \mathbf{a}_{ij} = \int_\Omega \nabla\phi_i : \nabla\phi_j, \tag{2.35}$$

$$B = (b_{kj})_{k,j=1}^{N_p,3N_v}, \qquad b_{kj} = -\int_\Omega \psi_k \nabla\cdot\phi_j. \tag{2.36}$$

The matrix $\mathbf{N}$ is the vector-convection matrix, and the matrix $\mathbf{W}$ is called the Newton derivative matrix. Both matrices depend on the current estimate of the discrete velocity $\mathbf{u}_h$, and their entries are

$$\mathbf{N} = (\mathbf{n}_{ij})_{i,j=1}^{3N_v}, \qquad \mathbf{n}_{ij} = \int_\Omega (\mathbf{u}_h\cdot\nabla\phi_j)\cdot\phi_i, \tag{2.37}$$

$$\mathbf{W} = (\mathbf{w}_{ij})_{i,j=1}^{3N_v}, \qquad \mathbf{w}_{ij} = \int_\Omega (\phi_j\cdot\nabla\mathbf{u}_h)\cdot\phi_i. \tag{2.38}$$

Notice that the Newton derivative matrix is symmetric. The right-hand side vectors in (2.34) are the nonlinear residuals associated with the current discrete solution estimates $\mathbf{u}_h$ and p_h, expanded via (2.30) and (2.33):

$$\mathfrak{f} = (\mathfrak{f}_i)_{i=1}^{3N_v}, \qquad \mathfrak{f}_i = \int_\Omega \mathbf{f}_i\cdot\phi_i - \int_\Omega \mathbf{u}_h\cdot\nabla\mathbf{u}_h\cdot\phi_i - \nu\int_\Omega \nabla\mathbf{u}_h : \nabla\phi_i + \int_\Omega p_h(\nabla\cdot\phi_i), \tag{2.39}$$

$$\mathbf{g} = (\mathbf{g}_k)_{k=1}^{N_p}, \qquad \mathbf{g}_k = -\int_\Omega \psi_k(\nabla\cdot\mathbf{u}_h). \tag{2.40}$$

The system (2.34) is referred to as *the discrete Newton problem.*

For Picard iteration, the Newton derivative matrix is omitted to give *the discrete Oseen problem*:

$$\begin{bmatrix} \nu\mathbf{A}+\mathbf{N} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u} \\ \Delta\mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix}. \tag{2.41}$$

The algebraic systems that arise from the Navier-Stokes equations belong to the type of linear systems in saddle point form and have block structure.

The numbers of blocks in the velocity-velocity couplings that have to be stored are different for Picard and Newton methods. Using Newton's method requires the storage of more velocity-velocity matrix blocks. As a consequence, the computational costs for matrix assembling are larger. Therefore in this work the Picard's iteration method is used.

The matrix $\mathbf{K} = diag(K_1,\ldots,K_d) = \nu\mathbf{A}+\mathbf{N}$ is a block diagonal matrix, where each block corresponds to a discrete convection-diffusion operator with appropriate boundary conditions. In general $\mathbf{K}$ is non-symmetric, but the symmetric part of $\mathbf{K}$, $H = \frac{1}{2}(\mathbf{K}+\mathbf{K}^T)$, is positive semi-definite, when an appropriate (conservative) discretization is used.

The rectangular matrix B^T represents the discrete gradient operator while B represents its adjoint, the (negative) divergence operator.

Properties of saddle point matrices such as invertibility, spectral properties, conditioning and solution methods will be considered in Chapter 4.

3

Time-Dependent Navier-Stokes Equations

The purposes of this chapter are analysis of the time-dependent Navier-Stokes equations and the obtaining of the corresponding linear algebraic system. The presentation of this section follows [8] and [9].

3.1 Basic Equations

Consider the time-dependent incompressible flow of a Newtonian, viscous fluid, with constant viscosity. This flow is governed by the incompressible Navier-Stokes equations given (in dimensionless form) by

$$\begin{aligned}
\mathbf{u}_t - \nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } (0,T]\times\Omega, && (3.1)\\
\nabla\cdot\mathbf{u} &= \mathbf{0} \quad \text{in } [0,T]\times\Omega, && (3.2)\\
\mathbf{u}(0,\cdot) &= \mathbf{u_0} \quad \text{in } \Omega, && (3.3)\\
\mathbf{u} &= \mathbf{g} \quad \text{on } [0,T]\times\Gamma, && (3.4)\\
\int_\Omega p(\mathbf{x})\,\mathrm{d}\mathbf{x} &= 0 \quad \text{in } [0,T], && (3.5)
\end{aligned}$$

where

$\Omega\subset\mathbb{R}^d$	– is a bounded domain with a Lipschitz boundary $\Gamma=\partial\Omega, d\in\{2,3\}$,
$\mathbf{u}$	– is the fluid velocity,
$\mathbf{u}_0$	– is the initial velocity,
p	– is the pressure field,
$\nu = 1/Re > 0$	– is the dimensionless viscosity coefficient,
$\mathbf{f}\in L^2(\Omega)$	– represents body forces,
$\mathbf{g}$	– is the given Dirichlet boundary data,
$[0,T]$	– is a given time interval.

3.2 Weak Formulation

To simplify the presentation, also in the case of instationary Navier-Stokes equations, consider only the problems with homogeneous Dirichlet boundary conditions on the whole boundary

$$\mathbf{u} = \mathbf{0} \qquad \text{on } [0,T] \times \Gamma. \tag{3.6}$$

One obtains a weak formulation from the strong problem (3.1)-(3.5) by usual steps: multiplication (3.1), (3.2) with a corresponding test function $(\mathbf{v}, q)$ and integration over $(0,T) \times \Omega$, as well as application of integration by parts in order to transfer the derivatives from the solution onto the test function. The result is the following variational problem

find $\mathbf{u} : (0,T] \to V$ and $p : [0,T] \to Q$ such that for all $(\mathbf{v}, q) \in (V, Q)$

$$\begin{aligned} -\int_\Omega \mathbf{u}_t \cdot \mathbf{v} + \nu \int_\Omega (\nabla \mathbf{u}) : (\nabla \mathbf{v}) + \int_\Omega (\mathbf{u} \cdot \nabla \mathbf{u}) \cdot \mathbf{v} - \int_\Omega p \nabla \cdot \mathbf{v} &= \int_\Omega \mathbf{f} \cdot \mathbf{v} \\ \int_\Omega q \nabla \cdot \mathbf{u} &= 0. \end{aligned} \tag{3.7}$$

or using the bilinear forms (2.12), introduced in Chapter 2, the weak formulation of the non-stationary Navier-Stokes problem with homogeneous Dirichlet boundary conditions can be written

find $\mathbf{u} : (0,T] \to V$ and $p : [0,T] \to Q$ such that for all $\mathbf{v} \in V$ and $q \in Q$

$$\begin{aligned} -(\mathbf{u}_t, \mathbf{v}) + a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) + c(\mathbf{u}, \mathbf{u}, \mathbf{v}) &= \langle \mathbf{f}, \mathbf{v} \rangle \\ b(\mathbf{u}, q) &= 0. \end{aligned} \tag{3.8}$$

Theorem 3.1 (**Existence and uniqueness for weak solutions of the non-stationary Navier-Stokes equations with homogeneous Dirichlet boundary conditions).** *For given* $\mathbf{f}$ *and* $\mathbf{u}_0$ *such that*

$$\mathbf{f} \in L^2(0,T; H^{-1}(\Omega)), \qquad \mathbf{u}_0 \in H_0^1(\Omega) \tag{3.9}$$

there exists a weak solution $\mathbf{u}$ *to the Navier-Stokes equations (3.1)-(3.5) satisfying*

$$\mathbf{u} \in L^\infty(0,T; H_0^1(\Omega)). \tag{3.10}$$

Furthermore, if $d = 2$, $\mathbf{u}$ *is unique.*

Proof. The proof can be found in [12], Chapter 3, §3.1, §3.2. □

3.3 Discretization

The numerical solution of the time-dependent Navier-Stokes equations requires their discretization in time and space as well as a linearization. There are different approaches for all of these components. In this thesis the following strategy is used:

(*i*) *Semi-discretization of (3.1)-(3.5) in time.* An implicit time stepping scheme will be applied first. The semi-discretization in time leads in each discrete time step to a non-linear system of equations.

(*ii*) *Variational formulation and linearization.* The non-linear system of equations is reformulated as a variational problem and the non-linear variational problem is linearized by a Picard iteration.

(*iii*) *Discretization of the linear systems in space.* The linear system of equations arising in each step of the fixed point iteration is discretized by a finite element method using an inf-sup stable pair of finite element spaces.

Below all these steps are discussed in detail.

3.3.1 Semi-Discretization in Time

Let Δt_k be the current time step from t_{k-1} to t_k, i.e., $\Delta t_k = t_k - t_{k-1}$. Then the time stepping Crank-Nicolson scheme is given by

$$\begin{aligned}\mathbf{u}_k + \frac{1}{2}\Delta t_k\left(-\nu\Delta\mathbf{u}_k + (\mathbf{u}_k\cdot\nabla)\mathbf{u}_k\right) + \Delta t_k\nabla p_k = \\ = \mathbf{u}_{k-1} - \frac{1}{2}\Delta t_k\left(-\nu\Delta\mathbf{u}_{k-1} + (\mathbf{u}_{k-1}\cdot\nabla)\mathbf{u}_{k-1}\right) + \frac{1}{2}\Delta t_k\left(\mathbf{f}_{k-1}+\mathbf{f}_k\right), \\ \nabla\cdot\mathbf{u}_k = 0. \qquad (3.11)\end{aligned}$$

One can prove, under a number of assumptions on the smoothness of the data, that the error between the time discrete and the continuous velocity in $L^\infty(0,T;L^2(\Omega))$ behaves like $(\Delta t)^2$ for the equidistant time step Δt. This scheme is A-stable and may lead to numerical oscillations in problems with rough initial data or boundary conditions. These oscillations are damped out only if sufficiently small time steps are used [8].

3.3.2 Variational Formulation and Linearization

The solution of (3.11) will be approximated by a finite element method. The basis of the finite element method is a variational formulation of (3.11), which can be formulated as:

find $(\mathbf{u}_k, p_k) \in (V, Q)$ such that for all $(\mathbf{v}, q) \in (V, Q)$

$$\begin{aligned}(\mathbf{u}_k,\mathbf{v}) + \frac{1}{2}\Delta t_k\left(a(\mathbf{u}_k,\mathbf{v}) + c(\mathbf{u}_k,\mathbf{u}_k,\mathbf{v})\right) + \Delta t_k b(p_k,\mathbf{v}) = \\ = (\mathbf{u}_{k-1},\mathbf{v}) - \frac{1}{2}\Delta t_k\left(a(\mathbf{u}_{k-1},\mathbf{v}) + c(\mathbf{u}_{k-1},\mathbf{u}_{k-1},\mathbf{v})\right) \\ + \frac{1}{2}\Delta t_k\left((\mathbf{f}_{k-1},\mathbf{v}) + (\mathbf{f}_k,\mathbf{v})\right), \\ b(\mathbf{u}_k, q) = 0. \qquad (3.12)\end{aligned}$$

The nonlinear system (3.12) is solved iteratively, starting with an initial guess $(\mathbf{u}_k^0, p_k^0)$. Given $(\mathbf{u}_k^m, p_k^m)$, the iterate $(\mathbf{u}_k^{m+1}, p_k^{m+1})$ is computed by solving

$$\begin{aligned}(\mathbf{u}_k^{m+1}, \mathbf{v}) + \frac{1}{2}\Delta t_k \left(a(\mathbf{u}_k^{m+1}, \mathbf{v}) + c(\mathbf{u}_k^m, \mathbf{u}_k^{m+1}, \mathbf{v})\right) + \Delta t_k b(p_k^{m+1}, \mathbf{v}) = \\ = (\mathbf{u}_{k-1}, \mathbf{v}) - \frac{1}{2}\Delta t_k \left(a(\mathbf{u}_{k-1}, \mathbf{v}) + c(\mathbf{u}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v})\right) + \\ + \frac{1}{2}\Delta t_k \left((\mathbf{f}_{k-1}, \mathbf{v}) + (\mathbf{f}_k, \mathbf{v})\right), \\ 0 = b(\mathbf{u}_k^{m+1}, q) \qquad (3.13)\end{aligned}$$

for all $(\mathbf{v}, q) \in (V, Q)$, $m = 0, 1, 2 \ldots$ That means, the linearization is done by a Picard iteration. The initial guess is chosen to be the solution of the previous time step $(\mathbf{u}_k^0, p_k^0) = (\mathbf{u}_{k-1}, p_{k-1})$.

3.3.3 Discretization of the Linear Systems in Space

The equations (3.13) are discretized by a finite element method. Let (V^h, Q^h) be a pair of finite element spaces which fulfill the inf-sup stability condition. Then, the finite element problem has the following form (indices k, m, $m+1$ will be neglected):

find $(\mathbf{u}^h, p^h) \in V^h \times Q^h$ such that for all $(\mathbf{v}^h, q^h) \in V \times Q$

$$\begin{aligned}(\mathbf{u}^h, \mathbf{v}^h) + \frac{1}{2}\Delta t_k \left(a(\mathbf{u}^h, \mathbf{v}^h) + c(\mathbf{u}_{old}^h, \mathbf{u}^h, \mathbf{v}^h)\right) + \Delta t_k b(p^h, \mathbf{v}^h) = \\ = (\mathbf{u}_{k-1}^h, \mathbf{v}^h) - \frac{1}{2}\Delta t_k \left(a(\mathbf{u}_{k-1}^h, \mathbf{v}^h) + c(\mathbf{u}_{k-1}^h, \mathbf{u}_{k-1}^h, \mathbf{v}^h)\right) + \\ + \frac{1}{2}\Delta t_k \left((\mathbf{f}_{k-1}^h, \mathbf{v}) + (\mathbf{f}_k^h, \mathbf{v})\right), \\ 0 = b(\mathbf{u}_k^h, q^h), \qquad (3.14)\end{aligned}$$

where $\mathbf{u}_{old}^h \in V^h$ is the current approximation of the velocity; $\mathbf{f}_{k-1}^h$ and $\mathbf{f}_k^h$ are finite element representations of the right-hand side at the times t_{k-1} and t_k respectively. For more efficiency it is sufficient to solve the system (3.14) only approximately in each time step of the fixed point iteration instead of solving it always accurately [8].

3.4 Matrix-Vector Form

To get the approximation of the solution, the functions $\mathbf{u}^h(\mathbf{x}, t)$, $p^h(\mathbf{x}, t)$ are written as linear combinations of time-independent basis functions with time-dependent coefficients

$$\mathbf{u}^h(\mathbf{x}, t) = \sum_{j=1}^{dN_v} u_j^h(t)\phi_j(x) \qquad (3.15)$$

$$p^h(\mathbf{x}, t) = \sum_{j=1}^{N_p} p_j^h(t)\psi_j(x), \qquad (3.16)$$

where the basis functions $\{\phi_i\}_{i=1}^{dN_v}$, $\{\psi_i\}_{i=1}^{N_p}$ are exactly the same as in the steady case of Chapter 2, Subsection 2.6; N_v is the number of degrees of freedom for one component of the velocity, N_p is the number of pressure degrees of freedom, $d \in \{2, 3\}$ is the dimension of the problem.

Inserting (3.15), (3.16) into (3.14) leads to a linear system of algebraic equations

$$\begin{bmatrix} \mathbf{M} + \frac{1}{2}\Delta t_k(\nu\mathbf{A} + \mathbf{N}) & \Delta t_k B^T \\ B & O \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u} \\ \Delta\mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix}. \tag{3.17}$$

The matrix $\mathbf{M}$ is a mass matrix and its entries are given by

$$\mathbf{M} = (\mathbf{m}_{ij})_{i,j=1}^{dN_v}, \quad \mathbf{m}_{ij} = \int_\Omega \phi_i : \phi_j. \tag{3.18}$$

The vector-Laplacian matrix $\mathbf{A}$, the divergence matrix B and the vector-convection matrix $\mathbf{N}$ were introduced in steady case by (2.35)-(2.37).

The right-hand side vectors are

$$\begin{aligned} \mathfrak{f} = (\mathfrak{f}_i)_{i=1}^{dN_v}, \qquad \mathfrak{f}_i = &\int_\Omega \mathbf{u}_{old}^h \phi_i - \frac{1}{2}\Delta t_k \left[\nu \int_\Omega \nabla\mathbf{u}_{old}^h : \nabla\phi_i + \int_\Omega (\mathbf{u}_{old}^h \cdot \nabla)\mathbf{u}_{old}^h \cdot \phi_i\right] + \\ &+ \frac{1}{2}\Delta t_k \left[\int_\Omega \mathbf{f}_i \cdot \phi_i + \int_\Omega \mathbf{f}_{old}^h \cdot \phi_i\right], \end{aligned} \tag{3.19}$$

$\mathbf{g} = \mathbf{0}$.

The algebraic system (3.17) belongs to the class of saddle point type systems.

4

Solvers for Linear Saddle Point Problems

This chapter is devoted to present:

- basic algebraic properties of the saddle point matrix $\mathcal{A}$ such as existence of various factorizations, invertibility, spectral properties, and conditioning;
- overview of solution algorithms and detailed presentation of the Schur complement reduction method;
- general strategies for preconditioning of the saddle point system arising from the Navier-Stokes equations;
- two popular techniques of block preconditioners as *Least Squares Commutator (LSC) Preconditioner* and *Semi-Implicit Method for Pressure-Linked Equations (SIMPLE)*, based on the Schur complement approximation;
- overview of multigrid methods and
- overview of the used sparse direct solvers.

The presentation of this chapter is based on the works listed below. Sections 4.1, 4.2 and 4.3 follow [1]. The works [4], [5] and [10] are sources for the presentation of Section 4.4. Section 4.5 follows [7].

4.1 Properties of Saddle Point Matrices

Throughout this section, we will assume that the systems of equations that have to be solved have the form arising from Picard iteration (2.41) applied to the Navier-Stokes equations

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix}, \tag{4.1}$$

with $\mathbf{K} \in \mathbb{R}^{n_v \times n_v}$, $B \in \mathbb{R}^{n_p \times n_v}$, $\mathfrak{f} \in \mathbb{R}^{n_v}$ and $\mathbf{g} \in \mathbb{R}^{n_p}$. Here $n_v = d \times N_v$, $n_p = N_p$, where N_v is the number of degrees of freedom for one component of the velocity, N_p is the number of pressure degrees of freedom, d is the dimension of the problem. The saddle point matrix $\mathcal{A} \in \mathbb{R}^{(n_v+n_p)\times(n_v+n_p)}$ has the form

$$\mathcal{A} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix}. \tag{4.2}$$

Assume that the matrix $\mathbf{K}$ is nonsingular, then the saddle point matrix $\mathcal{A}$ admits the following block triangular factorization

$$\mathcal{A} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} = \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & O \\ O & S \end{bmatrix} \begin{bmatrix} I & \mathbf{K}^{-1}B^T \\ O & I \end{bmatrix}, \tag{4.3}$$

where $S = -B\mathbf{K}^{-1}B^T$ is the Schur complement of $\mathbf{K}$ in $\mathcal{A}$.

Also useful are the equivalent factorizations

$$\mathcal{A} = \begin{bmatrix} \mathbf{K} & O \\ B & S \end{bmatrix} \begin{bmatrix} I & \mathbf{K}^{-1}B^T \\ O & I \end{bmatrix} \tag{4.4}$$

and

$$\mathcal{A} = \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}. \tag{4.5}$$

From any of the block decompositions (4.3)-(4.5) it follows, that $\mathcal{A}$ is nonsingular, if and only if S is, because exactly in this situation, all factors in (4.3)-(4.5) are nonsingular.

4.1.1 Solvability Conditions

Consider the case, where $\mathbf{K}$ is symmetric positive definite. From linear algebra it is known that the inverse of $\mathbf{K}$ is also symmetric positive definite and if B^T has full columnn rank, the matrix $B\mathbf{K}^{-1}B^T$ is a symmetric positive definite matrix. Thus, the Schur complement S is a symmetric negative definite matrix.

It is obvious that S, and thus $\mathcal{A}$, is invertible if and only if B^T has a full column rank, since in this case the Schur complement is symmetric negative definite. Then the problem (4.1) has a unique solution.

If $\mathbf{K}$ is indefinite, then $\mathcal{A}$ may be singular, even if B has a full rank. However, $\mathcal{A}$ will be invertible if $\mathbf{K}$ is positive definite on $ker(B)$.

Theorem 4.1 (**Necessary and sufficient condition for non-singularity of the saddle point matrix).** *Assume that* $\mathbf{K}$ *is symmetric positive semidefinite and* B *has full rank. Then a necessary and sufficient condition for the saddle point matrix* $\mathcal{A}$ *to be nonsingular is*

$$ker(\mathbf{K}) \cap ker(B) = \{0\}.$$

Proof. Let $\mathbf{u} = \begin{pmatrix} x \\ y \end{pmatrix}$ be such that

$$\mathcal{A}\mathbf{u} = 0.$$

Hence

$$\begin{aligned} \mathbf{K}x + B^T y &= 0 \\ Bx &= 0. \end{aligned}$$

It follows that

$$x^T\mathbf{K}x = -x^T B^T y = -(Bx)^T y = 0.$$

Since $\mathbf{K}$ is symmetric positive semidefinite,

$$x^T\mathbf{K}x = x^T\mathbf{K}^{1/2}\mathbf{K}^{1/2}x = 0$$

implies

$$\left\|\mathbf{K}^{1/2}x\right\|_2 = 0,$$

and therefore

$$\mathbf{K}^{1/2}x = 0 \quad \Rightarrow \quad \mathbf{K}x = 0.$$

This means that

$$x \in ker(\mathbf{K}) \cap ker(B),$$

thus

$$x = 0.$$

Also $y = 0$ since $B^T y = 0$ and B^T has a full rank. Therefore $\mathbf{u} = 0$, and $\mathcal{A}$ is nonsingular. This proves the sufficiency of the condition.

Assume now that $ker(\mathbf{K}) \cap ker(B) \neq \{0\}$. Take

$$x \in ker(\mathbf{K}) \cap ker(B), \quad x \neq 0.$$

Let

$$\mathbf{u} = \begin{pmatrix} x \\ 0 \end{pmatrix}.$$

Then we have

$$\mathcal{A}\mathbf{u} = 0.$$

It implies that $\mathcal{A}$ is singular. Hence, the condition is also necessary. □

In general case a necessary condition for invertibility is provided by the following theorem.

Theorem 4.2 (**Necessary condition for non-singularity of the saddle point matrix**). *If the matrix*

$$\mathcal{A} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix}$$

is nonsingular, then $rank(B) = n_p$ *and* $rank \begin{pmatrix} \mathbf{K} \\ B \end{pmatrix} = n_v + n_p$.

Proof. If $rank(B) < n_p$ then there exists a nonzero vector $y \in \mathbb{R}^{n_p}$ with

$$B^T y = 0.$$

Therefore, letting

$$\mathbf{u} = \begin{pmatrix} 0 \\ y \end{pmatrix},$$

we get

$$\mathcal{A}\mathbf{u} = 0,$$

a contradiction.

If $rank \begin{pmatrix} \mathbf{K} \\ B \end{pmatrix} < n_v + n_p$ then there exists a nonzero vector $x \in \mathbb{R}^{(n_v+n_p)}$ such that

$$\begin{pmatrix} \mathbf{K} \\ B \end{pmatrix} x = 0.$$

Let

$$\mathbf{u} = \begin{pmatrix} x \\ 0 \end{pmatrix},$$

then we get

$$\mathcal{A}\mathbf{u} = 0,$$

a contradiction. □

It is easy to show that these conditions are not sufficient to ensure the invertibility of $\mathcal{A}$. Some additional conditions are needed. Recall that for any matrix $A \in \mathbb{R}^{n\times n}$ one can write $A = H + H'$, where $H = \frac{1}{2}(A + A^T)$ and $H' = \frac{1}{2}(A - A^T)$ are symmetric and skew-symmetric part of A, respectively.

Theorem 4.3 (**Necessary and sufficient condition of invertibility of $\mathcal{A}$**). *Assume that H, the symmetric part of $\mathbf{K}$, is positive semidefinite and B has full rank. Then*

(*i*) $ker(H) \cap ker(B) = \{0\} \Rightarrow \mathcal{A}$ *invertible,*

(*ii*) $\mathcal{A}$ *invertible* $\Rightarrow ker(\mathbf{K}) \cap ker(B) = \{0\}$.

The converses of (i)-(ii) do not hold in general.

Proof. The proof is similar to the proof of the Theorem 4.1, and can be found in [2], Lemma 1.1. □

4.1.2 The Inverse of a Saddle Point Matrix

If $\mathbf{K}$ is nonsingular, then $\mathcal{A}$ is invertible if and only if $S = -B\mathbf{K}^{-1}B^T$ is nonsingular.

Lemma 4.4 (**General formula for matrix inversion in block form**). *Let the $(m + n) \times (m + n)$ matrix M be partitioned into a block form*

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix},$$

where the $m \times m$ matrix A and $n \times n$ matrix D are invertible. Then

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X & Y \\ Z & U \end{bmatrix} = \begin{bmatrix} I_m & O \\ O & I_n \end{bmatrix}$$

$$\Rightarrow \quad \begin{bmatrix} X & Y \\ Z & U \end{bmatrix} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix}.$$

Using the factorization (4.5) one can write the following expression for the $\mathcal{A}^{-1}$

$$\mathcal{A}^{-1} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix}^{-1} = \left(\begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix} \right)^{-1} = \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}^{-1} \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix}^{-1}. \tag{4.6}$$

Find the inverse for both matrices in (4.6) using the Lemma 4.4

$$\begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{K}^{-1} & -\mathbf{K}^{-1}B^T S^{-1} \\ O & S^{-1} \end{bmatrix}, \tag{4.7}$$

$$\begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix}^{-1} = \begin{bmatrix} I & O \\ -B\mathbf{K}^{-1} & I \end{bmatrix}. \tag{4.8}$$

Multiplying (4.7) and (4.8), one rewrites (4.6)

$$\mathcal{A}^{-1} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{K}^{-1} + \mathbf{K}^{-1}B^T S^{-1} B\mathbf{K}^{-1} & -\mathbf{K}^{-1}B^T S^{-1} \\ -S^{-1}B\mathbf{K}^{-1} & S^{-1} \end{bmatrix}. \tag{4.9}$$

However, such an expression is of limited interest in the numerical solution of saddle point problems.

In the finite element context the nonsingularity of $\mathcal{A}$ is not sufficient to ensure meaningful computed solutions. In [9] is proved that B has full column rank if and only if

$$\inf_{\mathbf{q} \in \mathbb{R}^{n_Q}, \mathbf{q} \neq \mathbf{0}} \sup_{\mathbf{v} \in \mathbb{R}^{n_V}, \mathbf{v} \neq \mathbf{0}} \frac{\mathbf{v}^T B^T \mathbf{q}}{\|\mathbf{v}\|_2 \|\mathbf{q}\|_2} \geq \beta > 0.$$

In order for the discrete problem to be well-posed it is essential that the saddle point matrices remain *uniformly invertible* as h, the mesh size parameter, goes to zero. This means that an appropriate condition number of $\mathcal{A}$ remains bounded as $h \to 0$. Sufficient conditions for this to happen include the already-mentioned discrete inf-sup (LBB) condition.

4.1.3 Spectral Properties of Saddle Point Matrices

Theorem 4.5 (Rusten and Winther (1992). Eigenvalue bounds for the symmetric case). *Assume* $\mathbf{K}$ *is symmetric positive definite and* B *has full rank. Let* μ_1 *and* μ_n *denote the largest and smallest eigenvalues of* $\mathbf{K}$*, and let* σ_1 *and* σ_m

denote the largest and smallest singular values of B. *Let* $\sigma(\mathbf{K})$ *denote the spectrum of* $\mathcal{A}$. *Then*

$$\sigma(\mathcal{A}) \subset I^- \cap I^+,$$

where

$$I^- = \left[\frac{1}{2}\left(\mu_n - \sqrt{\mu_n^2 + 4\sigma_1^2}\right), \frac{1}{2}\left(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2}\right)\right]$$

and

$$I^+ = \left[\mu_n, \frac{1}{2}\left(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2}\right)\right].$$

In the general case, not much can be said about the eigenvalues of $\mathcal{A}$.

4.1.4 Conditioning Issues

Saddle point systems that arise in practice can be very badly conditioned. In some cases the special structure of the saddle point matrix $\mathcal{A}$ can be exploited to avoid or mitigate the effect of ill-conditioning.

Consider, for the simplicity, a standard saddle point problem, where $\mathbf{K}$ is symmetric positive definite and B has full rank. In this case $\mathcal{A}$ is symmetric and its spectral condition number is given by

$$\kappa(\mathcal{A}) = \frac{\max|\lambda(\mathcal{A})|}{\min|\lambda(\mathcal{A})|}.$$

From Theorem 4.5 one can see that the condition number of $\mathcal{A}$ grows unboundedly as either $\mu_n = \lambda_{min}(\mathbf{K})$ or $\sigma_n = \sigma_{min}(B)$ goes to zero (assuming that $\lambda_{max}(\mathbf{K})$ and $\sigma_{max}(B)$ are kept constant). This growth of the condition number of $\mathcal{A}$ means that the rate of convergence of most iterative solvers (like Krylov subspace methods) deteriorates as the problem size increases. Preconditioning may be used to reduce or even eliminate this dependency on h in many cases. Similar considerations apply to nonsymmetric saddle point problems.

4.2 Overview of Solution Algorithms

Besides the usual classification on direct and iterative methods, solution algorithms for linear saddle point problems can be subdivided into two categories, which are called *segregated* and *coupled* (or "all at once") methods.

Segregated methods compute the two unknown vectors $\mathbf{u}$ and $\mathbf{p}$ separately. In some cases it is $\mathbf{u}$ to be computed first, in others it is $\mathbf{p}$. This approach involves the solution of two linear systems of a size smaller than $n_v + n_p$ (called reduced systems), one for each of $\mathbf{u}$ and $\mathbf{p}$. Segregated methods can be either direct or iterative, or involve a combination of the two. For example, one of the reduced systems could be solved by a direct method and the other iteratively. One of the main representatives of the segregated approach is the Schur complement reduction method, which is based on a block LU factorization of $\mathcal{A}$.

Coupled methods, on the other hand, deal with the system (4.1) as a whole, computing $\mathbf{u}$ and $\mathbf{p}$ (or approximations to them) simultaneously and without making explicit use of reduced systems. These methods include both direct solvers based on

triangular factorizations of the global matrix $\mathcal{A}$, and iterative algorithms like Krylov subspace methods applied to the entire system, typically with some form of preconditioning. Preconditioning tends to blur the distinction between direct and iterative solvers, and also that between segregated and coupled schemes. This is because direct solvers may be used to construct preconditioners, and also preconditioners for coupled iterative schemes are frequently based on segregated methods.

4.3 The Schur Complement Reduction Method

Consider the saddle point system (4.1), or

$$\begin{aligned} \mathbf{K}\mathbf{u} + B^T\mathbf{p} &= \mathfrak{f} \\ B\mathbf{u} &= \mathbf{g}. \end{aligned} \tag{4.10}$$

We assume that both $\mathbf{K}$ and $\mathcal{A}$ are nonsingular. By (4.3) this implies that $S = -B\mathbf{K}^{-1}B^T$ is also nonsingular. Multyplying both sides of the first equation from the right-hand side with $B\mathbf{K}^{-1}$, one obtains

$$B\mathbf{u} + B\mathbf{K}^{-1}B^T\mathbf{p} = B\mathbf{K}^{-1}\mathfrak{f}.$$

Using $B\mathbf{u} = \mathbf{g}$ and rearranging, one finds

$$B\mathbf{K}^{-1}B^T\mathbf{p} = B\mathbf{K}^{-1}\mathfrak{f} - \mathbf{g}, \tag{4.11}$$

a reduced system of order $m = N_p$ for $\mathbf{p}$ involving the (negative) Schur complement $-S = B\mathbf{K}^{-1}B^T$. Once $\mathbf{p}^*$ has been computed from (4.11), $\mathbf{u}^*$ can be obtained by solving

$$\mathbf{K}\mathbf{u} = \mathfrak{f} - B^T\mathbf{p}^*, \tag{4.12}$$

a reduced system of order $n = N_v$ for $\mathbf{u}$ involving the $(1,1)$ block, $\mathbf{K}$. Note that this is just block Gaussian elimination applied to (4.1). Indeed, using the block LU factorization (4.5) one gets the transformed system

$$\begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} I & O \\ -B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix},$$

that is,

$$\begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} - B\mathbf{K}^{-1}\mathfrak{f} \end{bmatrix}.$$

Solving this block upper triangular system by block backsubstitution leads to the two reduced systems (4.11) and (4.12) for $\mathbf{p}$ and $\mathbf{u}$. These systems can be solved either directly or iteratively. In the important special case where $\mathbf{K}$ and $-S$ are symmetric positive definite, highly reliable methods such as Cholesky factorization or the conjugate gradient (CG) method can be applied.

This approach is attractive if the order m of the reduced system (4.11) is small and if linear systems with coefficient matrix $\mathbf{K}$ can be solved efficiently.

The main disadvantage is the fact that the Schur complement $S = -B\mathbf{K}^{-1}B^T$ may be completely full and too expensive to compute or to factor. Numerical instabilities may also be a concern when forming S, especially when $\mathbf{K}$ is ill-conditioned.

If S is too expensive to form or factor, Schur complement reduction can still be applied by solving (4.11) by iterative methods that do not need access to individual entries of S, but only need S in the form of matrix-vector products. The Schur complement system (4.11) may be rather ill-conditioned, in which case preconditioning will be required. Preconditioning the system (4.11) is nontrivial when S is not explicitly available.

In the next section technologies of efficient preconditioning algorithms for the Navier-Stokes equations are discussed.

4.4 Preconditioning

The term *preconditioning* refers to transforming the linear system $\mathcal{A}x = b$ into another system with more favourable properties for its iterative solution. A preconditioner is a matrix $\mathcal{P}$ or $\mathcal{P}^{-1}$ that effects such a transformation. Generally speaking, preconditioning attempts to improve the spectral properties of the system matrix.

For symmetric problems, the rate of convergence of Krylov subspace methods like CG or MINRES depends on the distribution of the eigenvalues of $\mathcal{A}$. Ideally, the preconditioned matrix will have a smaller spectral condition number, and/or eigenvalues clustered around 1.

For nonsymmetric problems the situation is more complicated, and the eigenvalues may not describe the convergence of nonsymmetric matrix iterations like GMRES. Nevertheless, a clustered spectrum (away from 0) often results in rapid convergence.

For saddle point problems, the construction of high-quality preconditioners necessitates exploiting the block structure of the problem, together with detailed knowledge about the origin and structure of the various blocks. The choice of a preconditioner is strongly problem-dependent. For example, techniques that give excellent results for the time-dependent flow problem may be completely inadequate for the steady-state case.

In the context of the Navier-Stokes equations, block preconditioners are mostly based on the block LDU factorization (4.3)

$$\mathcal{A} = \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} = \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & O \\ O & S \end{bmatrix} \begin{bmatrix} I & \mathbf{K}^{-1}B^T \\ O & I \end{bmatrix},$$

where S is the Schur complement disccused above. Most preconditioners are based on a combination of these blocks and a suitable approximation of the Schur complement matrix.

Consider the following strategy [4], that is derived from the block factorization:

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} = \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}. \tag{4.13}$$

This implies that

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}^{-1} = \begin{bmatrix} I & O \\ B\mathbf{K}^{-1} & I \end{bmatrix}, \tag{4.14}$$

which suggests a preconditioning strategy for (4.1). If it were possible to use the matrix

$$\mathcal{P} = \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix} \tag{4.15}$$

as a right-oriented preconditioner, then the preconditioned operator would be the one given in (4.14).

Consider the following generalized eigenvalue problem, one can determine the eigenvalues of the preconditioned system

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix}. \tag{4.16}$$

From the first row of (4.16) one obtains

$$(1-\lambda)(\mathbf{K}\mathbf{u} + B^T\mathbf{p}) = 0.$$

This is only possible if $\lambda = 1$ or $\mathbf{K}\mathbf{u} + B^T\mathbf{p} = 0$. In the first case one has an eigenvalue equal to 1 of multiplicity N_v. For the second case it is

$$\mathbf{u} = -\mathbf{K}^{-1}B^T\mathbf{p}.$$

From the second row of (4.16) it follows

$$B\mathbf{u} - \lambda S\mathbf{p} = 0.$$

Substituting $\mathbf{u} = -\mathbf{K}^{-1}B^T\mathbf{p}$ in the previous equation gives

$$-B\mathbf{K}^{-1}B^T\mathbf{p} = \lambda S\mathbf{p}. \tag{4.17}$$

It means that one has $\lambda = 1$ with multiplicity N_p. In practice, one cannot use $S = -B\mathbf{K}^{-1}B^T$ in (4.11), but (4.17) shows that a good approximation of the Schur complement matrix will influence the good convergence of the preconditioned system with $\mathcal{P}$.

Following [4], when any preconditioner $\mathcal{P}$ is used in a Krylov subspace iteration, each step requires the application of $\mathcal{P}^{-1}$ to a vector. To see the computational issues, it is useful to express $\mathcal{P}^{-1}$ in factored form

$$\begin{bmatrix} \mathbf{K} & B^T \\ O & S \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{K}^{-1} & O \\ O & I \end{bmatrix} \begin{bmatrix} I & -B^T \\ O & I \end{bmatrix} \begin{bmatrix} I & O \\ O & S^{-1} \end{bmatrix}. \tag{4.18}$$

This shows that two nontrivial operations are required to apply $\mathcal{P}^{-1}$: application of S^{-1} to a vector in the discrete pressure space, and application of $\mathbf{K}^{-1}$ to a vector in the discrete velocity space. These tasks are too expensive for a practical computation due to the expensive calculations and storage of such matrices. In general, $\mathbf{K}^{-1}$ is approximated by a matrix $\hat{\mathbf{K}}^{-1}$ obtained by a small number of iterations with an iterative method.

Thus, the block triangular preconditioning (4.13) involves the solution of the transformated system

$$\mathcal{P}\mathbf{z} = \mathbf{r},$$

where

$$\mathbf{z} = \begin{pmatrix} z_u \\ z_p \end{pmatrix} \text{ and } \mathbf{r} = \begin{pmatrix} r_u \\ r_p \end{pmatrix}.$$

The algorithm below describes the process of solving.

Algorithm 4.1 (Basic Form of the Preconditioner $\mathcal{P}$).

1. Solve $Sz_p = r_p$.
2. Update $r_u = r_u - B^T z_p$.
3. Solve $\mathbf{K}z_u = r_u$.

As one can see the preconditioner belongs to segregated approach and involves the solution of two subproblems, one associated with the pressure and the other with the velocity problem.

As mentioned above, the Schur complement matrix is not formed, but approximated by a simple matrix $\hat{S}$. How this approximation is done defines the various block preconditioners. Now two popular techniques of block preconditioners are reviewed.

4.4.1 The Least Squares Commutator Preconditioner

The method *Least Squares Commutator (LSC) preconditioner* was presented by Elman et al. [4] in 2006. The method is based on approximating the commutator of the convection-diffusion operator with the gradient operator.

The convection-diffusion operator defined on the velocity space can be written as

$$\mathcal{L} = -\nu\Delta + \mathbf{w}_h \cdot \nabla, \tag{4.19}$$

where $\mathbf{w}_h$ is the approximation to the discrete velocity, computed in the most recent Picard iteration. Suppose that there is an analogous operator

$$\mathcal{L}_p = (-\nu\Delta + \mathbf{w}_h \cdot \nabla)_p, \tag{4.20}$$

defined on the pressure space. It is not necessary to ascribe any physical meaning to this operator, it will only be used to construct an algorithm. Consider the commutator of the convection-diffusion operators with the gradient operator, as follows

$$\varepsilon = \mathcal{L}\nabla - \nabla\mathcal{L}_p = (-\nu\Delta + \mathbf{w}_h \cdot \nabla)\nabla - \nabla(-\nu\Delta + \mathbf{w}_h \cdot \nabla)_p. \tag{4.21}$$

If $\mathbf{w}_h$ is constant, then this expression would be zero on the interior of Ω, and it is small for smooth $\mathbf{w}$. So if K_p is a discretization of (4.20), then the discrete commutator

$$\varepsilon_h = \mathbf{K}B^T - B^T K_p \tag{4.22}$$

will also be small. This means that

$$\mathbf{K}B^T \approx B^T K_p. \tag{4.23}$$

To isolate the Schur complement one has to multiply (4.23) from left by $B\mathbf{K}^{-1}$ and from right by K_p^{-1}. Thus, an approximation to the Schur complement matrix takes the form

$$S = -B\mathbf{K}^{-1}B^T \approx \hat{S} = -(BB^T)K_p^{-1}. \tag{4.24}$$

To approximate the matrix operator K_p also the idea that the discrete commutator (4.22) becomes small is used. This is done by solving a least squares problem.

Suppose k_j is the j-th column of the matrix K_p, then the least squares problem with respect to the Euclidean norm has the form

$$\min \left\| B^T k_j - \left[\mathbf{K} B^T \right]_j \right\|_2 , \tag{4.25}$$

where $\left[\mathbf{K} B^T \right]_j$ is the jth column of the matrix $\mathbf{K} B^T$. The normal equations associated with this problem are

$$\begin{aligned} \left(B^T \right)^T B^T k_j &= \left(B^T \right)^T \left[\mathbf{K} B^T \right]_j \\ \left(B B^T \right) k_j &= B \left[\mathbf{K} B^T \right]_j . \end{aligned} \tag{4.26}$$

An equivalent formulation is that K_p minimizes the Frobenius norm of the error in the complete system

$$\min \left\| B^T K_p - \mathbf{K} B^T \right\|_F . \tag{4.27}$$

According to (4.26) the solution is

$$K_p = \left(B B^T \right)^{-1} B \mathbf{K} B^T. \tag{4.28}$$

The resulting Schur complement preconditioner becomes

$$S = -B \mathbf{K}^{-1} B^T \approx -(B B^T)^{-1} B \mathbf{K} B^T (B B^T)^{-1}. \tag{4.29}$$

To accelerate convergence Elman et al. [4] proposed alternate solutions of (4.23). These modifications include the use of diagonal scaling in (4.27). Consider

$$M_2^{-1} B^T K_p \approx M_2^{-1} \mathbf{K} M_1^{-1} B^T, \tag{4.30}$$

where M_2 and M_1 are diagonal matrices. The operator M_2^{-1} can be thought of as a weight matrix that transforms (4.27) into a weighted least-squares problem. The introduction of M_1^{-1} can be viewed as a way to precondition $\mathbf{K}$ and B^T so that they are more amenable to commuting. Multiplying (4.30) from the left by $B \mathbf{K}^{-1} M_2$, one obtains

$$\begin{aligned} B \mathbf{K}^{-1} M_2 M_2^{-1} B^T K_p &\approx B \mathbf{K}^{-1} M_2 M_2^{-1} \mathbf{K} M_1^{-1} B^T \\ B \mathbf{K}^{-1} B^T K_p &\approx B \mathbf{K}^{-1} \mathbf{K} M_1^{-1} B^T \\ B \mathbf{K}^{-1} B^T K_p &\approx B M_1^{-1} B^T \\ B \mathbf{K}^{-1} B^T &\approx (B M_1^{-1} B^T) K_p^{-1}. \end{aligned}$$

Then the inverse Schur complement is approximated by

$$\left(B \mathbf{K}^{-1} B^T \right)^{-1} \approx K_p (B M_1^{-1} B^T)^{-1}, \tag{4.31}$$

where

$$K_p = \left(B M_2^{-2} B^T \right)^{-1} B M_2^{-2} \mathbf{K} M_1^{-1} B^T. \tag{4.32}$$

The net effect of "preconditioning" the commutator equation in this way is that a new definition of K_p is given and the inverse discrete Poisson operator is now replaced by a discrete variable-coefficient diffusion operator, when M_1 is a diagonal matrix.

The strategy for choosing of a diagonal matrix M_1 consists in the idea that a version of the commutator is small on some components of the pressure space. In [4] Elman suggested to choose as M_1 the diagonal matrix whose entries are those on the diagonal of the velocity mass matrix $\mathbf{Q}_v$

$$M_1 = \hat{\mathbf{Q}}_v = diag(\mathbf{Q}_v), \tag{4.33}$$

with

$$\mathbf{Q}_v = \mathbf{q}_{ij}, \quad \mathbf{q}_{ij} = \int_\Omega \phi_j \cdot \phi_i,$$

where $\{\phi_i\}$ is the basis for the discrete velocity space. As M_2 is chosen the matrix $M_1^{1/2}$ to make the two variable-coefficient Poisson operators identical.

The resulting approximation to the Schur complement matrix S with these choices for M_1 and M_2 looks like

$$\hat{S} = -(B\hat{\mathbf{Q}}_v^{-1}B^T)(B\hat{\mathbf{Q}}_v^{-1}\mathbf{K}\hat{\mathbf{Q}}_v^{-1}B^T)^{-1}(B\hat{\mathbf{Q}}_v^{-1}B^T), \tag{4.34}$$

and the corresponding inverse matrix is

$$\hat{S}^{-1} = -(B\hat{\mathbf{Q}}_v^{-1}B^T)^{-1}(B\hat{\mathbf{Q}}_v^{-1}\mathbf{K}\hat{\mathbf{Q}}_v^{-1}B^T)(B\hat{\mathbf{Q}}_v^{-1}B^T)^{-1}. \tag{4.35}$$

Using a block triangular matrix preconditioner

$$\mathcal{P} = \begin{bmatrix} \mathbf{K} & B^T \\ O & \hat{S} \end{bmatrix} \tag{4.36}$$

with $\hat{S}$ from (4.34) in the $(2,2)$ block is called least-squares commutator preconditioning. Implementing the strategy involves two discrete Poisson solves and matrix-vector productes with the matrices B, B^T, $\mathbf{K}$ and $\hat{\mathbf{Q}}_v^{-1}$. The main advantage of the least-square approach is that it is fully automated, it is defined in terms of matrices such as $\mathbf{K}$ and B available in the statement of the problem.

The algorithm for the LSC preconditioner reads

Algorithm 4.2 (LSC preconditioner).

1. Solve $S_P z_p^* = r_p$, where $S_P = B\hat{Q}_v^{-1}B^T$.
2. Update $r_p = B\hat{Q}_v^{-1}\mathbf{K}\hat{Q}_v^{-1}B^T z_p^*$.
3. Solve $S_P z_p = -r_p$.
4. Update $r_u = r_u - B^T z_p$.
5. Solve $\mathbf{K} z_u = r_u$.

4.4.2 The SIMPLE Preconditioner

The SIMPLE method (Semi-Implicit Method for Pressure-Linked Equations) has been introduced by Patankar & Spalding [11] as an iterative method to solve the finite volume discretized incompressible Navier-Stokes equations. The algorithm is based on the following steps

- First the pressure is assumed to be known from the previous iteration.
- Then the velocity is solved from the momentum equations. The newly obtained velocities do not satisfy the continuity equation since the pressure is only a guess.
- In the next substeps the velocities and pressures are corrected in order to satisfy the discrete continuity equation.

The algorithm follows from a block LU decomposition (4.4)

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & O \\ B & -B\mathbf{K}^{-1}B^T \end{bmatrix} \begin{bmatrix} I & \mathbf{K}^{-1}B^T \\ O & I \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix}. \tag{4.37}$$

The approximation $\mathbf{K}^{-1}$ as $D^{-1} = \mathrm{diag}(\mathbf{K})^{-1}$ leads to the SIMPLE algorithm. In this case the approximation of the Schur complement matrix is given by

$$\hat{S} = -BD^{-1}B^T,$$

and the decomposition looks like

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \approx \begin{bmatrix} \mathbf{K} & O \\ B & \hat{S} \end{bmatrix} \begin{bmatrix} I & D^{-1}B^T \\ O & I \end{bmatrix}. \tag{4.38}$$

Thus, one iteration of SIMPLE corresponds to the solving of the following system

$$\begin{bmatrix} \mathbf{K} & O \\ B & \hat{S} \end{bmatrix} \begin{bmatrix} I & D^{-1}B^T \\ O & I \end{bmatrix} \begin{bmatrix} \delta\mathbf{u} \\ \delta p \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix} - \begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \delta\mathbf{u}^{(k)} \\ \delta p^{(k)} \end{bmatrix} = \begin{bmatrix} r_u \\ r_p \end{bmatrix}, \tag{4.39}$$

which can be represented as

$$\begin{bmatrix} \mathbf{K} & O \\ B & \hat{S} \end{bmatrix} \begin{bmatrix} \delta\mathbf{u}^* \\ \delta p^* \end{bmatrix} = \begin{bmatrix} r_u \\ r_p \end{bmatrix}, \tag{4.40}$$

and

$$\begin{bmatrix} I & D^{-1}B^T \\ O & I \end{bmatrix} \begin{bmatrix} \delta\mathbf{u} \\ \delta p \end{bmatrix} = \begin{bmatrix} \delta\mathbf{u}^* \\ \delta p^* \end{bmatrix}. \tag{4.41}$$

After recursively solving these two systems, one has to update the velocities and pressure from the previous iteration

$$\begin{bmatrix} \mathbf{u}^{(k+1)} \\ p^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{u}^{(k)} \\ p^{(k)} \end{bmatrix} + \omega \begin{bmatrix} \delta\mathbf{u} \\ \delta p \end{bmatrix}, \tag{4.42}$$

where ω is a parameter in $(0, 1]$ that damps the pressure update.

The main attraction of SIMPLE is that it is easy to implement. SIMPLE does not affect the terms that operate on the velocity, but it perturbs the pressure operator in the momentum equation. When D^{-1} is a good approximation to $\mathbf{K}^{-1}$, then the error of this method is closed to a zero matrix. The results of the work [5] show that the diagonal approximation can yield poor results because the diagonal approximation does not capture enough information about the convection operator. This means that effectiveness of SIMPLE diminishes for flows that are convection-dominated. The next disadvantage is that the method performs worse when the spatial mesh size becomes small.

Algorithm 4.3 (**SIMPLE preconditioner**).

1. $u^{(k)}$, $p^{(k)}$ are given.
2. Set $r_u = \mathfrak{f} - \mathbf{K}u^{(k)} - B^T p^{(k)}, \quad r_p = g - Bu^{(k)}$.
3. Solve $\mathbf{K}\delta u^* = r_u$.
4. Solve $\hat{S}\delta p^* = r_p - B\delta u^*$.
5. Update $\delta p = \delta p^*$.
6. Update $\delta u = \delta u^* - D^{-1}B^T\delta p^*$.
7. Update $u^{(k+1)} = u^{(k)} + \omega_u \delta u, \quad p^{(k+1)} = p^{(k)} + \omega_p \delta p$.

4.5 Coupled Multigrid

The next preconditioner, which is studied in this work, is a coupled multigrid method and it will be briefly described in this subsection.

A multigrid method is defined by

- the grid hierarchy,
- the grid transfer operators (function prolongation, defect restriction and function restriction),
- the smoother on finer levels,
- the coarse grid solver.

4.5.1 Transfer Between the Levels of the Multigrid Hierarchy

The function prolongation

Consider the transfer (prolongation) from a finite element space V_{l-1}^h to a finite element space V_l^h. Let $\mathcal{T}_{l-1}$ and $\mathcal{T}_l$ be the corresponding triangulations of the domain Ω such that $\mathcal{T}_l$ originates either from refinement of $\mathcal{T}_{l-1}$ or $\mathcal{T}_{l-1} = \mathcal{T}_l$. The second case is relevant in the multiple discretisation multilevel method.

Let Σ_l^h be a discontinuous finite element space defined on $\mathcal{T}_l$

$$\Sigma_l^h = \left\{ w \in L^2(\Omega) : w|_K \in S_l^h(K) \quad K \in \mathcal{T}_l \right\}.$$

The choice of the local spaces $S_l^h(K)$ depends on V_{l-1}^h and V_l^h. It has to be done such that the inclusion

$$V_{l-1}^h + V_l^h \subset \Sigma_l^h \tag{4.43}$$

holds.

The transfer operator is based on the concept of nodal functionals. For each mesh cell $K \in \mathcal{T}_l$ and for the finite element space Σ_l^h exists a local finite element basis $\left\{ \psi_{l,j}^h|_K \right\}$ and a dual basis $\left\{ N_{l,j}^K \right\}$ of local nodal functionals such that

$$N_{l,j}^K \left(\psi_{l,j}^h|_K \right) = \delta_{ij}, \quad 0 \le i,j \le \dim\left(S_l^h(K) \right),$$

where δ_{ij} is the Kroneker delta.

Let $\left\{ \varphi_{ij}^h \right\}$ be a finite element basis of V_l^h. The indices j are called nodes or degrees of freedom. The set of all nodes of V_l^h is denoted by $I_l(V_l^h)$. The set of local nodes with respect to the mesh cell K is given by

$$I_l\left(K, V_l^h\right) = \left\{ i \in I_l\left(V_l^h\right) : \quad \operatorname{supp}\left(\varphi_{l,i}^h\right) \cap K \neq 0 \right\}. \tag{4.44}$$

Furthermore, for any node $j \in I_l\left(V_l^h\right)$ is defined

$$\mathcal{T}_{l,j} = \left\{ K \in \mathcal{T}_l : \quad j \in I_l\left(K, \Sigma_l^h\right) \right\}, \tag{4.45}$$

the set of all mesh cells who are connected to the node j. Then, the global nodal functional which is associated with a node $j \in I_l\left(V_l^h\right)$ and whose argument is a function $w^h \in \Sigma_l^h$ is defined by the arithmetic mean of local nodal functionals

$$N_{l,j}\left(w^h\right) = \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,j}\right)} \sum_{K \in \mathcal{T}_{i,j}} N_{l,j}^K\left(w^h\big|_K\right), \quad w^h \in \Sigma_l^h,$$

where $\operatorname{card}\left(\mathcal{T}_{l,j}\right)$ denotes the number of mesh cells in $\mathcal{T}_{l,j}$. The transfer operator for the prolongation is defined with the help of the global nodal functionals:

$$P_{l-1}^l : \Sigma_l^h \to V_l^h \quad P_{l-1}^l\left(w^h\right) = \sum_{i=1}^{\dim\left(V_l^h\right)} N_{l,i}(w^h)\varphi_{l,i}^h. \tag{4.46}$$

From the inclusion (4.43) follows that this operator is defined especially for functions from V_{l-1}^h.

Let $\left\{\varphi_{l-1,i}^h\right\}$ be a finite element basis of V_{l-1}^h and

$$w_{l-1}^h = \sum_{i=1}^{\dim\left(V_l^h\right)} w_{l-1,i}\varphi_{l-1,i}^h \in V_{l-1}^h.$$

For evaluating the coefficient of $\varphi_{l,i}^h$ for the prolongated function, one has to compute

$$\begin{aligned} N_{l,i}\left(w_{l-1}^h\right) &= \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,i}\right)} \sum_{K \in \mathcal{T}_{l,i}} N_{l,i}^K\left(w_{l-1}^h\big|_K\right) \\ &= \frac{1}{\operatorname{card}\left(\mathcal{T}_{l,i}\right)} \sum_{K \in \mathcal{T}_{l,i}} \sum_{j=1}^{\dim\left(V_{l-1}^h\right)} w_{l-1,j} N_{l,i}^K\left(\varphi_{l-1,j}^h\big|_K\right). \end{aligned}$$

An algorithm for computing the prolongation (4.46) looks as follows.

Algorithm 4.4 (Prolongation). *Given the coefficient vector* $\mathbf{w}_{l-1}$ *of the finite element function* $w_{l-1}^h \in V_{l-1}^h$.

```
1:  w_l = 0
2:  card = 0
3:  for K ∈ T_l do
4:      for i ∈ I_l(K, V_l^h) do
5:          for j := 0; j < dim(V_{l-1}^h); j++ do
6:              if supp(φ_{l-1,j})^h |_K ∩ K = 0 then
7:                  continue
8:              end if
9:              w_l(i) := w_l(i) + w_{l-1}(j) N_{l,i}^K (φ_{l-1,j}^h |_K)
10:             card(i) := card(i) + 1
11:         end for
12:     end for
13: end for
14: for i := 0; i < dim(V_l^h); i++ do
15:     w_l(i) := w_l(i)/card(i)
```

16: ***end for***

The defect restriction

The definition of the operator for the defect restriction $R_l^{*,l-1} : (V_l^h)^* \to (V_{l-1}^h)^*$ uses the prolongation operator given in (4.46). Let $d_l \in (V_l^h)^*$ be a given defect functional, its restriction to $(V_{l-1}^h)^*$ is defined by

$$\int_\Omega R_l^{*,l-1}(d_l)\varphi_{l-1}^h d\mathbf{x} = \int_\Omega d_l P_l^{l-1}(d_l)(\varphi_{l-1}^h) d\mathbf{x} \quad \forall \varphi_{l-1}^h \in V_{l-1}^h.$$

The function restriction

In the multigrid approaches for solving the linear saddle point problem, the matrix of this problem has to be assembled also on the coarse levels. Therefore, the finite element functions $\mathbf{w}_{old}^h$ and $\mathbf{w}_{k-1}^h$ must be available on the coarse grids. A restriction operator $R_l^{l-1} : V_l^h \to V_{l-1}^h$ which maps a finite element function from the finite element space connected to level l in the multilevel hierarchy to a finite element function connected to level $l-1$ is necessary. A function restriction which is based on local L^2-projections and averaging is used.

The bases of V_l^h, V_{l-1}^h are again denoted by $\{\varphi_{l,j}^h\}$, $\{\varphi_{l-1,i}^h\}$. Let $\mathbf{w}_l^h \in V_l^h$ with

$$\mathbf{w}_l^h = \sum_{i=1}^{\dim\left(V_l^h\right)} w_{l,i}\varphi_{l,i}^h$$

be given. The goal is to compute a function

$$R_l^{l-1}(\mathbf{w}_l^h) = \sum_{i=1}^{\dim\left(V_{l-1}^h\right)} w_{l-1,i}\varphi_{l-1,i}^h.$$

Consider a mesh cell K on the geometric grid which is connected with V_{l-1}^h and assume that K possesses an affine reference transformation. Local values of the unknown coefficients $w_{l-1,i}$ are determined by the local L^2-projection

$$\sum_{i=1}^{\mathrm{card}\left(I_l(K,V_l^h)\right)} w_{l,i}|_K \left(\varphi_{l,i}^h, \varphi_{l-1,j}^h\right)_K = \sum_{i=1}^{\mathrm{card}\left(I_{l-1}(K,V_{l-1}^h)\right)} w_{l-1,i}|_K \left(\varphi_{l-1,i}^h, \varphi_{l-1,j}^h\right)_K$$

for all $j \in I_{l-1}(K, V_{l-1}^h)$, where $I_l(K, V_l^h)$ is defined in(4.44). The transformation to the reference cell $\hat{K}$ gives

$$\sum_{i=1}^{\mathrm{card}\left(I_l(K,V_l^h)\right)} w_{l,i}|_K \int_{\hat{K}} \hat{\varphi}_{l,i}^h \hat{\varphi}_{l-1,j}^h \left|\det J_K(\hat{\mathbf{x}})\right| \mathrm{d}\hat{\mathbf{x}} =$$

$$= \sum_{i=1}^{\mathrm{card}\left(I_{l-1}(K,V_{l-1}^h)\right)} w_{l-1,i}|_K \int_{\hat{K}} \hat{\varphi}_{l-1,i}^h \hat{\varphi}_{l-1,j}^h \left|\det J_K(\hat{\mathbf{x}})\right| \mathrm{d}\hat{\mathbf{x}}$$

for all $j \in I_{l-1}(K, V_{l-1}^h)$. Since $|\det J_K(\hat{\mathbf{x}})|$ is constant, this relation simplifies to

$$\sum_{i=1}^{\mathrm{card}\left(I_l(K,V_l^h)\right)} w_{l,i}|_K \int_{\hat{K}} \hat{\varphi}_{l,i}^h \hat{\varphi}_{l-1,j}^h \mathrm{d}\hat{\mathbf{x}} = \sum_{i=1}^{\mathrm{card}\left(I_{l-1}(K,V_{l-1}^h)\right)} w_{l-1,i}|_K \int_{\hat{K}} \hat{\varphi}_{l-1,i}^h \hat{\varphi}_{l-1,j}^h \mathrm{d}\hat{\mathbf{x}} \tag{4.47}$$

for all $j \in I_{l-1}(K, V_{l-1}^h)$. This is a linear system of the form

$$G w_l|_K = M w_{l-1}|_K .$$

Thus, the local values of the unknown coefficients are given by

$$w_{l-1}|_K = M^{-1} G w_l\big|_K = R w_l|_K . \tag{4.48}$$

The matrix R is independent of K. That means, for all other mesh cells whose bases on the reference mesh cell have the same form as for K, one also needs the matrix R. This will be the case very often. E.g., if the grids are uniformly refined and the same finite element space is used on every level, the matrix R is needed for each mesh cell on each level. This matrix R will be computed once and then stored in a data base. Then, only a local matrix-vector product has to be computed in (4.48) which leads to a very fast algorithm. The final restriction is computed by an averaging

$$w_{l-1,i} = \frac{1}{\mathrm{card}(\mathcal{T}_{l-1,i})} \sum_{K \in \mathcal{T}_{l-1,i}} w_{l-1,i}|_K .$$

For mesh cells with a non-affine reference transformation, it is used for simplicity also (4.47) such that they are handled in the same way as mesh cells with an affine transformation. One can consider this approach also as a function restriction which is a local L^2-pojection on the reference mesh cell and which is an approximation of a L^2-projection on the original mesh cell.

4.5.2 The Vanka Smoothers

The coupled multigrid method is used with local smoothers, so-caled Vanka-type smoothers. Vanka-type smoothers can be considered as block Gauss-Seidel methods. Let $\mathcal{V}^h$ and $\mathcal{Q}^h$ be the set of velocity and pressure degrees of freedom, respectively. These sets are decomposed into

$$\mathcal{V}^h = \cup_{j=1}^J \mathcal{V}_j^h, \quad \mathcal{Q}^h = \cup_{j=1}^J \mathcal{Q}_j^h. \tag{4.49}$$

The subsets are not required to be disjoint.

Let A_j be the block of the matrix $\mathcal{A}$ which is connected with the degrees of freedom of $\mathcal{W}_j^h = \mathcal{V}_j^h \cup \mathcal{Q}_j^h$, i.e. the intersection of the rows and columns of $\mathcal{A}$ with the global indices belonging to $\mathcal{W}_j^h$,

$$\mathcal{A} = \begin{pmatrix} A_j & B_j \\ C_j & 0 \end{pmatrix} \in \mathbb{R}^{\dim(\mathcal{W}_j^h) \times \dim(\mathcal{W}_j^h)}.$$

In addition, define

$$\mathcal{D}_j = \begin{pmatrix} \mathrm{diag}(A_j) & B_j \\ C_j & 0 \end{pmatrix} \in \mathbb{R}^{\dim(\mathcal{W}_j^h)\times\dim(\mathcal{W}_j^h)}.$$

Similarly, denote by $(\cdot)_j$ the restriction of a vector on the rows corresponding to the degrees of freedom on $\mathcal{W}_j^h$. Each smoothing step with a Vanka-type smoother consists in a loop over all sets $\mathcal{W}_j^h$, where for each $\mathcal{W}_j^h$ a local system of equations connected with the degrees of freedom in this set is solved. The local solutions are updated in a Gauss-Seidel manner. The diagonal Vanka smoother computes the velocity and pressure values connected to $\mathcal{W}_j^h$ by

$$\begin{pmatrix} w \\ r \end{pmatrix}_j := \begin{pmatrix} w \\ r \end{pmatrix}_j + \mathcal{D}^{-1} \left(\begin{pmatrix} f \\ g \end{pmatrix} - \mathcal{A} \begin{pmatrix} w \\ r \end{pmatrix} \right)_j .$$

The full Vanka smoother computes new velocity and pressure values by

$$\begin{pmatrix} w \\ r \end{pmatrix}_j := \begin{pmatrix} w \\ r \end{pmatrix}_j + \mathcal{A}^{-1} \left(\begin{pmatrix} f \\ g \end{pmatrix} - \mathcal{A} \begin{pmatrix} w \\ r \end{pmatrix} \right)_j .$$

The general strategy for choosing the sets $\mathcal{V}_j^h$ and $\mathcal{Q}_j^h$ is as follows. First, pick some pressure degrees of freedom which define $\mathcal{Q}_j^h$. Second, $\mathcal{V}_j^h$ is formed by all velocity degrees of freedom which are connected with the pressure degrees of freedom from $\mathcal{Q}_j^h$ by non-zero entries in the matrix C.

4.5.3 The Multiple Discretisation Multilevel Method

A multilevel method for higher order finite element discretisations is based on a multilevel method with a stable lowest order non-conforming finite element discretisation. This approach is called multiple discretisation multilevel method. In this approach, the multilevel hierarchy possesses one level more than the geometric grid hierarchy. On the finest geometric grid, level L, two discretisations are applied. One of them, which forms the finest level of the multilevel hierarchy, is the discretisation which we are interested in, e.g., a higher order discretisation. The second discretisation on the geometric level L is a lowest order non-conforming discretisation with upwind. On all coarser geometric levels, also a stabilized lowest order non-conforming discretisation is applied.

Algorithm 4.5 (**Multigrid method for the solution (4.1)**).

1: **multigrid(level)**
2: **if** $level == 0$ **then** //coarsest level
3: compute norm of the initial residual $res0 = res$
4: **while** $res >$ `coarse_red_factor`$\cdot res0$ **do**
5: apply `coarse_smoother` and compute update
6: compute new iterate by adding the old iterate and the update damped with `smooth_damp_factor`

```
 7:         compute norm of the residual res
 8:         if coarse_maxit reached then
 9:             break
10:         end if
11:     end while
12: else//finer levels
13:     for j = 0; j < pre_smooth; j++ do
14:         apply smoother and compute update
15:         compute new iterate by adding the old iterate and the update damped
    with smooth_damp_factor
16:     end for
17:     compute defect
18:     restrict defect to level-1
19:     for i = 0; i < recursion(i); i++ do multigrid(level-1)
20:         prolongate update from level-1
21:         compute new iterate by adding the old iterate and the update damped
    with prolo_damp_factor
22:     end for
23:     for j = 0; j < post_smooth; j++ do
24:         apply smoother and compute update
25:         compute new iterate by adding the old iterate and the update damped
    with smooth_damp_factor
26:     end for
27: end if
28: return
```

The multigrid method can be controlled with the following parameters:

- `mg_type`: type of the multigrid method (standard or multiple discretization).
- `mg_cycle`: type of the multigrid cycle. This parameter defines the array `recursion` in line 21, e.g., `recursion`$(i) = 1$ for the V-cycle and `recursion`$(i) = 2$ for the W-cycle.
- `smoother`: type of the smoother (diagonal or full, mesh cell or pressure node oriented).
- `pre_smooth`: number of pre-smoothing steps on the finer levels, line 14.
- `post_smooth`: number of post-smoothing steps on the finer levels, line 26.
- `smooth_damp_factor_fine`: damping factor for smoothing iteration on the finest level, this parameter replaces on the finest level `smooth_damp_factor` in lines 15 and 25.
- `smooth_damp_factor`: damping factor for smoothing iteration on all coarser levels, lines 15 and 25.
- `prolo_damp_factor_fine`: damping factor for the update on the finest level, this parameter replaces on the finest level `prolo_damp_factor` in lines 21.

- `prolo_damp_factor`: damping factor for the update on all coarser levels, line 21.
- `coarse_smoother`: smoother on the coarsest grid.
- `coarse_maxit`: maximal number of iterations on the coarsest grid.
- `coarse_red_factor`: factor for the reduction of the Euclidean norm of the initial residual after which the iteration on the coarsest grid stops.

4.6 Sparse Direct Solvers

In this thesis, besides the methods explained above, two standard sparse direct solvers are used for solving the linear system with nonsymmetric matrix. These solvers are popular in the scientific-technical community and are free accessible on the internet.

4.6.1 UMFPACK

The first of the solvers is UMFPACK. The principal author is Timothy A. Davis of the University of Florida [13]-[14]. The tested version (version 4.1) is written in C; the original code was developed by Davis and Duff in Fortran 77 [15]. UMFPACK is a code for the direct solution of the systems of linear equations

$$Ax = b$$

using the Unsymmetric MultiFrontal method. The matrix A is assumed to be a square, sparse and unsymmetric. It is decomposed into:

$$PAQ = LU,$$

where L and U are sparse lower and upper triangular matrices, respectively. The column reordering Q is chosen to give a good a priory upper bound on fill-in and to refine the numeric factorization. The row ordering P is determined during the numeric factorization to maintain numeric stability and to preserve sparsity.

The solution of the linear system involves the following phases:

- column pre-ordering: the initial ordering Q is determined to reduce fill-in without regard to numeric values;
- symbolic factorization: this phase determines upper bounds on the memory usage, the floating-point operation count, and the number of non-zeros in the LU factors;
- numeric factorization: the column reordering Q is refined to reduce fill-in, and the row ordering P is computed based on sparsity-preserving criteria as well as numeric considerations (relaxed threshold partial pivoting);
- solution of linear system: given the LU factors and the right-hand side b, the linear system is solved by forward and backward substitution. Iterative refinement is performed optionally.

UMFPACK is available at http://www.suitesparse.com.

4.6.2 PARDISO

The second solver is PARDISO by Schenk and Gärtner, 2005. This package is a software for solving large sparse symmetric and non-symmetric linear systems on shared-memory and distributed-memory architectures. PARDISO calculates the solution of a set of sparse linear equations with multiple right-hand sides, using

a parallel LU, LDL or LLT factorization. It is written using a combination of Fortran 77 and C source code and is included in Intel Math Kernel Library (see https://software.intel.com/en-us/intel-mkl/details).

The solver uses a combination of left- and right-looking Level-3 BLAS super nodes techniques. In order to improve sequential and parallel sparse numerical factorization performance, the algorithms are based on a Level-3 BLAS update, and pipelining parallelism is exploited with a combination of left- and right-looking super node techniques. The parallel pivoting methods allow complete super node pivoting in order to balance numerical stability and scalability during the factorization process.

Further details can be found in the literature [16]-[18].

5

Numerical Studies for Steady-State Equations

This chapter deals with numerical studies of different solvers for the steady-state case. Special attention is paid to the study of the effectiveness of methods based on block preconditioning and the Schur complement approximation. At first the limit case of the Navier-Stokes equations - the Stokes problem - is considered. Afterwards three benchmark problems - the driven cavity problem, the backward facing step problem and the flow around a cylinder - are presented.

The numerical simulations were performed using the program package MooNMD by John and Matthies, 2004 [7]. The main purpose of the computational analysis was the comparison of the performance of the solvers, which already are part of the "classical code" of MooNMD

- direct solvers UMFPACK,
- generalized minimal residual method GMRES, used without preconditioning,
- GMRES with the standard multigrid method as preconditioner,
- GMRES with the multiple discretization multilevel method as preconditioner (see [8], section 3.2, p. 166)

with block preconditioning methods such as

- GMRES with Least Squares Commutator (LSC) Preconditioner with direct solver UMFPACK to solve the Poisson subproblems at each step and
- GMRES with Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) with direct solver UMFPACK to solve the linear system at each step.

5.1 The Stokes Problem

5.1.1 Governing Equations and Their Discretization

The Stokes equations describe very slow flows with a high viscosity. They are a limit case of the Navier-Stokes equations, where the viscous term $Re^{-1}\Delta\mathbf{u}$ dominates the convective term $(\mathbf{u}\cdot\nabla)\mathbf{u}$ and the convective term can be neglected. The

momentum equation of the Stokes equations becomes a linear equation and the resulting problem reads as follows:
find $(\mathbf{u}, p) : \Omega \times \Omega \to \mathbb{R}^d \times \mathbb{R}$, such that,

$$\begin{aligned} -\nu\Delta\mathbf{u} + \nabla p &= \mathbf{f} \qquad \text{in } \Omega, && (5.1)\\ \nabla \cdot \mathbf{u} &= \mathbf{0} \qquad \text{in } \Omega. && (5.2) \end{aligned}$$

On the boundary consider Dirichlet and Neumann-type conditions. Therefore split $\Gamma = \partial\Omega$ into two disjoint parts Γ_D, Γ_N such that $\Gamma = \bar{\Gamma}_D \cup \bar{\Gamma}_N$, $|\Gamma_D| > 0$, then the boundary conditions are given by

$$\begin{aligned} \mathbf{u} &= \mathbf{0} \qquad \text{on } \Gamma_D, && (5.3)\\ (\nu\nabla\mathbf{u} - p\mathbb{I}) \cdot \mathbf{n} &= \mathbf{s} \qquad \text{on } \Gamma_N, && (5.4) \end{aligned}$$

where $\mathbf{n}$ is the outward-pointing normal to the boundary, and $\mathbf{s} \in H^{-1/2}(\Gamma_N)$.

The weak form of the Stokes problem is: given $\mathbf{f} \in L^2(\Omega)$,
find $(\mathbf{u}, p) \in (V \times Q)$ such that

$$\begin{aligned} (\nabla\mathbf{u}, \nabla\mathbf{v}) - (\nabla \cdot \mathbf{v}, p) &= (\mathbf{f}, \mathbf{v}) + \langle \mathbf{s}, \mathbf{v} \rangle_{\Gamma_N} \qquad \forall \mathbf{v} \in V, && (5.5)\\ -(\nabla \cdot \mathbf{u}, q) &= 0 \quad \forall q \in Q, && (5.6) \end{aligned}$$

where

$$\langle \mathbf{s}, \mathbf{v} \rangle_{\Gamma_N} = \int_{\Gamma_N} \mathbf{s} \cdot \mathbf{v}, \tag{5.7}$$

and

$$V = \left\{ \mathbf{v} \in (H^1(\Omega))^d : \quad \mathbf{v}|_{\Gamma_D} = \mathbf{0} \right\}, \qquad Q = L^2_0(\Omega). \tag{5.8}$$

The Stokes problem has unique solution and this solution depends continuously on the right-hand side of (5.5)-(5.6) (see [20]). The discrete version of (5.5)-(5.6) is formulated as:

Given the finite-dimensional subspaces $V^h \subset V$ and $Q^h \subset Q$,
find $(\mathbf{u}^h, p^h) \in V^h \times Q^h$ such that

$$\begin{aligned} (\nu\nabla\mathbf{u}^h, \mathbf{v}^h) - (p^h, \nabla \cdot \mathbf{v}^h) &= (\mathbf{f}, \mathbf{v}^h) + \left\langle \mathbf{s}, \mathbf{v}^h \right\rangle_{\Gamma_N} \qquad \forall \mathbf{v}^h \in V^h, && (5.9)\\ (q^h, \nabla \cdot \mathbf{u}^h) &= 0 \quad \forall q^h \in Q^h. && (5.10) \end{aligned}$$

Following the standard steps of the Galerkin method one obtains the linear system of equations, which has the same form as in the general case of the Navier-Stokes equations

$$\begin{bmatrix} \mathbf{K} & B^T \\ B & O \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathfrak{f} \\ \mathbf{g} \end{bmatrix},$$

and to which the methods discussed in Chapter 4 can be applied.

5.1.2 Analytic Example

Consider the square domain $\Omega = (0,1)^2$, $Re = 1$ ($\nu = \frac{1}{Re} = 1$) and the source term

$$\mathbf{f} = \begin{pmatrix} \frac{\pi^2}{2}\sin(\frac{\pi}{2}x)\cos(\frac{\pi}{2}y) - 1 \\ -\frac{\pi^2}{2}\cos(\frac{\pi}{2}x)\sin(\frac{\pi}{2}y) \end{pmatrix} \in L^2(\Omega).$$

Figure 5.1 illustrates the sketch of the domain with four disjoint boundary parts. The boundary conditions look as follows

- On Γ_0 and Γ_1 the Dirichlet boundary conditions prescribe the inflow and outflow into Ω

$$\mathbf{u}|_{\Gamma_0} = \begin{pmatrix} -\sin(\frac{\pi}{2}x) \\ -0.5 + x \end{pmatrix}, \tag{5.11}$$

$$\mathbf{u}|_{\Gamma_1} = \begin{pmatrix} -\cos(\frac{\pi}{2}y) \\ 0.5 \end{pmatrix}. \tag{5.12}$$

- On Γ_2 and Γ_3 the Neumann conditions specify the natural boundary conditions

$$\mathbf{s}|_{\Gamma_2} = \begin{pmatrix} \frac{\pi}{2}\sin(\frac{\pi}{2}x) \\ -0.5 + x \end{pmatrix}, \tag{5.13}$$

$$\mathbf{s}|_{\Gamma_3} = \begin{pmatrix} 0.5 + \frac{\pi}{2}\cos(\frac{\pi}{2}y) \\ -1 \end{pmatrix}. \tag{5.14}$$

The functions

$$\mathbf{u} = \begin{pmatrix} -\sin(\frac{\pi}{2}x)\cos(\frac{\pi}{2}y) \\ \cos(\frac{\pi}{2}x)\sin(\frac{\pi}{2}y) + x - 0.5 \end{pmatrix}, \qquad p = 0.5 - x \tag{5.15}$$

are the analytic solution of the problem (5.1)-(5.4), representing steady-state horizontal flow in a channel driven by a pressure difference between the two ends. The streamfunctions, velocity and pressure are presented in Figure 5.2.

5.1.3 Numerical Results

The simulations were carried out with the standard Galerkin method for the inf-sup stable pairs of the finite element spaces P_2/P_1 on the triangular grids and Q_2/Q_1

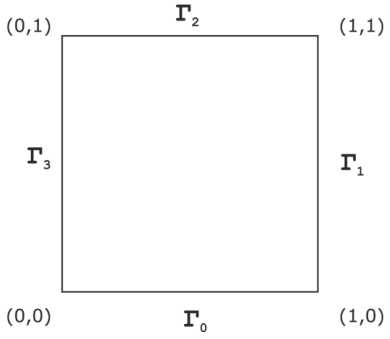

Figure 5.1 – 2D Stokes problem, domain.

on the quadrilateral grids with two uniform refinement steps at the beginning. Table 5.1 illustrates the corresponding geometry levels.

Table 5.1 – 2D Stokes problem, geometry levels

Level	Cells		d.o.f.	
	P_2/P_1	Q_2/Q_1	P_2/P_1	Q_2/Q_1
4	512	256	2467	2467
5	2048	1024	9539	9539
6	8192	4096	37507	37507

The iteration is terminated when either the Euclidean norm of the residual is smaller than 10^{-8} or the maximum number of iterations at a level is achieved (it has been set to 100000).

Tables 5.2-5.4 illustrate the actual results. The trends are as follows. For the both discretizations the direct solver UMFPACK was the fastest method, whereas GMRES without preconditioning was the worst. On the coarse grid LSC preconditioner was better than multigrid preconditioners, but on the fine meshes the multiple discretization multilevel method was faster. The CPU time for SIMPLE preconditioner was much better than for GMRES without preconditioning, but worse in comparison with other methods.

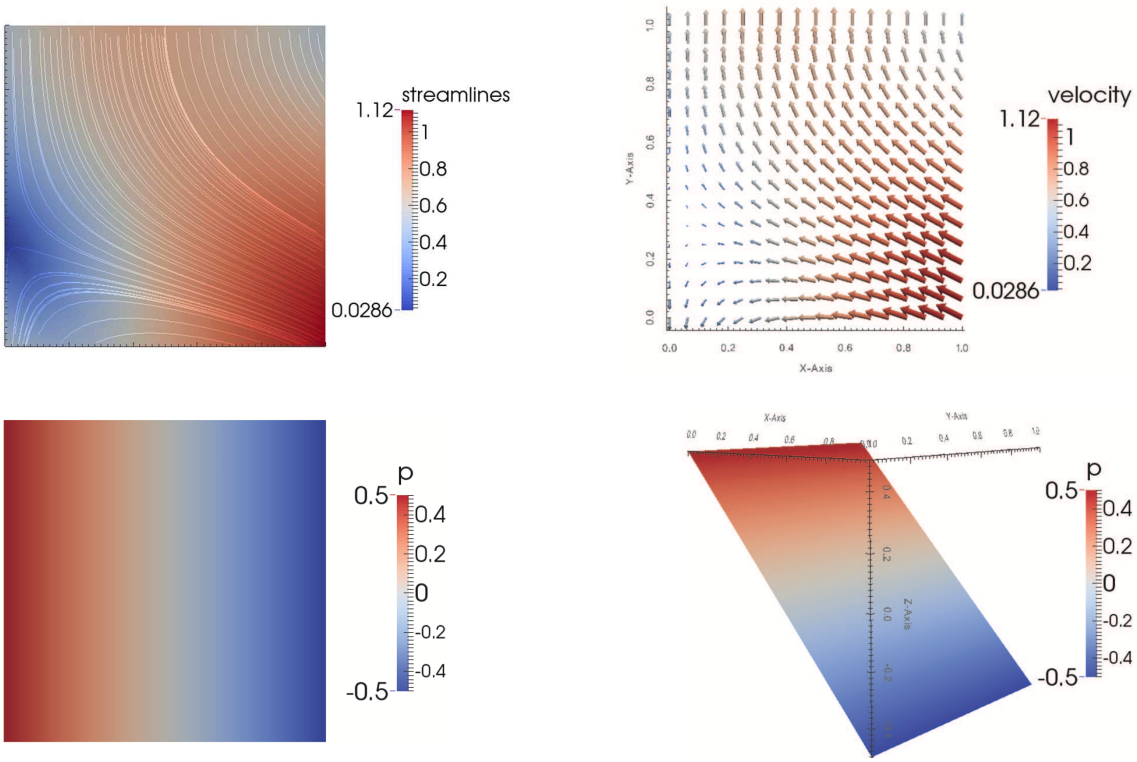

Figure 5.2 – 2D Stokes problem, streamfunction (top left), velocity (top right) and pressure.

Table 5.2 – 2D Stokes problem, P_2/P_1 and Q_2/Q_1 discretizations, CPU times (s) for direct solver UMFPACK.

Level	P_2/P_1	Q_2/Q_1
4	0.036	0.1
5	0.17	0.3
6	1	1.4

Table 5.3 – 2D Stokes problem, P_2/P_1 discretization, GMRES without preconditioning and with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners.

Level	Parameters	GMRES	GMRES+ MG	GMRES+ MDML	GMRES+ SIMPLE	GMRES+ LSC
4	Iterations	19189	6	6	268	28
	Time (total), s	10	0.53	0.44	0.5	0.1
5	Iterations	74596	6	7	520	42
	Time (total), s	171	1.8	1.55	5	0.9
6	Iterations	497223	6	7	1172	64
	Time (total), s	4160	5.9	5.4	80.5	10.4

Table 5.4 – 2D Stokes problem, Q_2/Q_1 discretization, GMRES without preconditioning and with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners.

Level	Parameters	GMRES	GMRES+ MG	GMRES+ MDML	GMRES+ SIMPLE	GMRES+ LSC
4	Iterations	17166	8	8	239	11
	Time (total), s	9.6	0.99	0.69	0.6	0.17
5	Iterations	64774	8	8	454	13
	Time (total), s	147.8	3.35	2.5	5.7	0.98
6	Iterations	837597	8	8	984	16
	Time (total), s	7603	13.2	8.8	59.6	9.7

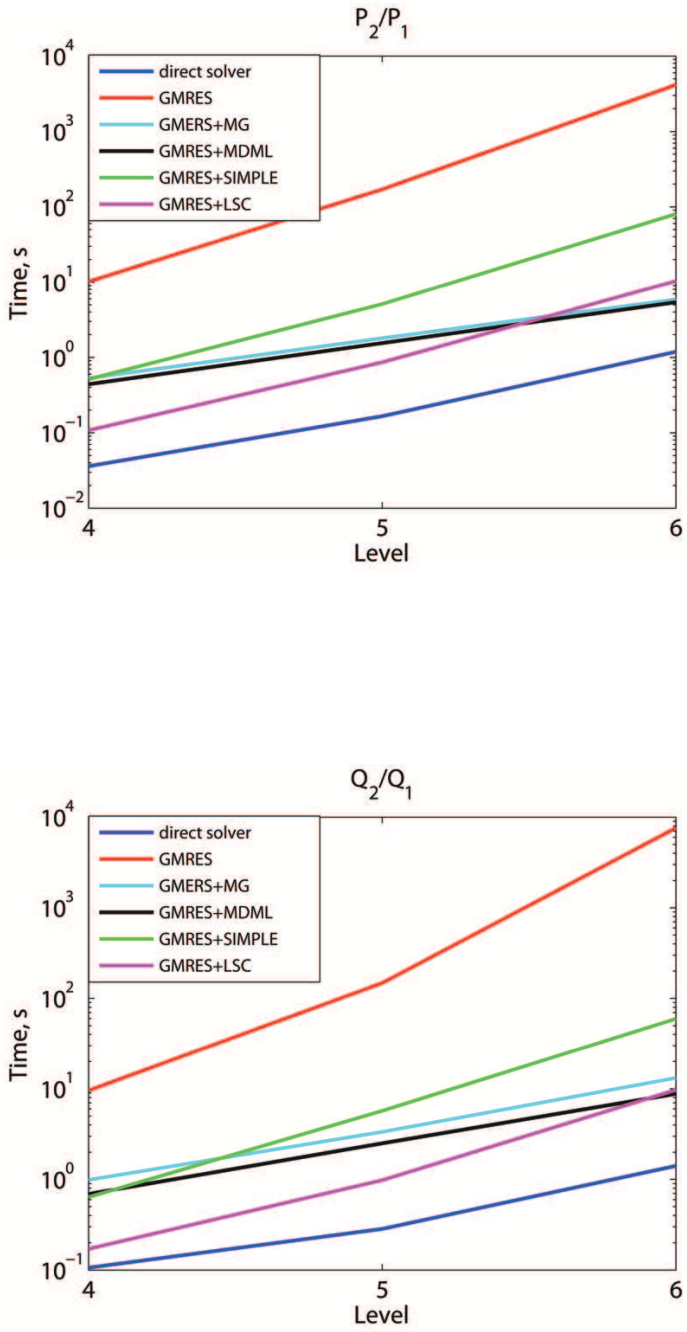

Figure 5.3 – 2D Stokes problem, CPU times (s) for P_2/P_1 and Q_2/Q_1 discretizations for direct solver UMFPACK, GMRES without preconditioning and with SIMPLE and LSC preconditioners.

5.2 The Steady-State Driven Cavity Problem with $Re = 1000$

5.2.1 Implemented Example

Numerical experiments in the steady-state case of the Navier-Stokes equations are presented for the widely studied benchmark problem - 2D driven cavity problem. Consider incompressible flow in a square domain $\Omega = (0,1)^2$ (cavity) with an upper lid moving with a velocity U as shown in Figure 5.4. On the other boundaries zero Dirichlet boundary conditions are used.

Let the Reynolds number of the flow be $Re = 1000$.

Figure 5.5 shows the velocity field and pressure field for an example solution to a 2D lid driven cavity on the geomtery level 5.

5.2.2 Numerical Results

The solution was carried out with the standard Galerkin method for the inf-sup stable pair of the finite element spaces Q_2/P_1^{disc} on the quadrilateral mesh. Table 5.5 illustrates the corresponding geometry levels.

Table 5.5 – 2D driven cavity problem, geometry levels

Level	Cells	d.o.f.
4	256	2946
5	1024	11522
6	4096	45570

The Picard iteration is terminated when either the euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved (it has been set to 500).

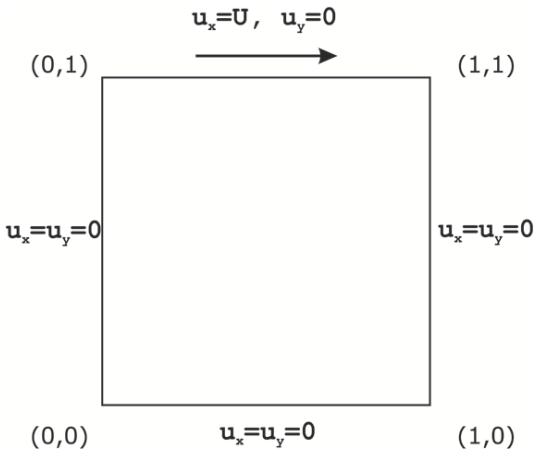

Figure 5.4 – 2D driven cavity problem, domain.

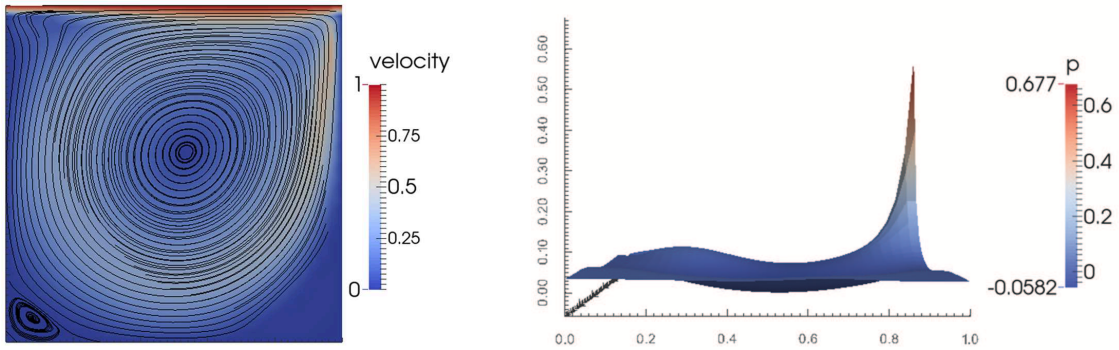

Figure 5.5 – 2D driven cavity problem, streamfunctions and 3D pressure rendering.

The stopping criteria for the GMRES iterations are either the Euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved (it has been set to 100000).

The stopping criteria for the case of the flexible GMRES with coupled multigrid and the multiple discretization multilevel method are either at most 5 iterations for solution of linear system were performed or the euclidean norm of the residual is reduced by factor 10. The multigrid methods were applied with the $F(2,2)$-cycle, where the number of pre and post smoothing steps is 2. The damping factor of Vanka smoother is 0.8.

Table 5.6 – 2D driven cavity problem, Q_2/P_1^{disc} discretization, $Re = 1000$, comparison of the direct solver, GMRES without preconditioning and GMRES with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners.

Level	Parameters	UMFPACK	GMRES	GMRES +MG	GMRES +MDML	GMRES +SIMPLE	GMRES +LSC
4	Nonlinear iterations	13	69	32	37	24	44
	GMRES iterations	-	753279	155	180	240000*	1361
	Time (total), s	0.5	461	3.8	4.3	7127	6.1
5	Nonlinear iterations	24	106	27	28	17	36
	GMRES iterations	-	10234567	128	133	170000*	939
	Time (total), s	4	3389	13.2	12	25273	31.9
6	Nonlinear iterations	23	38	29	27	14	32
	GMRES iterations	-	3513510*	138	120	1400000*	830
	Time (total), s	23.1	46683	56.4	42.1	97899	279

* - the method did not converge

Table 5.6 and Figure 5.6 illustrate the actual results. The direct solver UMFPACK demonstrated again the fastest CPU time. On the fine grid GMRES without preconditioning did not converge within the prescribed maximal number of steps, whereas the SIMPLE preconditioner did not converge on all geometry levels. The LSC preconditioner showed good convergence although was inferior to the direct solver and multigrid preconditioners.

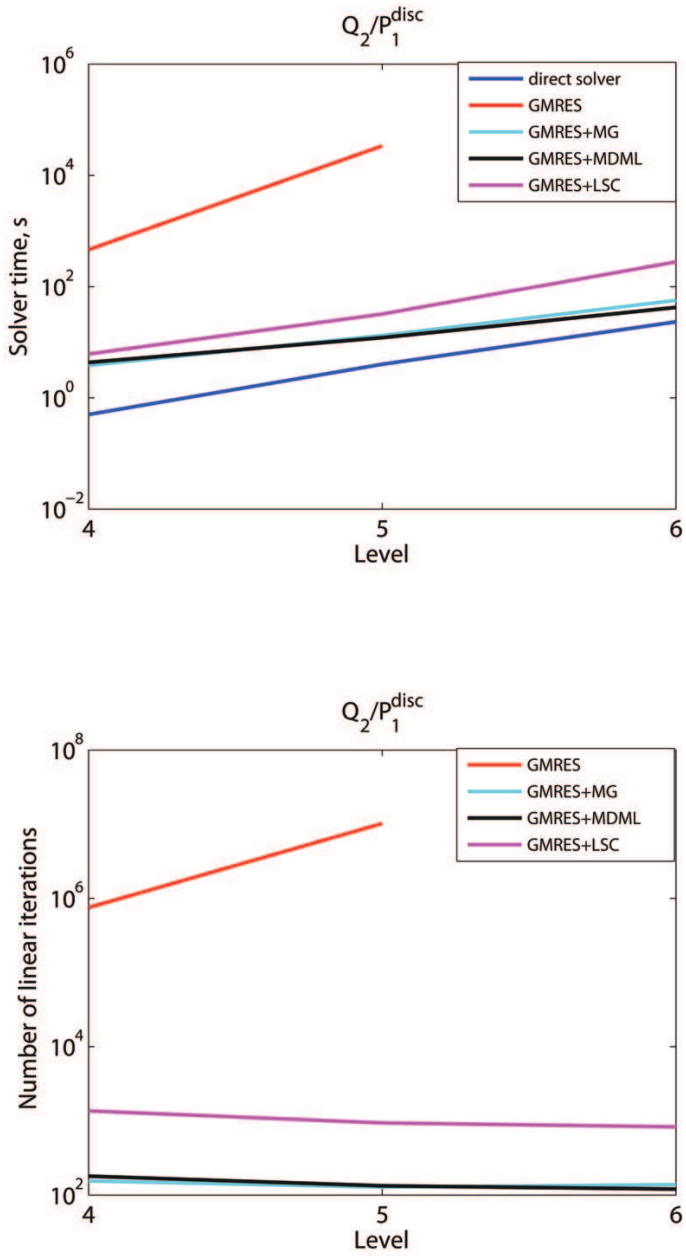

Figure 5.6 – 2D driven cavity problem, $Re = 1000$, Q_2/P_1^{disc} discretization, CPU times (s) for direct solver UMFPACK, GMRES without preconditioning and GMRES with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners. Number of the linear iterations for GMRES methods.

5.3 The Backward Facing Step Problem with $Re = 100$

5.3.1 Implemented Example

Consider L-shaped domain Ω shown in Figure 5.7. This problem geometry represents a flow in a rectangular duct with a sudden expansion.

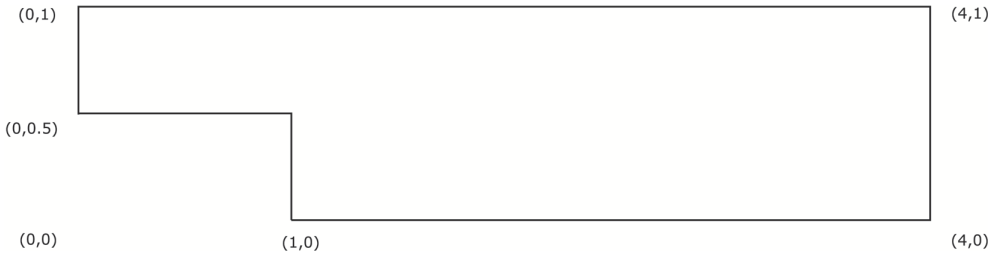

Figure 5.7 – 2D backward facing step problem, domain.

On the inflow boundary ($x = 0, 0.5 \leq y \leq 1$) a Poiseuille flow profile is imposed

$$u_x = 12(1 - y)(2y - 1), \quad u_y = 0, \quad p = -2\nu x,$$

which is a parabolic inflow boundary condition.

On the outflow boundary ($x = 4, 0 \leq y \leq 1$) the do-nothing condition

$$\mathbb{S}\mathbf{n} = 0$$

is considered.

On the walls the no-slip boundary conditions

$$\mathbf{u}(\mathbf{x}) = \mathbf{0}$$

are used.

Let the Reynolds number of the flow be $Re = 100$.

Figures 5.8-5.9 show velocity, pressure, streamlines and a three-dimensional rendering of the pressure solution, which was obtained by numerical simulations on the geometry level 5.

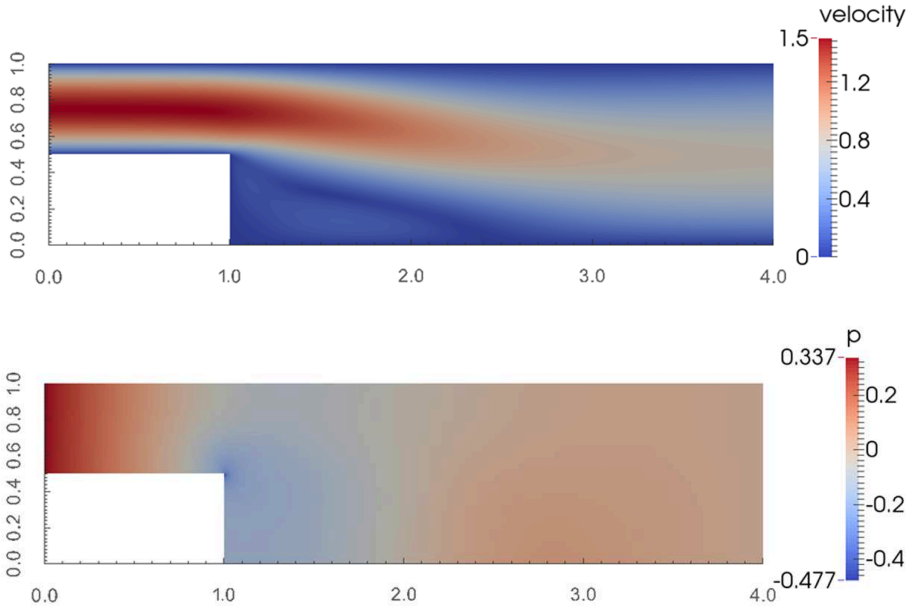

Figure 5.8 – 2D backward facing step problem, velocity (top) and pressure (bottom).

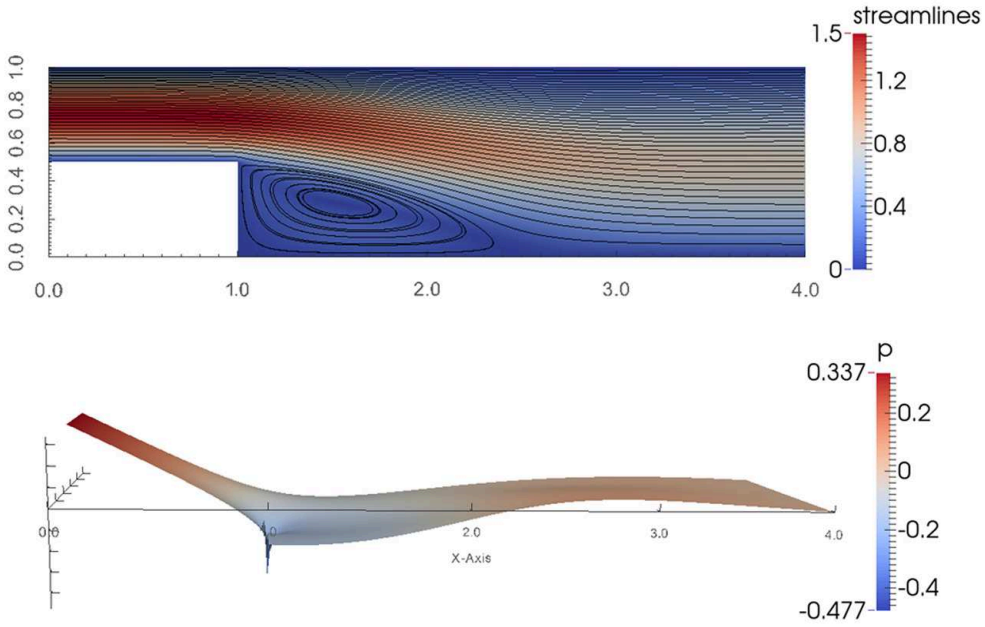

Figure 5.9 – 2D backward facing step problem, streamlines (top) and 3D pressure rendering (bottom).

5.3.2 Numerical Results

The solution was carried out with the standard Galerkin method for the inf-sup stable pair of the finite element spaces Q_2/Q_1 on the quadrilateral mesh. Table 5.7

illustrates the corresponding geometry levels.

Table 5.7 – 2D backward facing step problem, geometry levels

Level	Cells	d.o.f.
4	1792	16611
5	7168	65475
6	28672	259971

The Picard iteration is terminated when either the euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved (it has been set to 500).

The stopping criteria for the GMRES iterations are either the Euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved (it has been set to 100000).

The stopping criteria for the case of the flexible GMRES with coupled multigrid and the multiple discretization multilevel method are either at most 5 iterations for solution of linear system were performed or the euclidean norm of the residual is reduced by factor 10. The multigrid methods were applied with the $F(2,2)$-cycle, where the number of pre and post smoothing steps is 2. The damping factor of Vanka smoother is 0.8.

Table 5.8 – 2D backward facing step problem, Q_2/Q_1 discretization, $Re = 100$, comparison of the direct solver and GMRES with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners.

Level	Parameters	UMFPACK	GMRES	GMRES +MG	GMRES +MDML	GMRES +SIMPLE	GMRES +LSC
4	Nonlinear iterations	16	11	25	25	11	12
	GMRES iterations	-	85065	72	47	850	45
	Time (total), s	5.7	415	56.6	22.5	38	11
5	Nonlinear iterations	16	15	24	25	12	14
	GMRES iterations	-	1500000*	45	45	1806	67
	Time (total), s	36	23772	142.6	87	421	130
6	Nonlinear iterations	16	14	23	23	11	11
	GMRES iterations	-	1400000*	40	42	3283	75
	Time (total), s	269	92343	580	329	6156	1368

* - the method did not converge

We observe, that on the coarse grids the LSC preconditioner was worse than the direct solvers, but better than the preconditioner based on the coupled multigrid methods. While on the fine grids the LSC method was slower than these two solvers. Based on the outcomes of this and previous simulations, GMRES without preconditioning and with SIMPLE preconditioner will no longer be considered, since both of these methods showed the worst result.

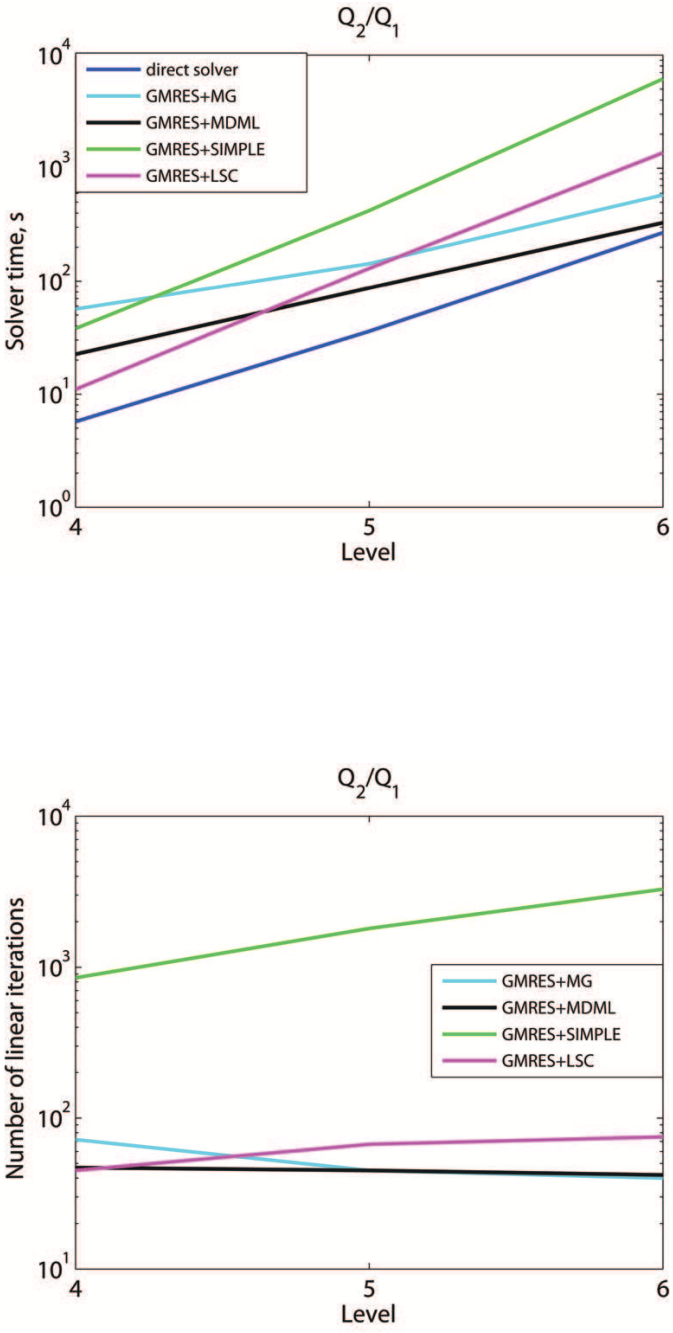

Figure 5.10 – 2D backward facing step problem, $Re = 100$, Q_2/Q_1 discretization, CPU times (s) for direct solver UMFPACK and GMRES with coupled multigrid, the multiple discretization multilevel, SIMPLE and LSC preconditioners. Number of the linear iterations for GMRES methods.

5.4 The Steady-State Flow Around a Cylinder with $Re = 20$

5.4.1 Implemented Example

This benchmark problem was defined by Schäfer and Turek in [19]. Consider a flow in a two-dimensional domain with a two-dimensional cylinder (circle), see Figure 5.11 for a sketch of this domain.

The dynamic viscosity of the fluid is given by $\mu = 10^{-3} Pa\,s$ and its density by $\rho = 1 kg/m^3$. These values are approximately the coefficients for water. The parabolic inflow profile is given by

$$\mathbf{v}(0\,m, y) = \frac{1}{0.41^2} \begin{pmatrix} 1.2y(0.41 - y) \\ 0 \end{pmatrix} m/s, \quad 0\,m \leq y \leq 0.41\,m. \tag{5.16}$$

At the top and the bottom of the channel and at the surface Γ_{body} of the cylinder, no-slip boundary conditions are prescribed. With respect to the outlet do-nothing boundary conditions are used

$$\mathbb{S}\mathbf{n} = (-\mu \nabla \mathbf{v} + P\mathbb{I})\mathbf{n} = \mathbf{0} N/m^2 \qquad \text{on } \Gamma_{\text{outfl,}} \tag{5.17}$$

where $\mathbf{n}$ is the outward pointing unit normal vector. The mean inflow velocity is given by

$$U_{mean} = \frac{1}{0.41^2} \frac{\int_0^{0.41} 1.2y(0.41 - y)dy}{\int_0^{0.41} dy} m/s = \frac{1}{5} \frac{0.41^3}{0.41^3} = 0.2\,m/s. \tag{5.18}$$

Based on the mean inflow, the diameter $d = 0.1\,m$ of the cylinder, and the kinematic viscosity μ/ρ, the Reynolds number of the flow is $Re = 20$. There are no external forces acting on the flow, i.e., $\mathbf{f}_{ext} = \mathbf{0} N/m^3$.

Using the characteristic length scale $L = 1\,m$ and the characteristic velocity scale $U = 1\,m/s$, one obtains the steady-state Navier-Stokes equations (2.1)-(2.4) with $\nu = \mu/(\rho UL) = 10^{-3}$ and accordingly $Re = (UL)/\nu = 1000$, the inflow condition

$$\mathbf{v}(0, y) = 0.41^{-2} \begin{pmatrix} 1.2y(0.41 - y) \\ 0 \end{pmatrix} \quad 0 \leq y \leq 0.41,$$

and the outflow condition

$$\mathbb{S}\mathbf{n} = (-\nu \nabla \mathbf{v} + p\mathbb{I})\mathbf{n} = \mathbf{0} \qquad \text{on } \Gamma_{\text{outfl}}.$$

Figures 5.12-5.13 present the solution, which was obtained by numerical simulations with do-nothing outflow boundary conditions.

5.4.2 Numerical Results

As in previous cases the solution was carried out with the standard Galerkin method. The inf-sup stable pair of the finite element spaces P_2/P_1 on the triangle

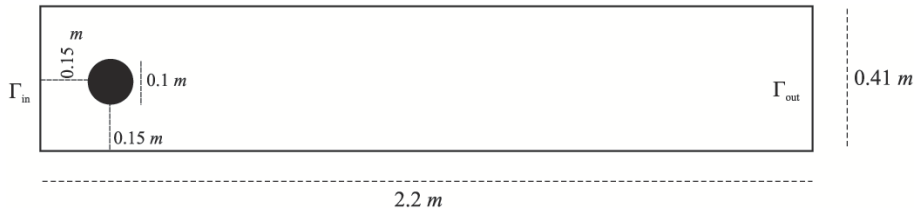

Figure 5.11 – 2D stationary flow around cylinder, domain

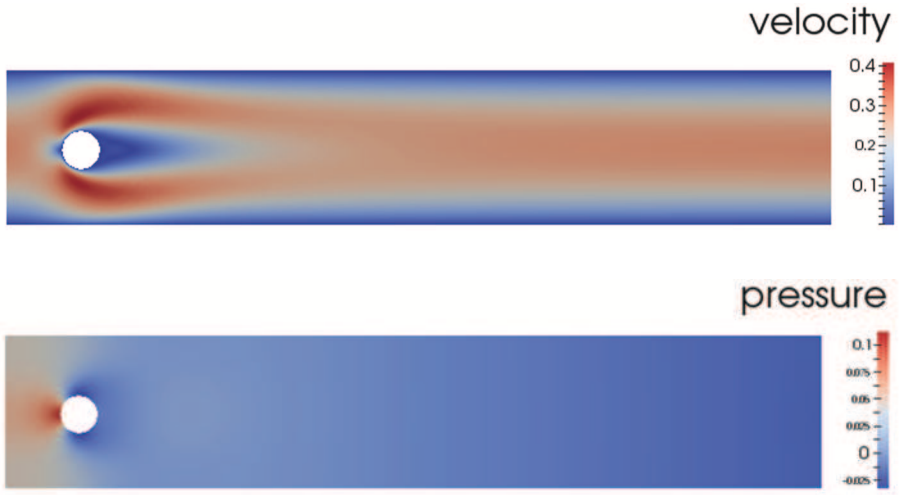

Figure 5.12 – 2D stationary flow around cylinder, velocity (top) and pressure (bottom).

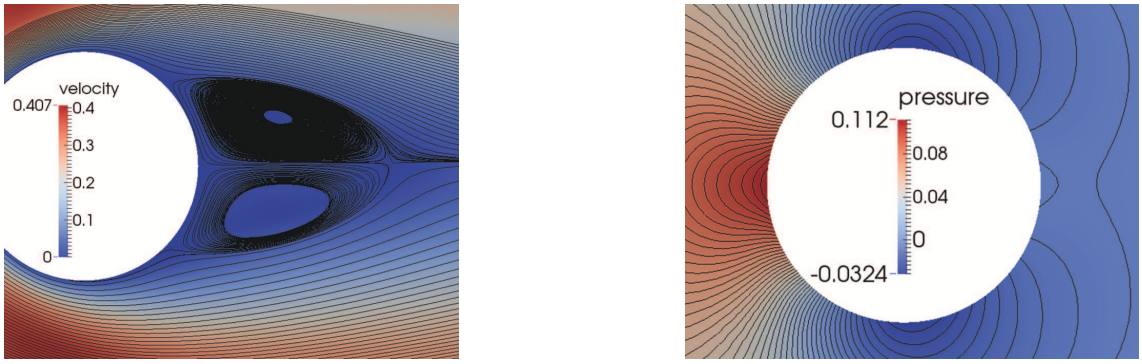

Figure 5.13 – 2D stationary flow around cylinder, streamfunction (left) and pressure isosurfaces (right).

grids and Q_2/P_1^{disc} on the quadrilateral grids were used. Table 5.9 illustrates the corresponding geometry levels.

The Picard iteration is terminated when either the Euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved

Table 5.9 – 2D stationary flow around cylinder, geometry levels

Level	Cells		d.o.f.	
	P_2/P_1	Q_2/P_1^{disc}	P_2/P_1	Q_2/P_1^{disc}
0	388	208	1926	2440
1	1552	832	7344	9456
2	6208	3328	28656	37216
3	24832	13312	113184	147648
4	99328	53248	449856	588160

(it has been set to 500).

The stopping criteria for the GMRES iterations are either the Euclidean norm of the residual is smaller than 10^{-10} or the maximum number of iterations at a level is achieved (it has been set to 100000). The rest simulation parameters are the same as for the driven cavity and backward facing step problems.

All simulations were implemented for two initial guesses: zero initial guess and extrapolated solution from the coarse grid.

Table 5.10 – 2D stationary flow around cylinder, $Re = 20$, P_2/P_1, comparison of the iteration counts and the total CPU times (s) at each level for zero initial guess.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
0	Nonlinear iterations	21	21	22	22	24
	Linear iterations	-	-	101	95	440
	Time (total)	0.4	0.8	7.23	3.64	0.7
1	Nonlinear iterations	16	16	17	16	21
	Linear iterations	-	-	79	75	5941
	Time (total)	1.3	2.6	14.4	12.3	4.8
2	Nonlinear iterations	15	15	16	16	17
	Linear iterations	-	-	72	72	6183
	Time (total)	7.4	12.9	58.4	45.5	34
3	Nonlinear iterations	15	15	15	15	17
	Linear iterations	-	-	66	50	2429
	Time (total)	74.3	69.3	218.3	127.9	259
4	Nonlinear iterations	14	14	15	14	19
	Linear iterations	-	-	58	42	18833
	Time (total)	656.9	372.9	760	440.8	2337

Table 5.10 and Table 5.11 illustrate the performance of the above mentioned methods for the case of P_2/P_1: the number of iterations and the total computing times for two initial guesses respectively.

Table 5.12 and Table 5.13 give the same information for the finite element spaces Q_2/P_1^{disc}.

Table 5.11 – 2D stationary flow around cylinder, $Re = 20$, P_2/P_1, comparison of the iteration counts and the total CPU times (s) at each level for initial guess as extrapolated solution from the coarse grid.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
0	Nonlinear iterations	21	21	22	22	24
	Linear iterations	-	-	101	95	4440
	Time (total)	0.38	0.68	6.81	3.46	0.7
1	Nonlinear iterations	12	12	12	11	14
	Linear iterations	-	-	57	52	2850
	Time (total)	0.99	1.8	10.3	8.3	4
2	Nonlinear iterations	8	8	10	9	11
	Linear iterations	-	-	46	37	3108
	Time (total)	4.1	6.1	38.7	24.1	25
3	Nonlinear iterations	5	5	7	6	9
	Linear iterations	-	-	25	15	3405
	Time (total)	24.3	21.4	83.3	40.1	176
4	Nonlinear iterations	2	3	5	4	7
	Linear iterations	-	-	16	72	4276
	Time (total)	139.3	71.7	211.6	75.7	1503

Table 5.12 – 2D stationary flow around cylinder, $Re = 20$, Q_2/P_1^{disc}, comparison of the iteration counts and the total CPU times (s) at each level for zero initial guess.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
0	Nonlinear iterations	17	17	22	17	18
	Linear iterations	-	-	108	81	1333
	Time (total)	0.36	0.6	2.74	1.4	0.7
1	Nonlinear iterations	16	16	19	16	17
	Linear iterations	-	-	93	77	1127
	Time (total)	1.8	2.7	5.9	5	4
2	Nonlinear iterations	15	15	15	15	20
	Linear iterations	-	-	65	70	2185
	Time (total)	11.9	13	18.6	19.2	36
3	Nonlinear iterations	15	15	15	15	17
	Linear iterations	-	-	59	67	2429
	Time (total)	93	65	69	74.8	259
4	Nonlinear iterations	failed	14	14	471	21
	Linear iterations	-	-	44	2353	4977
	Time (total)	-	356	208	10460	2257

Table 5.13 – 2D stationary flow around cylinder, $Re = 20$, Q_2/P_1^{disc}, comparison of the iteration counts and the total CPU times (s) at each level for initial guess as extrapolated solution from the coarse grid.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
0	Iterations	17	17	22	17	18
	Linear iterations	-	-	108	81	1333
	Time (total)	0.4	0.6	2.7	1.4	0.7
1	Iterations	11	11	14	12	12
	Linear iterations	-	-	67	53	551
	Time (total)	1.5	2.1	4.3	3.5	3
2	Iterations	8	8	9	8	10
	Linear iterations	-	-	37	32	485
	Time (total)	7.8	7.6	10.6	8.7	20
3	Iterations	6	6	6	6	10
	Linear iterations	-	-	19	18	584
	Time (total)	55.2	31.6	22.3	20.8	159
4	Iterations	failed	3	5	250	6
	Linear iterations	-	-	14	1249	279
	Time (total)	-	96.7	67	5586	1462

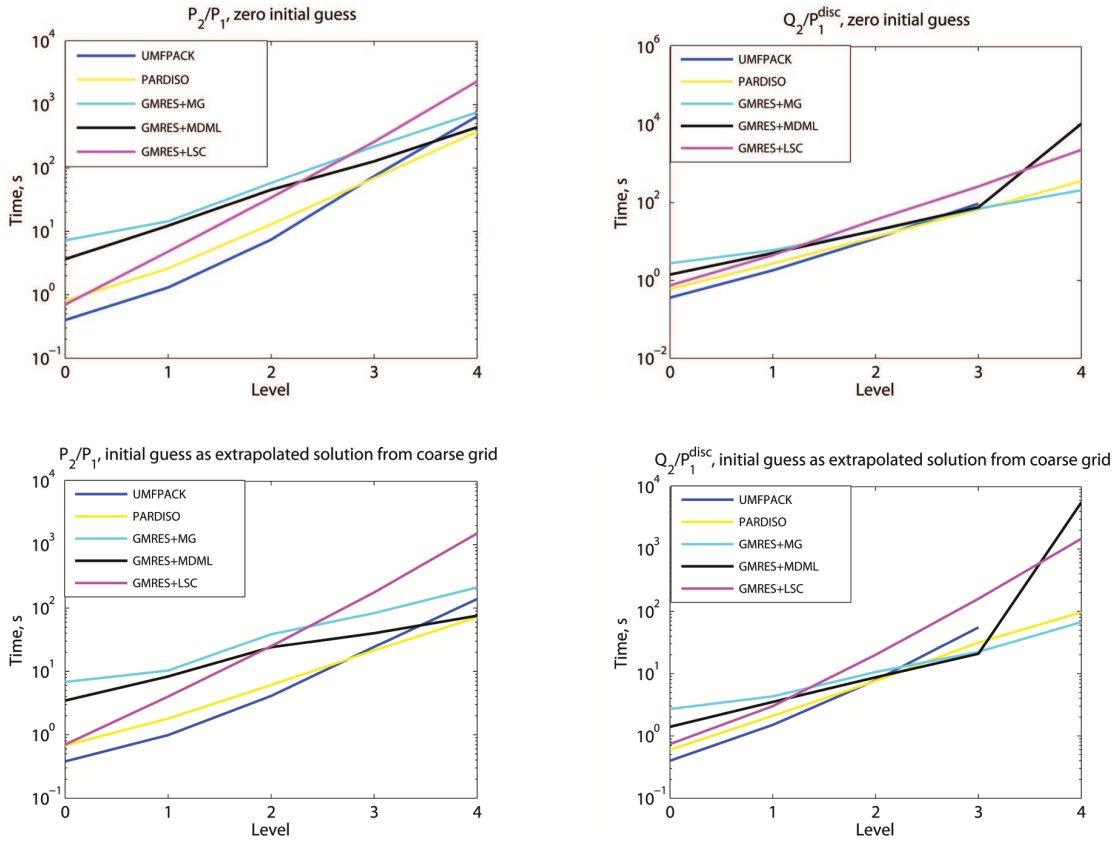

Figure 5.14 – 2D stationary flow around cylinder, $Re = 20$, CPU times (s) for direct solvers UMFPACK, PARDISO and GMRES with coupled multigrid, the multiple discretization multilevel and LSC preconditioners.

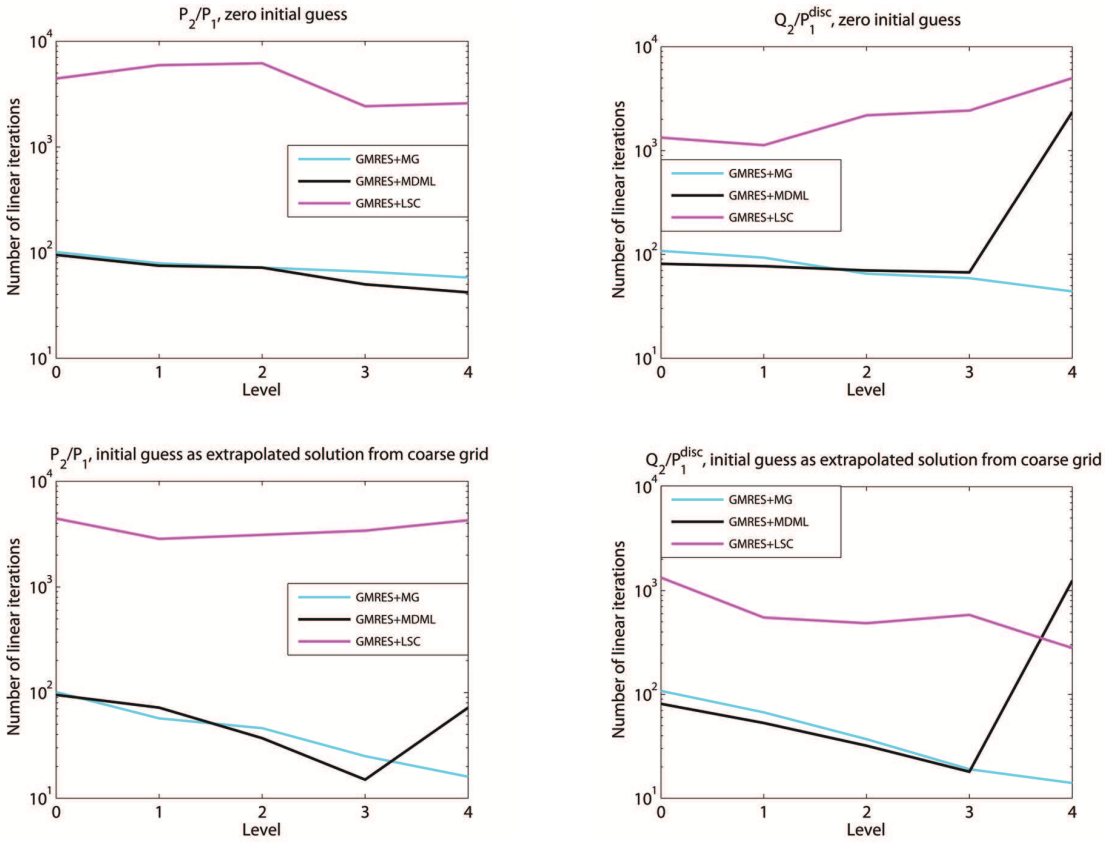

Figure 5.15 – 2D stationary flow around cylinder, $Re = 20$, number of linear iterations for GMRES with coupled multigrid, the multiple discretization multilevel and LSC preconditioners.

5.5 Summary of Results

In the stationary examples presented in this thesis, we could generally observe

- the iterative method GMRES without preconditioning has showed the worst results in all examples.
- The direct solvers UMFPACK and PARDISO were the fastest methods. An exception made only the examples on finer grids with the large numbers of degrees of freedom.
- Considering only GMRES with block preconditioning based on the Schur complement approximation such as SIMPLE and LSC preconditioners one can see that LSC preconditioner proved to be cheaper. This concerns both the number of nonlinear and linear iterations and CPU times. In all presented examples this method demonstrates the most acceptable results. Its computing time, for example for 2D Stokes's problem, was more than one hundred times better than the computing time of GMRES and it is about 6-7 times better than SIMPLE. For Navier-Stokes examples we have similar results.
- But the comparison of the block preconditioning with the preconditioning based on the coupled multigrid methods leads us to the conclusion that the last methods are much faster. Their results are comparable with direct solvers and in cases with the large numbers of degrees of freedom even better.

6

Numerical Studies for Time-Dependent Navier-Stokes Equations

This chapter presents a numerical study of different solvers for the unsteady Navier-Stokes equations on the two examples - a problem with a known analytic solution and laminar flow around a cylinder. The numerical simulations also were performed using the program package MooNMD [7] and the following methods were tested

- direct solvers UMFPACK and PARDISO,
- generalized minimal residual method GMRES, used without preconditioning,
- GMRES with the standard multigrid method as preconditioner,
- GMRES with the multiple discretization multilevel method as preconditioner,
- GMRES with Least Squares Commutator (LSC) Preconditioner and
- GMRES with Semi-Implicit Method for Pressure-Linked Equations (SIMPLE).

6.1 The Example with a Known Analytic Solution

6.1.1 Implemented Example

Consider the square domain $\Omega = (0,1)^2$ and the solution of (3.1)-(3.5) given by

$$\begin{aligned} u_1 &= 2\pi \sin(t)\sin^2(\pi x)\sin(\pi y)\cos(\pi y), \\ u_2 &= -2\pi \sin(t)\sin(\pi x)\cos(\pi x)\sin^2(\pi y), \\ p &= 20\sin(t)\left(x^2 y - \frac{1}{6}\right), \end{aligned} \tag{6.1}$$

with $\mathbf{u} = (u_1, u_2)^T$.

The right-hand side $\mathbf{f}$ is chosen such that $(u_1, u_2, p)^T$ is the solution of (3.1)-(3.5) with the initial conditions

$$\mathbf{u}(0,\cdot) = \mathbf{0}$$

and the homogeneous Dirichlet conditions on the boundary.

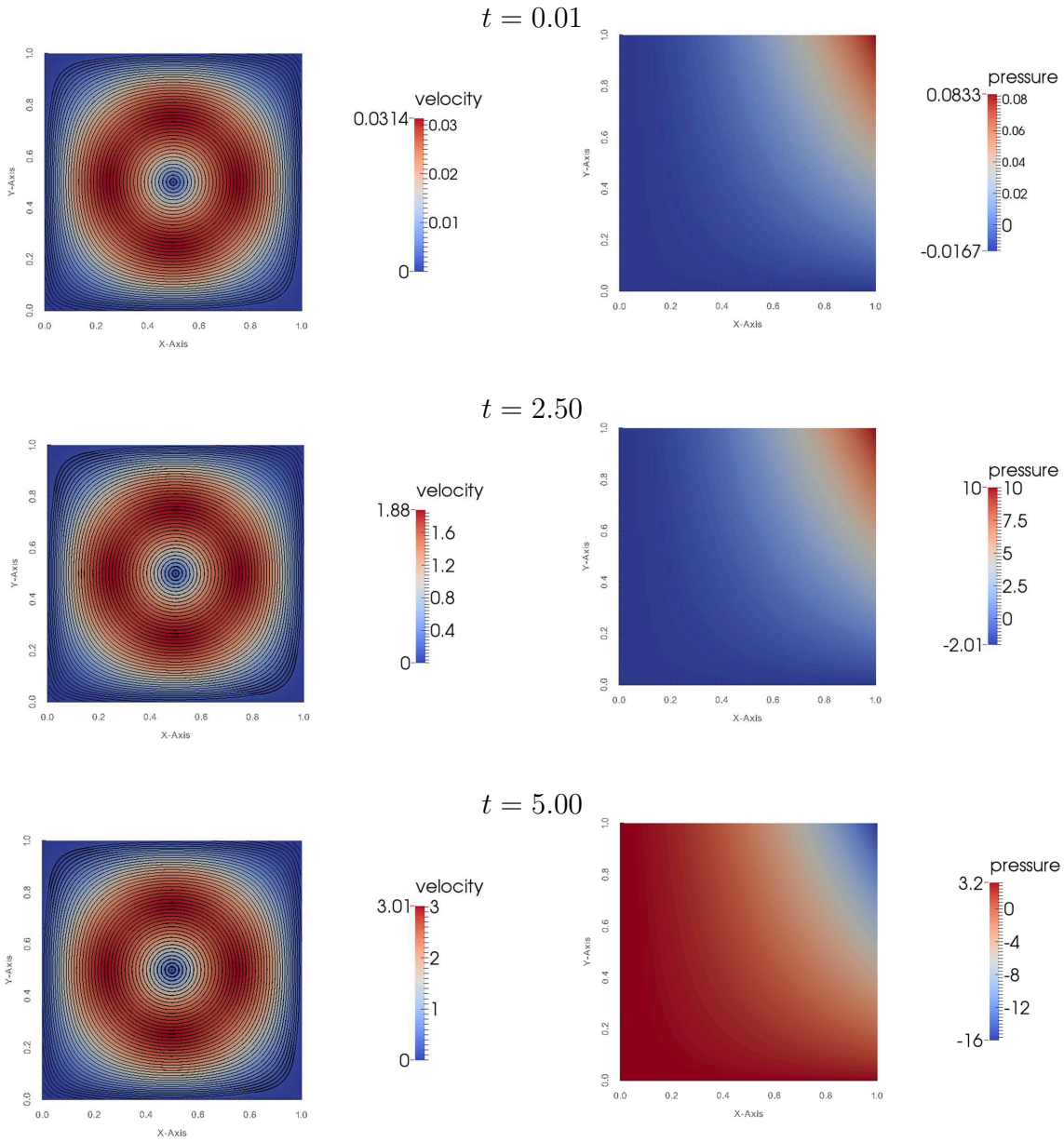

Figure 6.1 – 2D instationary analytic example, Q_2/P_1^{disc} discretization, $Re = 1000$, streamfunctions (left) and pressure (right).

The Reynolds number of the flow is $Re = 1000$ and the final time is set to be $T = 5$. The streamfunctions, velocity and pressure are presented in Figure 6.1 for the different time points.

6.1.2 Numerical Results

As usual the solution was carried out with the standard Galerkin method. Discretization in time was done with help of the Crank-Nicolson scheme (3.11) and computations were done for the time step $\Delta t = 0.01$.

The discretization in space was performed with quadrilateral finite elements,

using the inf-sup stable Q_2/P_1^{disc} pair of finite elements spaces. Table 6.1 illustrates the corresponding geometry levels.

Table 6.1 – 2D instationary analytic example, geometry levels

Level	velocity d.o.f.	pressure d.o.f.	total d.o.f.
3	578	192	770
4	2178	768	2946
5	8450	3072	11522
6	33282	12288	45570

The Picard iteration is terminated when either the Euclidean norm of the residual is smaller than 10^{-8} or the maximum number of iterations at a level is achieved (it has been set to 500).

The stopping criteria for the GMRES iterations are either the Euclidean norm of the residual is smaller than 10^{-8} or the maximum number of iterations at a level is achieved (it has been set to 10000).

The stopping criteria for the case of the flexible GMRES with coupled multigrid and the multiple discretization multilevel method are either at most 5 iterations for solution of linear system were performed or the euclidean norm of the residual is reduced by factor 10. The multigrid methods were applied with the $F(1,1)$-cycle without damping.

Table 6.2 – 2D instationary analytic example, Q_2/P_1^{disc} discretization, $Re = 1000$, $\Delta t = 0.01$, solvers' comparison at $t = 0.01$.

Level	Parameters	UMFPACK	GMRES	GMRES +MG	GMRES +MDML	GMRES +SIMPLE	GMRES +LSC
3	Nonlinear iterations	2	1	3	4	4	1
	GMRES iterations	-	4080	5	9	40000*	5
4	Nonlinear iterations	2	2	3	4	27	1
	GMRES iterations	-	14666	5	9	270000*	4
5	Nonlinear iterations	2	10	3	4	227	1
	GMRES iterations	-	96352	5	9	2270000*	4
6	Nonlinear iterations	1	6	4	4	42	1
	GMRES iterations	-	57376	4	7	420000*	3

* - the method did not converge

Table 6.3 and Figure 6.2 show the following result. GMRES without preconditioning was the worse. The SIMPLE preconditioner did not converge on all geometry levels. The behavior of the LSC preconditioner in the instationary case was the same as in the case of the steady-state problems. It worked cheaper on the coarse grids and was slower than direct solvers and multigrid methods on the fine grids. GMRES with coupled multigrid preconditioners were the fastest methods on the fine grids.

Table 6.3 – 2D instationary analytic example, Q_2/P_1^{disc} discretization, $Re = 1000$, total time (s) of studying solvers for $t = 0 \ldots 5$, $\Delta t = 0.01$.

Level	UMFPACK	GMRES	GMRES +MG	GMRES +MDML	GMRES +LSC
3	11	187	14	21	13
4	51	1914	54	77	64
5	258	39497	179	304	416
6	1549	117106	715	1205	4114

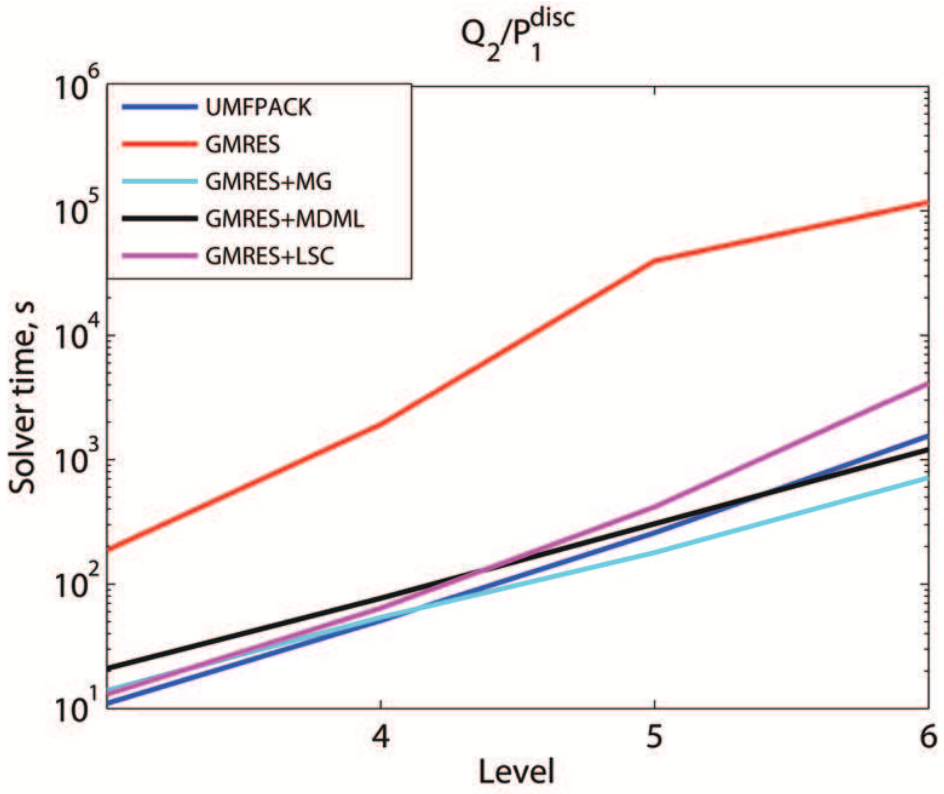

Figure 6.2 – 2D instationary analytic example, $Re = 1000$, Q_2/P_1^{disc} discretization, CPU times (s) for direct solver UMFPACK, GMRES without preconditioning and GMRES with coupled multigrid, the multiple discretization multilevel and LSC preconditioners.

6.2 The Instationary Flow Around a Cylinder with $Re = 100$

6.2.1 Implemented Example

Back to the problem discussed in Section 5.4, but in the case of unsteady flow and with five times stronger inflow. This problem is given in

$$\Omega = \{(0, 2.2) \times (0, 0.41)\} \setminus \left\{\mathbf{x} : (\mathbf{x} - (0.15, 0.15))^2 \leq 0.05^2\right\},$$

see Figure 5.11.

The kinematic viscosity of the fluid is given by $\nu = 10^{-3} m^2/s$. At the boundary $x = 0$ the steady-state inflow condition

$$\mathbf{v}(0\,m, y) = \frac{1}{0.41^2} \begin{pmatrix} 6y(0.41 - y) \\ 0 \end{pmatrix} m/s, \quad 0\,m \leq y \leq 0.41\,m \tag{6.2}$$

is used. At the boundary $x = 2.2$ the outflow condition (do-nothing condition)

$$\mathbb{S}\mathbf{n} = (-\nu\nabla\mathbf{v} + P\mathbb{I})\mathbf{n} = \mathbf{0} N/m^2 \qquad \text{on } \Gamma_{\text{outfl}} \tag{6.3}$$

is applied. On other parts of the boundary no-slip conditions are prescribed.

The initial condition is a fully developed flow field that has to be computed in a preprocessing step.

Based on the mean inflow velocity $U = 1m/s$, the diameter of the cylinder $L = 0.1m$ and the kinematic viscosity, the Reynolds number of the flow is $Re = 100$. The final time is set to be $T = 0.34$. In the fully developed periodic regime, a vortex shedding (von Kármán vortex street) can be observed behind the obstacle, see Figure 6.3.

6.2.2 Numerical Results

The Navier-Stokes equations were discretized in space, using the inf-sup stable pair of the finite element spaces P_2/P_1 on the triangle grids and Q_2/P_1^{disc} on the quadrilateral grids. In Table 6.4 the corresponding geometry levels are described.

Table 6.4 – 2D instationary flow around cylinder, geometry levels

Level	velocty d.o.f.		pressure d.o.f.		total d.o.f.	
	P_2/P_1	Q_2/P_1^{disc}	P_2/P_1	Q_2/P_1^{disc}	P_2/P_1	Q_2/P_1^{disc}
3	25408	27232	3248	9984	28656	37216
4	100480	107712	12704	39936	113184	147648
5	399616	428416	50240	159744	449856	588160

Discretization in time was done with help of the Crank-Nicolson scheme (3.11) and computations were done for the time step $\Delta t = 0.005$.

The Picard iteration is terminated when either the Euclidean norm of the residual is smaller than 10^{-8} or the maximum number of iterations at a level is achieved (it has been set to 30).

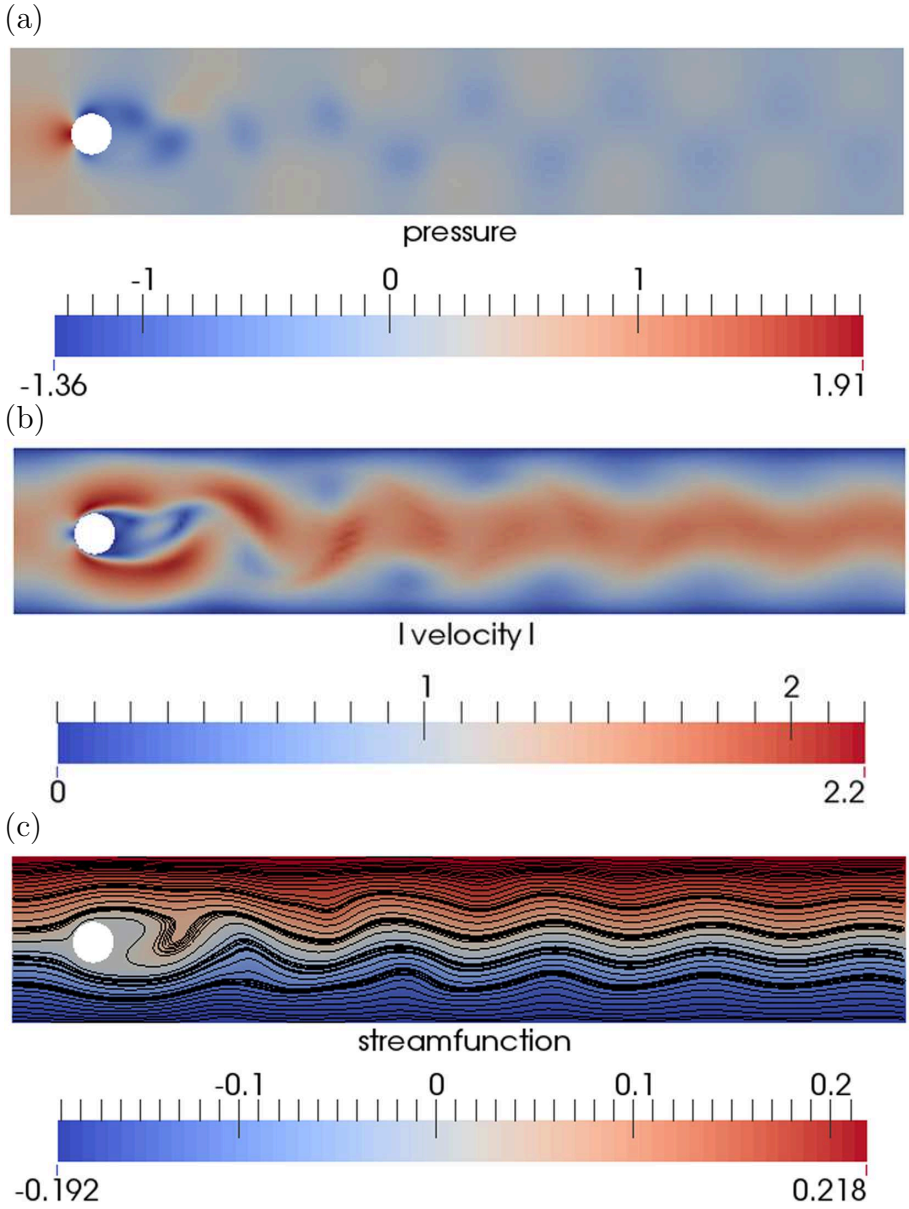

Figure 6.3 – 2D instationary flow around cylinder, $Re = 100$, (a) - pressure, (b) - magnitude of velocity and (c) - streamfunctions at $t = 0.005$.

The stopping criterion for the GMRES iterations are either the Euclidean norm of the residual is smaller than $8 \cdot 10^{-9}$ or the maximum number of iterations at a level is achieved (it has been set to 10).

The stopping criterion for the case of the flexible GMRES with coupled multigrid and the multiple discretization multilevel method is either at most 10 iterations for solution of linear system were performed or the Euclidean norm of the residual is reduced by factor 10. The multigrid methods were applied with the $F(1,1)$-cycle without damping.

Table 6.5 and Table 6.6 illustrate the performance of the above mentioned methods for the case of P_2/P_1: the number of iterations and the total computing times. Table 6.7 and Table 6.8 give the same information for the finite element discretiza-

tion Q_2/P_1^{disc}.

Table 6.5 – 2D instationary flow around cylinder, P_2/P_1 discretization, $Re = 100$, $\Delta t = 0.005$, solvers' comparison at $t = 0.005$.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
3	Nonlinear iterations	3	3	4	6	7
	Linear iterations	-	-	16	20	55
4	Nonlinear iterations	3	3	5	6	7
	Linear iterations	-	-	7	19	92
5	Nonlinear iterations	3	3	4	5	7
	Linear iterations	-	-	6	13	167

Table 6.6 – 2D instationary flow around cylinder, P_2/P_1 discretization, $Re = 100$, $\Delta t = 0.005$, total time (s) of studying solvers for $t = 0 \ldots 0.34$.

Level	UMFPACK	PARDISO	GMRES +MG	GMRES +MDML	GMRES +LSC
3	84	108	558	984	382
4	759	989	1564	3423	2506
5	8876	9517	5604	10145	20236

Table 6.7 – 2D instationary flow around cylinder, Q_2/P_1^{disc} discretization, $Re = 100$, $\Delta t = 0.005$, solvers' comparison at $t = 0.005$.

Level	Parameters	UMFPACK	PARDISO	FGMRES +MG	FGMRES +MDML	GMRES +LSC
3	Nonlinear iterations	3	3	7	6	7
	Linear iterations	-	-	31	25	26
4	Nonlinear iterations	3	3	5	5	7
	Linear iterations	-	-	13	17	37
5	Nonlinear iterations	3	3	5	5	6
	Linear iterations	-	-	12	14	44

Table 6.8 – 2D instationary flow around cylinder, Q_2/P_1^{disc} discretization, $Re = 100$, $\Delta t = 0.005$, total time (s) of studying solvers for $t = 0 \ldots 0.34$.

Level	UMFPACK	PARDISO	GMRES +MG	GMRES +MDML	GMRES +LSC
3	145	143	819	683	464
4	1171	1085	1448	1555	2731
5	9781	9363	5582	7343	14048

In the case of both discretizations GMRES with standard multigrid preconditioner was the fastest solver on the fine grids. On the coarse grid the best results

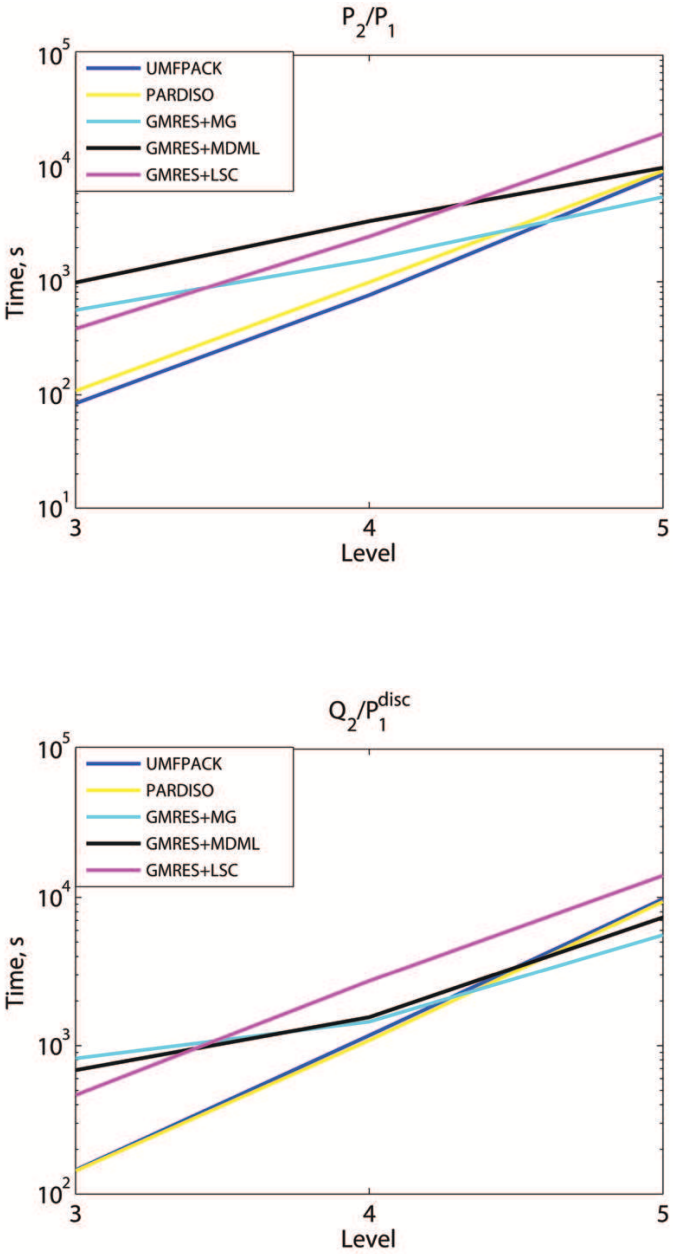

Figure 6.4 – 2D instationary flow around cylinder, $Re = 100$, $\Delta t = 0.005$, total time (s) of studying solvers for $t = 0 \ldots 0.34$.

had the direct solvers. The LSC preconditioner demonstrated the same results as in the previous examples.

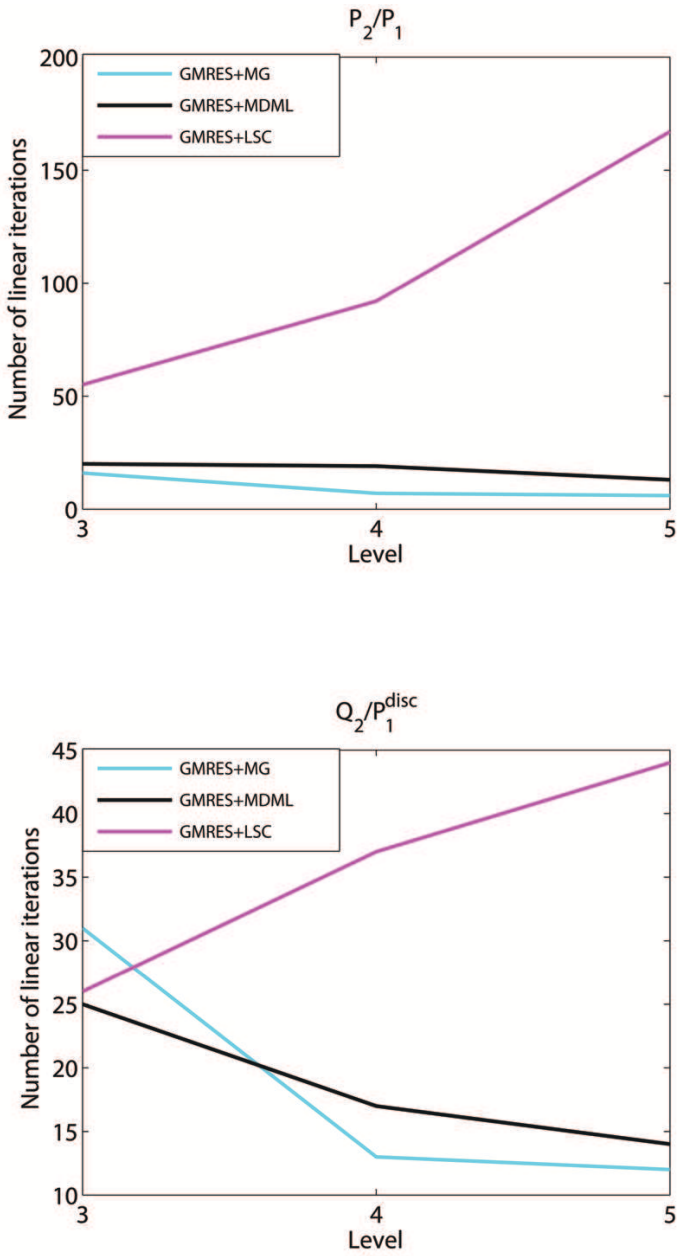

Figure 6.5 – 2D instationary flow around cylinder, $Re = 100$, $\Delta t = 0.005$, $t = 0.005$, number of linear iterations for GMRES with coupled multigrid, the multiple discretization multilevel and LSC preconditioners.

6.3 Summary of Results

Considering the solvers for the time-dependent Navier-Stokes equations one can observe:

- the iterative method GMRES with Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) preconditioner had the worst result;
- from the remaining methods GMRES without preconditioning was the slowest;
- on the grids with a small number of degrees of freedom the results of the direct solver and GMRES with the studied preconditioners (the standard multigrid, the multiple discretization multilevel method and LSC) were comparable and not much different from each other;
- on finer grids ($d.o.f. > 400000$) the best result demonstrated the flexible GMRES with the coupled multigrid preconditioners.

7

Conclusion and Outlooks

The subject of this project was to analyze different solution methods for linear systems in saddle point form, arising in finite element discretizations of incompressible flow problems.

Firstly the mathematical foundations for the incompressible Navier-Stokes equations and the corresponding saddle point systems were presented. Afterwards the basic properties of the saddle point matrices and the solution algorithms were studied. Special attention was paid to different techniques of preconditioning and the following preconditioners were considered in detail

- the standard multigrid method,
- the multiple discretization multilevel method,
- Least Squares Commutator (LSC) preconditioner and
- Semi-Implicit Method for Pressure-Linked Equations (SIMPLE).

As a numerical experiment the analysis of the above-mentioned methods and the comparison of their results with direct methods were carried out. The numerical tests showed:

- the direct solvers are the best choice for the problems with a number of degrees of freedom smaller than 400000;
- for the problems on the fine grids with a large number of degrees of freedom the one of the Krylov subspace methods for non-symmetric matrices such as flexible GMRES used with the preconditioner based on the coupled multigrid methods is optimal;
- SIMPLE preconditioner shows adequate results only in the case, if $diag(\mathbf{K})^{-1}$ is a good approximation to $\mathbf{K}^{-1}$. But diagonal approximation can yield poor results because the diagonal approximation does not capture enough information about the convection operator. This means that effectiveness of SIMPLE diminishes for flows that are convection-dominated. This fact was well demonstrated with our computational experiments. For the convection-dominated problems with a large Reynolds number (see Table 5.6 and Table 6.2) the flexible GMRES with SIMPLE preconditioner did not converge within the prescribed maximal number of steps. While in the problems with a small

Reynolds number the SIMPLE preconditioner demonstrates the convergence still worse than the other preconditioners;

- LSC preconditioner reveals the acceptable results on all studied examples, but its performance differs on coarse and fine grids. For example, on the coarse grids LSC preconditioner is worse than the direct solvers, but better than the preconditioner based on the coupled multigrid methods. While on the fine grids it has demonstrated the results, which are worse than the results of multigrid preconditioners and of direct solvers.

In a future one can extend the results to the three-dimensional case and check whether the conclusions, made above, are confirmed in this case.

Also the further investigations could consider the other block preconditioner based on the Schur complement approximation such as the exact and inexact pressure convection-diffusion preconditioners or modifications of SIMPLE algorithm as SIMPLEC and SIMPLER in order to compare their performance with other approaches.

List of Tables

List of Figures

Bibliography

[1] M. Benzi, G.H. Golub and J. Liesen: *Numerical Solution of Saddle Point Problems*, Acta Numerica, Cambridge University Press/ Cambridge, 1-137, 2005

[2] M. Benzi, G.H. Golub: *A preconditioner for generalized saddle point problems*, SIAM J. Matrix Anal. Appl., 26: 20-41, 2004

[3] H.C. Elman, D.J. Silvester and A.J. Wathen: *Finite Elements and Fast Iterative Solvers*, Numerical Mathematics and Scientific Computation, Oxford University Press/ Oxford, 2005

[4] H.C. Elman, V.E. Howle, J. Shadid, R. Shutterworth and R. Tumirano: *Block Preconditioner Based on Approximate Commutators*, SIAM J. Sci. Comput., 27 (5): 1651-1668, 2006

[5] H.C. Elman, V.E. Howle, J. Shadid, R. Shutterworth and R. Tumirano: *A Taxonomy and Comparison of Parallel Block Multi-Level Preconditioners for the Incompressible Navier-Stokes Equations*, J. of Computational Physics, 227: 1790-1808, 2008

[6] V. Girault, P.A. Raviart: *Finite Element Methods for Navier-Stokes Equations - Theory and Algorithms*, Volume 5 of Springer Series in Computational Mathematics, Springer Verlag/ Berlin, 1986

[7] V. John, G. Matthies: *Moonmd - a program package based on mapped finite element methods*, Computing and Visualization in Science, 6: 163-170, 2004

[8] V. John, G. Matthies, J. Rang: *A comparison of time-discretization/linearization approaches for the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg., 195: 5995-6010, 2006

[9] V. John: *Numerical Methods for Incompressible Flow Problems*, Lecture notes/ Berlin, 2014

[10] A. Segal, M. ur Rehman, C. Vuik: *Preconditioners for Incompressible Navier-Stokes Solvers*, Numerical Mathematics, 3(3): 245-275, 2010

[11] S. V. Patankar, D.A. Spalding: *A Calculation Procedure for Heat, Mass and Momentum Transfer in Three Dimensional Parabolic Flows*, Int. J. Heat and Mass Trans., 15: 1787-1806, 1972

[12] R. Temam: *Navier-Stokes Equations - Theory and Numerical Analysis*, North-Holland Publishing Company/ Amsterdam - New York - Oxford, 1977

[13] T.A. Davis: *Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method*, ACM Trans. Mathematical Software, 30(2): 196-199, 2004

[14] T.A. Davis: *A column pre-ordering strategy for the unsymmetric-pattern multifrontal method*, ACM Trans. Mathematical Software, 34(2): 165-195, 2004

[15] T.A. Davis, I.S. Duff: *An unsymmetric-pattern multifrontal method for sparse LU factorization*, Technical Report RAL-93-036, Rutherford Appleton Laboratory, 1993

[16] O. Schenk, A. Waechter, and M. Hagemann: *Matching-based Preprocessing Algorithms to the Solution of Saddle-Point Problems in Large-Scale Nonconvex Interior-Point Optimization*, Journal of Computational Optimization and Applications, 36(2-3): 321-341, 2007

[17] O. Schenk, M. Bollhoefer, and R. Roemer: *On large-scale diagonalization techniques for the Anderson model of localization*, SIAM Review 50, 91-112, 2008

[18] M. Luisier, O. Schenk et.al.: *Fast Methods for Computing Selected Elements of the Green's Function in Massively Parallel Nanoelectronic Device Simulations*, Euro-Par, LNCS 8097, 2013 F. Wolf, B. Mohr, and D. an Ney (Eds.), Springer-Verlag/ Berlin - Heidelberg, 533-544, 2013

[19] M. Schaf̈er, S. Turek: *The benchmark problem "Flow around a cylinder"*. Flow Simulation with HighPerformance, Computers II. Notes on Numerical Fluid Mechanics, vol. 52, Hirschel EH (ed.). Vieweg: Wiesbaden, 547-566, 1996

[20] A. Ern, J.L. Guermond: *Theory and Practice of Finite Elements*, Volume 159 of Springer Series in Applied Mathematical Sciences, Springer Verlag/ New York, 2004