Deutsche Forschungsgemeinschaft

Priority Program 1253

Optimization with Partial Differential Equations

P. Benner, J. Saak, M. Stoll and H. K. Weichelt

Efficient Solution of Large-Scale Saddle Point Systems Arising in Riccati-Based Boundary Feedback Stabilization of Incompressible Stokes Flow

June 2012

Preprint-Number SPP1253-130

http://www.am.uni-erlangen.de/home/spp1253

EFFICIENT SOLUTION OF LARGE-SCALE SADDLE POINT SYSTEMS ARISING IN RICCATI-BASED BOUNDARY FEEDBACK STABILIZATION OF INCOMPRESSIBLE STOKES FLOW

PETER BENNER[†‡], JENS SAAK[†‡], MARTIN STOLL[‡], AND HEIKO K. WEICHELT[†]

Abstract. We investigate numerical methods for solving large-scale saddle point systems which arise during the feedback control of flow problems. We focus on the Stokes equations that describe instationary, incompressible flows for low and moderate Reynolds numbers. After a mixed finite element discretization [23] we get a differential-algebraic system of differential index two [45]. To reduce this index, we follow the analytic ideas of Raymond [34, 35, 36] coupled with the projection idea of Heinkenschloss et al. [22]. Avoiding this explicit projection leads to solving a series of large-scale saddle point systems. In this paper we construct iterative methods to solve such saddle point systems by deriving efficient preconditioners based on the approaches of Wathen et al. [19, 42]. In addition, the main results can be extended to the non-symmetric case of linearized Navier-Stokes equations. We conclude with numerical examples showcasing the performance of our preconditioned iterative saddle point solver.

Key words. Flow control, Stokes equations, Riccati-based feedback, saddle point systems, Schur complement approximation

AMS subject classifications. 65F08, 65F10, 93D15, 49M15, 76D55

1. Introduction.

Stabilization of flow problems is crucial for many areas of engineering, for example the automotive and aerospace industries, as well as nanotechnology. In the latter case of microfluidic structures, we often encounter flow problems at moderate Reynolds numbers that do not require turbulence modeling [24]. In this paper, we are concerned with such a setting; therefore, we follow the Riccati-based feedback approach for stabilization of incompressible flow problems [6]. In contrast to the common idea of distributed control, we consider boundary control, which is more natural in a technical implementation. The analytic approach for feedback boundary stabilization of Stokes and linearized Navier-Stokes equations was given by Raymond in [34, 35, 36]. Raymond used the *Leray* projector to project the velocity functions onto the space of divergence-free vector functions [18], in order to deal with the algebraic constrains imposed by the incompressibility condition. Following Raymond's analytic approach, Bänsch and Benner investigated several ideas for the numerical treatment of the Leray projection in [6]. One of these ideas was to use the projection from balanced truncation model order reduction as discussed by Heinkenschloss et. al in [22].

This paper is a first step towards a thorough numerical treatment of the Leray projection, which is the key to robust implementations of optimal control for flow problems. We consider a symmetric and linear approach for instationary, incompressible flow

[†]Research group Mathematics in Industry and Technology (MiIT), Chemnitz University of Technology, Reichenhainer Str. 39/41, 09126 Chemnitz, Germany

[‡]Research Group Computational Methods in Systems and Control Theory (CSC), Max Planck Institute for Dynamics of Complex Technical Systems Magdeburg, Sandtorstr. 1, 39106 Magdeburg, Germany

problems, i.e., the Stokes equations,

$$\left.\begin{aligned} \frac{\partial}{\partial t}\mathbf{v}(t,\mathbf{x}) - \frac{1}{\mathrm{Re}}\Delta\mathbf{v}(t,\mathbf{x}) + \nabla p(t,\mathbf{x}) &= \mathbf{0}, \\ \nabla\cdot\mathbf{v}(t,\mathbf{x}) &= \mathbf{0}, \end{aligned}\right\} \text{ on } (0,\infty)\times\Omega, \tag{1.1}$$

with the time $t \in (0,\infty)$, the spatial variable $\mathbf{x} \in \Omega$, the velocity field $\mathbf{v}(t,\mathbf{x}) \in \mathbb{R}^2$, the pressure $p(t,\mathbf{x}) \in \mathbb{R}$ and the Reynolds number $\mathrm{Re} \in \mathbb{R}^+$. Additionally, we have $\Omega \subset \mathbb{R}^2$ as a bounded and connected domain with boundary $\Gamma = \partial\Omega$, some Dirichlet boundary conditions and appropriate initial conditions.

First, we show in Section 2 that the discretized system from the feedback control approach leads to differential algebraic equations. After showing why the projection idea of [22] can be used as numerical realization of the *Leray* projector, we end up with large-scale saddle point systems as the major ingredients to compute the Riccati feedback via this projection approach. In Section 3, we introduce the solution strategy for these saddle point systems based on the ideas of [19, 42]. Afterwards, we show numerical examples in Section 4 and summarize the ideas and results in Section 5.

2. Discretization. As it is common in instationary control problems [15, 22, 4, 3] we apply the *method of lines* [41] to the *Stokes equations*, which means we discretize (1.1) with a mixed finite element method [23] in space and get the following system of differential-algebraic equations

$$M\frac{d}{dt}\mathbf{z}(t) = A\mathbf{z}(t) + G\mathbf{p}(t) + \mathbf{f}(t), \tag{2.1a}$$

$$\mathbf{0} = G^T\mathbf{z}(t), \tag{2.1b}$$

with the discretized velocity $\mathbf{z}(t) \in \mathbb{R}^{n_v}$ and pressure $\mathbf{p}(t) \in \mathbb{R}^{n_p}$, the symmetric positive definite mass matrix $M \in \mathbb{R}^{n_v\times n_v}$, the symmetric positive semi-definite system matrix $A \in \mathbb{R}^{n_v\times n_v}$ and the discretized gradient $G \in \mathbb{R}^{n_v\times n_p}$ of rank n_p. For $P_2 - P_1$ *Taylor-Hood* finite elements, we have $n_v > n_p$ [23]. The source term $\mathbf{f}(t)$ describes the feedback influence via the boundary and can be expressed as $\mathbf{f}(t) = B\mathbf{u}(t)$ [6, Section 2], with the boundary control $\mathbf{u}(t) \in \mathbb{R}^{n_r}$ and the input operator $B \in \mathbb{R}^{n_v\times n_r}$ that maps the control onto the corresponding boundary nodes. Since in general, one can only observe the velocity in parts of the domain we add the output equation

$$\mathbf{y}(t) = C\,\mathbf{z}(t), \tag{2.1c}$$

with the output $\mathbf{y}(t) \in \mathbb{R}^{n_a}$ and the output operator $C \in \mathbb{R}^{n_a\times n_v}$ that selects the part of the domain where we want to measure the velocity, which is in our case a part of the outflow boundary.

Equations (2.1a)-(2.1b) represent a differential-algebraic system (DAE) of differential index two [45], written in a compact form as the matrix pencil

$$\left(\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}\right) \tag{2.2}$$

with a singular left hand side coefficient matrix $\begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}$. Because the solution set does not lie in an affine subspace but on a (hidden) manifold of the Euclidean space, we face some additional difficulties referring to the solvability (see, e.g., [45]). To avoid this problem we use the idea of index reduction described in [22, Section 3] which is demonstrated in the next subsection for *descriptor systems* like (2.1).

2.1. Projection Method. First, we show how the idea of index reduction by Heinkenschloss et al. [22], used for balanced truncation model order reduction of descriptor systems (2.1), represents a numerical realization of the *Leray* projector. Namely, we can convert (2.1) into a generalized state space system by using the projector

$$\Pi := I - G(G^T M^{-1} G)^{-1} G^T M^{-1},$$

defined in [22, Section 3]. To apply the analytic approach by Raymond [35] using the *Leray* projection, we need a self-adjoint projector onto the space of divergence free velocity functions. Hence, if $\mathbf{w}(t)$ lies in the range of such a projector, the algebraic constraint $G^T \mathbf{w}(t) = 0$ is fulfilled. Since range $\left(\Pi^T\right) = \text{null}\left(G^T\right)$, Π^T has the property $\Pi^T \mathbf{w}(t) = \mathbf{w}(t)$. We call

$$(\mathbf{x}, \mathbf{y})_{\mathbb{R}^{n_v}} := \mathbf{x}^T \mathbf{y}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{n_v}$$

the inner product of the Euclidean space $\mathbb{R}^{n_v}$ and

$$< \mathbf{x}, \mathbf{y} >_M := (M\mathbf{x}, \mathbf{y})_{\mathbb{R}^{n_v}} = \mathbf{x}^T M \mathbf{y}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{n_v}$$

the M-inner product for a symmetric matrix $M \in \mathbb{R}^{n_v \times n_v}$. It is easily shown that

$$< \Pi^T \mathbf{x}, \mathbf{y} >_M = (M\Pi^T \mathbf{x}, \mathbf{y})_{\mathbb{R}^{n_v}} = (M\mathbf{x}, \Pi^T \mathbf{y})_{\mathbb{R}^{n_v}} = < \mathbf{x}, \Pi^T \mathbf{y} >_M \ \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{n_v},$$

such that Π^T is self-adjoint with respect to the M-inner product. Additionally, null $\left(\Pi^T\right)$ = range $\left(M^{-1}G\right)$ which represents the curl-free components in the nullspace of Π^T. Finally, Π^T decomposes $\mathbb{R}^{n_v}$ into the direct sum of the two spaces $\mathbf{H}_{\text{div0}}(\mathbb{R}^{n_v})$, the space of divergence-free vector functions, and $\mathbf{H}_{\text{curl0}}(\mathbb{R}^{n_v})$, the space of curl-free vector functions [18]. Because projectors are unique we have Π^T as the discrete version of the *Leray* projector with respect to the M-inner product, which is the discrete version of the L_2-inner product.

The projector Π^T ensures that the solution fulfills the algebraic equation (2.1b) and simultaneously resides in the correct solution manifold, the so called *hidden manifold* [45] defined by

$$\mathbf{0} = G^T M^{-1} A \mathbf{z}(t) + G^T M^{-1} G \mathbf{p}(t) + G^T M^{-1} B \mathbf{u}(t).$$

Thus, the system (2.1) reduces to

$$\Pi M \Pi^T \frac{d}{dt} \mathbf{z}(t) = \Pi A \Pi^T \mathbf{z}(t) + \Pi B \mathbf{u}(t), \tag{2.3a}$$

$$\mathbf{y}(t) = C \Pi^T \mathbf{z}(t), \tag{2.3b}$$

with $\Pi^T \mathbf{z}(t) = \mathbf{z}(t)$. Because the nullspace of Π is non-trivial the matrix $\Pi M \Pi^T$ is not invertible. Therefore, for $\Theta_l, \Theta_r \in \mathbb{R}^{n_v \times (n_v - n_p)}$ satisfying

$$\Theta_l^T \Theta_r = I_{(n_v - n_p)},$$

we consider the decomposition

$$\Pi = \Theta_l \Theta_r^T. \tag{2.4}$$

Since the pencil (2.2) has $n_v - n_p$ finite eigenvalues [17, Theorem 2.1], (2.4) is computable using, for example the singular value decomposition [1]. If we substitute this decomposition into (2.3) we obtain

$$\Theta_r^T M \Theta_r \frac{d}{dt}\tilde{\mathbf{z}}(t) = \Theta_r^T A \Theta_r \tilde{\mathbf{z}}(t) + \Theta_r^T B \mathbf{u}(t), \tag{2.5a}$$

$$\mathbf{y}(t) = C\Theta_r^T \tilde{\mathbf{z}}(t), \tag{2.5b}$$

with $\tilde{\mathbf{z}} = \Theta_l^T \mathbf{z} \in \mathbb{R}^{n_v - n_p}$. After replacing $\mathcal{M} = \Theta_r^T M \Theta_r, \mathcal{A} = \Theta_r^T A \Theta_r, \mathcal{B} = \Theta_r^T B$, and $\mathcal{C} = C\Theta_r^T$, (2.5) yields

$$\mathcal{M}\frac{d}{dt}\tilde{\mathbf{z}}(t) = \mathcal{A}\tilde{\mathbf{z}}(t) + \mathcal{B}\mathbf{u}(t), \tag{2.6a}$$

$$\mathbf{y}(t) = \mathcal{C}\tilde{\mathbf{z}}(t), \tag{2.6b}$$

as a generalized state space system with a symmetric positive definite mass matrix $\mathcal{M}$.

In [22], Heinkenschloss et al. apply balanced truncation model order reduction to the system in (2.6), which requires computing the controllability and observability Gramians $\tilde{P}, \tilde{Q} \in \mathbb{R}^{(n_v - n_p)\times(n_v - n_p)}$, which solve the generalized Lyapunov equations

$$\mathcal{A}\tilde{P}\mathcal{M}^T + \mathcal{M}\tilde{P}\mathcal{A}^T = -\mathcal{B}\mathcal{B}^T, \tag{2.7}$$

$$\mathcal{A}^T\tilde{Q}\mathcal{M} + \mathcal{M}^T\tilde{Q}\mathcal{A} = -\mathcal{C}^T\mathcal{C}, \tag{2.8}$$

[22, Section 4]. In the next subsections we show that we have to solve similar equations for the Riccati-based feedback approach. It is clear that for computational purposes we do not want to form Π *explicitly* and we certainly cannot compute the decomposition presented in (2.4). We will later see how we can work with (2.6) *implicitly* by solving certain saddle point problems.

2.2. The Feedback Control Approach. For an asymptotic stabilization of the Stokes equations (1.1), we apply the *linear quadratic regulator approach (LQR)* to the system (2.6). An introduction to LQR for state space systems can be found in [28]. As we consider the generalized state space system (2.6) with $\mathcal{M} \neq I$, we have to modify the results as carried out in [40, Chapter 5.2]. In summary, we minimize

$$\mathcal{J}(\tilde{\mathbf{z}}(t), \mathbf{u}(t)) = \frac{1}{2}\int_0^\infty \tilde{\mathbf{z}}(t)^T\mathcal{C}^T\mathcal{C}\tilde{\mathbf{z}}(t) + \mathbf{u}(t)^T\mathbf{u}(t)\,\mathrm{dt}, \tag{2.9}$$

with subject to (2.6), which can be achieved by the optimal control defined by

$$\mathbf{u}_*(t) = -\underbrace{\mathcal{B}^T X \mathcal{M}}_{\mathcal{K}}\tilde{\mathbf{z}}_*(t) = -\mathcal{K}\tilde{\mathbf{z}}_*(t), \tag{2.10}$$

with the feedback operator $\mathcal{K}$ and X as the solution of the *Generalized Algebraic Riccati Equation (GARE)*

$$\mathbf{0} = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X\mathcal{M} + \mathcal{M}^T X\mathcal{A} - \mathcal{M}^T X\mathcal{B}\mathcal{B}^T X\mathcal{M} =: \mathfrak{R}(X). \tag{2.11}$$

Thus, we have to solve such a non-linear matrix equation to get the optimal control $\mathbf{u}_*(t)$. One way to solve this GARE is described in the next subsection.

Algorithm 1 Generalized low-rank Cholesky factor ADI iteration (G-LRCF-ADI)

Input: $\mathcal{A}^{(m)}, \mathcal{M}, \mathcal{W}^{(m)}$ and shift parameters $\{q_1, \ldots, q_{i_{max}}\}$
Output: $Z = Z_{i_{max}} \in \mathbb{C}^{n \times t_{i_{max}}}$, such that $ZZ^H \approx X^{(m+1)}$

1: $V_1 = \sqrt{-2\,\mathrm{Re}\,(q_1)}\left(\mathcal{A}^{(m)} + q_1\mathcal{M}\right)^{-T}(\mathcal{W}^{(m)})^T$
2: $Z_1 = V_1$
3: **for** $i = 2, 3, \ldots, i_{max}$ **do**
4: $\quad V_i = \sqrt{\mathrm{Re}\,(q_i)/\,\mathrm{Re}\,(q_{i-1})}\left(V_{i-1} - (q_i + \overline{q_{i-1}})\left(\mathcal{A}^{(m)} + q_i\mathcal{M}\right)^{-T}(\mathcal{M}^T V_{i-1})\right)$
5: $\quad Z_i = [Z_{i-1}\, V_i]$
6: **end for**

2.3. Solving Generalized Algebraic Riccati Equation. A common way to solve the non-linear matrix equation (2.11) is a Newton-type iteration as described in [2, 26]. The Newton system at step m is given by

$$X^{(m+1)} = X^{(m)} + N^{(m)}, \tag{2.12a}$$

with the update computed via

$$\mathfrak{R}'|_{X^{(m)}}(N^{(m)}) = -\mathfrak{R}(X^{(m)}), \tag{2.12b}$$

where $\mathfrak{R}'|_{X^{(m)}}$ is the Frechét derivative of the Riccati operator (2.11) at $X^{(m)}$ defined as

$$\mathfrak{R}'|_{X^{(m)}}: \quad N^{(m)} \mapsto (\mathcal{A} - \mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M})^T N^{(m)}\mathcal{M} + \mathcal{M}^T N^{(m)}(\mathcal{A} - \mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M}).$$

Therefore, we have to solve

$$\begin{aligned}&(\mathcal{A} - \mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M})^T N^{(m)}\mathcal{M} + \mathcal{M}^T N^{(m)}(\mathcal{A} - \mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M})\\ &= -\mathcal{C}^T\mathcal{C} - \mathcal{A}^T X^{(m)}\mathcal{M} - \mathcal{M}^T X^{(m)}\mathcal{A} + \mathcal{M}^T X^{(m)}\mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M}\end{aligned} \tag{2.13}$$

to compute the update $N^{(m)} = X^{(m+1)} - X^{(m)}$. If we plug this expression into (2.13), we obtain the generalized Lyapunov equation

$$(\mathcal{A}^{(m)})^T X^{(m+1)}\mathcal{M} + \mathcal{M}^T X^{(m+1)}\mathcal{A}^{(m)} = -(\mathcal{W}^{(m)})^T\mathcal{W}^{(m)}, \tag{2.14}$$

with $\mathcal{A}^{(m)} = \mathcal{A} - \mathcal{B}\mathcal{B}^T X^{(m)}\mathcal{M}$ and a right hand side split into the low-rank factors $\mathcal{W}^{(m)} = \begin{bmatrix} C \\ \mathcal{B}^T X^{(m)}\mathcal{M} \end{bmatrix}$ (see [25]). Equation (2.14) has the same structure as (2.8) and has to be solved at each Newton step. Hence, the index reduction method in [22] is applicable.

A solution strategy for this kind of equation is the *low-rank ADI iteration* [27, 8], which is extended for the generalized case in [7] as shown in Algorithm 1 using the above notation. For details of an effective implementation we refer to [40].

Combining the Newton iteration (2.12), as an outer iteration, and the G-LRCF-ADI (Algorithm 1), as an inner iteration yields the *generalized low-rank Cholesky factor Newton method (G-LRCF-NM)* [8, 6, 9].

Algorithm 2 Generalized low-rank Cholesky factor Newton method for Stokes

Input: M, A, G, B, C, shift parameters $\{q_1, \ldots, q_{n_{\mathrm{ADI}}}\}$
Output: feedback operator K

1: $K^0 = [\,]$
2: **for** $m = 1, 2, \ldots, n_{\mathrm{Newton}}$ **do**
3: $W^{(m)} = \begin{bmatrix} C^T & (K^{(m-1)})^T \end{bmatrix}$
4: Get V_1 by solving

$$\begin{bmatrix} A^T - (K^{(m-1)})^T B^T + q_1 M^T & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ * \end{bmatrix} = \begin{bmatrix} \sqrt{-2\,\mathrm{Re}\,(q_1)}\, W^{(m)} \\ 0 \end{bmatrix}$$

5: $K_1^{(m)} = B^T V_1 V_1^T M$
6: **for** $i = 2, 3, \ldots, n_{\mathrm{ADI}}$ **do**
7: Get $\tilde{V}$ by solving

$$\begin{bmatrix} A^T - (K^{(m-1)})^T B^T + q_i M^T & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{V} \\ * \end{bmatrix} = \begin{bmatrix} M^T V_{i-1} \\ 0 \end{bmatrix}$$

8: $V_i = \sqrt{\mathrm{Re}\,(q_i)/\,\mathrm{Re}\,(q_{i-1})}\left(V_{i-1} - (q_i + \overline{q_{i-1}})\tilde{V}\right)$
9: $K_i^{(m)} = K_{i-1}^{(m)} + B^T V_i V_i^T M$
10: **if** $\left(\frac{||K_i^{(m)} - K_{i-1}^{(m)}||_F}{||K_i^{(m)}||_F} < tol_{\mathrm{ADI}}\right)$ **then**
11: break
12: **end if**
13: **end for**
14: $K^{(m)} = K_{n_{\mathrm{ADI}}}^{(m)}$
15: **if** $\left(\frac{||K^{(m)} - K^{(m-1)}||_F}{||K^{(m)}||_F} < tol_{\mathrm{Newton}}\right)$ **then**
16: break
17: **end if**
18: **end for**
19: $K = K^{n_{\mathrm{Newton}}}$

In lines 1 and 4 of Algorithm 1 large-scale linear systems of equations involving the projected matrices have to be solved. Both have the following structure

$$\left(\mathcal{A}^{(m)} + q_i \mathcal{M}\right)^T \Lambda = \mathcal{Y}, \tag{2.15}$$

with different right hand sides $\mathcal{Y}$. Because one neither wants to build the dense projector Π^T nor its Θ-decomposition, we recall the results in [22, Section 5] which state that the solution of the Θ-projected equation (2.15) is also a solution of the Π-projected equation

$$\Pi\left(A^T - M^T X^{(m)} B B^T + q_i M^T\right)\Pi^T \Lambda = \Pi Y. \tag{2.16}$$

With [22, Lemma 5.2], one has to solve the equivalent linear system

$$\underbrace{\begin{bmatrix} A^T - M^T X^{(m)} BB^T + q_i M^T & G \\ G^T & 0 \end{bmatrix}}_{=:\tilde{\mathbf{A}}} \begin{bmatrix} \Lambda \\ * \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix}, \tag{2.17}$$

instead of system (2.16). The complete process for computing the feedback operator $K = B^T X M$ is shown in Algorithm 2.

The linear system (2.17) has to be solved in every ADI step. Note that despite the fact that the discretized diffusion operator A is symmetric, the $(1,1)$-block of the saddle point problem (2.17) is non-symmetric. Furthermore, note that every Newton step consists of several ADI steps. In the remainder of this paper we show how we can efficiently solve (2.17).

3. Solving Large-Scale Saddle Point Systems. Linear systems of the form (2.17) are often referred to as *saddle point systems*. A comprehensive overview about the numerical solution of saddle point systems is given in [10]. In the following we discuss the properties of the linear system (2.17) and our strategy for its efficient solution.

3.1. Properties of the Saddle Point System. The saddle point system arising from the feedback control approach for the Stokes equations is of the form (2.17). Although the matrices A, M and G are sparse, the low rank product $K^T B^T$ is dense, and the $(1,1)$-block of $\tilde{\mathbf{A}}$ also becomes dense, making Algorithm 2 inefficient. To avoid this, we can write system (2.17) in the form of a low rank update

$$\left(\underbrace{\begin{bmatrix} A^T + q_i M^T & G \\ G^T & 0 \end{bmatrix}}_{\mathbf{A}} - \underbrace{\begin{bmatrix} K^T \\ 0 \end{bmatrix}}_{\mathbf{K}^T} \underbrace{\begin{bmatrix} B^T & 0 \end{bmatrix}}_{\mathbf{B}^T} \right) \underbrace{\begin{bmatrix} \Lambda \\ * \end{bmatrix}}_{\mathbf{\Lambda}} = \underbrace{\begin{bmatrix} Y \\ 0 \end{bmatrix}}_{\mathbf{Y}},$$

or, more compactly,

$$(\mathbf{A} - \mathbf{K}^T \mathbf{B}^T) \mathbf{\Lambda} = \mathbf{Y}. \tag{3.1}$$

We then use the *Sherman-Morrison-Woodburry* formula [20] for the evaluation of (3.1), that is

$$(\mathbf{A} - \mathbf{K}^T \mathbf{B}^T)^{-1} = \left(I_{n_v} + \mathbf{A}^{-1} \mathbf{K}^T (I_{n_r} - \mathbf{B}^T \mathbf{A}^{-1} \mathbf{K}^T)^{-1} \mathbf{B}^T \right) \mathbf{A}^{-1}.$$

This means, we have to solve for $\mathbf{A}$ and the small dense matrix $(I_{n_r} - \mathbf{B}^T \mathbf{A}^{-1} \mathbf{K}^T)$ of size $n_r \ll n_v$ instead of solving with $\tilde{\mathbf{A}}$. Furthermore, we have to solve for $\mathbf{A}$ with the right hand side $\mathbf{K}^T$. Therefore, we just add $\mathbf{K}^T$ as additional n_r columns in the right hand side matrix $\mathbf{Y}$, which results in $\begin{bmatrix} Y & K^T \\ 0 & 0 \end{bmatrix} =: \begin{bmatrix} \tilde{Y} \\ 0 \end{bmatrix}$. Because we consider 2 boundary control parameters, $n_r = 2$. Finally, we obtain the saddle point system

$$\begin{bmatrix} A^T + q_i M^T & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \Lambda \\ * \end{bmatrix} = \begin{bmatrix} \tilde{Y} \\ 0 \end{bmatrix}, \tag{3.2}$$

with $M = M^T \succ 0, A = A^T \preceq 0$ and $q_i < 0$, which means the $(1,1)$-block of $\mathbf{A}$ is negative definite, whereas the whole matrix $\mathbf{A}$ is indefinite. This system has to be solved for many different right hand sides and a different shift parameter q_i in every ADI-step. Our strategy for computing the solution of (3.2) is discussed next.

3.2. Preconditioned Iterative Solvers. To solve the system (3.2), we use iterative methods, instead of direct solvers, because the size of the whole system $n = n_v + n_p$ becomes too large for usual finite element discretizations. For three-dimensional problems the fill-in generated by a direct method often allows only small problems to be solved, meaning that the use of iterative methods is imperative. Nevertheless, in their basic form the performance of iterative methods will deteriorate with decreasing mesh-size. This can be avoided if a preconditioner $\mathbf{P} \in \mathbb{R}^{n\times n}$ is introduced and the modified system

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{x} = \mathbf{P}^{-1}\mathbf{b}$$

is solved instead (see [38, 19]).

If we want to use a symmetric iterative solver (e.g., MINRES [32]) we have to use a symmetric positive definite preconditioner, such as the one given by

$$\mathbf{P} = \begin{bmatrix} -P_F & 0 \\ 0 & -P_{SC} \end{bmatrix}$$

[19, Section 6.1], where P_F approximates $F := A^T + p_i M^T \prec 0$ and $P_{SC} := G^T F^{-1} G$ is the *Schur complement*, which we can not form explicitly as this would require the dense matrix $F^{-1} \in \mathbb{R}^{n_v \times n_v}$. A good approximation for the steady Stokes case with moderate Reynolds numbers is $P_{SC} \approx -\frac{1}{\nu} M_p$ with M_p, the mass matrix, defined on the pressure space and $\nu := \frac{1}{\mathrm{Re}}$. We could also use a simpler operator, such as $P_{SC} \approx \mathrm{diag}\,(-\frac{1}{\nu} M_p)$. Both can be found in [19, Section 8.2]. We will call these versions

$$\hat{\mathbf{P}}_{\mathbf{M}} := \begin{bmatrix} -P_F & 0 \\ 0 & \mathrm{diag}\,(\frac{1}{\nu} M_p) \end{bmatrix} \quad \text{and} \quad \mathbf{P}_{\mathbf{M}} := \begin{bmatrix} -P_F & 0 \\ 0 & \frac{1}{\nu} M_p \end{bmatrix}.$$

Note that the matrix F has a form similar to the matrix obtained when the transient Stokes equation is discretized. Hence, the derivation presented now will somehow resemble preconditioning for transient Stokes systems.

Another way to approximate P_{SC} will be discussed next. Beforehand, we will show how to use a non-symmetric iterative method. The obvious advantage is that we can extend our method to non-symmetric systems that arise for more general Navier-Stokes systems, which is the long term goal of these investigations. Due to this fact we denote explicitly A^T, M^T in all algorithms, although both matrices are symmetric in the Stokes case.

A non-symmetric iterative method for the block structured and potentially non-symmetric matrix

$$\mathbf{F} := \begin{bmatrix} F & G \\ G^T & 0 \end{bmatrix}, \text{ with } F = A^T + q_i M^T$$

is GMRES [39]. Based on the ideas in [19, Section 8.1] we consider the block structured non-symmetric left preconditioner

$$\mathbf{P}_{\mathbf{G}} := \begin{bmatrix} P_F & 0 \\ G^T & -P_{SC} \end{bmatrix} \Rightarrow \mathbf{P}_{\mathbf{G}}^{-1} = \begin{bmatrix} P_F^{-1} & 0 \\ P_{SC}^{-1} G^T P_F^{-1} & -P_{SC}^{-1} \end{bmatrix}.$$

Note that in the Stokes case it is possible to use the block-triangular preconditioner $\mathbf{P}_{\mathbf{G}}$ within a symmetric iterative method. Namely, a variant of the CG-method [38]

introduced by Bramble and Pasciak in [13] can be used, which is only slightly more expensive than MINRES for the same problem.

Applying $\mathbf{P}_\mathbf{G}^{-1}$ from the left to $\mathbf{F}$ gives

$$\mathbf{P}_\mathbf{G}^{-1}\mathbf{F} = \begin{bmatrix} P_F^{-1}F & 0 \\ P_{SC}^{-1}G^T P_F^{-1}F - P_{SC}^{-1}G^T & P_{SC}^{-1}G^T P_F^{-1}G \end{bmatrix}. \tag{3.3}$$

For $P_F = F$ and $P_{SC} = G^T F^{-1} G$, (3.3) yields

$$\begin{bmatrix} F^{-1}F & 0 \\ P_{SC}^{-1}G^T F^{-1}F - P_{SC}^{-1}G^T & P_{SC}^{-1}G^T F^{-1}G \end{bmatrix} = \begin{bmatrix} I_{n_v} & 0 \\ 0 & I_{n_p} \end{bmatrix}. \tag{3.4}$$

As before, we cannot form $P_{SC} \in \mathbb{R}^{n_p \times n_p}$. Instead we apply the least squares commutator approach based on [19, Section 8.2]. Therefore, we consider a shifted diffusion operator on the velocity space

$$\mathcal{F} = -\nu\nabla^2 + q \cdot I$$

and suppose that the analogous operator on the pressure space also exists, that is

$$\mathcal{F}_p = (-\nu\nabla^2 + q \cdot I)_p.$$

We then define the least squares commutator of the shifted diffusion operators with the gradient operator

$$\mathcal{E} = (\mathcal{F})\nabla - \nabla(\mathcal{F}_p),$$

which should become small in some sense. If we plug in the discrete versions of the operators we obtain

$$\mathcal{E}_h = (M^{-1}F)M^{-1}G - M^{-1}G(M_p^{-1}F_p).$$

Pre-multiplying this by $G^T F^{-1} M$ and post-multiplying by $F_p^{-1} M_p$ yields

$$G^T M^{-1} G F_p^{-1} M_p \approx G^T F^{-1} G = P_{SC}.$$

In general it is infeasible to work with $G^T M^{-1} G$ as it is a large dense matrix. In [19, Section 5.5.1] it is shown that this matrix is spectrally equivalent to the Laplacian S_p defined on the pressure space. Hence, we get

$$P_{SC} \approx S_p F_p^{-1} M_p \quad \Rightarrow \quad P_{SC}^{-1} \approx M_p^{-1} F_p S_p^{-1},$$

with F_p, the shifted system matrix, defined on the pressure space. Thus, one has to solve a pure Neumann problem, defined on the pressure space and formally denoted as S_p^{-1} (as in [12]), multiply this with the system matrix F_p once, and solve a linear system with symmetric positive definite M_p, to apply the Schur complement approximation. A similar approach is applied to the incompressible Navier- Stokes equations in [11].

Note that for the symmetric Stokes case $F_p = A_p + qM_p$ the Schur complement approximation becomes slightly easier [42, Section 3], because the system matrix A_p is just the negative and scaled stiffness matrix $-\nu S_p$, such that

$$P_{SC} \approx S_p(-\nu S_p + qM_p)^{-1} M_p, \tag{3.5}$$

$$\Rightarrow \quad P_{SC}^{-1} \approx M_p^{-1}(-\nu S_p + qM_p)S_p^{-1} = -\nu M_p^{-1} + qS_p^{-1}. \tag{3.6}$$

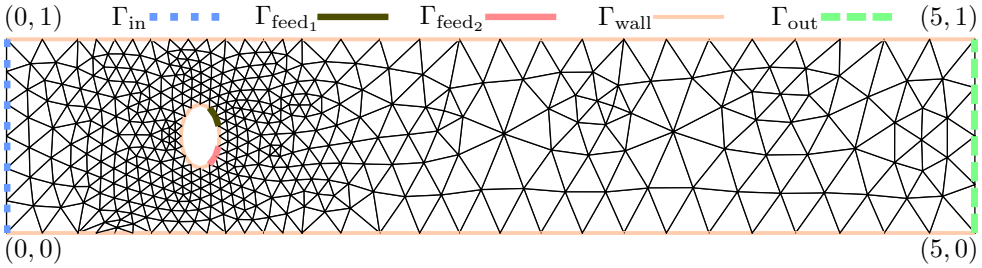


Figure 4.1: Coarsest discretization of *von Kármán vortex street* with coordinates and boundary conditions.

Summarizing, we just solve a pure Neumann problem S_p^{-1} [12] and a linear system with the mass matrix M_p. To increase the efficiency of this method we plan to explore *multigrid* methods [21, 37] to apply S_p and a *Chebyshev* semi iteration for M_p in the future [44]. At the moment we just pin a boundary node in S_p to fix the rank deficiency and use a direct solver (see, e.g., [12]).

Further details regarding this kind of Schur-complement approximation can be found in [16] for generalized Stokes systems, in [29, 14] for steady and unsteady Stokes systems, and in [30] for general partial differential equations. For unsteady Stokes problems the above approximation is known as the Cahouet-Chabard preconditioner [16].

In (3.4) we assumed F as the best choice for P_F. To get an efficient preconditioner we want to apply a *multigrid* approximation of F in the future as well [19, Section 8.3.2]. For the proof of concept we use a sparse direct solver in MATLAB® (via the "backslash" operator) in the numerical experiments for our proposed preconditioner.

Again, we can use the simple approximation $P_{SC} \approx -\frac{1}{\nu} M_p$ and define the preconditioners for the non-symmetric iterative method as

$$\hat{\mathbf{P}}_{\mathbf{G}} := \begin{bmatrix} P_F & 0 \\ G^T & \frac{1}{\nu} M_p \end{bmatrix} \quad \text{and} \quad \mathbf{P}_{\mathbf{G}} := \begin{bmatrix} P_F & 0 \\ G^T & -S_p(-\nu S_p + q M_p)^{-1} M_p \end{bmatrix}.$$

We compare both methods with the appropriate preconditioner in the next section. Additionally, we discuss some problems with the nested iterations and remark on the numerical realization.

4. Numerical Examples. The example we use for our domain Ω is the *von Kármán vortex street* depicted in Figure 4.1. We have a parabolic inflow with a maximum value of 1.0 at Γ_{in}, which has a diameter $d_{\text{in}} = 1$. The fluid passes an elliptic obstacle with the center at the coordinates $(1, 0.5)$, which has a width of $\frac{1}{5}\, d_{\text{in}}$ and a height of $\frac{1}{3}\, d_{\text{in}}$. The fluid flows out of the domain at Γ_{out} without any barrier. To influence the flow field we can blow in or exhaust fluid on the back of the obstacle via $\Gamma_{\text{feed}_1}, \Gamma_{\text{feed}_2}$. Naturally, we impose no-slip conditions on Γ_{wall}.

The matrices for the numerical tests arise from a standard mixed finite element discretization (e.g., P_2-P_1 *Taylor-Hood* elements like in Figure 4.2) of Ω. Using a *Bänsch-refinement* [5] we get five different magnitudes for n_v and n_p, where every second

level corresponds to one level of global uniform refinement (see Table 4.1). Figure 4.1 shows the coarsest grid Level 1.

Level	n_v	n_p
1	3 452	453
2	8 726	1 123
3	20 512	2 615
4	45 718	5 783
5	99 652	12 566

Table 4.1: Levels of refinement

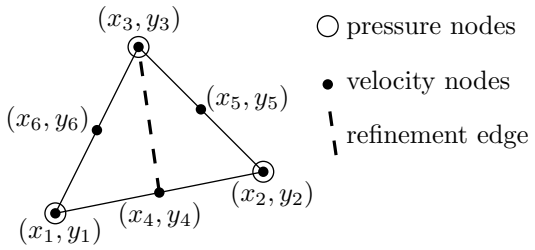


Figure 4.2: P_2-P_1 *Taylor-Hood* element

All computations were done with MATLAB R2010b on a 64-bit server with CPU type Intel® Xeon® X5650 @2.67GHz, with 2 CPUs, 12 Cores (6 Cores per CPU) and 48 GB main memory available.

First, we compare the iterative methods and their preconditioners, defined in Section 3.2, to explain the choice for the tests concerning the parameter dependence of the nested iteration.

4.1. Solving the Saddle Point System. First, we compare MINRES and GMRES with the preconditioners defined before. Therefore, the MATLAB implementations of both methods are used and the preconditioning is realized as function handles to increase the efficiency.

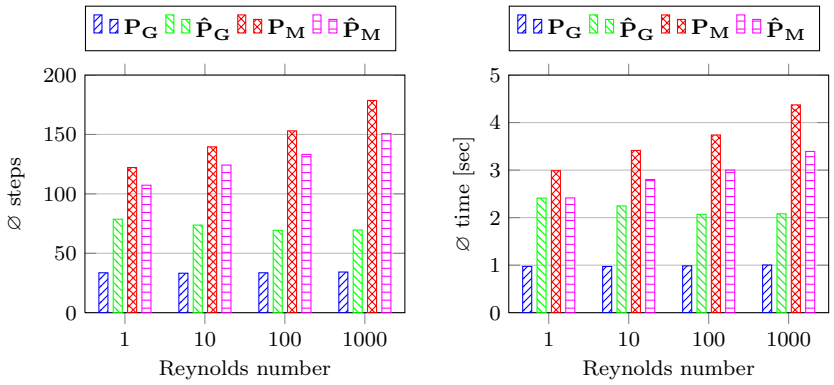
We use

$$\hat{\mathbf{P}}_{\mathbf{M}} := \begin{bmatrix} -P_F & 0 \\ 0 & \operatorname{diag}(\frac{1}{\nu} M_p) \end{bmatrix}, \quad \mathbf{P}_{\mathbf{M}} := \begin{bmatrix} -P_F & 0 \\ 0 & \frac{1}{\nu} M_p \end{bmatrix}$$

for MINRES as symmetric preconditioners and

$$\hat{\mathbf{P}}_{\mathbf{G}} := \begin{bmatrix} P_F & 0 \\ G^T & \frac{1}{\nu} M_p \end{bmatrix}, \qquad \mathbf{P}_{\mathbf{G}} := \begin{bmatrix} P_F & 0 \\ G^T & -S_p(-\nu S_p + q M_p)^{-1} M_p \end{bmatrix}$$

for GMRES and solve the first Newton step with all methods for an iteration tolerance of $tol_{\text{solve\_SPS}} = 10^{-12}$. This computation includes roughly fifteen ADI steps with seven right hand sides in system (3.2) in every ADI step. In Figure 4.3 we show the average number of steps over all these right hand sides on the left, and the average time over all these right hand sides on the right. The plots are clustered for different Reynolds numbers and the order of methods is the same as in the legend. GMRES with $\mathbf{P}_{\mathbf{G}}$ wins this comparison for each configuration, although every GMRES step ($0.029 - 0.030$ seconds) needs a little bit more time than a MINRES step ($0.024 - 0.025$ seconds). Our tests also showed that after a certain point all methods, except GMRES with $\mathbf{P}_{\mathbf{G}}$, showed stagnation of the residual norm. This effect occurs due to the Schur complement approximation not being sufficiently accurate (see [31]). Moreover, we observed that this high accuracy is not needed to achieve our main goal, which is to solve the Newton-ADI iteration (Algorithm 2). The influence of the accuracy of the saddle point solvers on the convergence of the whole algorithm is presented in the next subsection.



(a) Average number of steps over all right hand sides during the first Newton step for different Reynolds numbers (Level 1).

(b) Average time over all right hand sides during the first Newton step for different Reynolds numbers (Level 1).

Figure 4.3: Compare the different methods with desired $tol_{\text{solve\_SPS}} = 10^{-12}$.

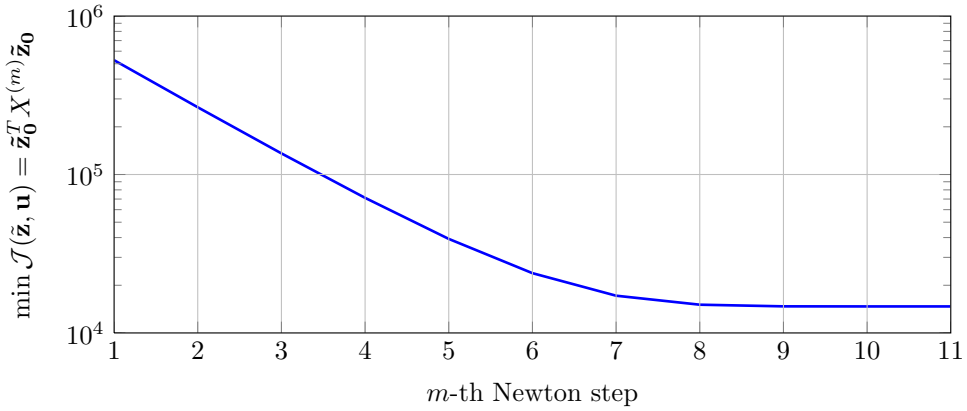


Figure 4.4: Evolution of optimal costs during the Newton iteration (Level 1).

4.2. Nested Iteration. In Algorithm 2 we have a nested iteration with the outermost Newton iteration, the central ADI iteration, and finally the innermost iteration, where the saddle point system (3.2) has to be solved at every ADI step. To compute one Newton step (2.12b) we need a number of ADI steps to reach the stopping criterion for the Newton iteration $tol_{\text{Newton}} \approx 5 \cdot 10^{-5}$.

To visualize the convergence we show the evolution of the optimal costs, that is the value of the cost functional (2.9), in Figure 4.4. This convergence only depends on the chosen cost functional (2.9) and thereby it is dependent on the problem conditions.

The convergence of the ADI iteration depends heavily on the shift parameters q_i. We use the heuristic *Penzl* shifts [33] or the *Wachspress* shifts [43]. Both approaches achieve similar results and computation times are lower for the *Penzl* shifts, so we use this method for the remaining computations with $tol_{\text{ADI}} \approx 5 \cdot 10^{-7}$.

$tol_{\text{solve\_SPS}}$	GMRES $\mathbf{P_G}$		GMRES $\hat{\mathbf{P}}_\mathbf{G}$		MINRES $\mathbf{P_M}$		MINRES $\hat{\mathbf{P}}_\mathbf{G}$	
	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time
10^{-4}	> 50	> 159	28	148	**29**	**51**	**29**	**59**
10^{-5}	26	99	28	189	28	80	26	72
$\mathbf{10^{-6}}$	**15**	**65**	20	154	32	133	32	141
10^{-7}	15	72	15	143	29	187	34	215
10^{-8}	15	79	15	171	34	394	34	330
10^{-9}	15	86	15	201	28	389	28	321
10^{-10}	15	92	15	225	20	303	18	223
10^{-11}	15	99	15	245	15	266	15	218
10^{-12}	15	105	15	265	15	315	15	255
direct	$n_{\text{ADI}} = 15$							

(a) Reynolds number 1.

$tol_{\text{solve\_SPS}}$	GMRES $\mathbf{P_G}$		GMRES $\hat{\mathbf{P}}_\mathbf{G}$		MINRES $\mathbf{P_M}$		MINRES $\hat{\mathbf{P}}_\mathbf{G}$	
	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time
10^{-4}	> 50	> 157	37	181	28	122	35	157
10^{-5}	23	84	28	169	28	175	35	207
10^{-6}	24	101	24	155	28	219	26	188
$\mathbf{10^{-7}}$	**15**	**72**	15	127	37	429	37	367
10^{-8}	15	78	15	147	28	393	28	328
10^{-9}	15	85	15	172	24	362	24	308
10^{-10}	15	91	15	196	15	276	15	227
10^{-11}	15	97	15	220	15	317	15	262
10^{-12}	15	105	15	240	15	359	15	295
direct	$n_{\text{ADI}} = 15$							

(b) Reynolds number 10

$tol_{\text{solve\_SPS}}$	GMRES $\mathbf{P_G}$		GMRES $\hat{\mathbf{P}}_\mathbf{G}$		MINRES $\mathbf{P_M}$		MINRES $\hat{\mathbf{P}}_\mathbf{G}$	
	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time
10^{-4}	> 50	> 166	29	69	25	117	> 50	> 230
10^{-5}	22	82	36	118	43	269	> 50	> 312
$\mathbf{10^{-6}}$	**15**	**64**	27	123	43	416	43	369
10^{-7}	15	71	25	164	27	344	27	288
10^{-8}	15	78	20	154	20	290	20	245
10^{-9}	15	85	15	142	15	262	15	214
10^{-10}	15	91	15	168	15	304	15	248
10^{-11}	15	98	15	195	15	349	15	283
10^{-12}	15	106	15	221	15	394	15	316
direct	$n_{\text{ADI}} = 15$							

(c) Reynolds number 100

$tol_{\text{solve\_SPS}}$	GMRES $\mathbf{P_G}$		GMRES $\hat{\mathbf{P}}_\mathbf{G}$		MINRES $\mathbf{P_M}$		MINRES $\hat{\mathbf{P}}_\mathbf{G}$	
	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time	n_{ADI}	time
10^{-4}	> 50	> 172	**25**	**59**	34	266	> 50	> 287
10^{-5}	25	98	34	116	34	428	> 50	> 383
$\mathbf{10^{-6}}$	**14**	**60**	34	177	31	500	22	239
10^{-7}	14	67	31	220	20	319	20	256
10^{-8}	14	74	20	153	14	260	14	207
10^{-9}	14	80	14	128	14	305	14	241
10^{-10}	14	87	14	153	14	350	14	274
10^{-11}	14	93	14	178	14	391	14	306
10^{-12}	14	101	14	206	14	430	14	334
direct	$n_{\text{ADI}} = 14$							

(d) Reynolds number 1000

Table 4.2: Overview of the influence of $tol_{\text{solve\_SPS}}$ to ADI convergence for the first Newton step (Level 1).

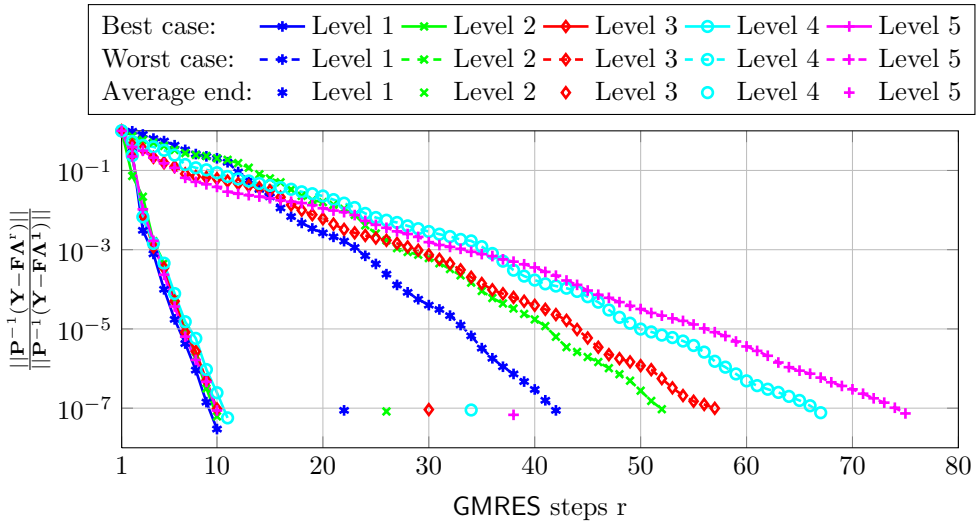


Figure 4.5: Relatively preconditioned residuals of GMRES with $\mathbf{P_G}$ for refinement levels of Table 4.1 during the first Newton step (Re = 10).

Depending on how accurately the saddle point system is solved at every ADI step, we need more or less ADI steps to reach this tolerance. In Table 4.2 we compare the number of ADI steps and the required computation time in relation to the achieved tolerance $tol_{\text{solve\_SPS}}$ for the methods and preconditioners defined before. We computed these results for different Reynolds numbers and summarized them in different sub-tables. In the bottom line of every table we put the number of ADI steps for a direct solver as a reference level, while assuming the direct solver reaches the highest accuracy in the ADI iteration. The best configurations are plotted in boldface. We notice that the configuration of GMRES with $\mathbf{P_G}$ is the best choice on average, although the other methods are sometimes faster. This configuration reaches the smallest number of ADI steps first that we would obtain using a direct solver. At this point the number of ADI steps does not decrease if we increase the accuracy further. So we do not necessarily need the highest accuracy we could achieve.

Additionally, there is a lower bound for the accuracy to have convergence at all ($\max n_{\text{ADI}} = 50$). For this level of refinement the direct solver is of course much faster than our iterative method, because the number of unknowns is comparably small. Based on this observation and the fact that we want to deal with the Navier-Stokes equations in the future, we use GMRES as the iterative method in all further considerations and show more parameters which influence convergence.

4.3. Parameter Dependence. First, in Figure 4.5 the relative preconditioned residuals of GMRES with $\mathbf{P_G}$ are listed for the different levels of refinement in Table 4.1. We plotted the best case and the worst case as residual vectors, and the average over the final residuals as markers. Notice that, the best cases need nearly the same numbers of steps (around 10 steps). However, in the worst case the numbers of steps varies between 42 and 75 steps. On average, the number of steps for various levels of refinement seem almost constant (between 21 and 37).

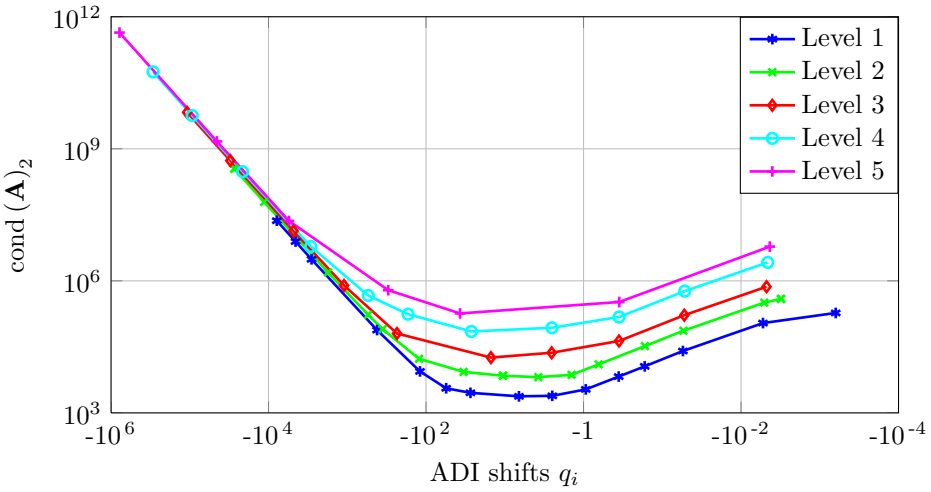
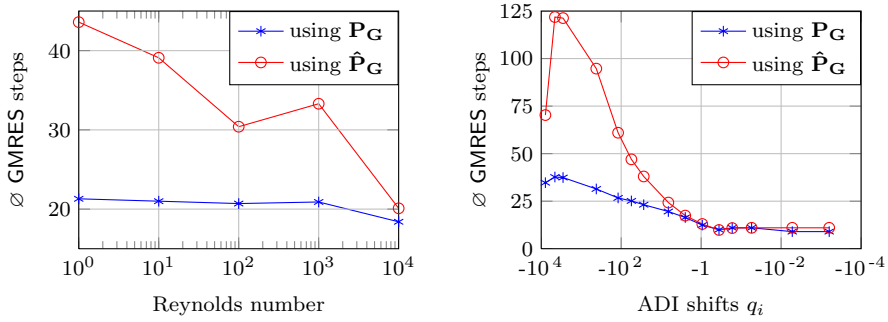


Figure 4.6: Condition number of $\mathbf{A}$ concerning the ADI shifts during the first Newton step for refinement levels of Table 4.1 (Re = 10).

Figure 4.6 gives an explanation for the larger variation in the worst case. The condition number of matrix $\mathbf{A}$, for which we solve the saddle point systems, increases as we refine our mesh. In detail, the condition number roughly scales up with a factor of 10 for each refinement level. Considering this fact, we have a robust (with respect to the mesh parameter), preconditioned, iterative method to solve saddle point systems like (3.2).

Two further parameters which influence the properties of the saddle point system are the Reynolds number and the ADI shift. For different Reynolds numbers we get a different system matrix A and also different Schur complement approximations (3.6). The influence of the Reynolds number concerning the various methods was already shown in Figure 4.3a, where we evaluated the performance of the various preconditioners. Figure 4.7a shows the influence of the Reynolds number for refinement Level 1. We averaged the number of GMRES steps over all right hand sides, which appear during the first Newton step. We see that we need nearly the same amount of iterations if we vary Re for $\mathbf{P_G}$. In contrast, we need more iterations for smaller Reynolds numbers if we use $\hat{\mathbf{P}}_\mathbf{G}$. This behavior seems non natural if we observe the construction of the preconditioners and needs further investigation.

The second parameter, which changes during Algorithm 2 in each ADI step, is the ADI shift q_i. Again, this change influences the saddle point system itself and also the least squares Schur complement approximation (3.6). Figure 4.7b shows the dependence of the number of iterations on ADI shifts q_i for refinement Level 1 and Re = 10, where q_i are the shifts used in the first Newton step. In contrast to the Reynolds number, we notice that we need more iterations if we increase the absolute value of q_i for $\mathbf{P_G}$. But again we have worse results if we use $\hat{\mathbf{P}}_\mathbf{G}$. Unfortunately, we cannot influence the absolute value of the shift too much, because both shift concepts create a similar distribution of shifts.



(a) Average number of GMRES steps for different Reynolds numbers during the first Newton step (Level 1).

(b) Average number of GMRES steps for the ADI shifts q_i during the first Newton step (Re = 10, Level 1).

Figure 4.7: Number of GMRES iterations influenced by different parameters.

Finally, we note that all results shown here point out that the use of GMRES with the $\mathbf{P_G}$ preconditioner is the best choice for solving the saddle point systems (3.2) arising in the Riccati feedback stabilization approach (2.10) for the Stokes equations (1.1). We believe that this preconditioner in combination with the Bramble-Pasciak CG-method would perform equally well [13].

5. Conclusion and Outlook. In this paper we have shown how we can use the idea of index reduction for balanced truncation model order reduction [22] to derive a numerical realization of the *Leray* projector. This projector is the main tool of the analytical approach for Riccati-based boundary feedback stabilization of flow problems in [35]. We have considered the Stokes flow as a first example towards a thorough numerical investigation. To compute the optimal control we have to solve an algebraic Riccati equation corresponding to the projected system. We have applied a Newton type method for solving this Riccati equation. The Lyapunov equations arising in each Newton step have been solved with a low-rank version of the Alternating Direction Implicit (ADI) method. Avoiding the explicit projection leads to large-scale saddle point systems, which have to be solved, in the central loop of this nested iteration.

We have pointed out the properties of these resulting saddle point systems and have introduced an efficient way to solve such equations. To this end, we have investigated preconditioners to be used in iterative methods based on the ideas of Wathen et al. [19, 42]. As major result we could adapt known Schur complement approximations to our problem setting.

We have illustrated the influence of the arising parameters with numerical examples to compare the different variants of preconditioned Krylov solvers. We have shown the interactions of the nested iterations. Summarizing the test results, we could point out a fast set up for these kinds of problems.

In the future, we will deal with the non-symmetric Navier-Stokes flow and explore in detail the additional difficulties arising there. The most important facts will be the ADI shift computation for the non-symmetric Navier Stokes case, as well as the instability of these systems.

We will further investigate the acceleration of the iterative solvers in order to deal with multiple right hand sides, which is an advantage of the direct solvers that solve them almost simultaneously.

Acknowledgments. The work presented in this paper is developed within the project *Optimal Control-Based Feedback Stabilization in Multi-Field Flow Problems*. The project is a joint work with Eberhard Bänsch and is part of the DFG priority program 1253: *Optimization with Partial Differential Equations*. We thank Andy Wathen for the inspiring discussions about preconditioning of iterative methods.

REFERENCES

[1] E. Anderson, Z. Bai, and C. Bischof, *LAPACK Users' Guide*, Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics. XV, 235 p. , 1992.

[2] W. Arnold and A. J. Laub, *Generalized Eigenproblem Algorithms and Software for Algebraic Riccati Equations*, Proc. IEEE, 72 (1984), pp. 1746–1754.

[3] H. Banks and K. Ito, *A Numerical Algorithm for Optimal Feedback Gains in High Dimensional Linear Quadratic Regulator Problems*, SIAM J. Cont. Optim., 29 (1991), pp. 499–515.

[4] H. Banks and K. Kunisch, *The Linear Regulator Problem for Parabolic Systems*, SIAM J. Cont. Optim., 22 (1984), pp. 684–698.

[5] E. Bänsch, *An adaptive finite-element strategy for the three-dimensional time-dependent Navier-Stokes equations*, Journal of Computational and Applied Mathematics, 36 (1991), pp. 3–28.

[6] E. Bänsch and P. Benner, *Stabilization of Incompressible Flow Problems by Riccati-Based Feedback*, in Constrained Optimization and Optimal Control for Partial Differential Equations, G. Leugering, S. Engell, A. Griewank, M. Hinze, R. Rannacher, V. Schulz, M. Ulbrich, and S. Ulbrich, eds., vol. 160 of International Series of Numerical Mathematics, Birkhäuser, 2012, pp. 5–20.

[7] P. Benner, *Solving large-scale control problems*, IEEE Control Systems Magazine, 14 (2004), pp. 44–59.

[8] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems*, Numer. Lin. Alg. Appl., 15 (2008), pp. 755–777.

[9] P. Benner and J. Saak, *A Galerkin-Newton-ADI Method for Solving Large-Scale Algebraic Riccati Equations*, Preprint SPP1253 090, 2010.

[10] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numerica, 14 (2005), pp. 1–137.

[11] M. Benzi, M. A. Olshanskii, and Z. Wang, *Modified augmented Lagrangian preconditioners for the incompressible Navier-Stokes equations.*, Int. J. Numer. Methods Fluids, 66 (2011), pp. 486–508.

[12] P. Bochev and R. Lehoucq, *On the finite element solution of the pure Neumann problem*, SIAM Review, 47 (2005), pp. 50–66.

[13] J. H. Bramble and J. E. Pasciak, *A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems*, Math. Comp, 50 (1988), pp. 1–17.

[14] ———, *Iterative techniques for time dependent Stokes problems*, Computers & Mathematics with Applications, 33 (1997), pp. 13–30.

[15] J. A. Burns, E. W. Sachs, and L. Zietsman, *Mesh Independence of Kleinman–Newton Iterations for Riccati Equations in Hilbert Space*, SIAM Journal on Control and Optimization, 47 (2008), pp. 2663–2692.

[16] J. Cahouet and J.-P. Chabard, *Some fast 3D finite element solvers for the generalized Stokes problem*, International Journal for Numerical Methods in Fluids, 8 (1988), pp. 869–895.

[17] K. A. Cliffe, T. J. Garratt, and A. Spence, *Eigenvalues of Block Matrices Arising from Problems in Fluid Mechanics*, SIAM Journal on Matrix Analysis and Applications, 15 (1994), pp. 1310–1318.

[18] E. Deriaz and V. Perrier, *Orthogonal Helmholtz decomposition in arbitrary dimension using divergence-free and curl-free wavelets*, Applied and Computational Harmonic Analysis, 26 (2009), pp. 249–269.

[19] H. Elman, D. Silvester, and A. Wathen, *Finite Elements and Fast Iterative Solvers: with applications in incompressible fluid dynamics*, Oxford University Press, Oxford, 2005.
[20] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, third ed., 1996.
[21] W. Hackbusch, *Multi-Grid Methods and Applications*, vol. 4 of Springer Series in Computational Mathematics, Springer Verlag, 1985.
[22] M. Heinkenschloss, D. C. Sorensen, and K. Sun, *Balanced truncation model reduction for a class of descriptor systems with application to the Oseen equations*, SIAM Journal on Scientific Computing, 30 (2008), pp. 1038–1063.
[23] P. Hood and C. Taylor, *Navier-Stokes equations using mixed interpolation*, in Finite Element Methods in Flow Problems, J. T. Oden, R. H. Gallagher, C. Taylor, and O. C. Zienkiewicz, eds., University of Alabama in Huntsville Press, 1974, pp. 121–132.
[24] B. Kirby, *Micro- and Nanoscale Fluid Mechanics: Transport in Microfluidic Devices*, Cambridge University Press, 2010.
[25] D. Kleinman, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.
[26] P. Lancaster and L. Rodman, *The Algebraic Riccati Equation*, Oxford University Press, Oxford, 1995.
[27] J.-R. Li and J. White, *Low rank solution of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280.
[28] A. Locatelli, *Optimal Control: An Introduction*, Birkhäuser Verlag, Basel, Boston, Berlin, 2001.
[29] Y. Maday, D. Meiron, A. T. Patera, and E. M. Rønquist, *Analysis of Iterative Methods for the Steady and Unsteady Stokes Problem: Application to Spectral Element Discretizations*, SIAM Journal on Scientific Computing, 14 (1993), pp. 310–337.
[30] K.-A. Mardal and R. Winther, *Preconditioning discretizations of systems of partial differential equations*, Numerical Linear Algebra with Applications, 18 (2011), pp. 1–40.
[31] M. Olshanskii and V. Simoncini, *Acquired clustering properties and solution of certain saddle point systems.*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2754–2768.
[32] C. C. Paige and M. A. Saunders, *Solutions of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal, 12 (1975), pp. 617–629.
[33] T. Penzl, Lyapack *Users Guide*, Tech. Rep. SFB393/00-33, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, Germany, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.
[34] J. Raymond, *Local boundary feedback stabilization of the Navier-Stokes equations*, in Control Systems: Theory, Numerics and Applications, Rome, 30 March – 1 April 2005, Proceedings of Science, SISSA, `http://pos.sissa.it`, 2005.
[35] ———, *Feedback boundary stabilization of the two-dimensional Navier-Stokes equations*, SIAM Journal on Control and Optimization, 45 (2006), pp. 790–828.
[36] ———, *Feedback boundary stabilization of the three-dimensional incombressible Navier-Stokes equations*, J.Math. Pures Appl. (9), 87 (2007), pp. 627–669.
[37] J. Ruge and K. Stüben, *Algebraic multigrid (amg)*, in Multigrid Methods, E. S. McCormich, ed., vol. 5 of Frontiers in Applied Mathematics, SIAM, 1987, pp. 73–130.
[38] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, USA, 2003.
[39] Y. Saad and M. H. Schultz, *Gmres: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 856–869.
[40] J. Saak, *Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction*, PhD thesis, Chemnitz University of Technology, July 2009.
[41] W. E. Schiesser, *Numerical Methods of Lines*, Academic Press, 1991.
[42] M. Stoll and A. Wathen, *All-at-once solution of time-dependent Stokes control*, Max Planck Institute Magdeburg Preprints 11-03, (2011). `http://www.mpi-magdeburg.mpg.de/preprints/index.php`.
[43] E. Wachspress, *The ADI model problem*, 1995. Available from the author.
[44] A. Wathen and T. Rees, *Chebyshev semi-iteration in preconditioning for problems including the mass matrix.*, ETNA, Electron. Trans. Numer. Anal., 34 (2008), pp. 125–135,.
[45] J. Weickert, *Navier-Stokes equations as a differential-algebraic system*, Preprint SFB393/96-08, Preprint Series of the SFB 393, Department of Mathematics, Chemnitz University of Technology, August 1996.