E. Bänsch, P. Benner, J. Saak and H. K. Weichelt

Riccati-Based Boundary Feedback Stabilization of Incompressible Navier-Stokes Flow

September 2013

Preprint-Number SPP1253-154

Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flow

Eberhard Bänsch[†] Peter Benner[‡§] Jens Saak[‡§]
Heiko K. Weichelt[‡]

September 19, 2013

Abstract

In this article a boundary feedback stabilization approach for incompressible Navier-Stokes flow is studied. One of the main difficulties encountered is the fact that after space discretization by a mixed finite element method (because of the solenoidal condition) we end up with a differential-algebraic system (DAE) of index 2. The remedy here is to use a discrete realization of the *Leray projection* used by Raymond [J.-P. Raymond, *SIAM J. Control Optim.*, 45 (2006), pp. 790–828] used to analyze and stabilize the continuous problem.

Using the discrete projection, a linear quadratic regulator approach can be applied to stabilize the (discrete) linearized flow field with respect to small perturbations from a stationary trajectory. We show that the discrete Leray projector is nothing else but the numerical projection method proposed by Heinkenschloss et al. [M. Heinkenschloss, D. C. Sorensen, and K. Sun, *SIAM J. Sci. Comput.*, 30 (2008), pp. 1038–1063].

The nested iteration resulting from applying this approach within the Newton-ADI method to solve the LQR algebraic Riccati equation is the key to compute a feedback matrix that in turn can be applied within a closed-loop simulation.

Numerical examples for various parameters influencing the different levels of the nested iteration are given. Finally, the stabilizing property of the computed feedback matrix is demonstrated using the "von Kármán vortex shedding" within a finite element based flow solver.

[†]Research Group Applied Mathematics III (AMIII), Friedrich-Alexander-Universität Erlangen-Nürnberg, Cauerstr. 11, 91058 Erlangen, Germany
[‡]Research Group Mathematics in Industry and Technology (MiIT), Technische Universität Chemnitz, 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

Keywords. Flow control, Navier-Stokes equations, Riccati-based feedback

AMS subject classification. 93D15, 49M15, 76D55

1 Introduction

Incompressible flow problems are encountered in many technical fields such as chemical industry, biological research, or microfluids in micro- and nanosystems. These flow fields often deal with systems with moderate Reynolds numbers that do not require turbulence models. Many applications require a stable and controlled velocity field that is the basis for an ongoing reaction or production processes. The well known open-loop control approach [27] is not stable regarding small perturbations, which notorously occur in real life processes. Distributed control that basically would require the interaction of the controller on every point of the flow field (or in parts of it) is often impractical In contrast, the approach of boundary feedback stabilization avoids both problems. Although the feedback stabilization cannot manipulate flow fields in their general behavior, it can be used to stabilize existing open-loop controllers. This approach makes the open-loop controllers robust with respect to occurring perturbations. In [35, 36, 37] Raymond established the functional analytical setting and results for a *linear quadratic regulator* (LQR, see, e.g., [14, 31]) approach. With this approach Raymond was able to stabilize Stokes and Navier-Stokes equations regarding small perturbations from a stationary trajectory. Raymond's approach differs from most other ones (and consequently requires a much more involved analysis) in that the feedback contributes *via the boundary* (in contrast to distributed control), which is the natural procedure in most technical applications.

The main difficulty for the feedback stabilization of flow problems is the solenoidal condition, i.e., the local mass conservation. To this end, Raymond uses the *Leray projector* to project the whole system onto the space of divergence-free velocities in L^2. Using this projection, one ends up with a parabolic linear system, where well known techniques for Riccati-based feedback stabilization can be applied.

The Riccati-based boundary feedback stabilization of incompressible Stokes flow has been treated in [13]. There, the major interest was the efficient solution of the arising large-scale saddle point systems. In the present article we expand these techniques and ideas to the more general case of the Navier-Stokes equations and show numerical results. Most of the formulations in [13] can be extended straight forward to the Navier-Stokes case as well. We will upgrade some ideas and show more efficient realizations in the numerical treatment. Furthermore, we will describe the extension of the finite element flow solver NAVIER [5] regarding a closed-loop simulation.

In what follows, we consider the incompressible Navier-Stokes equations (NSE), reading in dimensionless form:

$$\frac{\partial}{\partial t}\vec{v}(t,\vec{x}) - \frac{1}{\mathrm{Re}}\Delta\vec{v}(t,\vec{x}) + (\vec{v}(t,\vec{x})\cdot\nabla)\vec{v}(t,\vec{x}) + \nabla\chi(t,\vec{x}) = \vec{f}(t,\vec{x})_{ext}, \tag{1a}$$

$$\operatorname{div}\vec{v}(t,\vec{x}) = 0, \tag{1b}$$

where $\vec{v}(t,\vec{x}) = \begin{bmatrix} v_{x_1}(t,\vec{x}) & v_{x_2}(t,\vec{x}) \end{bmatrix}^T \in \mathbb{R}^2$ denotes the velocity field and $\chi(t,\vec{x}) \in \mathbb{R}$ the pressure defined for $t \in [0,\infty)$ and $\vec{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \in \Omega \subset \mathbb{R}^2$. Here, Ω is a bounded domain with boundary $\Gamma = \partial\Omega$. The Reynolds number $\mathrm{Re} \in \mathbb{R}^+$ describes the ratio of inertial and viscous forces within the fluid. We will skip the arguments $t, \vec{x}$ hereafter for better readability.

We consider inflow-outflow problems, i.e., we assume that the boundary can be partitioned as

$$\Gamma = \Gamma_{in} \cup \Gamma_{out} \cup \Gamma_{wall} \cup \Gamma_{feed}.$$

On the respective parts of the boundary we thus impose the boundary conditions:

$$\vec{v} = \begin{cases} \vec{g}_{feed} & \text{on } \Gamma_{feed}, \\ \vec{g}_{in} & \text{on } \Gamma_{in}, \\ 0 & \text{on } \Gamma_{wall}. \end{cases} \tag{2a}$$

The control variable $\vec{g}_{feed}$ realizes the boundary control and will be explained in more detail in Section 2.4. As outflow condition the so called *do-nothing* condition [16, 26] is assumed:

$$-\frac{1}{\mathrm{Re}}\nabla\vec{v}\,\vec{n} + \chi\vec{n} = 0 \quad \text{on } \Gamma_{out} \tag{2b}$$

with $\vec{n}$ the outward normal to Γ_{out}. Finally, we define the initial condition

$$\vec{v}(0,\cdot) = 0 \quad \text{in } \Omega. \tag{2c}$$

The convection of the velocity field is a non-linear operator defined as

$$(\vec{v}\cdot\nabla)\vec{v} = \begin{bmatrix} v_{x_1}\frac{\partial v_{x_1}}{\partial x_1} + v_{x_2}\frac{\partial v_{x_1}}{\partial x_2} \\ v_{x_1}\frac{\partial v_{x_2}}{\partial x_1} + v_{x_2}\frac{\partial v_{x_2}}{\partial x_2} \end{bmatrix} \in \mathbb{R}^2,$$

see, e.g., [20].

The main idea of feedback stabilization is to stabilize stationary but possibly unstable solutions of the flow field. Such solutions may result from an open-loop control problem [27]. Usually these solutions are not robust with respect to small perturbations. The feedback controller measures the deviation that occurs due to perturbations and pushes the system back to the desired stationary trajectory. If these perturbations are small enough, one can show that the Riccati-based feedback stabilization is able to exponentially stabilize unstable solution trajectories, as it is described in, e.g., [6]. To this end, an LQR approach is used and we determine the unique stabilizing solution of the LQR problem while we solve the corresponding Riccati equation, see, e.g., [14, 31].

The rest of this paper is organized as follows. The complete work flow to compute the feedback is described in the next section. In Section 3 some special properties of the arising nested iteration are discussed and we illustrate the resulting algorithm by numerical experiments in Section 4. Section 5 concludes the paper and gives an outlook on open problems and future investigations.

2 Riccati-based feedback stabilization

We follow the analytic approaches for boundary feedback stabilization by Raymond [36] and seek for a numerical realization of those. Raymond applies the *Leray* projector that projects the velocity field onto the space of divergence-free functions in L^2 to the linearized flow problem; as a result Eq. (1b) is fulfilled automatically. This means that one can apply an LQR approach to the projected evolution equation. Raymond shows that one can find a stabilizing feedback that also stabilizes the non-linear system (1) via boundary control influence, see, e.g, [36]. In [13] the numerical realization of this approach is shown for the Stokes equation. In contrast to the Stokes case, the Navier-Stokes equations are non-linear. We explain our linearization approach in the next subsection.

2.1 Linearization of the NSE

Before we can apply the LQR approach to the (non-linear) Navier-Stokes equations (1) we consider $(\vec{w}(t,\vec{x}), \chi_s(t,\vec{x}))$ as velocity and pressure pair that fulfills the stationary Navier-Stokes equations

$$-\frac{1}{\mathrm{Re}}\Delta\vec{w} + (\vec{w}\cdot\nabla)\vec{w} + \nabla\chi_s = \vec{f}_{ext}, \tag{3a}$$

$$\operatorname{div}\vec{w} = 0, \tag{3b}$$

defined for $\vec{x} \in \Omega$ with the same boundary and initial conditions like Eqs. (2) and $\vec{g}_{feed} = 0$. The pair $(\vec{w}, \chi_s)$ depicts the desired stationary but (possibly) unstable solution of Eq. (1) that we want to stabilize against perturbations. The goal of the LQR approach is to force the pair $(\vec{v}, \chi)$ of Eq. (1) towards $(\vec{w}, \chi_s)$. That implies that the difference state $\vec{z} := \vec{v} - \vec{w}$ is asymptotically stabilized and $p = \chi - \chi_s$ up to a constant, respectively. This yields the linearized Navier-Stokes equations

$$\frac{\partial}{\partial t}\vec{z} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{w}\cdot\nabla)\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + \nabla p = 0, \tag{4a}$$

$$\operatorname{div}\vec{z} = 0, \tag{4b}$$

defined for $t \in [0,\infty)$ and $\vec{x} \in \Omega \subset \mathbb{R}^2$ with Dirichlet boundary conditions

$$\vec{z} = 0 \qquad \text{on } \Gamma_{in} \cup \Gamma_{wall}, \tag{4c}$$

$$\vec{z} = \vec{g}_{feed} \qquad \text{on } \Gamma_{feed}, \tag{4d}$$

do-nothing condition

$$-\frac{1}{\mathrm{Re}}\nabla\vec{z}\,\vec{n} + p\vec{n} = 0 \qquad \text{on } \Gamma_{out}, \tag{4e}$$

and the initial condition

$$\vec{z}(0,\cdot) = 0 \qquad \text{in } \Omega. \tag{4f}$$

The LQR approach we want to apply is based on finite dimensional matrix equations. To derive those equations, we semi-discretize the linearized Navier-Stokes equations (4) and introduce the finite dimensional LQR approach in detail in the next subsections.

2.2 Discretization

A common way of discretizing instationary control problems [3, 4, 17, 25] is the *method of lines* [40]. In our case we use a mixed finite element method [28] to discretize the linearized Navier-Stokes equations (4) in space and leave the time variable untouched. This yields the system of differential-algebraic equations:

$$M\frac{d}{dt}\mathbf{z}(t) = A\mathbf{z}(t) + G\mathbf{p}(t) + \mathbf{f}(t), \tag{5a}$$

$$0 = G^T\mathbf{z}(t), \tag{5b}$$

where $\mathbf{z}(t) \in \mathbb{R}^{n_v}$ denotes the nodal vector of the discretized velocity, $\mathbf{p}(t) \in \mathbb{R}^{n_p}$ the discretized pressure, and $\mathbf{f}(t) \in \mathbb{R}^{n_v}$ contains the control (see below), respectively. Furthermore, we denote by $M = M^T \succ 0 \in \mathbb{R}^{n_v \times n_v}$ the mass matrix, with $A \in \mathbb{R}^{n_v \times n_v}$ the system matrix [20, Section 7.3], and with $G \in \mathbb{R}^{n_v \times n_p}$ the discretized gradient, respectively. The system matrix can be split as follows:

$$A = -\frac{1}{\mathrm{Re}}S - K - R$$

with $-S\mathbf{z}$ the discrete Laplacian for $\Delta\vec{z}$, $-K\mathbf{z}$ the discrete convection for $(\vec{w} \cdot \nabla)\vec{z}$, and $-R\mathbf{z}$ a discrete reaction process for $(\vec{z} \cdot \nabla)\vec{w}$, respectively. Note that the system matrix and the gradient in the setting for the system theoretic framework are usually written on the right hand side of the equation, which leads to a switch of the sign in contrast to the convention in the finite element community.

Furthermore, we assume that the velocity can be observed only in parts of the domain and, therefore, add the output equation

$$\mathbf{y}(t) = C\mathbf{z}(t) \tag{ 5c}$$

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 measures the behavior of the velocity by using the information of the inner nodes. The particular properties of C are discussed in Section 4.2.

Using inf-sup stable finite elements (like $\mathcal{P}_2 - \mathcal{P}_1$ *Taylor-Hood* finite elements [28] in our case) the discretized gradient G is of rank n_p [20, Section 5.3], in particular $n_v > n_p$

Considering the structure of Eqs. (5a)-(5b) one gets a DAE of differential index 2 [43] defined by the matrix pencil

$$\left(\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}\right). \tag{6}$$

Here, the right-hand side coefficient matrix $\begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}$ is singular and the matrix pencil has $n_v - n_p$ finite eigenvalues $\lambda_i \in \mathbb{C} \setminus \{0\}$ and $2n_p$ infinite eigenvalues $\lambda_\infty = \infty$ [18, Theorem 2.1].

DAE systems are more involved than ordinary differential equations, since the solution set of the DAE lies on a (hidden) manifold. In the case of incompressible flows this manifold is the space of (discretely) divergence-free functions. We use the index reduction idea described in [25, Section 3] to avoid this problem and demonstrate the feasibility of this idea for descriptor systems like Eq. (5) in the next subsection.

Note that the whole system Eq. (5) is a descriptor system with multiple inputs (n_r) (see Section 2.4 below) and multiple outputs (n_a) (short MIMO system). It is well known that such systems are getting more complicated if the number of inputs n_r or outputs n_a gets larger.

After defining all components in the descriptor system Eq. (5) we show an index reduction technique that is a numerical realization of the *Leray* projection in the next subsection.

2.3 Projection method

In [6] Bänsch/Benner proposed to use the index reduction method defined in [25] as numerical realization of the *Leray* projection in the case of linearized Navier-Stokes equations. In [13] it is shown that the projector

$$\mathit{\Pi}^T := I_{n_v} - M^{-1}G(G^T M^{-1} G)^{-1} G^T \tag{7}$$

used in [25] turns out to be nothing else but the discrete Leray operator (as needed for our LQR approach) in our setting. To this end, it is necessary to view $\mathit{\Pi}^T$ as orthogonal projector in the appropriate setting, i.e., the discrete L^2 inner product, in contrast to the interpretation of $\mathit{\Pi}^T$ as oblique projection in the Euclidean inner product as in [25]. In this section we recall the proof of this fact. Note that we can use the one sided projection using Π^T from (7) since our FEM provides the feature that the discretized gradient on the pressure space and the discrete constraint equation are transposes of each other. For a more general discretization that does not provide this feature, however, the basic projection idea stays valid, but requires using the more general two sided projection approach discussed in the interpolatory model reduction framework in [23, Section 6].

Following the description in [19, 21], $L^2(\Omega)^2$ can be decomposed into

$$L^2(\Omega)^2 = \mathbf{H}(\mathrm{div}, 0) \perp \mathbf{H}(\mathrm{div}, 0)^\perp,$$

where the spaces $\mathbf{H}(\mathrm{div}, 0), \mathbf{H}(\mathrm{div}, 0)^\perp$ are characterized by

$$\mathbf{H}(\mathrm{div}, 0) := \{\vec{v} \in L^2(\Omega)^2 \mid \mathrm{div}\, \vec{v} = 0, \vec{v} \cdot \vec{n}_{|\Gamma} = 0\}, \quad \mathbf{H}(\mathrm{div}, 0)^\perp = \{\nabla p \mid p \in H^1(\Omega)\}.$$

This splitting is equivalent to $\vec{v} = \vec{v}_{\text{div}} + \nabla p$, where $\vec{v}_{\text{div}}$ and p fulfill

$$\begin{aligned} \vec{v}_{\text{div}} + \nabla p &= \vec{v} \quad \text{in } \Omega, \\ \operatorname{div} \vec{v}_{\text{div}} &= 0 \quad \text{in } \Omega, \\ \vec{v}_{\text{div}} \cdot \vec{n} &= 0 \quad \text{on } \Gamma. \end{aligned}$$

The operator $P : L^2(\Omega)^2 \to \mathbf{H}(\text{div}, 0)$ with $P\vec{v} := \vec{v}_{\text{div}}$ is called the *Leray* projection.

The discrete version of the above splitting for the finite element discretization can be written in terms of nodal value vectors as:

$$M\mathbf{v}_{\text{div}} + G\mathbf{p} = M\mathbf{v}, \tag{8a}$$

$$G^T\mathbf{v}_{\text{div}} = 0 \tag{8b}$$

with the discrete velocity field $\mathbf{v} \in \mathbb{R}^{n_v}$, the discretely divergence-free velocity field $\mathbf{v}_{\text{div}} \in \mathbb{R}^{n_v}$, and the discrete pressure $\mathbf{p} \in \mathbb{R}^{n_p}$. M, G are defined as above. Multiplying Eq. (8a) from the left by $G^T M^{-1}$ and using Eq. (8b) reveals

$$\mathbf{p} = (G^T M^{-1} G)^{-1} G^T \mathbf{v}.$$

Using this expression for $\mathbf{p}$ in Eq. (8a) yields

$$\mathbf{v}_{\text{div}} = (I_{n_v} - M^{-1}G(G^T M^{-1} G)^{-1} G^T)\mathbf{v} \quad \Rightarrow \quad \mathbf{v}_{\text{div}} = \mathit{\Pi}^T \mathbf{v}.$$

This shows, together with the properties described in [13], that multiplication of a discretized velocity field from the left with $\mathit{\Pi}^T$ is equivalent to applying the discretized *Leray* projection to a discretized vector function. Of course, building $\mathit{\Pi}^T$ explicitly is prohibitive, since the matrix would become dense. A numerically suitable way to apply $\mathit{\Pi}^T$ is to write the system (8) as saddle point problem

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_{\text{div}} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} M\mathbf{v} \\ 0 \end{bmatrix}$$

that has to be solved for a given velocity field $\mathbf{v}$ to get the (discretely) divergence-free velocity field $\mathbf{v}_{\text{div}}$. Nevertheless, we want to avoid the projection in general in the process of computing the optimal control $\mathbf{u}(t)$ as it is shown next.

Following the projection steps in [13, Section 2.1] and using the substitutions with the identical notation the descriptor system (5) can be written in terms of $\tilde{\mathbf{z}} = \mathit{\Pi}^T \mathbf{z} \in \mathbb{R}^{n_v - n_p}$ as the generalized state space system

$$\mathcal{M} \frac{d}{dt} \tilde{\mathbf{z}}(t) = \mathcal{A}\tilde{\mathbf{z}}(t) + \mathcal{B}\mathbf{u}(t), \tag{9a}$$

$$\mathbf{y}(t) = \mathcal{C}\tilde{\mathbf{z}}(t) \tag{9b}$$

with $\mathcal{M} = \mathcal{M}^T \succ 0 \in \mathbb{R}^{(n_v - n_p) \times (n_v - n_p)}$ that fits the scheme of [25, equations 4.1].

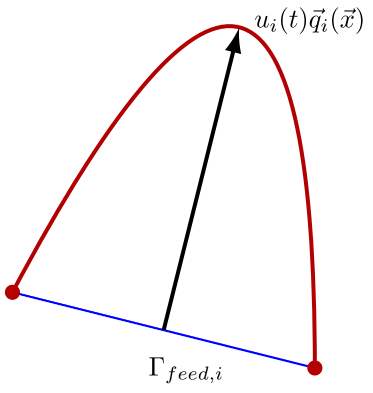


Figure 1: The parabolic inflow profile at the control boundary part $\Gamma_{feed,i}$.

2.4 Boundary control

In this section we describe how to incorporate *normal* boundary control for system (9). Note that $\tilde{\mathbf{z}}$ has zero normal components in a discrete sense.

We mimic the approach in [38], where an operator was constructed that distributed the boundary control into the interior of Ω. This construction was necesarry, since Raymond worked with the space $\mathbf{H}(\mathrm{div}, 0)$. Note that functions in $\mathbf{H}(\mathrm{div}, 0)$ have vanishing normal components on the boundary.

We consider n_r different parts of the control boundary Γ_{feed} , i.e.,

$$\vec{g}_{feed}(t, \vec{x}) = \sum_{i=1}^{n_r} u_i(t)\vec{q}_i(\vec{x}),$$

where $\vec{q}_i$ has support on $\Gamma_{feed,i}$ and $\bigcup_{i=1}^{n_r} \Gamma_{feed,i} = \Gamma_{feed}$. In the example in Section 4 we consider a parabolic in-/outflow $\vec{q}_i$ and $u_i(t)$ controls the intensity, see Fig. 1.

The control operator $\mathcal{B}$ in Eq. (9) has n_r columns, each of which is computed in the following way.

For $i = 1, \dots n_r$ do:

1. *Solve the linearized Navier-Stokes equations with homogeneous boundary condition except for $\vec{g}_{feed}$ on $\Gamma_{feed,i}$ where the Dirichlet condition $1 \cdot \vec{q}_i(\vec{x})$ is imposed. Denote the resulting velocity field by $\mathbf{v}_i$.*

2. *Project $\tilde{\mathbf{v}}_i := \Pi^T \mathbf{v}_i$.*

3. *Multiply $\tilde{\mathbf{v}}_i = \Pi^T \mathbf{v}_i$ by the discrete linearized Navier-Stokes operator A to get $\hat{\mathbf{v}}_i$.*

4. *Set the i-th column $\mathbf{b}_i$ of $\mathcal{B}$ to $\mathbf{b}_i := \Pi^T \hat{\mathbf{v}}_i$.*

In the next subsection, we formulate the LQR approach for the projected system (9) and show how to compute the optimal control $\mathbf{u}(t)$. In contrast to the Stokes formulation in [13] the system matrix $\mathcal{A}$ is non-symmetric. This yields minor changes in the procedure, but does not change the work flow.

2.5 The linear quadratic regulator approach

The LQR approach is supposed to minimize a given quadratic cost functional subject to a given linear system as it is described, e.g., in Locatelli's introduction about LQR in [33]. Because we consider the generalized state space system (9) with a mass matrix $\mathcal{M} \neq I$ on the left hand side, the procedure changes a little bit as it is shown in [39, Chapter 5.2].

The cost functional for our problem setting is defined as

$$\mathcal{J}(\tilde{\mathbf{z}}(t), \mathbf{u}(t)) := \frac{1}{2} \int_0^\infty \lambda \left(\tilde{\mathbf{z}}(t)^T \mathcal{C}^T \mathcal{C} \tilde{\mathbf{z}}(t)\right) + \mathbf{u}(t)^T \mathbf{u}(t) \, \mathrm{d}t \tag{10}$$

that measures the output $\mathbf{y} = \mathcal{C}\tilde{\mathbf{z}}$ and the cost of the optimal control $\mathbf{u}$ in the square of the Euclidean norm $||\mathbf{x}||_2 := \sqrt{\mathbf{x}^T\mathbf{x}}$. To minimize this integral over the infinite time horizon, the output, as well as the control, have to converge asymptotically to zero for $t \to \infty$. The output $\mathbf{y}$ measures on the discrete level the velocity field $\vec{z}$. In Subsection 2.1, $\vec{z}$ is defined as perturbation of our flow field $\vec{v}$ from the desired stationary flow field $\vec{w}$. A zero output for $t \to \infty$ implies that $\vec{v}$ approximates $\vec{w}$ for $t \to \infty$; our flow field $\vec{v}$ achieves the properties of the desired stationary flow field.

The factor λ in the first term of Eq. (10) is used as regularization. By varying λ one may also achieve qualitatively different results. For ease of notation, we set $\lambda = 1$ in the theoretical derivations. In the actual implementation reported in Section 3, we of course allow variations of λ, and we will also investigate the influence of these variations in the numerical experiments in Section 4.

Following the LQR approach, we know that the optimal control

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

minimizes the cost functional (10) subject to (9) with $\tilde{\mathbf{z}}_*(t)$ as the optimal trajectory and $X = X^T \succeq 0 \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$ as the unique stabilizing solution of the *generalized algebraic Riccati equation (GARE)*

$$0 = \mathcal{C}^T\mathcal{C} + \mathcal{A}^T X \mathcal{M} + \mathcal{M} X \mathcal{A} - \mathcal{M} X \mathcal{B}\mathcal{B}^T X \mathcal{M} =: \mathfrak{R}(X). \tag{11}$$

The key ingredient to compute the feedback $\mathcal{K} \in \mathbb{R}^{n_r\times(n_v-n_p)}$ that asymptotically stabilizes the generalized state space system (9) is to solve the GARE (11). A solution approach is shown in the next subsection.

2.6 Solving the generalized algebraic Riccati equation

The generalized algebraic Riccati equation (11) is a quadratic matrix equation for the unknown matrix $X \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$ that can be solved by a Newton-type iteration; this iteration is given at step m by

$$X^{(m+1)} = X^{(m)} + N^{(m)},$$

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

```
Input: $\mathcal{A}^{(m)}, \mathcal{M}, \mathcal{W}^{(m)}$, and shift parameters $q_i \in \mathbb{C}^- : i = 1, \ldots, i_{\max}$
Output: $Z = Z_{i_{\max}} \in \mathbb{C}^{n \times t_{i_{\max}}}$, such that $ZZ^H \approx X^{(m+1)}$
1: $V_1 = \sqrt{-2\,\mathrm{Re}\,(q_1)}\left((\mathcal{A}^{(m)})^T + q_1\mathcal{M}\right)^{-1}(\mathcal{W}^{(m)})^T$
2: $Z_1 = V_1$
3: for $i = 2, 3, \ldots, i_{\max}$ do
4:     $V_i = \sqrt{\mathrm{Re}\,(q_i)/\,\mathrm{Re}\,(q_{i-1})}\left(V_{i-1} - (q_i + \overline{q_{i-1}})\left((\mathcal{A}^{(m)})^T + q_i\mathcal{M}\right)^{-1}(\mathcal{M}V_{i-1})\right)$
5:     $Z_i = [Z_{i-1}\, V_i]$
6: end for
```

with the update $N^{(m)}$ computed via

$$\mathfrak{R}'|_{X^{(m)}}(N^{(m)}) = -\mathfrak{R}(X^{(m)}).$$

$\mathfrak{R}'|_{X^{(m)}}$ is the Fréchet derivative of the Riccati operator (11) at $X^{(m)}$ defined as

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

see, e.g., [2, 30]. Following the procedure in [13, Section 2.3] the main task to compute the iterate $X^{(m+1)}$ in the Newton step $m+1$ is to solve a generalized Lyapunov equation

$$(\mathcal{A}^{(m)})^T X^{(m+1)}\mathcal{M} + \mathcal{M}X^{(m+1)}\mathcal{A}^{(m)} = -(\mathcal{W}^{(m)})^T\mathcal{W}^{(m)}. \tag{12}$$

The closed-loop system matrix $\mathcal{A}^{(m)} = \mathcal{A} - \mathcal{B}\mathcal{K}^{(m)}$ includes the previous feedback $\mathcal{K}^{(m)} = \mathcal{B}^T X^{(m)}\mathcal{M}$ from step m, and the low-rank right hand side factor $\mathcal{W}^{(m)} = \begin{bmatrix} \mathcal{C} \\ \mathcal{K}^{(m)} \end{bmatrix}$, see, e.g., [29]. The linear matrix equation (12) has to be solved in every Newton step. We apply the *low-rank ADI iteration* [11, 32], more precisely, we use the extended formulation for the generalized case as described in [8] to compute the iterate $X^{(m+1)}$ in Eq. (12). Using our notation we end up with Algorithm 1.

The spectrum of the projected state space system (9) is equivalent to the finite spectrum of the DAE system (6). The major part of this finite spectrum lies in the open left complex half plane $\mathbb{C}^-$, but with increasing Reynolds number a few finite eigenvalues with positive real part occur [1]. Because of this fact the system is not asymptotically stable [33], which is, however, necessary to ensure the uniqueness of the solution of the Lyapunov equation [8]. Therefore, we need to compute an initial stabilizing feedback K_0 such that the finite spectrum of the closed-loop system

$$\left(\begin{bmatrix} A - BK_0 & G \\ G^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}\right) \tag{13}$$

is a subset of $\mathbb{C}^-$, as it is shown in Subsection 2.7. Details about the ADI shift parameters $q_i \in \mathbb{C}^-$ are described in Section 3.2.

The combination of the Newton iteration, as an outer iteration, with the G-LRCF-ADI (Algorithm 1), as an inner iteration, is known as the *generalized low-rank Cholesky factor Newton method (G-LRCF-NM)* [11, 12]. The main computational work is done in lines 1 and 4 of Algorithm 1 by solving linear systems of equations with large-scale and projected matrices of the form

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

Solving these linear systems for different right hand sides $\mathcal{Y}$ would involve the dense projection matrix Π^T that we want to avoid using the main results of [25, Section 5]. There it is shown that Λ can be computed as the solution of

$$\Pi\left((A - BK^{(m)})^T + q_iM\right)\Pi^T\Lambda = \Pi Y,$$

which is equivalent to the solution of the saddle point system

$$\underbrace{\begin{bmatrix} (A - BK^{(m)})^T + q_iM & G \\ G^T & 0 \end{bmatrix}}_{=:\tilde{\mathbf{A}}} \begin{bmatrix} \Lambda \\ * \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix} \tag{14}$$

with $K^{(m)} = B^TXM$, see [25, Lemma 5.2]. The saddle point system (14) consists of the original matrices and does not include any projection. We summarize the entire work flow to compute the feedback matrix K via the generalized low-rank Cholesky factor Newton method avoiding the use of any projection in Algorithm 2.

After a short note about the initial feedback in the next subsection, we will discuss important properties of the nested iteration in Algorithm 2 in Section 3.

2.7 Initial feedback

As mentioned above, the matrix pencil (6) is not stable in general. But in order to use the ADI method for solving the Lyapunov equation (12) we need a stable pencil. Following the ideas in [24], we show how to construct an initial feedback K_0. Note that this concept is also used in [1] as a numerical method to stabilize the Navier-Stokes equations.

The majority of the $n_v - n_p$ finite eigenvalues of (6) are stable and only n_{us} eigenvalues are unstable with $n_{us} \ll n_v$. First, we need all unstable finite eigenvalues $\lambda_{us}^{(i)} \in \mathbb{C}^+$ of the pencil (6) together with their corresponding left and right eigenvectors $\omega^i, \eta^i \in \mathbb{C}^{n_v+n_p}$ for $i = 1, \ldots, n_{us}$. Note that solving these large scale generalized eigenvalue problems in an efficient way is not an element of this paper. We use the implicitly restarted shift-and-invert Arnoldi method as implemented in the `eigs` function of MATLAB® with the matrix shifting strategy described in [18]; using this strategy all infinite eigenvalues are transformed to a fixed finite eigenvalue. This technique to calculate all finite and unstable eigenvalues, is not the most robust method for unknown problem settings but it is sufficient in our case.

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

```
Input: $M, A, G, B, C$, initial feedback $K_0$, and ADI shift parameters
       $q_i \in \mathbb{C}^- : i = 1, \dots, n_{\text{ADI}}$, $tol_{ADI}$, $tol_{Newton}$, $\lambda$
Output: feedback operator $K$
 1: for $m = 1, 2, \dots, n_{\text{Newton}}$ do
 2:   $W^{(m)} = \begin{bmatrix} \sqrt{\lambda}\, C \\ (K^{(m-1)}) \end{bmatrix}$
 3:   Get $V_1$ by solving
          $$\begin{bmatrix} A^T - (K^{(m-1)})^T B^T + q_1 M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ * \end{bmatrix} = \begin{bmatrix} \sqrt{-2\,\mathrm{Re}\,(q_1)}\,(W^{(m)})^T \\ 0 \end{bmatrix}$$
 4:   $K_1^{(m)} = B^T V_1 \overline{V}_1^T M$
 5:   for $i = 2, 3, \dots, n_{\text{ADI}}$ do
 6:     Get $\tilde{V}$ by solving
          $$\begin{bmatrix} A^T - (K^{(m-1)})^T B^T + q_i M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{V} \\ * \end{bmatrix} = \begin{bmatrix} M V_{i-1} \\ 0 \end{bmatrix}$$
 7:     $V_i = \sqrt{\mathrm{Re}\,(q_i) / \mathrm{Re}\,(q_{i-1})} \left( V_{i-1} - (q_i + \overline{q_{i-1}}) \tilde{V} \right)$
 8:     $K_i^{(m)} = K_{i-1}^{(m)} + B^T V_i \overline{V}_i^T M$
 9:     if $\left( \frac{||K_i^{(m)} - K_{i-1}^{(m)}||_F}{||K_i^{(m)}||_F} < tol_{\text{ADI}} \right)$ then
10:       break
11:     end if
12:   end for
13:   $K^{(m)} = K_{n_{\text{ADI}}}^{(m)}$
14:   if $\left( \frac{||K^{(m)} - K^{(m-1)}||_F}{||K^{(m)}||_F} < tol_{\text{Newton}} \right)$ then
15:     break
16:   end if
17: end for
18: $K = K^{(n_{\text{Newton}})}$
```

We define the matrices

$$H := \left[\eta^{(1)}, \ldots, \eta^{(n_{us})}\right] \in \mathbb{C}^{(n_v+n_p)\times n_{us}} \text{ and}$$
$$W := \left[\omega^{(1)}, \ldots, \omega^{(n_{us})}\right] \in \mathbb{C}^{(n_v+n_p)\times n_{us}}$$

for the left and right eigenvectors, respectively. Using these matrices we project our system on the subspace that is spanned by the eigenvectors of the unstable eigenvalues λ_{us}:

$$\tilde{E} := W^* \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} H, \quad \tilde{A} := W^* \begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix} H, \quad \tilde{B} := W^* \begin{bmatrix} B \\ 0 \end{bmatrix}.$$

After solving the n_{us}-dimensional generalized Bernoulli equation

$$\tilde{A}^* X_0 \tilde{E} + \tilde{E}^* X_0 \tilde{A} - \tilde{E}^* X_0 \tilde{B} \tilde{B}^* X_0 \tilde{E} = 0, \tag{15}$$

we can define our initial feedback $K_0 \in \mathbb{R}^{n_r \times n_v}$ as

$$K_0 := B^T W X_0 W^T M.$$

We solve the (generalized) Bernoulli equation (15) with an algorithm based on [7]. Using the initial feedback K_0 the resulting closed-loop pencil

$$\left(\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix} - \begin{bmatrix} B \\ 0 \end{bmatrix} \cdot \begin{bmatrix} K_0 & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right)$$

is stable and all initially unstable eigenvalues $\lambda_{us}^{(i)}$ appear as stabilized eigenvalues

$$\lambda_{stab}^{(i)} := -\operatorname{Re}(\lambda_{us}^{(i)}) + i \operatorname{Im}(\lambda_{us}^{(i)}) \in \mathbb{C}^-, \forall i = 1, \ldots, n_{us}$$

that are mirrored at $i\mathbb{R}$ [7].

Note that some algorithms compute the eigenvectors in such way that $\tilde{E} = I$, which means V and W are orthogonal in $(.,.)_{\tilde{E}}$. For this constellation one has to solve an ordinary Bernoulli equation.

Using the initial feedback K_0 we can use the nested Algorithm 2 to compute the feedback K that defines $\mathbf{u}(t)$ minimizing the cost functional (10). Some details of the nested iteration are shown in the next section.

3 Nested iteration

The iterative process to compute the feedback matrix K is described in Algorithm 2. It is a nested iteration embedded in a Newton type method, as outermost iteration, an ADI iteration, as central iteration, and a method to solve the large-scale and sparse saddle point systems (14), as innermost iteration.

For moderate problem sizes of $n = n_v + n_p = O(10^6)$ a direct solver is the best choice, but if the dimension of the underlying finite element discretization gets larger or one deals with three-dimensional problems, the direct solvers generate a considerable fill-in such that the use of iterative methods is imperative, see, e.g., [13]. For the problem size of the numerical example in Section 4 a direct solver is suitable. Extending the iterative solution techniques of [13] to the more general Navier-Stokes flows is part of our current research.

We show some properties of the saddle point matrix $\tilde{\mathbf{A}}$ in (14) and discuss different parameters of the nested iteration in the next subsection.

3.1 Properties of the saddle point system

Although the matrices A, M, G in $\tilde{\mathbf{A}}$ are sparse, the low-rank product $BK^{(m)}$ is a dense block such that nearly the whole matrix $\tilde{\mathbf{A}}$ becomes dense. To avoid this problem we write (14) in the form of a low-rank update

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

that is in a compact notation

$$(\mathbf{A} - \mathbf{K}^T\mathbf{B}^T)\mathbf{\Lambda} = \mathbf{Y}. \tag{16}$$

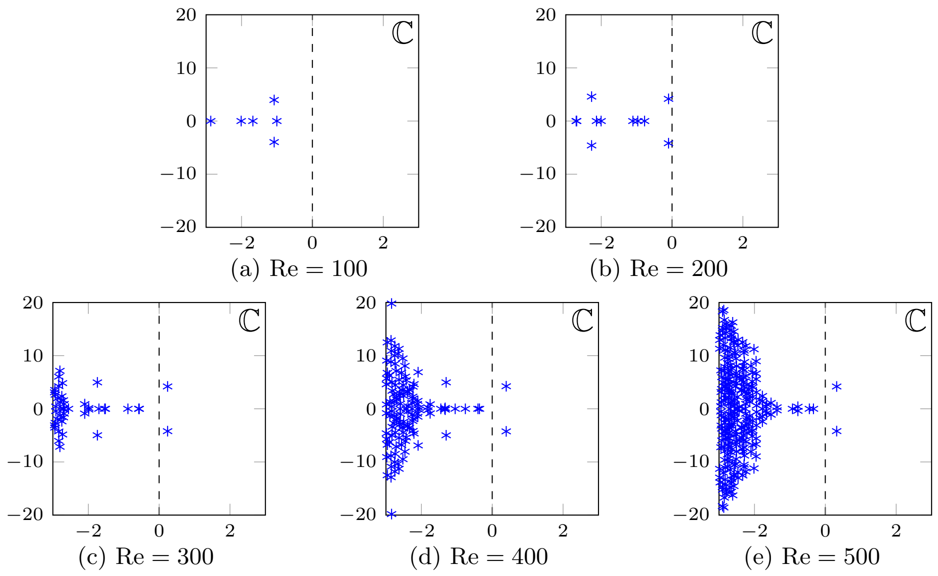
To evaluate Eq. (16) we recall [13, Section 3.1], where the *Sherman-Morrison-Woodbury* formula (see, e.g., [22])

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

is used. In addition to solving with $\mathbf{A}$ we need to solve with a small dense matrix $(I_{n_r} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{K}^T)$ of dimension $n_r \ll n_v$ to perform the solve with $\tilde{\mathbf{A}}$. The extra solve for $\mathbf{A}$ with the right hand side $\mathbf{K}^T$ can be easily done by adding $\mathbf{K}^T$ as n_r additional columns to the matrix $\mathbf{Y}$, such that the new right hand side becomes $\begin{bmatrix} Y & K^T \\ 0 & 0 \end{bmatrix} =: \begin{bmatrix} \tilde{Y} \\ 0 \end{bmatrix}$. In our configuration with two boundary control parameters, $n_r = 2$. The resulting saddle point system that has to be solved in every ADI step during each Newton step is of the form:

$$\begin{bmatrix} A^T + q_i M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \Lambda \\ * \end{bmatrix} = \begin{bmatrix} \tilde{Y} \\ 0 \end{bmatrix}, \tag{17}$$

where $M = M^T \succ 0, q_i \in \mathbb{C}^-$, and a non-symmetric matrix A with eigenvalues in $\mathbb{C}$. Note that for $|q_i|$ large enough all eigenvalues of $A^T + q_i M$ are located in $\mathbb{C}^-$, which is relevant for some preconditioning techniques of the iterative solver. We will skip the details concerning these influences in this paper and refer the reader to [13]. In contrast, the whole saddle point matrix $\mathbf{A}$ will in general have eigenvalues in all of $\mathbb{C}\, \forall q_i \in \mathbb{C}^-$.



(a) Re = 100 (b) Re = 200 (c) Re = 300 (d) Re = 400 (e) Re = 500

Figure 2: Eigenvalues of matrix pencil (6) that are close to $i\mathbb{R}$ for different Reynolds numbers.

3.2 ADI shifts

The ADI shifts q_i influence the eigenvalues of the matrix $\mathbf{A}$ and they are crucial for the convergence of the ADI method. There are many different ways and methods to choose q_i. We focused on the use of the heuristic *Penzl* shifts [34]. A basic ADI result tells that (optimal) ADI shifts q_i have to be included in the convex hull of the finite spectrum of the closed-loop DAE system (13) as it is described in, e.g., [41]. Using the efficient implementation to compute $q_i \in \mathbb{C}^-$ as it is described in [39] we run into problems, because the infinite eigenvalues of (13) destroy the convergence for eigenvalues with large, however, finite magnitude. We can avoid this difficulty with the matrix shifting techniques of [18] as used in Subsection 2.7. Using the shifts q_i in a cyclic way provides acceptable convergence results in our cases. Further investigations to improve the convergence by an optimized selection of the ADI shifts q_i is part of our current research.

3.3 Reynolds number

The second parameter that influences the matrices and, as a result, the convergence behavior of the nested iteration is the Reynolds number. For high Reynolds numbers the system gets more convection dominated and stable finite eigenvalues are getting closer to the imaginary axis. At a certain point, a few eigenvalues cross the imaginary axis and we end up with unstable eigenvalues. As described in Subsection 2.7, we need to compute an initial feedback to ensure the convergence of the ADI Algorithm 1 in this case.

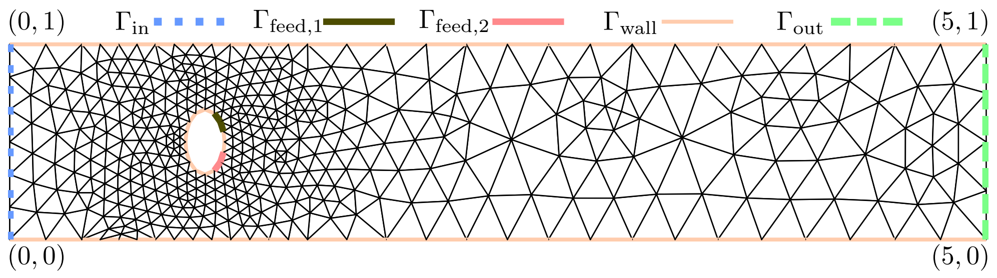


Figure 3: Discretization of *von Kármán vortex street* with coordinates and boundary conditions.

The eigenvalue behavior is depicted in Fig. 2 for different Reynolds numbers, where the eigenvalues of the matrix pencil (6) are plotted for the example of Section 4.

4 Numerical examples

In order to test our numerical method for boundary feedback stabilization of Navier-Stokes flows we use the *von Kármán vortex shedding* as depicted in Fig. 3. We consider the flow through a channel $\Omega \subset \mathbb{R}^2$ with diameter $d_{in} = 1$ that flows round an obstacle of elliptic shape centered at the coordinates $(1, 0.5)$, $\frac{1}{5}d_{in}$ wide, and $\frac{1}{3}d_{in}$ high. The boundary conditions of the Navier-Stokes equations (1) are taken as described in Section 1. We define the parabolic inflow condition from (2a) as

$$\vec{v}(t,\vec{x}) = \vec{g}_{in}(\vec{x}) := \begin{bmatrix} 4\cdot(1-x_2)\cdot x_2 \\ 0 \end{bmatrix}, \text{ on } \Gamma_{in}$$

with a maximum of 1.0 at $x_2 = 0.5$. This condition is consistent with the *no-slip* condition (2a) at $(0,0)$ and $(0,1)$ that are parts of Γ_{wall} as well.

We use the finite element flow solver NAVIER [5] for our numerical tests to assemble the matrices. NAVIER uses a standard mixed finite element discretization of Ω (i.e., $\mathcal{P}_2$-$\mathcal{P}_1$ *Taylor-Hood* elements [28]) as it is shown in Fig. 4. NAVIER is implemented in FORTRAN90 and the matrices are saved using the so called *Matrix Market* format [15]. The dimensions of the computational example are $n_v = 3452$ and $n_p = 453$.

The computations for the resulting matrix equations are performed with MATLAB on a a 64-bit server with Intel® Xeon® X5650 @2.67GHz, with 2 CPUs, 12 Cores (6 Cores per CPU) and 48 GB main memory available.

In Section 3 we discussed various parameters and properties of the nested iteration. Using the above configuration we show different results regarding these parameters and properties. At the end of this section we show the realization of a closed-loop simulation within NAVIER.

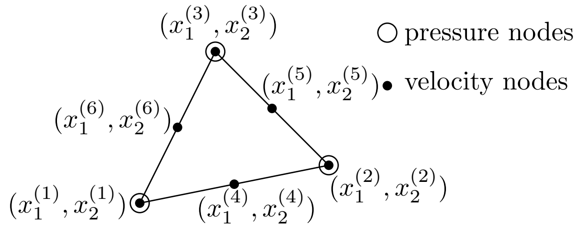


Figure 4: $\mathcal{P}_2$-$\mathcal{P}_1$ *Taylor-Hood* element.

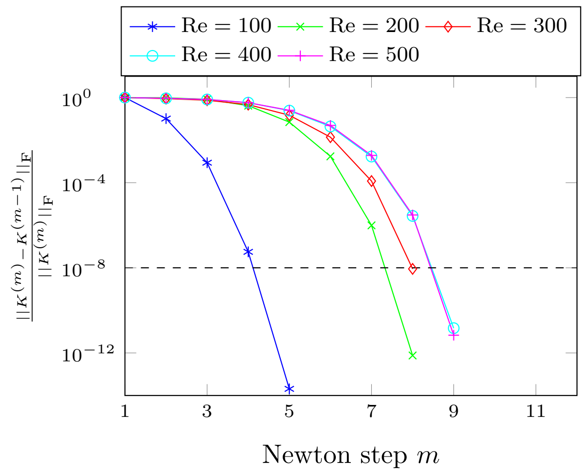
4.1 Parameters in nested iteration

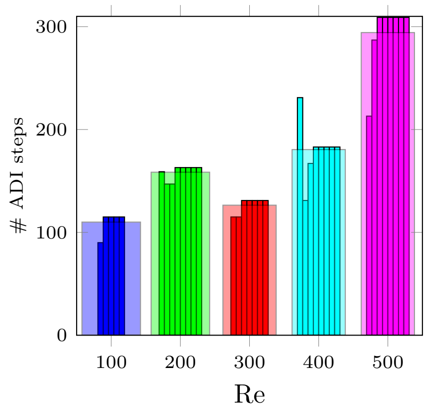
We show the influence of the different parameters by comparing the convergence behavior for different problem settings. Unless otherwise stated, we change only one parameter for every configuration. The two parameters that mostly influence the convergence of the outer Newton-ADI are the Reynolds number $\mathrm{Re} \in \mathbb{R}^+$ and the regularization parameter $\lambda \in \mathbb{R}^+$. We do not further discuss the convergence influence of the ADI shifts because we use the *Penzl* shifts in all configurations. However, the ADI shifts play a certain role within the iterative solution process for the saddle point systems (17) as it is shown for Stokes flow in [13].

4.1.1 Reynolds number

We use five different Reynolds numbers for our tests and fix $\lambda = 1$. Note that for $\mathrm{Re} < 100$ the influence of the convection term $(\vec{v} \cdot \nabla)\vec{v}$ in Eq. (1) is negligible and the flow behaves like a Stokes flow [13]. If we use $\mathrm{Re} > 500$, numerical difficulties with the FEM arise, since the mesh is not sufficiently fine. In this case we would need to use stabilization techniques for the FEM, but this does not fit in the theoretical approach, because some additional components would occur in the DAE pencil (6).

Fig. 5 shows the influence of the Reynolds number on the Newton-ADI iteration. From Fig. 5a one can see that the number of Newton steps increases for increasing Reynolds numbers. However, there is an upper bound at $m = 9$. Although the number of Newton steps remains constant for $\mathrm{Re} \geq 400$, the number of ADI steps per Newton step still increases, as depicted in Fig. 5b. Fig. 5b shows details about the number of ADI steps per Newton steps depicted as small bars clustered for the different Reynolds numbers. A wider and slightly transparent block denotes the average number of ADI steps for each Reynolds number. Thus, the computing time to evaluate the Newton-ADI iteration increases for increasing Reynolds numbers. Note that we additionally need to compute the initial feedback K_0 for $\mathrm{Re} \geq 300$ that is represented by one additional Newton step. Hence, the system with $\mathrm{Re} = 300$ needs less ADI steps in average but the Newton convergence is slower than for $\mathrm{Re} = 200$. To summarize, the nested iteration is able to compute the feedback K for all considered Reynolds numbers with a reasonable computational effort. This behavior does not qualitatively change if one changes the regularization parameter λ. The quantitative influence of λ is shown below.

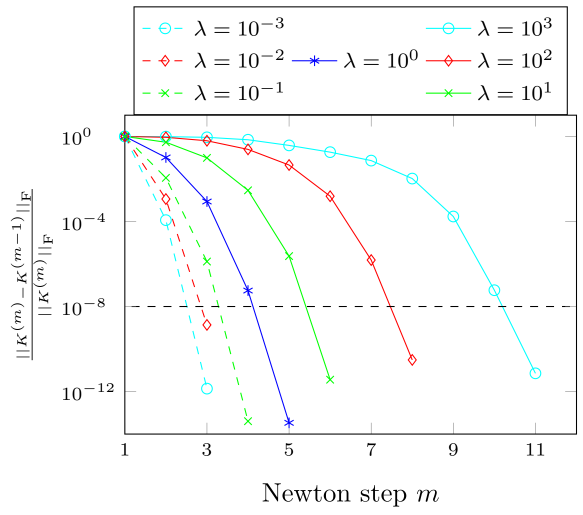


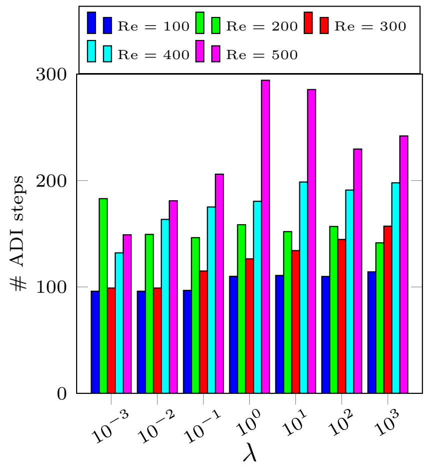


(a) Convergence behavior of Newton iteration for different Reynolds numbers.

(b) Number of ADI steps per Newton step (small bars) and average over all Newton steps (wide bars).

Figure 5: Influence of Reynolds number for Newton and ADI convergence ($\lambda = 10^0$, $tol_{Newton} = 10^{-8}$, $tol_{ADI} = 10^{-7}$).





(a) Convergence behavior of Newton iteration for different λ and Re = 100.

(b) Average number of ADI steps during Newton iteration.

Figure 6: Influence of regularization parameter λ for Newton and ADI convergence ($tol_{Newton} = 10^{-8}$, $tol_{ADI} = 10^{-7}$).

4.1.2 Regularization parameter λ

The regularization parameter λ is introduced in the cost functional (10). For $\lambda > 1$ the output $\mathbf{y} = \mathcal{C}\tilde{\mathbf{z}}$ is penalized; the feedback K should force our system with more effort to the desired state. It is natural that we need higher control cost to achieve this.

In contrast, for $\lambda < 1$ the influence of the output is reduced compared to the control cost. The control cost $\mathbf{u}^T\mathbf{u}$ is penalized in an indirect way in that the controller K tries to force our system to the desired state but with a reduced amount of control effort.

The convergence behavior of the Newton-ADI is illustrated for $10^{-3} \leq \lambda \leq 10^3$ in Fig. 6. Fig. 6a shows the convergence of the Newton iteration for Re = 100. The number of Newton steps increases monotonically with respect to λ. This seems to be natural, because the feedback K has to work more effective if the output is more penalized; the feedback K has to provide more "information" to achieve its aim in a shorter amount of time.

For small λ there exists a lower bound at $m = 3$. The Newton convergence for different λ stays qualitatively the same for different Reynolds numbers. The number of steps varies quantitatively as shown in Fig. 5a.

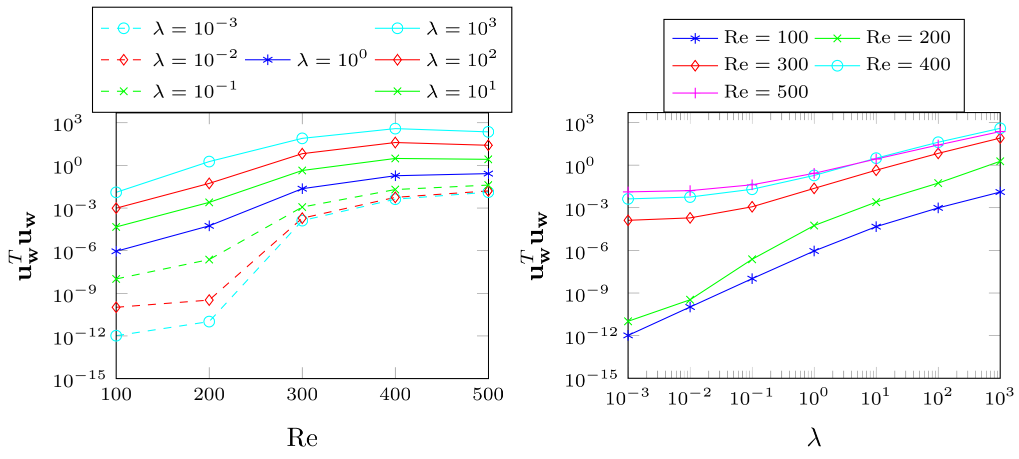
Fig. 6b illustrates the average number of ADI steps within the Newton iteration for varying Reynolds number and regularization parameter λ. The number of ADI steps is nearly constant for Re = 100 and Re = 200, although the Newton steps for Re = 200 need a distinct number of steps more. For Re $\geq$ 300 the number of steps increases as λ gets larger. For Re = 500 there are two configurations that do not perfectly fit in this scheme. The influence of the initial feedback K_0 for Re > 200 influences the number of steps as well, such that some configurations with supposed higher complexity need less ADI steps, because we start with a quite good initial guess.

The best choice of λ in the closed-loop simulation is not obvious. Moreover, the underlying finite element discretization gives us natural bounds for the input $\mathbf{u}$ and, hence, for λ as well. Furthermore, the best choice of λ is a separate optimization task and depends on the design of the test configuration, but will not be discussed further in this paper.

4.1.3 Control cost

The number of Newton steps associates in a natural way to the difficulty of the optimization problem. The feedback matrices computed by using more Newton steps force the system more strictly to the desired state. Such a strict forcing requires larger control cost, which is implicitly shown in Fig. 6. To show this in a more detailed way we compute the stationary optimal control $\mathbf{u_w}$ with respect to the linearization point $\vec{w}$ of system (3) and its corresponding discretized velocity field $\mathbf{w}$. We measure $\mathbf{u_w}$ as the square of the Euclidean norm:

$$||\mathbf{u_w}||_2^2 = \mathbf{u_w}^T\mathbf{u_w} = \mathbf{w}^T K^T K\mathbf{w}.$$



(a) Various Reynolds number and fixed λ. (b) Various λ and fixed Re.

Figure 7: Initial control cost with $\mathbf{u}_\mathbf{w} = -K\mathbf{w}$, where $\mathbf{w}$ is the discretized version of the stationary velocity field $\vec{w}$ in (3).

Using the same configurations as above, Fig. 7a shows that the control cost increases for higher Reynolds numbers up to an upper bound; that is the same behavior like we have seen for the number of Newton steps in Fig. 5a. It is natural that the control cost is smaller for lower λ, because the control cost is implicitly penalized for $\lambda < 1$. To illustrate the monotone increase for increasing λ, we fix the Reynolds number and show the influence of the varying λ in Fig. 7b. There is a Noticeable gap between $\mathrm{Re} \leq 200$ and $\mathrm{Re} \geq 300$ that is related to the instability of systems with higher Reynolds numbers as explained in Section 3.3.

4.2 Closed-loop simulation

In this subsection we verify the usability of the feedback matrix K computed via Algorithm 2 to stabilize the flow problem (1) during a forward simulation. Here, by stabilization we mean to smooth vortexes behind the obstacle that occur for Reynolds numbers $\mathrm{Re} > 200$. We consider a laminar stationary flow field $\vec{w}$ as linearization point.

The matrix K, computed in MATLAB is imported into our finite element flow solver NAVIER using the *Matrix Market* format.

In [36] Raymond shows that the optimal control $\mathbf{u}(t)$ computed for the linearized system (4) also stabilizes the non-linear system (1) if we assume $\vec{v} \approx \vec{w}$; that means the deviations of the velocity field $\vec{v}$ from the stationary velocity field $\vec{w}$ are small enough [6].

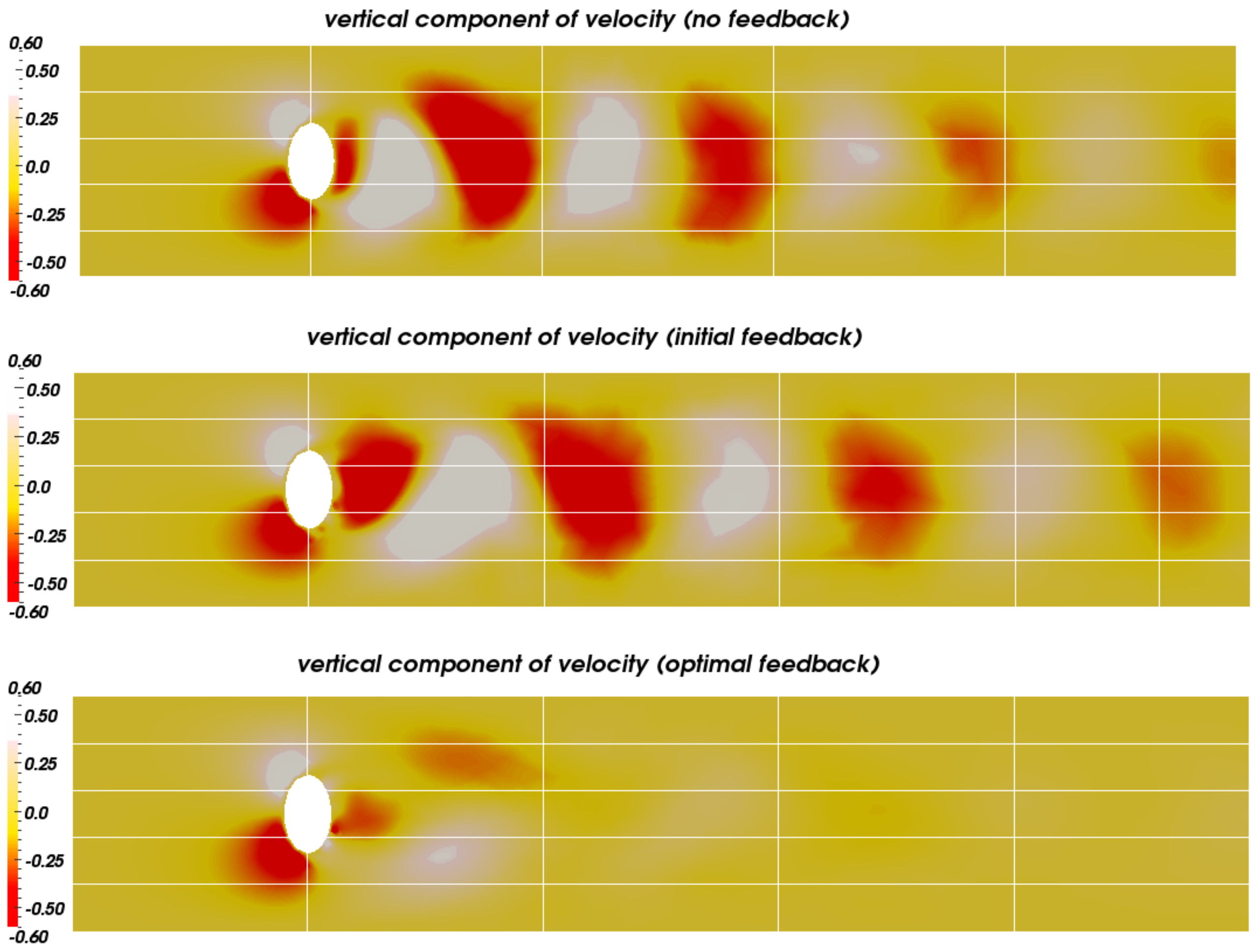


Figure 8: Screenshots of closed-loop simulation for $\mathrm{Re} = 300, \lambda = 2$.

For the controlled forward simulation we proceed as follows. We use the same finite element discretization as above, such that the spatially discretized non-linear system can be written as

$$M\frac{d}{dt}\mathbf{v}(t) = A(\mathbf{v}(t))\mathbf{v}(t) + G\tilde{\mathbf{p}}(t) + \tilde{B}\mathbf{u}(t), \tag{18a}$$

$$0 = G^T\mathbf{v}(t), \tag{18b}$$

$$\mathbf{y}(t) = C\mathbf{v}(t), \tag{18c}$$

with the discrete velocity $\mathbf{v}(t) \in \mathbb{R}^{n_v}$, the discrete pressure $\tilde{\mathbf{p}}$, and the non-linear Navier-Stokes operator $A(\mathbf{v}(t))$ that depends on the current velocity field $\mathbf{v}(t)$. Additionally, we have the discrete versions of the boundary conditions (2). We start with an initial condition

$$\mathbf{v}(0) = \mathbf{w},$$

such that the perturbations from the stationary trajectory are zero at the beginning. We refer the reader to [5] for more details on solving the non-linear system (18).

The actual $\mathbf{u}(t)$ is computed on the discrete level via

$$\mathbf{u}(t) = -K\mathbf{z}(t) = -(K\mathbf{v}(t) - K\mathbf{w});$$

that means the feedback matrix K is applied to the stationary linearization point $\mathbf{w}$ and the recent velocity field $\mathbf{v}(t)$. The values of $\mathbf{u}(t)$ describe the magnitude of the parabolic inflow or outflow conditions via the control boundaries $\Gamma_{feed,i} \forall i = 1, 2$, compare Fig. 1 and Fig. 3. In this setting the input operator $\tilde{B}$ maps the parabolic control inflow at the feedback boundaries $\Gamma_{feed,i}, \forall i = 1, 2$ with the magnitudes $u_i(t)$ to the control Dirichlet conditions

$$\mathbf{v}(t) = \mathbf{g}_{feed}(t) := \tilde{B}u_i(t) \text{ on } \Gamma_{feed,i}, \forall i = 1, 2.$$

Using these boundary conditions the closed-loop forward simulation can be solved.

The output operator C measures the vertical velocity component v_{x_2} at seven different nodes of the FE grid behind the obstacle. Getting these components as small as possible is a way to measure how laminar the flow field is.

The choice of λ influences the quality of the results. To find the optimal value λ one would need to solve an optimization process; we do not show details regarding this problem in this paper. Another difficulty that occurs is that the computed control $\mathbf{u}(t)$ is not directly constrained. Thus, the required inflow via the control boundaries $\Gamma_{feed,i}$ can exceed the resolution capabilities of the underlying finite element grid.

Fig. 8 shows a flow field for Re = 300. The vertical component of the velocity is depicted in red, as maximal value downwards, and white, as maximal value upwards. As explained above, the observation matrix C measures the vertical velocities only in a few nodes behind the obstacle. We divided our domain into equi-sized rectangles of five rows and five columns for better illustration. The measurement nodes are located in the vertical center row in the third and fourth horizontal sections from the left. To achieve the goal of smoothing all vortexes, the flow field should not consist of vertical velocity components; a zero vertical velocity corresponds to yellow in the applied color scale. Due to the nature of our cost function, we focus especially on the above-mentioned measuring rectangles on which deviations from zero are penalized.

The top picture shows the velocity field at $t = 25$ without any feedback influence. The occurring vortexes are shown by the red and white areas that move away from the obstacle in an alternating order.

The middle picture visualizes the influence of the initial feedback K_0. Although K_0 numerically stabilizes the matrix pencil, the forward simulation is not stabilized with respect to occurring vortexes.

Finally, the bottom picture represents the closed-loop simulation for our feedback matrix K computed with Algorithm 2 using the regularization parameter $\lambda = 2$. Nearly all vortexes are smoothed and the flow field is laminar, especially in the rectangles in our focus. Here, the value $\lambda = 2$ is an experimentally determined value. For $\lambda < 2$ the flow field still consists of vortexes. For $\lambda > 2$ the finite element grid cannot handle the in-/outflow conditions required for the computed feedback. More drastic feedback influence would require a local mesh refinement. Unfortunately, this would require to compute a new feedback matrix. Recently, we investigate the connection between coarse grid computed feedback influences applied to fine grid forward simulations. A video of the closed-loop forward simulation is available at: `https://zenodo.org/record/7109?ln=en#.Ujq9fKxMI_E` [42].

5 Conclusions and outlook

In this paper we have investigated the numerical realization of a boundary feedback stabilization for instationary Navier-Stokes flow. Based on an approach by Raymond [35, 36], the linearization around a given stationary trajectory is the starting point to apply a linear quadratic regulator approach. Raymond uses the *Leray* projector to define the evolution equation for the regulator approach on the divergence-free vector field. The stabilizing feedback operator is based on the solution of an operator Riccati equation. Using a finite element discretization in space yields an algebraic Riccati equation that has to be solved numerically. These large scale and non-linear equations can be solved with a Newton type method. Starting with a suitable initial guess, the Newton iteration converges superlinearly to the unique root that defines the stabilizing feedback. Every Newton step consists of solving a Lyapunov equation, which is solved by applying a low-rank factor ADI method. To this end, a linear solve has to be done in each ADI step.

We have used a Taylor-Hood [28] mixed spatial discretization as a stable conforming finite element method to semi discretize the evolution equation. Thus, the incompressibility condition is not automatically built into the trial space and requires additional effort. In [25] Heinkenschloss and colleagues proposed a trick to overcome this difficulty. We have extended the ideas in [13] that demonstrate the equivalence of this approach with the use of a discrete *Leray* projection for discretized flow fields. We have shown that the Newton-ADI method to solve an algebraic Riccati equation can be reformulated for such a DAE setting. The crucial step to use the Newton-ADI method without the explicit projection is the linear solve of large-scale non-symmetric saddle point systems. The occurring structure is common for Navier-Stokes flow related problems. Again, we have extended the approaches in [13] to derive a block preconditioner to solve such saddle point systems efficiently within an iterative Krylov method.

Compared to the Stokes case [13], the Navier-Stokes case has the need of an initial feedback to guarantee convergence of the Newton-ADI method. In this paper we have shown a way to compute such an initial feedback based on the solution of a Bernoulli equation. Using this initial feedback, the threefold nested iteration determines a stabilizing feedback matrix.

We have illustrated the influence of the occurring parameters with some numerical examples based on the forward simulation of the "von Kármán vortex street". A closed-loop simulation for this example has verified the stabilizing property of the feedback matrix.

In the future we are going to extend this approach to coupled multi-field flow problems. Furthermore, the parameter dependence within the nested iteration, as well as the handling of complex ADI shifts, will be a challenging task. We hope to benefit from some recent ADI improvements such as avoiding complex arithmetic [9] or efficient low-rank residual computations [10].

Moreover, we plan to expand the iterative solution strategies in [13] to the Navier-Stokes and coupled multi-field flow problems on finer meshes. To this end, we will investigate efficient block preconditioning methods. To deal with the special structure of the saddle point systems arising in the Newton-ADI algorithm we will investigate recycling and block Krylov methods to accelerate the time-consuming handling of multiple right hand sides for iterative linear solvers.

Acknowledgments

The work reported in this paper is part of the project *Optimal Control-Based Feedback Stabilization of Multi-Field Flow Problems.* This project is embedded in the DFG priority program 1253: *Optimization with Partial Differential Equations.* We thank Martin Stoll and Andy Wathen for the inspiring discussions about preconditioning of iterative methods. Furthermore, we thank Stephan Weller for the help with implementation issues in NAVIER.

References

[1] L. Amodei and J.-M. Buchot, *A stabilization algorithm of the Navier–Stokes equations based on algebraic Bernoulli equation*, Numer. Lin. Alg. Appl., 19 (2012), pp. 700–727.

[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, *Simulation of instationary, incompressible flows*, Acta Math. Univ. Comenianae, 67 (1998), pp. 101–114.

[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] S. Barrachina, P. Benner, and E. Quintana-Ortí, *Efficient algorithms for generalized algebraic Bernoulli equations based on the matrix sign function*, Numer. Algorithms, 46 (2007), pp. 351–368.

[8] P. Benner, *Solving large-scale control problems*, IEEE Contr. Syst. Mag., 14 (2004), pp. 44–59.

[9] P. Benner, P. Kürschner, and J. Saak, *Efficient Handling of Complex Shift Parameters in the Low-Rank Cholesky Factor ADI Method*, Numer. Algorithms, 62 (2013), pp. 225–251. 10.1007/s11075-012-9569-7.

[10] ———, *An improved numerical method for balanced truncation for symmetric second order systems*, Math. Comp. Model. Dyn., (2013). Early Version available from `http://www.mpi-magdeburg.mpg.de/preprints/2012/20/`.

[11] 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.

[12] P. Benner and J. Saak, *A Galerkin-Newton-ADI method for solving large-scale algebraic Riccati equations*, Preprint SPP1253-090, DFG-SPP1253, `http://www.am.uni-erlangen.de/home/spp1253/wiki/images/2/28/Preprint-SPP1253-090.pdf`, 2010.

[13] 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*, SIAM J. Sci. Comput. To appear. Early Version available from `http://www.am.uni-erlangen.de/home/spp1253/wiki/images/b/bc/Preprint-SPP1253-130.pdf`.

[14] A. Bensoussan, G. D. Prato, M. Delfour, and S. Mitter, *Representation and control of infinite dimensional systems, Volume I*, Systems & Control: Foundations & Applications, Birkäuser, Boston, Basel, Berlin, 1992.

[15] R. F. Boisvert, R. Pozo, and K. A. Remington, *The matrix market exchange formats: Initial design*, NIST Interim Report, 5935 (1996).

[16] M. O. Bristeau, R. Glowinski, and J. Périaux, *Numerical methods for the Navier-Stokes equations. Applications to the simulation of compressible and incompressible viscous flows*, in Finite Elements in Physics (Lausanne, 1986), North-Holland, Amsterdam, 1987, pp. 73–187.

[17] J. A. Burns, E. W. Sachs, and L. Zietsman, *Mesh independence of Kleinman–Newton iterations for Riccati equations in Hilbert space*, SIAM J. Control Optim., 47 (2008), pp. 2663–2692.

[18] K. A. Cliffe, T. J. Garratt, and A. Spence, *Eigenvalues of block matrices arising from problems in fluid mechanics*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 1310–1318.

[19] E. Deriaz and V. Perrier, *Orthogonal Helmholtz decomposition in arbitrary dimension using divergence-free and curl-free wavelets*, Appl. Comput. Harmon. Anal., 26 (2009), pp. 249–269.

[20] H. Elman, D. Silvester, and A. Wathen, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*, Oxford University Press, Oxford, 2005.

[21] V. Girault and P. Raviart, *Finite element methods for Navier-Stokes equations*, Springer-Verlag Berlin, 1986.

[22] G. Golub and C. Van Loan, *Matrix computations*, Johns Hopkins University Press, Baltimore, third ed., 1996.

[23] S. Gugercin, T. Stykel, and S. Wyatt, *Model reduction of descriptor systems by interpolatory projection methods*, tech. rep., arXiv, 2013. `http://arxiv.org/abs/1301.4524`.

[24] S. Hein, *MPC-LQG-based optimal control of parabolic PDEs*, PhD thesis, TU Chemnitz, February 2009. Available from `http://archiv.tu-chemnitz.de/pub/2010/0013`.

[25] 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 J. Sci. Comput., 30 (2008), pp. 1038–1063.

[26] J. Heywood, R. Rannacher, and S. Turek, *Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations*, Internat. J. Numer. Methods Fluids, 22 (1996), pp. 325–352.

[27] M. Hinze and K. Kunisch, *Second order methods for boundary control of the instationary Navier-Stokes system.*, Z. Angew. Math. Mech., 84 (2004), pp. 171–187.

[28] 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.

[29] D. Kleinman, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.

[30] P. Lancaster and L. Rodman, *The algebraic Riccati equation*, Oxford University Press, Oxford, 1995.

[31] I. Lasiecka and R. Triggiani, *Control theory for partial differential equations: Continuous and approximation theories I. Abstract parabolic systems*, Cambridge University Press, Cambridge, UK, 2000.

[32] J.-R. Li and J. White, *Low rank solution of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280.

[33] A. Locatelli, *Optimal control: An introduction*, Birkhäuser Verlag, Basel, Boston, Berlin, 2001.

[34] 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`.

[35] J.-P. 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.

[36] ——, *Feedback boundary stabilization of the two-dimensional Navier-Stokes equations*, SIAM J. Control Optim., 45 (2006), pp. 790–828.

[37] ——, *Feedback boundary stabilization of the three-dimensional incombressible Navier-Stokes equations*, J.Math. Pures Appl. (9), 87 (2007), pp. 627–669.

[38] ——, *Stokes and Navier–Stokes equations with nonhomogeneous boundary conditions*, Ann. I. H. Poincaré – AN, 24 (2007), pp. 921–951.

[39] 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.

[40] W. E. Schiesser, *Numerical methods of lines*, Academic Press, 1991.

[41] E. L. Wachspress, *The ADI model problem*, Springer New York, 2013.

[42] H. K. Weichelt, *Riccati-based feedback stabilized Navier-Stokes flow*, (2013). `https://zenodo.org/record/7110?ln=en#.UjrAEaxMI_F`.

[43] 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.