Numerical Aspects of Flow Stabilization by Riccati Feedback

Dissertation

zur Erlangung des akademischen Grades

doctor rerum naturalium
(Dr. rer. nat.)

von **Dipl.-Math. techn. Heiko K. Weichelt**

geb. am **23.02.1986** in **Zwickau**

genehmigt durch die Fakultät für Mathematik
der Otto-von-Guericke-Universität Magdeburg

Gutachter:	**Prof. Dr. Peter Benner**
	Prof. Dr. Ekkehard Sachs
	Prof. Dr. Eberhard Bänsch

eingereicht am:	**28. Januar 2016**
Verteidigung am:	**7. Juli 2016**

Numerical Aspects of Flow Stabilization by Riccati Feedback

Dissertation

by

Dipl.-Math. techn. Heiko K. Weichelt

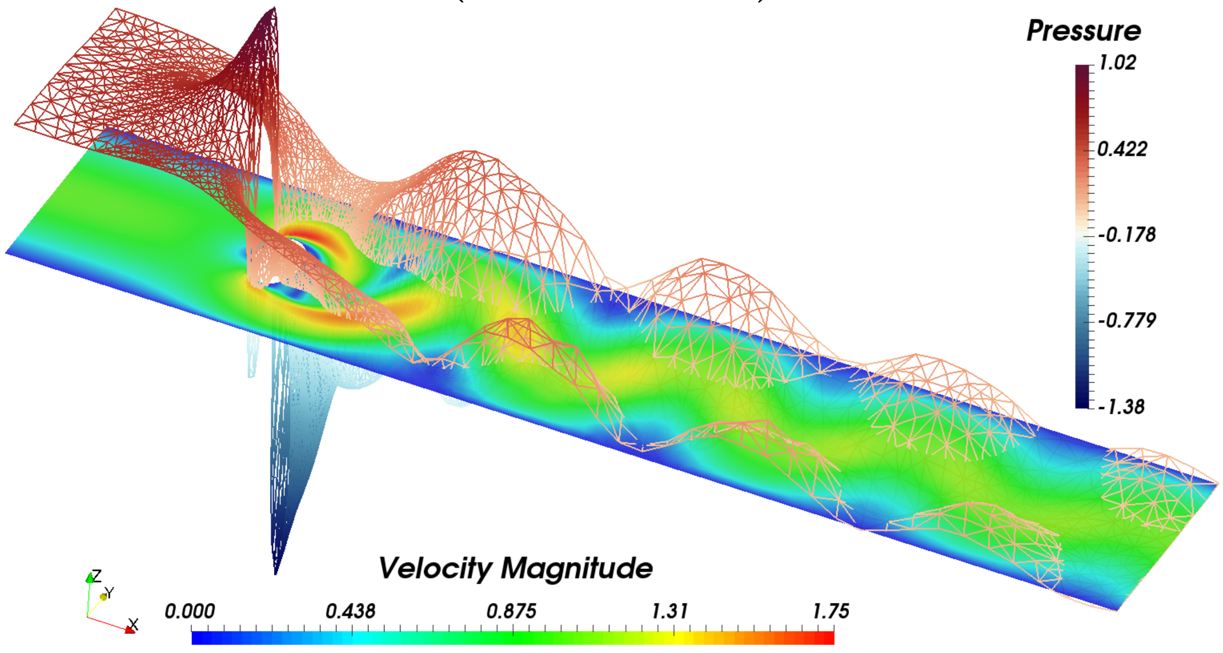
Submitted to **Faculty of Mathematics**

at **Otto-von-Guericke-Universität Magdeburg**

in accordance with the requirements for the degree

doctor rerum naturalium

(Dr. rer. nat.)



Advisor:	Prof. Dr. Peter Benner
2nd Reviewer:	Prof. Dr. Ekkehard Sachs
3rd Reviewer:	Prof. Dr. Eberhard Bänsch

Magdeburg, 28th of January, 2016

Pentru iubita mea soţie Ioana.

Acknowledgments

Finishing this thesis closes the most engaging chapter of my life so far. Many people have contributed to my work and to my life.

I want to thank my supervisor Peter Benner who gave me this great opportunity to work in his group and to get in touch with so many researchers and interesting people all over the world. I also want to thank him for all his constructive corrections throughout the process of writing this thesis.

Many colleagues contributed either by their knowledge or their guidance to the success of this thesis and I cannot name each single person. However, a special thanks goes to Eberhard Bänsch, Martin Stoll, Matthias Heinkenschloss, Stephan Weller, and Jan Heiland.

Many of my colleagues became close friends and brightened up life at work substantially. I am thankful to call them my friends: Matthias Voigt, Thilo Muth, Jonas Denißen, Tobias Breiten, Norman Lang, Cosmin Ioniţă, Jessica Bosch, Oliver Hädicke, Martin Köhler, and André and Judith Schneider.

A special thank for proofreading the entire thesis goes to my friends Tom Barker, Manuela Krones, and Patrick Kürschner. The latter one is not only a colleague but became my best friend here in Magdeburg.

Natürlich möchte ich nicht vergessen, mich bei meiner Familie für die jahrelange Unterstützung zu bedanken. Ihr alle habt die Grundlage gelegt, dass ich heute diese Arbeit abschließen kann.

The person who helped and influenced my scientific life most, is my dear colleague and friend Jens Saak. Without his help, many of the ideas would have never been discovered and many LaTeX problems would still be unsolved.

Last but not least, I want to say thank you to my beloved wife Ioana. She moved across the ocean for me and made my life so much better. Mulţumesc!

Abstract

In this thesis, we examine important numerical aspects of a Riccati-based feedback stabilization approach in order to stabilize incompressible flow problems.

Various transport and flow problems are important in many technical applications. To control or influence these generally nonlinear problems, one usually uses an open-loop controller. Highly sophisticated solvers exist for this kind of problem. However, an open-loop controller is unstable with regard to small perturbations. Using a feedback stabilization approach for the linearization around the open-loop trajectory increases the robustness of these methods drastically.

The main issue in deriving such a Riccati-based feedback is the efficient solution of a large-scale generalized algebraic Riccati equation. The solution of this quadratic matrix equation is derived by applying a specially tailored version of Newton's method. Due to the natural divergence-free condition of incompressible flow problems, all systems occur as differential-algebraic systems. The solution of this kind of equations is highly demanding. We try to avoid general-purpose solution strategies for these differential-algebraic equations by using and extending an existing implicit projection method.

A highly efficient algorithm to determine the Riccati-based feedback for various flow scenarios has been established by modifying and combining various existing solution strategies. The combination of all these strategies is completely new and only possible because of current improvements. The key ingredient to enable the synergy of these methods are low-rank structures and specially tailored algorithms that exploit these structures. A convergence proof for our proposed method as well as thorough numerical experiments verify the usability of our approach.

By modifying and extending an existing finite element flow solver, we have been able to extract the arising finite dimensional matrices. These matrices are used within our MATLAB®-based algorithms to compute the Riccati-based feedback. The computed feedback is included into the flow solver such that a closed-loop simulation is able to validate the usability of our proposed approach to stabilize the Navier–Stokes equations over the *Kármán vortex street.*

Zusammenfassung

Diese Arbeit beschäftigt sich mit wichtigen numerischen Aspekten, um inkompressible Strömungsprobleme mit der Hilfe eines Riccati-basierten Feedbacks zu stabilisieren.

In vielen technischen Anwendungen ist die Lösung von Transport- und Strömungsproblemen von hoher Bedeutung. Um diese im Allgemeinen nichtlinearen Probleme zu beeinflussen, benutzt man für gewöhnlich *open-loop controller*. Für diese Klasse von Problemen existieren hoch entwickelte Lösungsmethoden. Ein *open-loop controller* ist jedoch instabil bezüglich kleinster Störungen. Die Robustheit dieser Methoden kann durch die Benutzung einer Feedback-Stabilisierung, basierend auf einer Linearisierung um die *open-loop* Trajektorie, erheblich verbessert werden.

Das Hauptproblem bei der Erzeugung solch eines Riccati-basierten Feedbacks ist die Lösung einer hochdimensionalen verallgemeinerten algebraischen Riccati-Gleichung. Die Lösung dieser quadratischen Matrixgleichung ist von der Anwendung eines speziell angepassten Newton-Verfahrens abgeleitet. Aufgrund der natürlichen Divergenzfreiheit der inkompressiblen Strömungen treten alle Systeme als differentiell-algebraische Gleichungen auf. Die Lösung solcher Systeme ist sehr anspruchsvoll. Daher versuchen wir in dieser Arbeit allgemeingültige Lösungsstrategien für diese differentiell-algebraischen Gleichungen zu vermeiden, indem wir eine existierende implizite Projektionsmethode anpassen.

Durch die Modifizierung verschiedener existierender Lösungsstrategien, die in dieser Kombination vorher nicht zusammen benutzt werden konnten, wird ein hoch effizienter Algorithmus zur Bestimmung des Riccati-basierten Feedbacks eingeführt und für verschiedene strömungsmechanische Szenarien angewandt. Die Hauptbestandteile, die das Zusammenwirken dieser Methoden ermöglichen, sind Niedrigrangstrukturen und speziell zugeschnittene Algorithmen, die diese Strukturen ausnutzen. Ein Konvergenzbeweis für unsere vorgeschlagene Methode sowie gründliche numerische Experimente bestätigen die Anwendbarkeit unserer Methode.

Durch die gezielte Anpassung und Erweiterung eines existierenden Finite-Elemente-Programms war es uns möglich, die entstehenden Matrizen zu extrahieren. Diese Matrizen werden in den MATLAB-basierten Algorithmen zur Berechnung des Riccati-basierten Feedbacks benutzt. Das dadurch berechnete Feedback wird in das Finite-Elemente-Programm eingebunden. Eine *closed-loop* Simulation ermöglicht es uns, die Verwendbarkeit unserer vorgeschlagenen Methode zur Stabilisierung der Navier–Stokes-Gleichungen in der Kármánschen Wirbelstraße zu überprüfen.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

(G)ABE	(generalized) algebraic Bernoulli equation
ADI	alternating directions implicit
AMG	algebraic multigrid
BC	boundary conditions
(G)CALE	(generalized) continuous-time algebraic Lyapunov equations
(G)CARE	(generalized) continuous-time algebraic Riccati equations
CFM	coupled flow model
CTP	combined transport process
DAE	differential-algebraic equations
DCE	diffusion-convection equation
ELSM	exact line search method
EVP	eigenvalue problem
FEM	finite element method
GMRES	generalized minimal residual
IC	initial condition
KNM	Kleinman–Newton method
KN-ADI	Kleinman–Newton-ADI
KVS	Kármán vortex street
LQR	linear-quadratic regulator
(G)-LRCF-ADI	(generalized) low-rank Cholesky factor ADI iteration
LTI	linear, time-invariant
LTV	linear, time-varying
MIMO	multiple input, multiple output
NSE	Navier–Stokes equations
ODE	ordinary differential equation
PDE	partial differential equation
RM	reactor model
SC	Schur complement
SISO	single input, single output
SPDC	shifted pressure diffusion-convection
sp(s)d	symmetric, positive-(semi)definite

splr	sparse plus low-rank
SPS	saddle point system
UCD	unit cube domain

List of Symbols

General Notation and Spaces

$\mathbb{K} \in \{\mathbb{N}, \mathbb{R}, \mathbb{C}\}$	set of natural, real, or complex numbers
$\jmath$	imaginary unit with $\jmath^2 = -1$
$\overline{z} = a - \jmath b$	complex conjugate of $z = a + \jmath b \in \mathbb{C}$ with $a, b \in \mathbb{R}$
$\mathrm{Re}\,(z) = a$	real part of $z = a + \jmath b \in \mathbb{C}$
$\mathrm{Im}\,(z) = b$	imaginary part of $z = a + \jmath b \in \mathbb{C}$
$\mathbb{R}^-$ $(\overline{\mathbb{R}}^-)$	set of real numbers a with $a < 0$ $(a \leq 0)$
$\mathbb{R}^+$ $(\overline{\mathbb{R}}^+)$	set of real numbers a with $a > 0$ $(a \geq 0)$
$\mathbb{C}^-$ $(\overline{\mathbb{C}}^-)$	set of complex numbers z with $\mathrm{Re}\,(z) < 0$ $(\mathrm{Re}\,(z) \leq 0)$
$\mathbb{C}^+$ $(\overline{\mathbb{C}}^+)$	set of complex numbers z with $\mathrm{Re}\,(z) > 0$ $(\mathrm{Re}\,(z) \geq 0)$
$L^2(\Omega)$	space of all functions $f : \Omega \to \mathbb{R}$ that are square integrable
$H^1(\Omega)$	space of all functions $f \in L^2(\Omega)$ with $\nabla f \in L^2(\Omega)$
$\mathbb{K}^n$	space of n-dimensional vectors with entries in $\mathbb{K}$
$\mathbb{K}^{n \times m}$	space of $n \times m$ matrices with entries in $\mathbb{K}$
x	scalar variable or function in $\mathbb{K}$
$\vec{x} := \begin{bmatrix} x_1, \dots, x_d \end{bmatrix}^T$	vector-valued variable or function in $\mathbb{K}^d$
$\boldsymbol{x} := \begin{bmatrix} x_1, \dots, x_n \end{bmatrix}^T$	discretized version of variable x or $\vec{x}$ in $\mathbb{K}^n$
$A := (a_{i,j})_{i,j=1}^{n,m}$	constant matrix in $\mathbb{K}^{n \times m}$
$a_{i,j} \in \mathbb{K}$	matrix entry in i-th row and j-th column of A
$I_n = \begin{bmatrix} e_1 & \dots & e_n \end{bmatrix} \in \mathbb{K}^{n \times n}$	identity matrix of dimension $n \times n$

$e_i \in \mathbb{K}^n$	i-th unit vector of length n
$\mathbb{1}_n = [1,\dots,1]^T \in \mathbb{R}^n$	vector with all ones of dimension n
$A^T := (a_{j,i})_{i,j=1}^{n,m} \in \mathbb{K}^{m\times n}$	transpose of a matrix $A \in \mathbb{K}^{n\times m}$
$A^H := (\overline{a}_{j,i})_{i,j=1}^{n,m} \in \mathbb{C}^{m\times n}$	complex conjugate and transpose of a matrix $A \in \mathbb{C}^{n\times m}$
$\operatorname{diag}(A) := \{(a_{i,j})_{i,j=1}^{n,m} : a_{i,j} = 0,\ \forall i \neq j\}$	diagonal matrix of $A \in \mathbb{K}^{n\times m}$
$\Lambda(A) := \{\lambda : \det(\lambda I_n - A) = 0\}$	spectrum of quadratic matrix $A \in \mathbb{K}^{n\times n}$
$\lambda_i \in \Lambda(A)$	i-th eigenvalue of quadratic matrix $A \in \mathbb{K}^{n\times n}$
$\Lambda(A,B) := \{\lambda : \det(\lambda B - A) = 0\}$	spectrum of the matrix pencil (A,B) with $A,B \in \mathbb{K}^{n\times n}$
$\sigma(A,B) := \max\{\operatorname{Re}(\lambda) : \lambda \in \Lambda(A,B)\}$	spectral abscissa of matrix pencil (A,B) with $A,B \in \mathbb{K}^{n\times n}$
$\Lambda_\epsilon(A,B) := \{\lambda \in \mathbb{C} : \|\|(\lambda B - A)^{-1}\|\| > \epsilon^{-1}\}$	ϵ-pseudospectrum of matrix pencil (A,B) with $A,B \in \mathbb{K}^{n\times n}$ and $\epsilon > 0$
$\sigma_\epsilon(A,B) := \max\{\operatorname{Re}(\lambda) : \lambda \in \Lambda_\epsilon(A,B)\}$	ϵ-pseudospectral abscissa of matrix pencil (A,B) with $A,B \in \mathbb{K}^{n\times n}$ and $\epsilon > 0$
$\boldsymbol{A}$	block structured matrix related to DAE structure
$\mathcal{A}$	projected matrix related to A
$\mathcal{A}(.)$	matrix function

Operators

$(\boldsymbol{x},\boldsymbol{y}) := \boldsymbol{x}^H\boldsymbol{y} = \sum\limits_{i=1}^n x_i\, y_i$	Euclidean inner product for $\boldsymbol{x},\boldsymbol{y} \in \mathbb{K}^n$
$<\boldsymbol{x},\boldsymbol{y}>_M := (M\boldsymbol{x},\boldsymbol{y})$	M-inner product for $\boldsymbol{x},\boldsymbol{y} \in \mathbb{K}^n$ with spd $M \in \mathbb{K}^{n\times n}$
$\|\|\boldsymbol{x}\|\|_2 := \left(\sum\limits_{i=1}^n x_i^2\right)^{\frac{1}{2}}$	Euclidean norm of vector $\boldsymbol{x} \in \mathbb{K}^n$
$\|\|\boldsymbol{x}\|\|_\infty := \max\limits_{i=1,\dots,n} \|x_i\|$	maximum norm of vector $\boldsymbol{x} \in \mathbb{K}^n$
$\operatorname{tr}(A) := \sum\limits_{i=1}^n a_{i,i} = \sum\limits_{i=1}^n \lambda_i$	trace of matrix $A \in \mathbb{K}^{n\times n}$
$\langle A,B\rangle := \sum\limits_{i=1}^n \sum\limits_{j=1}^m a_{i,j} b_{i,j} = \operatorname{tr}\left(A^H B\right)$	inner product for $A,B \in \mathbb{K}^{n\times m}$
$\|\|A\|\|_F := \left(\sum\limits_{i=1}^n \sum\limits_{j=1}^m \|a_{i,j}\|^2\right)^{\frac{1}{2}} = \sqrt{\langle A,A\rangle}$	Frobenius norm of matrix $A \in \mathbb{K}^{n\times m}$
$\|\|A\|\|_2 := \sqrt{\lambda_{\max}(A^H A)}$	spectral norm of matrix $A \in \mathbb{K}^{n\times m}$

$\varkappa(A) := \|\|A\|\|\,\|\|A^{-1}\|\|$	condition number of $A \in \mathbb{R}^{n\times m}$ induced by the respective norm
$\operatorname{null}(A) := \{\boldsymbol{x} \in \mathbb{K}^m : A\boldsymbol{x} = 0\}$	null-space of matrix $A \in \mathbb{K}^{n\times m}$
$\operatorname{range}(A) := \{\boldsymbol{y} \in \mathbb{K}^n : \boldsymbol{y} = A\boldsymbol{x},\, \boldsymbol{x} \in \mathbb{K}^m\}$	range of matrix $A \in \mathbb{K}^{n\times m}$
$A^\perp := \{\boldsymbol{x} \in \mathbb{K}^n : (\boldsymbol{x}, \boldsymbol{y}) = 0,\, \forall \boldsymbol{y} \in \operatorname{range}(A)\}$	orthogonal complement of the range of $A \in \mathbb{K}^{n\times m}$
$\nabla := \left[\frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_d}\right]^T$	d-dimensional nabla operator
$\Delta := \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$	d-dimensional Laplace operator
$\operatorname{div} \vec{w} := \sum_{i=1}^{d} \frac{\partial w_i}{\partial x_i}$	divergence operator for $\vec{w} \in \mathbb{R}^d$ in d-dimensional space
$y_{x_i} := \frac{\partial y}{\partial x_i}$	partial derivative of $y \in \mathbb{R}$ with respect to x_i
$A = A^T \succ 0$ $(\succeq 0)$	matrix A is symmetric and positive-(semi)definite
$A \succ B$ $(A \succeq B)$	$A - B$ is positive-(semi)definite
$\mathscr{F} : B_1(\Omega_1) \to B_2(\Omega)$	operator function that maps between functional spaces $B_1(\Omega)$ and $B_2(\Omega)$

Problem Dependent Variables and Parameters

$d \in \{2, 3\}$	problem dimension
$t \in \mathbb{R}$	time
$\vec{x} = [x_1, \ldots, x_d]^T \in \mathbb{R}^d$	space coordinate
$\Omega \subset \mathbb{R}^d$	bounded domain
$\Gamma := \partial\Omega$	boundary of Ω
$Q_\infty := \Omega \times [t_0, \infty)$	space time cylinder with $\vec{x} \in \Omega$ and $t \in [t_0, \infty)$
$\Sigma_\infty := \Gamma \times [t_0, \infty)$	boundary time cylinder with $\vec{x} \in \Gamma$ and $t \in [t_0, \infty)$
$\vec{n}(\vec{x}) \in \mathbb{R}^d$	outward normal dependent on $\vec{x}$
$\vec{v}(t, \vec{x}) \in \mathbb{R}^d$	velocity
$\chi(t, \vec{x}) \in \mathbb{R}^+$	pressure
$\vec{f}(t, \vec{x}) \in \mathbb{R}^d$	external forces
$\rho \in \mathbb{R}^+$	density
$\eta \in \mathbb{R}^+$	dynamic viscosity
$c(t, \vec{x}) \in \mathbb{R}^+$	concentration
$D \in \mathbb{R}^+$	diffusion coefficient
$\nu := \frac{\eta}{\rho} \in \mathbb{R}^+$	kinematic viscosity
$d_{\text{ref}} \in \mathbb{R}^+$	reference length

$v_{\mathrm{ref}} \in \mathbb{R}^+$	reference velocity
$\mathrm{Re} := \dfrac{\rho \cdot v_{\mathrm{ref}} \cdot d_{\mathrm{ref}}}{\eta} \in \mathbb{R}^+$	Reynolds number
$\mathrm{Sc} := \dfrac{\nu}{D} = \dfrac{\eta}{\rho D} \in \mathbb{R}^+$	Schmidt number
$\varepsilon_{\mathrm{mach}} := 2.2204 \cdot 10^{-16}$	floating-point relative accuracy for double precision

1 Introduction

Contents

The physics of fluid mechanics is present in our daily life, be it by driving a car or airing an apartment. Both actions involve the flow of air with low velocity that can be described by an incompressible flow model. These models are one of the major subjects in fluid mechanics. The behavior of incompressible flows, which can be described mathematically by the Navier–Stokes equations, has been extensively researched for the past decades. In the year 2000, the prestigious *Clay Mathematical Institute* in the U.S.A. announced the question about the existence and smoothness of a solution of the Navier-Stokes equations as one of the seven *Millennium Price Problems* [58]. This unsolved problem investigates under which conditions solutions exist and if they are unique. The main obstacle in describing unique solutions are the naturally existing instabilities that yield chaotically appearing turbulences. These turbulences cannot be computed nor completely predicted, therefore, the stabilization of flow problems is an essential task in many engineering fields.

This thesis does not intend to solve the Millennium Price Problem [58], but it investigates various numerical aspects of flow stabilization by Riccati feedback approaches to avoid turbulences. After introducing the topic in more detail in this chapter, some mathematical basics as well as various physical models are established in the subsequent chapters. The main question of feedback stabilization for flow problems is addressed in Chapter 4, followed by two chapters about some important numerical aspects. The entire thesis concludes with a stabilized flow simulation in Chapter 7.

1.1. Motivation

Transport of any kind is one of the most fundamental dynamical processes in nature and it influences all areas of the natural sciences. For example, the spread of temperature in a room from a heat source into the entire room is based on the scalar transport of this temperature driven by diffusion and convection. Another example is the movement of water inside a river; each water particle has a certain velocity, which is described by a vector. Hence, the dispersion of this vector field defines a vector-valued transport problem, or in other words, a flow problem.

Combining and analyzing such simple connections plays an important role in many engineering areas such as ship, airplane, and car development, where the movement of the object is highly affected by the flow of the surrounding substance. Looking into rather small or even microscopically scaled processes, the biological and chemical industry uses various transport and flow models to set up as well as control certain reaction processes. The knowledge of the involved transport problems is essential to predict the outcome of such experiments.

However, not only the prediction of the resulting behavior is important nowadays. In order to develop new production methods or more safe and efficient vehicles, one needs to actively influence the system by certain inputs that yield the desired outputs. Therefore, it is essential to examine the relationship between the input and the output of a dynamical system as it is done in the research area of the control theory. A widely used approach in this context is the open-loop controller that forces the system towards a certain behavior, for example, the temperature in a room reaches a certain level as fast as possible. Unfortunately, this approach is not stable regarding perturbations that naturally occur in every real-world model and application. Influencing the dynamical system is often done by a so-called distributed control, where each particle in the considered domain or certain control subdomains can be influenced. Again, this is rather complicated to realize if one, for example, considers the flow of a river or the flow inside a closed reactor. A more physically feasible approach that overcomes both drawbacks is based on closed-loop feedback stabilization using boundary control as examined in detail in this thesis.

Certainly, the idea of a feedback stabilization using boundary control is not new. However, the application to incompressible flow problems has revealed various analytical problems that have to be overcome. One of the most inspiring articles in the author's career about these analytical problems is [106], which was published by J. P. Raymond in 2006. In this article, Raymond introduces an analytical examination of the feedback boundary stabilization of the two-dimensional Navier-Stokes equations. Together with further articles by various authors, the functional analytic basics were prepared, but a numerical treatment was still missing. In [13], Bänsch and Benner proclaimed a numerical feasible approach based on the derivation by Raymond that yields the foundation of the author's work.

In this thesis, the ideas from [13] are continued and deepened. Thereby, a numerical framework is derived that applies a Riccati-based boundary feedback stabilization approach to various flow problems. The arising numerically challenging tasks are addressed by algorithms that are able to handle large-scale setups efficiently.

The examples that are considered in this thesis are of relatively academic nature. Nevertheless, they show the feasibility of the proposed approach in the context of boundary feedback stabilization for various flow models. Thus, the examples provide a foundation to derive methods for real-world applications.

1.2. Feedback Control: State of the Art

In general, various approaches for feedback control can be considered. One concept that is widely used in industry is the PID controller. This feedback approach is a combination of the most crucial feedback concepts, namely, the *proportional, integral,* and *differential* feedback. The PID approach, as well as other combinations, are easy to implement and their behavior can be predicted accurately. Nevertheless, non of these methods can claim to work optimal in any way. The use of a proportional feedback for flow simulations has been investigated by the author in [131, 132].

Within this thesis, the Riccati-based feedback approach is considered. Thereby, the most numerically challenging task is the solution of the matrix-valued and quadratic Riccati equation. In detail, one seeks for a solution of dimension $n \times n$, where n is the size of the used discretization. The way of solving this kind of equation is extensively examined in, e.g., [43, 81, 85, 91]. Considering incompressible flow problems yields the so-called divergence-free condition as an additional time-invariant constraint. It is this constraint that makes it impossible to directly apply the standard Riccati approach.

The manner of incorporating this constraint into a feedback stabilization approach for incompressible flow problems has been addressed by various authors such as Fursikov [61], Barbu et al. [16–18], Badra [7, 8], Raymond [105–108], and most recently by Rodrigues [110] and Nguyen et al. [98]. Although these articles provide the basis for all numerical considerations, none of them contains numerical examples. This list of publications is by no means comprehensive. The only articles to be considered in more detail throughout this thesis are the articles [106, 108] by Raymond.

Based on these analytical approaches, a new numerical concept was developed by Bänsch and Benner in [13] using a Riccati-based feedback with an implicit projection method. Parallel to this concept, Amodei and Buchot introduced in [1] a stabilization algorithm based on a Bernoulli equation, which is a special Riccati equation, and examined its usability for different numerical test examples. Within the introduction of [1], it is mentioned that the ideas in [13] face important unsolved critical points, therefore, the authors of [1] adopt the Bernoulli approach for their article.

The essential contribution of this thesis is the development of a numerically feasible stabilization framework based on the analytical approaches in [106, 108]. It is this framework that also provides an innovative technique to solve the critical points in [13]. Therefore, a highly efficient method is derived that treats various numerical aspects of the flow stabilization by a Riccati feedback.

Notice that various authors directly address constraint dynamical systems by using methods for differential-algebraic equations as explained in detail in, e.g., [68, 69, 83, 127] and the references therein. This approach is not further considered within this thesis.

1.3. Problems and Peculiarities

The main problems and peculiarities in applying a Riccati feedback approach to stabilize flow problems are shortly summarized in this section.

In contrast to an open-loop control approach, the Riccati feedback stabilization approach cannot force a system towards a desired non-homogeneous state and can only be applied to linear systems. Nevertheless, it can be used to stabilize nonlinear open-loop trajectories regarding small perturbations. Therefore, one uses such a stationary but possible unstable open-loop trajectory as linearization point. Hence, the nonlinear systems can be rewritten as linear dynamical systems that need to be forced to zero. For this system definition, the Riccati-based linear-quadratic regulator approach can be applied, which means one needs to solve a continuous-time algebraic Riccati equation.

The central gap is to apply this approach to systems with time-invariant constraints without the use of explicit techniques for differential-algebraic equations. Therefore, existing numerical methods need to be adapted and new algorithms have to be developed. Once such a framework is derived, it can be applied to various multi-field flow problems if their discrete formulation fits into a certain scheme.

Various algorithmic improvements in related topics were achieved during the author's PhD career. This thesis incorporates these improvements and combines them with other existing methods that previously could not be used together. It is this combination that enhances the numerical treatment drastically and makes the proposed novel approach highly competitive when compared to other available options.

1.4. Outline

A detailed outline of this thesis is as follows. Chapter 2 reviews various mathematical basics and techniques from the existing literature that are used within the subsequent chapters. In Chapter 3, four physical models for scalar and vector-valued transport problems are introduced. These models serve as test scenarios in the numerical experiments for the derived methods. Each example has been used in the already published articles by the author. The principal contribution of this thesis is stated in Chapter 4, where the feedback stabilization approach for the scenarios from the previous chapter is derived. The statements in Chapter 4 combine the derivations from [14, 15, 35] in a generalized notation. At the end of Chapter 4, the setup for the following numerical tests is defined and various numerical results are presented. The core computational step within all derived algorithms is examined in detail in Chapter 5. Therefore, different linear solvers for specially structured and large-scale linear systems are compared. The theoretical results are generalized formulations of the derivations in [35, 36]. Additional numerical experiments regarding these different linear solvers are depicted at the end of each section in Chapter 5. In Chapter 6, the above mentioned algorithmic improvements are reviewed and incorporated into the methods from Chapter 4. The arising method expands the ideas from [26] to flow problems. It is this innovative algorithm that improves the methods from Chapter 4 significantly. To verify this proposition, the new method is compared with its predecessor from Chapter 4 at the end of Chapter 6. The overall combining result from this thesis is a closed-loop forward simulation of one of the test scenarios from Chapter 3 using the algorithm from Chapter 6 as depicted in Chapter 7. All results and observations, as well as open questions, are summarized in the conclusions in Chapter 8.

2 Mathematical Basics

Contents

The second chapter of this thesis introduces certain mathematical basics that are essential to understand the derivations, methods, and results from Chapters 3–7. Each section is, of course, only a quick introduction into the different topics. For more details, the interested reader is referred to the respective citations within each section.

The chapter is organized as follows. After introducing a spatial discretization technique in Section 2.1, the computation of various matrix quantities is examined in Section 2.2. The principal part of this chapter builds Section 2.3, where many useful aspects regarding optimal control and matrix equations are introduced. Section 2.4 gives an insight to the main iterative methods used within this thesis.

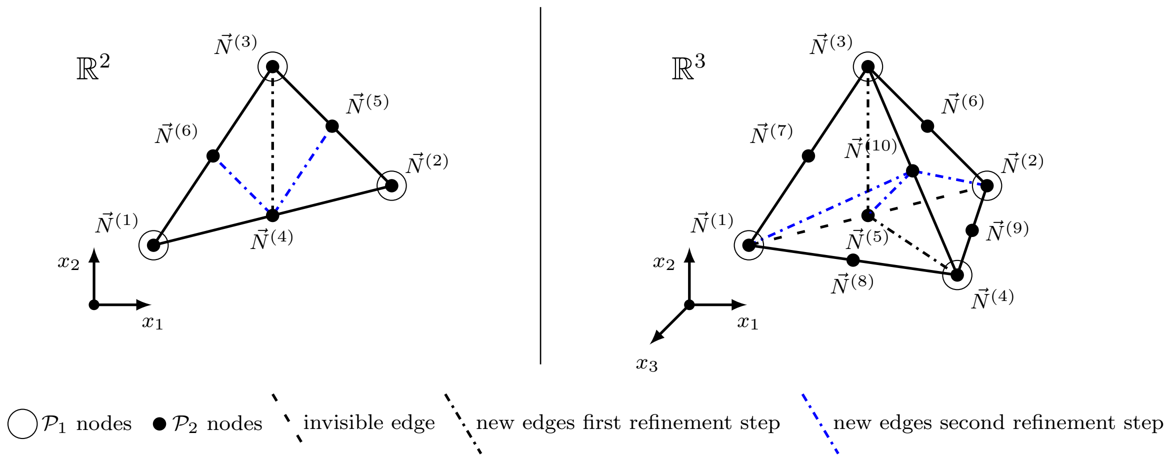


Figure 2.1.: Subdomain Ω_k with nodes of corresponding $\mathcal{P}_1$ and $\mathcal{P}_2$ elements including potential *bisection refinement* edges.

2.1. Finite Element Discretization

A widely used method to approximate the solution of differential equations is the *finite element method* (FEM), which "is today one of the major tools of *Computer Aided Engineering*" [6, p. 1]. To shortly introduce this method, an arbitrary differential equation $\mathscr{F}(a(t,\vec{x})) = 0$ is considered. The solution $a(t,\vec{x}) \in \mathbb{R}$ is continuous with respect to the time $t \in [0,\infty)$ and space $\vec{x} \in \Omega \subset \mathbb{R}^d$. The main idea is to discretize the differential equation in space using certain ansatz functions $\{\varphi_j(\vec{x})\}_{j=1}^n \in \mathbb{R}$ such that $a(t,\vec{x})$ can be approximated as

$$a(t,\vec{x}) \approx \sum_{j=1}^{n} a_j(t)\varphi_j(\vec{x}) \tag{2.1}$$

associated with the unique spatially discretized solution $\boldsymbol{a}(t) := [a_1(t),\ldots,a_n(t)]^T \in \mathbb{R}^n$ that is continuous in time.

Standard FEM ansatz functions have only a small local support, which is the subdomain $\Omega_k \subsetneq \Omega$, where $\varphi_j(\vec{x}) \neq 0$. These small subdomains "*tile (or tessellate)* the domain" [56, p. 20] Ω such that

$$\bigcup_{k=1}^{n_\mathcal{T}} \overline{\Omega}_k = \overline{\Omega} \qquad \text{and} \qquad \Omega_\ell \cap \Omega_m = \varnothing,\ \forall \ell \neq m.$$

The set $\{\Omega_k\}_{k=1}^{n_\mathcal{T}}$ is called *triangulation* of dimension $n_\mathcal{T}$. Depending on the choice of ansatz functions, different finite elements can be built. In this thesis the piecewise linear elements $\mathcal{P}_1$ and the piecewise quadratic elements $\mathcal{P}_2$ are considered, where the ansatz functions are either linear or quadratic, respectively. Furthermore, the subdomains Ω_k are triangles for $d = 2$ and tetrahedra for $d = 3$. The corners of these elements are called nodes.

The linear elements $\mathcal{P}_1$ are uniquely defined by

$$\begin{aligned} b_1x_1 + b_2x_2 + b_3 &= 0, \quad \text{for } d = 2,\\ b_1x_1 + b_2x_2 + b_3x_3 + b_4 &= 0, \quad \text{for } d = 3, \end{aligned}$$

using three (for $d = 2$) or four (for $d = 3$) coefficients $b_i \in \mathbb{R}$, one for each node of the subdomain Ω_k. To define the quadratic elements $\mathcal{P}_2$, the midpoints of each edge are added as additional nodes, such that one ends up with six (for $d = 2$) or ten (for $d = 3$) coefficients $b_i \in \mathbb{R}$ to define the quadratic elements $\mathcal{P}_2$ via

$$\begin{aligned} b_1x_1^2 + b_2x_1x_2 + b_3x_2^2 + b_4x_1 + b_5x_2 + b_6 = 0,& \quad \text{for } d = 2,\\ \left.\begin{aligned} b_1x_1^2 + b_2x_1x_2 + b_3x_1x_3 + b_4x_2^2 + b_5x_2x_3\\ +b_6x_3^2 + b_7x_1 + b_8x_2 + b_9x_3 + b_{10} = 0, \end{aligned}\right.& \quad \text{for } d = 3. \end{aligned}$$

An example for a subdomain Ω_k and the corresponding nodes for $\mathcal{P}_1$ and $\mathcal{P}_2$ elements are depicted in Figure 2.1. More details about these specific finite elements as well as further finite elements can be found in, e.g., [56, Sec. 1.3].

Using the ansatz functions $\{\varphi_j(\vec{x})\}_{j=1}^{n}$ together with the triangulation $\{\Omega_k\}_{k=1}^{n_\mathcal{T}}$, the FEM can be applied to discretize the differential equation $\mathscr{F}(a(t,\vec{x})) = 0$, which yields the matrix equation

$$F\boldsymbol{a}(t) = 0, \quad \text{with } F \in \mathbb{R}^{n\times n},\ \boldsymbol{a}(t) \in \mathbb{R}^n.$$

The actual matrix assembling process for scalar functions $a(t,\vec{x}) \in \mathbb{R}$ and vector-valued functions $\vec{a}(t,\vec{x}) \in \mathbb{R}^d$ is not within the scope of this thesis. The interested reader is referred to [56, Sec. 1.3]. A more detailed analysis of the FEM is given in [6].

To improve the quality of the approximation (2.1), the dimension of the triangulation $n_\mathcal{T}$ needs to be increased, which naturally leads to a larger system dimension n. Instead of creating new triangulations with smaller subdomains Ω_k, the existing triangulation is usually modified by dividing the existing subdomains into smaller subdomains. This process is called *refinement* of the triangulation. One possible refinement strategy, which is used within this thesis, is the *bisection refinement* as introduced in [11]. This means in detail that the longest edge of the element Ω_k becomes the refinement edge, which is split into half by inserting a new node at its midpoint. New edges are inserted between this midpoint and the nodes that are not part of the refinement edge. For $d = 2$, this is one edge and for $d = 3$, these are two edges, respectively. The resulting smaller subdomains are schematically depicted in Figure 2.1 for two bisection refinement steps. Notice that the new nodes are not drawn into the picture for reasons of clarity. Two bisection refinement steps correspond to one uniform refinement step [11].

The $\mathcal{P}_1$ and $\mathcal{P}_2$ elements are used in Section 3.5 to discretize the different partial differential equations (PDEs) from Chapter 3. For PDEs that depend on different variables, one can consider different discretization schemes for each variable. Hence, so-called mix finite elements are used. One of these mixed finite elements, the $\mathcal{P}_2$–$\mathcal{P}_1$ *Taylor–Hood* element [78], is used in Section 3.5. The nodes of the Taylor–Hood element are chosen identically to the example depicted in Figure 2.1. The *Taylor–Hood* element is an *inf-sup*-stable finite element discretization and, therefore, suits for the problems considered in this thesis. This *inf-sub*-stability is crucial for certain matrix properties as discussed in more detail in Subsections 3.5f. In the 2-dimensional examples of this thesis, the system dimension is roughly of order $2 \cdot n_{\mathcal{T}}$ for $\mathcal{P}_2$ and of order $0.5 \cdot n_{\mathcal{T}}$ for $\mathcal{P}_1$ elements.

To refine the triangulations from Section 3.5, the bisection refinement is used in Subsection 4.4.1 to setup the parameters for the numerical experiments.

2.2. Efficient Computation of Various Matrix Quantities

Throughout this thesis, the efficient computation of various matrix quantities is essential. In this section, some definitions are provided and frequently used methods to compute matrix quantities are explained. Through this a description of the case of real matrices will be given.

The following definition introduces various eigenvalue and -vector corresponding quantities.

Definition 2.1 *For A, $M \in \mathbb{R}^{n\times n}$, the following definitions hold.*

(a) If $\boldsymbol{x} \in \mathbb{C}^n$, we consider the ***eigenvalue problem*** *(EVP)*

$$A\boldsymbol{x} = \lambda \boldsymbol{x}, \quad \boldsymbol{x} \neq 0, \tag{2.2}$$

where $\lambda \in \mathbb{C}$ is a scalar. If a scalar λ and a nonzero vector $\boldsymbol{x}$ happen to satisfy this equation, then λ is called eigenvalue of A and x is called an ***[right] eigenvector*** *of A associated with λ. Notice that the two occur inextricably as a pair, and that an eigenvector cannot be the zero vector. The pair $(\lambda, \boldsymbol{x}) \in \mathbb{C} \times \mathbb{C}^n$ is called eigenpair (cf. [79, Def. 1.1.2]).*

(b) In some applications one is also interested in the eigenpair of the adjoint problem

$$\boldsymbol{y}^H A = \lambda \boldsymbol{y}^H, \quad \boldsymbol{y} \neq 0,$$

where $\boldsymbol{y} \in \mathbb{C}^n$ is called ***left eigenvector*** *regarding the eigenvalue λ. An* ***eigentriplet*** *of A is defined as $(\lambda, \boldsymbol{x}, \boldsymbol{y}) \in \mathbb{C} \times \mathbb{C}^n \times \mathbb{C}^n$.*

(c) The ***generalized*** *EVP is defined as*

$$A\boldsymbol{x} = \lambda M \boldsymbol{x}, \quad \boldsymbol{x} \neq 0 \tag{2.3}$$

and (A, M) is called a ***matrix pencil****. For M non-singular, (2.3) is equivalent to $M^{-1}A\boldsymbol{x} = \lambda \boldsymbol{x}$ that is of the form (2.2).*

(d) For M singular, the matrix pencil (A, M) is called ***regular*** *if the so-called characteristic polynomial ζ defined by*

$$\zeta(\lambda) = \det(\lambda M - A)$$

is not the zero polynomial. A matrix pencil which is not regular is called ***singular*** *(cf. [83, Def. 2.5]).*

(e) *The set of all $\lambda \in \mathbb{C}$ that are eigenvalues of (A, M) is called the* ***spectrum*** *of (A, M) and is denoted by $\Lambda(A, M)$. [79, Def. 1.1.4]*

(f) *For $\epsilon > 0$ and the regular matrix pencil (A, M),*

$$\Lambda_\epsilon(A, M) = \Lambda_\epsilon(M^{-1}A) := \{\lambda \in \mathbb{C} : \left\|(\lambda M - A)^{-1}\right\| > \epsilon^{-1}\}$$

[126, eqs. (2.1),(45.8)] defines the ***ϵ-pseudospectrum****.*

(g) *The supremum*

$$\sigma(A, M) := \max\{\mathrm{Re}\,(\lambda) : \lambda \in \Lambda(A, M)\}$$

is called the ***spectral abscissa*** *of (A, M). [126, Chap. 14]*

(h) *For $\epsilon > 0$, the* ***ϵ-pseudospectral abscissa*** *is defined by*

$$\sigma_\epsilon(A, M) := \max\{\mathrm{Re}\,(\lambda) : \lambda \in \Lambda_\epsilon(A, M)\},$$

[126, Chap. 14].

(i) *A matrix A is said to be* ***normal*** *if*

$$A^H A = A A^H,$$

[79, Def. 2.5.1]

(j) *A regular matrix pencil (A, M) is a* ***normal matrix pair*** *if*

$$A^H M^{-1} A = A M^{-1} A^H,$$

[87, Sec. 1].

The efficient computation of eigenvalues and eigenvectors of large-scale matrices is itself a numerically challenging task. In Subsections 2.2.1 and 2.2.2, various methods are shown which avoid the handling of the large-scale matrices if possible.

In many applications, including the examples in this thesis, the ranks of some of the important matrices are low, such that these matrices can be written as low-rank decompositions.

Definition 2.2 (Rank, cf. [79, Def. 0.4.1]) *If $A \in \mathbb{R}^{m \times n}$, $\mathrm{rank}\,(A)$ is the largest number of columns of A that constitute a linearly independent set. This set of columns is not, of course, unique, but the cardinality (number of elements) of this set is unique. Hence, $\mathrm{rank}\left(A^T\right) = \mathrm{rank}\,(A)$. Therefore, $\mathrm{rank}\,(A)$ may equivalently be defined in terms of linearly independent rows. This is often phrased as “row rank = column rank”.*

Definition 2.3 (Real-valued low-rank product) *Considering a rectangular matrix $W \in \mathbb{R}^{n\times m}$ with $m \ll n$ and* $\operatorname{rank}(W) \leq m$*, the symmetric low-rank product is defined as*

$$WW^T \in \mathbb{R}^{n\times n} \tag{2.4}$$

with $\operatorname{rank}\left(WW^T\right) \leq m$*. The product of the form (2.4) is by construction symmetric positive-semidefinite (spsd). It is often referred to as outer product or dyadic product.*

The use of low-rank matrix products is essential to adapt and improve various numerical methods that are used in this thesis. For that reason, the norm and trace computations of real-valued low-rank products are examined in what follows.

Subtracting two low-rank products yields possibly indefinite matrices which cannot be written in the form of a low-rank product (2.4) anymore. Nevertheless, one can write such a difference in a factorized version which is essential for the following norm and trace computations; compare the statements in [32, 86].

Definition 2.4 (Real-valued indefinite low-rank product) *Considering rectangular matrices $W \in \mathbb{R}^{n\times m}$ and $K \in \mathbb{R}^{n\times p}$ with $m+p \ll n$,* $\operatorname{rank}(W) \leq m$*, and* $\operatorname{rank}(K) \leq p$*, the symmetric indefinite low-rank product is defined as*

$$WW^T - KK^T =: U\mathcal{D}U^T \in \mathbb{R}^{n\times n} \tag{2.5}$$

with $U := [W \,|\, K] \in \mathbb{R}^{n\times(m+p)}$, $\mathcal{D} := \left[\begin{smallmatrix} I_m & 0 \\ 0 & -I_p \end{smallmatrix}\right]$, and $\operatorname{rank}\left(U\mathcal{D}U^T\right) \leq m+p$*. The product of the form (2.5) is symmetric by construction.*

2.2.1. Norm of Low-Rank Products

This subsection starts with computing the spectral and Frobenius norms of an outer product (2.4) and shows how this can be reduced to computations involving only the small product $W^TW \in \mathbb{R}^{m\times m}$. Notice that W^TW is often also called *inner product*. Within this thesis, however, the inner product for matrices is defined differently in the next subsection.

Since WW^T is real and symmetric, the norms can be defined as follows:

$$||WW^T||_2 := \max\{|\lambda| : \lambda \in \Lambda(WW^T)\}, \tag{2.6a}$$

$$||WW^T||_F := \sqrt{\sum_{i=1}^{n} \lambda_i^2} \quad \text{with} \quad \lambda_i \in \Lambda(WW^T). \tag{2.6b}$$

The following theorem can be seen as a more general result that presents the relation of eigenvalues between commuting products.

Theorem 2.5 (cf. [74, Thm. 1.32]) *Let $A \in \mathbb{C}^{n\times m}$ and $B \in \mathbb{C}^{m\times n}$. The non-zero eigenvalues of AB are the same as for BA and have the same Jordan structure. [...] If $m \neq n$, then the larger (in dimension) of AB and BA has a zero eigenvalue of geometric multiplicity at least $|n-m|$.*

This implies the norm equivalence $||WW^T|| = ||W^TW||$ for the spectral and Frobenius norm, i.e.,

$$||WW^T||_2 = \max\{|\lambda| : \lambda \in \Lambda(W^TW)\} =: ||W^TW||_2, \tag{2.7a}$$

$$||WW^T||_F = \sqrt{\sum_{i=1}^{m} \lambda_i^2} =: ||W^TW||_F \quad \text{with} \quad \lambda_i \in \Lambda(W^TW). \tag{2.7b}$$

In various applications, the norm of a difference of outer products, as defined in (2.5), is considered. The norm of this symmetric matrix difference can be defined as

$$||WW^T - KK^T||_2 := \max\{|\lambda| : \lambda \in \Lambda(WW^T - KK^T)\},$$
$$||WW^T - KK^T||_F := \sqrt{\sum_{i=1}^{m} \lambda_i^2} \quad \text{with} \quad \lambda_i \in \Lambda(WW^T - KK^T).$$

Using (2.5) and Theorem 2.5, yields that the non-zero eigenvalues of $U^TU\mathcal{D} \in \mathbb{R}^{(m+p)\times(m+p)}$ (nonsymmetric, but spectrally equivalent to a symmetric matrix) are real and equal to the non-zero eigenvalues of (2.5). Hence,

$$||WW^T - KK^T||_2 = \max\{|\lambda| : \lambda \in \Lambda(U^TU\mathcal{D})\} \neq ||U^TU\mathcal{D}||_2, \tag{2.8a}$$

$$||WW^T - KK^T||_F = \sqrt{\sum_{i=1}^{m} \lambda_i^2} \neq ||U^TU\mathcal{D}||_F \quad \text{with} \quad \lambda_i \in \Lambda(U^TU\mathcal{D}). \tag{2.8b}$$

Within all algorithms, the specifications in (2.7) and (2.8) are used to efficiently compute norms of low-rank matrices involving its smaller products.

2.2.2. Trace of Low-Rank Products

Similar to the norm computation of low-rank products using Theorem 2.5, the computation of the trace can be significantly improved by exploiting low-rank structures.

Definition 2.6 *The trace of a quadratic matrix $A \in \mathbb{R}^{n\times n}$ is defined as*

$$\operatorname{tr}(A) := \sum_{i=1}^{n} a_{i,i} \ \textit{or} \ \operatorname{tr}(A) := \sum_{i=1}^{n} \lambda_i, \ \textit{with} \ \lambda_i \in \Lambda(A), \forall i = 1,\dots,n. \tag{2.9a}$$

For matrices $A, B \in \mathbb{R}^{n\times m}$ the inner product is defined as

$$\langle A, B\rangle := \sum_{i=1}^{n}\sum_{j=1}^{m} a_{i,j}b_{i,j} = \operatorname{tr}\left(A^TB\right). \tag{2.9b}$$

The following lemma states efficient computational methods for the trace of various combinations of positive-semidefinite and indefinite low-rank products.

Lemma 2.7 *Consider the low-rank matrices $W \in \mathbb{R}^{n\times m}$, $K \in \mathbb{R}^{n\times p}$, $\widehat{W} \in \mathbb{R}^{n\times \widehat{m}}$, $\widehat{K} \in \mathbb{R}^{n\times \widehat{p}}$ with $m+\widehat{m}+p+\widehat{p} \ll n$. Using these matrices and $U := \left[\widehat{W} \mid \widehat{K}\right] \in \mathbb{R}^{n\times(\widehat{m}+\widehat{p})}$, $\mathcal{D} := \left[\begin{smallmatrix} I_{\widehat{m}} & 0 \\ 0 & -I_{\widehat{p}} \end{smallmatrix}\right]$, the following equalities hold:*

$$\operatorname{tr}\left((WW^T)^2\right) = \sum_{j=1}^{m}\sum_{i=1}^{m}(W^TW)_{i,j}^2 \tag{2.10a}$$

$$\operatorname{tr}\left((U\mathcal{D}U^T)^2\right) = \sum_{j=1}^{\widehat{m}}\sum_{i=1}^{\widehat{m}}(\widehat{W}^T\widehat{W})_{i,j}^2 + \sum_{j=1}^{\widehat{p}}\sum_{i=1}^{\widehat{p}}(\widehat{K}^T\widehat{K})_{i,j}^2 - 2\sum_{j=1}^{\widehat{m}}\sum_{i=1}^{\widehat{p}}(\widehat{K}^T\widehat{W})_{i,j}^2 \tag{2.10b}$$

$$\operatorname{tr}\left(WW^TKK^T\right) = \sum_{j=1}^{m}\sum_{i=1}^{p}(K^TW)_{i,j}^2, \tag{2.10c}$$

$$\operatorname{tr}\left(U\mathcal{D}U^TWW^T\right) = \sum_{j=1}^{m}\sum_{i=1}^{\widehat{m}}(\widehat{W}^TW)_{i,j}^2 - \sum_{j=1}^{m}\sum_{i=1}^{\widehat{p}}(\widehat{K}^TW)_{i,j}^2 \tag{2.10d}$$

In other words, computing the trace of combinations of low-rank products of size $n \times n$ involves only the small products of the defining low-rank matrices, the component-by-component squaring of them, and its summation.

Proof. For $A, \widetilde{A} \in \mathbb{R}^{n\times n}$, the properties $\Lambda(A^2) = \{\lambda_i^2 : \lambda_i \in \Lambda(A), \forall i = 1,\dots,n\}$ (see, e.g., [74, Theorem 1.13]) and $\Lambda(A) = \Lambda(\widetilde{A}) \Rightarrow \operatorname{tr}(A) = \operatorname{tr}\left(\widetilde{A}\right)$ (compare (2.9a)) yield

$$\Lambda(A) = \Lambda(\widetilde{A}) \quad \Rightarrow \quad \Lambda(A^2) = \Lambda(\widetilde{A}^2) \quad \Rightarrow \quad \operatorname{tr}\left(A^2\right) = \operatorname{tr}\left(\widetilde{A}^2\right). \tag{2.11}$$

Furthermore, (2.9a) and the definitions of inner and outer products yield

$$\operatorname{tr}\left(A^T\widetilde{A}\right) = \operatorname{tr}\left(A\widetilde{A}^T\right) = \operatorname{tr}\left(\widetilde{A}^TA\right) = \operatorname{tr}\left(\widetilde{A}A^T\right) = \sum_{j=1}^{n}\sum_{i=1}^{n} a_{ij}\widetilde{a}_{ij}. \tag{2.12}$$

The proof of (2.10a) is as follows:

$$\begin{aligned}
\operatorname{tr}\left((WW^T)^2\right) &= \operatorname{tr}\left((W^TW)^2\right) && \text{(compare [74, Theorem 1.32] and (2.11))}\\
&= \operatorname{tr}\left((W^TW)(W^TW)\right) && \text{(notice that } W^TW \in \mathbb{R}^{m\times m}\text{)}\\
&= \sum_{j=1}^{m}\sum_{i=1}^{m}(W^TW)_{i,j}(W^TW)_{i,j} && \text{(see (2.12))}\\
&= \sum_{j=1}^{m}\sum_{i=1}^{m}(W^TW)_{i,j}^2.
\end{aligned}$$

To prove (2.10b) consider

$$\begin{aligned}
\operatorname{tr}\left((U\mathcal{D}U^T)^2\right) &= \operatorname{tr}\left((U^TU\mathcal{D})^2\right) \quad \text{(compare [74, Theorem 1.32], (2.11))}\\
&= \operatorname{tr}\left(\left(\begin{bmatrix}\widehat{W}^T\\ \widehat{K}^T\end{bmatrix}\begin{bmatrix}\widehat{W} \mid -\widehat{K}\end{bmatrix}\right)^2\right) \quad \text{(cf. (2.5))}\\
&= \operatorname{tr}\left(\left(\begin{bmatrix}\widehat{W}^T\widehat{W} & -\widehat{W}^T\widehat{K}\\ \widehat{K}^T\widehat{W} & -\widehat{K}^T\widehat{K}\end{bmatrix}\right)^2\right) \quad \text{(cf. (2.12))}\\
&= \sum_{j=1}^{\widehat{m}+\widehat{p}}\sum_{i=1}^{\widehat{m}+\widehat{p}}\begin{bmatrix}\widehat{W}^T\widehat{W} & -\widehat{W}^T\widehat{K}\\ \widehat{K}^T\widehat{W} & -\widehat{K}^T\widehat{K}\end{bmatrix}_{i,j}\begin{bmatrix}\widehat{W}^T\widehat{W} & \widehat{W}^T\widehat{K}\\ -\widehat{K}^T\widehat{W} & -\widehat{K}^T\widehat{K}\end{bmatrix}_{i,j}\\
&= \sum_{j=1}^{\widehat{m}}\sum_{i=1}^{\widehat{m}}(\widehat{W}^T\widehat{W})^2_{i,j} + \sum_{j=1}^{\widehat{p}}\sum_{i=1}^{\widehat{p}}(\widehat{K}^T\widehat{K})^2_{i,j} - 2\sum_{j=1}^{\widehat{m}}\sum_{i=1}^{\widehat{p}}(\widehat{K}^T\widehat{W})^2_{i,j}.
\end{aligned}$$

Similar, (2.10c) can be proven as follows:

$$\begin{aligned}
\operatorname{tr}\left(WW^TKK^T\right) &= \operatorname{tr}\left((WW^TK)K^T\right) && ((WW^TKK^T) \in \mathbb{R}^{n\times n})\\
&= \operatorname{tr}\left(K^T(WW^TK)\right) && \text{(see (2.12))}\\
&= \operatorname{tr}\left((K^TW)(K^TW)^T\right) && ((K^TW) \in \mathbb{R}^{p\times m})\\
&= \sum_{j=1}^{m}\sum_{i=1}^{p}(K^TW)^2_{i,j}.
\end{aligned}$$

Finally, the proof of (2.10d) is as follows:

$$\begin{aligned}
\operatorname{tr}\left(U\mathcal{D}U^TWW^T\right) &= \operatorname{tr}\left((U^TW)(W^TU\mathcal{D})\right)\\
&= \operatorname{tr}\left(\left(\begin{bmatrix}\widehat{W}^T\\ \widehat{K}^T\end{bmatrix}W\right)\left(W^T\begin{bmatrix}\widehat{W} \mid -\widehat{K}\end{bmatrix}\right)\right)\\
&= \operatorname{tr}\left(\begin{bmatrix}\widehat{W}^TW\\ \widehat{K}^TW\end{bmatrix}\begin{bmatrix}W^T\widehat{W} \mid -W^T\widehat{K}\end{bmatrix}\right)\\
&= \sum_{j=1}^{m}\sum_{i=1}^{\widehat{m}+\widehat{p}}\begin{bmatrix}\widehat{W}^TW\\ \widehat{K}^TW\end{bmatrix}_{i,j}\begin{bmatrix}\widehat{W}^TW\\ -\widehat{K}^TW\end{bmatrix}_{i,j}\\
&= \sum_{j=1}^{m}\sum_{i=1}^{\widehat{m}}(\widehat{W}^TW)^2_{i,j} - \sum_{j=1}^{m}\sum_{i=1}^{\widehat{p}}(\widehat{K}^TW)^2_{i,j}.
\end{aligned}$$

□

***Remark* 2.8** *Using* (2.6a), (2.9a), (2.9b), *and [74, Thm. 1.13], it is obvious that for a symmetric* $A \in \mathbb{R}^{n \times n}$ *it holds that*

$$\langle A, A \rangle = \operatorname{tr}\left(A^2\right) = \sum_{i=1}^{n} \lambda_i^2 = ||A||_F^2. \tag{2.13}$$

In this way, the equations (2.10a) *and* (2.10b) *also describe the squared Frobenius norms* $||WW^T||_F^2$ *and* $||U\mathcal{D}U^T||_F^2$ *or the inner products* $\langle WW^T, WW^T \rangle$ *and* $\langle U\mathcal{D}U^T, U\mathcal{D}U^T \rangle$*, respectively.*

The results established in Lemma 2.7 and Remark 2.8 are used in all algorithms that involve a trace or norm computation of various low-rank products. Ultimately, this is one of the key ingredients for the methods explained in Section 6.3.

2.2.3. Eigenpairs of Regular Matrix Pencils

Throughout this thesis, the computation or approximation of eigenpairs of the arising matrix pencils is needed. In this subsection, some results from Cliffe et al. [50] are pointed out. Therefore, the matrix pencil

$$\left(\underbrace{\begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}}_{\boldsymbol{A}}, \underbrace{\begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}}_{\boldsymbol{M}} \right)$$

is defined with the non-singular matrices $A, M \in \mathbb{R}^{n \times n}$, $\widehat{G} \in \mathbb{R}^{n \times m}$ with $\operatorname{rank}\left(\widehat{G}\right) = m$, $\boldsymbol{A}, \boldsymbol{M} \in \mathbb{R}^{N \times N}$, and $N = n+m$. These matrix pencils typically arise in the discretization of incompressible flow problems and are defined in more detail in Section 3.5. If the system dimension is too large for dense eigenvalue solvers, iterative methods that only involve matrix-vector products are of interest. Because the efficient handling of large-scale eigenvalue problems is not within the scope of this thesis, it is preferred to use standard methods such as the MATLAB function `eigs`, which is an implicitly restarting shift-and-invert Arnoldi method, see, e.g., [112]. Unfortunately, if one is interested in finite eigenvalues with large magnitude, `eigs` is not suitable in the case of regular matrix pencils with singular $\boldsymbol{M}$, which means the pencil has infinite eigenvalues.

Consider generalized EVP of the form

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{M}\boldsymbol{x}, \tag{2.14}$$

for which the following theorems hold. The first theorem determines the amount of finite and infinite eigenvalues.

Theorem 2.9 (cf. [50, Thm. 2.1.(a)]) *The EVP (2.14) has precisely* $n-m$ *finite eigenvalues, that are those of the reduced EVP of dimension* $n-m$

$$Q_2^T(A - \lambda M)Q_2\boldsymbol{z} = 0,$$

where $Q_2 \in \mathbb{R}^{n\times(n-m)}$ is an orthonormal basis for $\widehat{G}^{\perp}$, i.e., the orthogonal complement of span $\left\{\widehat{G}\right\}$.

Furthermore, the second theorem introduces a shifting technique that differently modifies finite and infinite eigenvalues.

Theorem 2.10 (cf. [50, Thm. 3.1]) *Choose δ_1, δ_2, $\gamma \in \mathbb{R}$. Denote the finite eigenvalues of (2.14) by λ_i, $i = 1, \dots, n-m$. Assume A, M are non-singular and*

$$\text{(i) } \delta_2 \neq 0, \qquad \text{(ii) } \delta_1\delta_2^{-1} \neq \lambda_i - \gamma. \tag{2.15}$$

Then the EVP

$$\begin{bmatrix} A-\gamma M & \delta_1\widehat{G} \\ \delta_1\widehat{G}^T & 0 \end{bmatrix} \boldsymbol{w} = \mu \begin{bmatrix} M & \delta_2\widehat{G} \\ \delta_2\widehat{G}^T & 0 \end{bmatrix} \boldsymbol{w} \tag{2.16}$$

has eigenvalues μ_i, $i = 1, \dots, n+m$ with

$$\begin{aligned} &\text{(a)} \quad \mu_i = \lambda_i - \gamma,\ i = 1, \dots, n-m, \\ &\text{(b)} \quad \mu_i = \delta_1\delta_2^{-1},\ i = n-m+1, \dots, n+m. \end{aligned}$$

Some correlations of the eigenvectors between the original EVP (2.14) and the transformed EVP (2.16) are described in the next lemma.

Lemma 2.11 (cf. [50, Lem. 3.2]) (a) *Let λ be a finite eigenvalue of EVP (2.14). Assume A, M non-singular and (2.15). If $\binom{\widehat{\boldsymbol{x}}}{\widehat{\boldsymbol{p}}}$, $\widehat{\boldsymbol{x}} \in \mathbb{R}^n$, $\widehat{\boldsymbol{p}} \in \mathbb{R}^m$ is an eigenvector of EVP (2.14) associated with λ, then $\binom{\boldsymbol{w}_1}{\boldsymbol{w}_2}$, where $\boldsymbol{w}_1 = \widehat{\boldsymbol{x}}$, $\boldsymbol{w}_2 = (\delta_1 - (\lambda-\gamma)\delta_2)^{-1}\widehat{\boldsymbol{p}}$ is the corresponding eigenvector of EVP (2.16).*

(b) *Let $\binom{0}{\widehat{\boldsymbol{p}}}$, $\widehat{\boldsymbol{p}} \in \mathbb{R}^m$ be an eigenvector corresponding to an infinite eigenvalue of EVP (2.14). Then $\binom{0}{\boldsymbol{w}_2}$ with $\boldsymbol{w}_2 = \widehat{\boldsymbol{p}}$ is an eigenvector of EVP (2.16) corresponding to the eigenvalue $\delta_1\delta_2^{-1}$.*

Using these results, the regular matrix pencil (2.14) can be shifted to a regular pencil with no infinite eigenvalues without changing the finite eigenvalues by choosing $\gamma = 0$ in (2.16). Since the pencil (2.16) does not have infinite eigenvalues, standard eigenvalue solvers can be applied to calculate eigenvalues and -vectors. Furthermore, Lemma 2.11 can be used to recover the eigenvectors of the original pencil (2.14). To compute eigenvalues with small magnitude, `eigs` can be applied directly to the regular pencil. This is used, for example, to calculate all finite and unstable eigenvalues with `eigs` as explained in Subsection 4.2.3. Knowing the exact number of unstable eigenvalues beforehand yields a sufficient stopping criterion for the computation. This approach is not the most robust method for unknown problem settings but it is sufficient in our case.

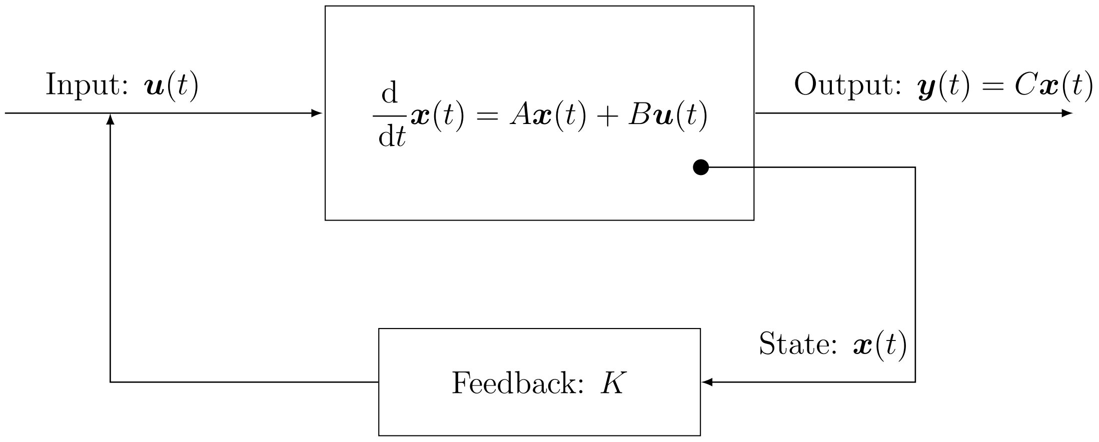

Input: $\boldsymbol{u}(t)$
$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t)$
Output: $\boldsymbol{y}(t) = C\boldsymbol{x}(t)$
State: $\boldsymbol{x}(t)$
Feedback: K

Figure 2.2.: Diagram of a linear closed-loop control system with state feedback.

2.3. Optimal Control and Matrix Equations

In this section the concept of optimal control for linear dynamical systems is introduced. After defining the basic framework for these optimal control problems and some necessary terms in the following subsection, important matrix equations are established in Subsection 2.3.2. These equations are essential to solve the linear-quadratic regulator (LQR) approach as explained in Subsection 2.3.3.

2.3.1. Basic Framework

Optimal control refers to methods that influence dynamic systems towards a desired behavior in an *optimal* way. Thereby, the optimality can describe a, for example, time- or cost-optimal process; the optimality is measured by a given cost functional (or performance index, cf. [91]) that evaluates the input, the output, and/or the state of the dynamical system. In what follows, linear dynamical systems are considered. The interested reader is referred to [91, Chap. 2] for some basic results of optimal control for nonlinear dynamical systems.

Linear dynamical systems are classified into *linear, time-varying* (LTV) and *linear, time-invariant* (LTI) systems depending on whether the matrices, which define the system, are time-varying or time-invariant. All statements of this thesis are related to LTI systems.

Using the internal description, see, e.g. [2, Sec. 4.2], a continuous-time and -invariant, linear dynamical system $\Phi(A, B, C)$ is defined by

$$\Phi: \qquad \begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t), &\quad (2.17a)\\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t) &\quad (2.17b)\end{aligned}$$

with the state variable $\boldsymbol{x}(t) \in \mathbb{R}^{n}$, the input function $\boldsymbol{u}(t) \in \mathbb{R}^{n_r}$, the output function $\boldsymbol{y}(t) \in \mathbb{R}^{n_a}$, the time $t \in [t_0, t_f]$, the system matrix $A \in \mathbb{R}^{n\times n}$, the input operator

$B \in \mathbb{R}^{n \times n_r}$, and the output operator $C \in \mathbb{R}^{n_a \times n}$. Thereby, continuous-time means that the state $\boldsymbol{x}(t)$ is continuous in time and time-invariant implies that the matrices A, B, C are constant over time. Notice that we omit a second term $D\boldsymbol{u}(t)$ in (2.17b), which usually describes the direct influence of $\boldsymbol{u}(t)$ on the output $\boldsymbol{y}(t)$, in our considerations (cf. [2, eq. (4.12)]). Figure 2.2 depicts schematically the LTI system (2.17).

Various possible system properties of Φ are defined in what follows.

Definition 2.12 (System properties of Φ) *For the linear dynamical system (2.17), the following definitions hold.*

(a) The order of the system is defined as the dimension of the associated state-space (cf. [2, Def. 4.2.(a)]):

$$\dim(\Phi) = n.$$

*(b) Φ is called **stable** if its poles have nonpositive real parts.*

*(c) The matrix A is **Hurwitz** or **(asymptotically) stable** if all its eigenvalues have strictly negative real part (cf. [77, p. 80-7]).*

Some further system properties can be checked using the following conclusions.

Conclusion 2.13 (Extended system properties of Φ) *Consider the linear dynamical system (2.17).*

*(a) The pair (A, B) is **controllable** if and only if*

$$\operatorname{rank}\left(\begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix}\right) = n, \tag{2.18}$$

i.e., when the rows of this $n \times n_r n$ matrix are linearly independent (cf. [85, eq. (4.1.3)] and Def. 2.2).

*(b) The pair (C, A) is **observable** if and only if (A^T, C^T) is controllable (cf. [85, Prop. 4.2.2]).*

*(c) A pair (A, B) is called **stabilizable** if there exists a (feedback) matrix $K \in \mathbb{R}^{n \times n_r}$ such that $A - BK^T$ is asymptotically stable. A pair (C, A) is said to be **detectable** if and only if (A^T, C^T) is stabilizable (cf. [85, pp. 90,91]).*

***Remark* 2.14** *"The notion **stabilizability** takes two forms originating with the theory of continuous (differential) systems and discrete (differential) systems. In these two cases, an $n \times n$ matrix is said to be **stable** according as its eigenvalues are all in the open left half-plane, or in the unit disc, respectively." [85, p. 90]*

*Since only continuous-time systems are considered within this thesis, the word **stable** is always used in the context of **asymptotically stable** as in part (c) of Def. 2.12.*

A more feasible test for stabilizability and detectability is given by the *Hautus-Popov* test.

Theorem 2.15 (Hautus-Popov Test [77, Sec. 80.3])

- *For a given LTI system, the following are equivalent:*
  - *(a) The LTI system (2.17) is stabilizable, i.e., $\exists K \in \mathbb{R}^{n \times n_r}$ such that $A - BK^T$ is Hurwitz.*
  - *(b)* ***(Hautus-Popov test)*** *If $\boldsymbol{x} \neq 0$, $\boldsymbol{x}^H A = \lambda \boldsymbol{x}^H$, and $\mathrm{Re}\,(\lambda) \geq 0$, then $\boldsymbol{x}^H B \neq 0$.*
  - *(c) $\mathrm{rank}\,([A - \lambda I \mid B]) = n$, $\forall \lambda \in \mathbb{C}$ with $\mathrm{Re}\,(\lambda) \geq 0$.*
- *For a given LTI system, the following are equivalent:*
  - *(a) The LTI system (2.17) is detectable.*
  - *(b) The matrix pair (A^T, C^T) defines a stabilizable system.*
  - *(c)* ***(Hautus-Popov test)*** *If $\boldsymbol{x} \neq 0$, $A\boldsymbol{x} = \lambda \boldsymbol{x}$, and $\mathrm{Re}\,(\lambda) \geq 0$, then $C\boldsymbol{x} \neq 0$.*
  - *(d) $\mathrm{rank}\left(\left[\begin{smallmatrix} \lambda I - A \\ C \end{smallmatrix}\right]\right) = n$, $\forall \lambda \in \mathbb{C}$ with $\mathrm{Re}\,(\lambda) \geq 0$.*

In thesis one is interested in an optimal control setup that uses an feedback control approach. Thus, the major task is to determine a feedback K for an unstable system Φ such that $A - BK^T$ is stable. The following lemma states an important assumption for the existence of such a feedback.

Lemma 2.16 (cf. [85, Lem. 4.5.4]) *If $N \succeq 0$ and (A, N) is stabilizable, then there is an $X \succeq 0$ such that $A - NX$ is stable.*

Throughout common literature the notation of the matrix pairs within the properties in Def. 2.12 and Concl. 2.13 varies. In the following proposition, the equivalence of both notations is shown.

Proposition 2.17 *For A, B, C as in (2.17), it holds that (A, B) is stabilizable if and only if (A, BB^T) is stabilizable and (C, A) is detectable if and only if $(C^T C, A)$ is detectable.*

Proof. If (A, BB^T) is stabilizable, then there exists a $X \in \mathbb{R}^{n \times n}$ such that the eigenvalues of $A - BB^T X$ are in $\mathbb{C}^-$. Hence, the eigenvalues of $A - BK^T$ with $K = XB$ are in the negative half-plane $\mathbb{C}^-$ and (A, B) is stabilizable.

If (A, B) is stabilizable, then there exists a $K \in \mathbb{R}^{n \times n_r}$ such that the eigenvalues of $A - BK^T$ are in $\mathbb{C}^-$. Let π be the orthogonal projection onto the null-space of B. Since $BK^T = B\pi K^T + B(I - \pi)K^T = B(I - \pi)K^T$ and $I - \pi$ is the orthogonal projection onto $\mathrm{null}\,(B)^\perp = \mathrm{range}\left(B^T\right)$, it holds that $\mathrm{range}\left((I - \pi)K^T\right) \subset \mathrm{range}\left(B^T\right)$. Therefore, there exists a $X \in \mathbb{R}^{n \times n}$ such that $(I - \pi)K^T = B^T X$. In detail, if e_i is the i-th unit vector, then the i-th column $(I - \pi)K^T e_i$ is a linear combination $B^T x_i$ of the columns of B^T, i.e., $(I - \pi)K^T e_i = B^T x_i$. Hence, the vectors x_i are the columns of X and the eigenvalues of $A - BB^T X = A - B(I - \pi)K^T = A - BK^T$ are in $\mathbb{C}^-$ which means that (A, BB^T) is stabilizable.

The proof for detectability follows by definition.

□

Before the specific optimal control problem dealt with in this thesis can be stated, some remarks about matrix equations are given in the next subsection.

2.3.2. Linear and Quadratic Matrix Equations

The second subsection introduces linear and quadratic matrix equations, which are used within this thesis.

In general, the subject of linear algebra studies various solution strategies for scalar functions $0 = f(x) \in \mathbb{K}$ or systems of equations $0 = \mathscr{F}(\boldsymbol{x}) \in \mathbb{K}^q$, where the solution is either a scalar $x \in \mathbb{K}$ or a vector $\boldsymbol{x} \in \mathbb{K}^n$, respectively, with $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$. Moreover, one mainly distinguishes between linear and nonlinear equations.

Scalar linear equations are commonly denoted by $ax = b$ and linear systems of equations by $A\boldsymbol{x} = \boldsymbol{b}$ with $a, b, x \in \mathbb{K}$, $\boldsymbol{b} \in \mathbb{K}^q$, $\boldsymbol{x} \in \mathbb{K}^n$, and $A \in \mathbb{K}^{q \times n}$. In contrast, nonlinear functions $f(.)$ or $\mathscr{F}(.)$ may contain combinations of more complicated terms such as, for example, power functions, exponential functions, logarithms, or trigonometric functions. In principal, nonlinear equations are harder to solve than linear equations. Moreover, analytical solutions are only available for selected nonlinear equations such that some nonlinear equations can only be solved approximately.

If the unknown variable is no longer a scalar or a vector but a matrix $X \in \mathbb{K}^{n \times q}$, one refers to matrix functions with $n \cdot q$ unknowns. For the remainder of this subsection, the notation is fixed to the set of real numbers $\mathbb{K} = \mathbb{R}$. As mentioned above, only continuous-time systems are considered and, hence, only continuous-time matrix equations are introduced. More details about their discrete counterparts can be found in, e.g., [84].

Lyapunov Equations

The first class of matrix equations that are considered are linear in the unknown X. Given the matrices $F, O \in \mathbb{R}^{n \times n}$, $P, E \in \mathbb{R}^{q \times q}$, and $H \in \mathbb{R}^{n \times q}$, the generalized *Sylvester* equation is defined as

$$FXE + OXP = -H$$

with $X \in \mathbb{R}^{n \times q}$. An overview about this kind of equations as well as various solution strategies and methods can be found in [120]. During this work, a more special type of equation is considered. For $n = q$, $P = F^T$, $O = E = I_n$, and $H = H^T$, the *continuous-time algebraic Lyapunov* equation (CALE) is specified as

$$FX + XF^T = -H. \tag{2.19}$$

This linear matrix equation is symmetric if a symmetric solution $X = X^T \in \mathbb{R}^{n\times n}$ is assumed. The solution X is unique if $\Lambda(F) \subset \mathbb{C}^-$, i.e., F is stable, and can be defined by [85, eq. (5.3.3)]

$$X = \int_0^\infty e^{Ft} H e^{F^T t}\,\mathrm{d}t.$$

The following theorem defines properties of the solution X depending on the right-hand side H.

Theorem 2.18 (cf. [85, Thm. 5.3.1]) *Assume that all eigenvalues of F lie in the open left half-plane and let X be the unique solution of (2.19). Then: (a) If $H \succ 0$ then $X \succ 0$ and, if $H \succeq 0$, then $X \succeq 0$. (b) Moreover, if $H \succeq WW^T$, where (F, W) is a controllable pair, then we have $X \succ 0$.*

Relaxing the controllability property gives the following theorem.

Theorem 2.19 (cf. [85, Thm. 5.3.4]) *Let (F, W) be a stabilizable pair with $F \in \mathbb{R}^{n\times n}$, $W \in \mathbb{R}^{n\times m}$ and suppose $H \succeq WW^T$. If the equation $FX + XF^T = -H$ has a solution $X \succeq 0$, then F has all its eigenvalues in the open left half-plane.*

In many applications, the right hand side H of the Lyapunov equation (2.19) has a low numerical rank and can be written as $H = WW^T$, as introduced in Theorem 2.19, with $W \in \mathbb{R}^{n\times r}$, $\operatorname{rank}(W) \le r$, and $r \ll n$. This low-rank right hand side leads to specially tailored solution strategies; one strategy, the *alternating directions implicit* (ADI) method [31, 88, 89], is explained in Subsection 2.4.2. Further detailed information about the numerical solution of large-scale CALEs can be found in, e.g., [103] for symmetric F, [3, 88, 89, 121] for non-symmetric F, and in [32, 65, 84] for the more general Sylvester equation.

Riccati Equations

The second class of matrix equations that are considered in this thesis are quadratic in the unknown X. Extending the Lyapunov equation (2.19) by a symmetric, quadratic term and setting $A = F^T$ yields the *continuous-time algebraic Riccati* equation (CARE)

$$\mathcal{R}(X) = H + A^T X + XA - XNX = 0 \tag{2.20}$$

with $N = N^T \in \mathbb{R}^{n\times n}$. A noticeably detailed analysis of CAREs can be found in [85]. Some statements from [85] that are relevant for the presentations of this thesis are given in what follows. Notice that in [85], the notation switches within the chapters. In the Riccati Chapters 7.1-7.8 and 8.1-8.4 the matrix A is $-A$ in our notation. In the Chapters 7.9, 7.10, 8.5, 8.6, and 9 the matrix A is used in our notation. Furthermore, the Riccati equations in [85] have switched signs such that $\mathcal{R}(X)$ is $-\mathcal{R}(X)$ in our notation. For better readability, we adapted all results to our notation.

Following the derivation in [85, Sec. 7.1/8.1], the Hamiltonian matrix Ξ is defined as

$$\Xi := \begin{bmatrix} A & -N \\ -H & -A^T \end{bmatrix} \in \mathbb{R}^{2n\times 2n} \tag{2.21}$$

and the n-dimensional subspace

$$\Psi(X) = \operatorname{Im}\left(\begin{bmatrix} I_n \\ X \end{bmatrix}\right) \subset \mathbb{R}^{2n} \tag{2.22}$$

is called a *graph* of $X \in \mathbb{R}^{n\times n}$. The *graph* $\Psi(X)$ is directly related to the solution of the CARE (2.20) as described in the following proposition.

Proposition 2.20 (real version of [85, Prop. 7.1.1]) *For a real $n \times n$ matrix X, the graph of X is Ξ-invariant if and only if X is a solution of (2.20).*

Proof. If $\Psi(X)$ is Ξ-invariant, then

$$\begin{bmatrix} A & -N \\ -H & -A^T \end{bmatrix}\begin{bmatrix} I_n \\ X \end{bmatrix} = \begin{bmatrix} I_n \\ X \end{bmatrix}\widetilde{P} \tag{2.23}$$

for a suitable matrix $\widetilde{P}$. The first row in this equality gives $\widetilde{P} = A - NX$, and the second block row gives

$$-H - A^T X = X(A - NX).$$

In other words, X solves (2.20). Conversely, if X solves (2.20), then (2.23) holds with $\widetilde{P} = A - NX$.

□

In optimal control, one is interested in solutions X such that $\widetilde{P} = A - NX$ is stable. If one finds a Ξ-invariant subspace such that

$$\begin{bmatrix} A & -N \\ -H & -A^T \end{bmatrix}\begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix}\widetilde{P}$$

with U non-singular and $\widetilde{P} \in \mathbb{R}^{n\times n}$ stable, a solution $X \in \mathbb{R}^{n\times n}$ of (2.20) can be obtained as

$$X := VU^{-1}.$$

The following theorems describe under which conditions such a solution exists. Notice that all formulations are restricted to the real-valued case.

Theorem 2.21 (cf. [85, Thm. 7.9.3]) *If $N \succeq 0$, $H = H^T$, (A, N) is stabilizable, and there exists a symmetric solution of (2.20), then there is a maximal symmetric solution, $X^{(*)}$. Moreover, $X^{(*)}$ coincides with the unique solution of (2.20) for which $\Lambda(A - NX^{(*)})$ lies in the closed left half-plane.*

Theorem 2.22 (cf. [85, Thm. 7.9.4]) *Suppose that $N \succeq 0$, $H = H^T$, (A, N) is stabilizable, and there is a symmetric solution of (2.20). Then for the maximal symmetric solution $X^{(*)}$ of (2.20), $A - NX^{(*)}$ is stable if and only if the matrix (2.21) has no eigenvalues on the imaginary axis.*

***Remark* 2.23 (cf. [85, p.196])** *Observe that from [85, Thm. 7.2.8], the matrix (2.21) has no eigenvalues on the imaginary axis if, in addition, we have $H \succeq 0$ and (H, A) is detectable.*

Theorem 2.24 (cf. [85, Thm. 8.5.1]) *Assume that $N \succeq 0$, $H = H^T$, (A, N) is stabilizable and that (2.20) has a real symmetric solution. Then there exists a maximal real symmetric solution, which can be characterized as the unique stabilizing solution.*

Theorem 2.25 (cf. [85, Thm. 9.1.2]) *If $N \succeq 0$, $H \succeq 0$ and the pair (A, N) is stabilizable, then there exist symmetric solutions of $\mathcal{R}(X) = 0$. Moreover, the maximal symmetric solution $X^{(*)}$ (which exists by [85, Thm. 7.9.1]) also satisfies $X^{(*)} \succeq 0$. If, in addition, (H, A) is detectable, then $A - NX^{(*)}$ is stable.*

The interesting case in optimal control is whenever the number of in- and outputs in (2.17) is limited. In detail, only a few outputs $\boldsymbol{y} = C\boldsymbol{x} \in \mathbb{R}^{n_a}$ with $n_a \ll n$ can be observed and the control, which acts on the states $\boldsymbol{x}$ via $B\boldsymbol{u} \in \mathbb{R}^n$, is given by $\boldsymbol{u} \in \mathbb{R}^{n_r}$ with $n_r \ll n$. Thus, the constant term and the middle part of the quadratic term in (2.20) can be written as low-rank products $N = BB^T$ and $H = C^TC$ that are spsd by definition. Using these low-rank products, Theorem 2.25 can be extended to the following theorem.

Theorem 2.26 (extended version of [26, Thm. 3]) *Assuming (A, B) is stabilizable and (C, A) is detectable (cf. Theorem 2.25 with $N = BB^T \succeq 0$ and $H = C^TC \succeq 0$), every spsd solution $X^{(*)} = (X^{(*)})^T \succeq 0$ of the CARE (2.20) is stabilizing.*

Proof. Since the assumptions of Theorem 2.25 are fulfilled, the CARE (2.20) has a spsd solution that stabilizes.

Let $X = X^T \succeq 0$ solve the CARE (2.20). We show that $A - BB^TX$ is stable by contradiction.

Assume that μ is an eigenvalue of $A - BB^TX$ with $\mathrm{Re}\,(\mu) > 0$ and let $\boldsymbol{x} \in \mathbb{C}^n \backslash \{0\}$ be a corresponding eigenvector. The CARE (2.20) can be written as

$$(A - BB^TX)^TX + X(A - BB^TX) = -C^TC - XBB^TX. \tag{2.24}$$

Multiply (2.24) with $\boldsymbol{x}^H$ from left and $\boldsymbol{x}$ from the right. The left-hand side of (2.24) yields

$$2\,\mathrm{Re}\,(\mu)\, \boldsymbol{x}^H X \boldsymbol{x} \geq 0, \text{ since } X = X^T \succeq 0,$$

and the right-hand side of (2.24) yields

$$-\boldsymbol{x}^H C^TC\boldsymbol{x} - \boldsymbol{x}^H XBB^TX\boldsymbol{x} \leq 0, \text{ since } C^TC \succeq 0 \text{ and } XBB^TX \succeq 0.$$

Hence, left- and right-hand sides of (2.24) multiplied by $\boldsymbol{x}^H$ from left and $\boldsymbol{x}$ from the right are equal to zero that is $\boldsymbol{x}^H X \boldsymbol{x} = 0$ and $\boldsymbol{x}^H C^T C \boldsymbol{x} + \boldsymbol{x}^H X B B^T X \boldsymbol{x} = 0$ which yields $C\boldsymbol{x} = 0$ and $B^T X \boldsymbol{x} = 0$. Since $(A - BB^T X)\boldsymbol{x} = \mu \boldsymbol{x}$, $\boldsymbol{x}$ is an eigenvector of A with eigenvalue μ and $\operatorname{Re}(\mu) > 0$. Thus, $C\boldsymbol{x} = 0$ contradicts the assumption that (C, A) is detectable (compare Theorem 2.15). This means that $\mu \in \mathbb{C}^+$ cannot be an eigenvalue of $A - BB^T X$.

Since $\Lambda(A - BB^T X) \subset \Lambda(\Xi) \subset \mathbb{C} \backslash i\mathbb{R}$ (compare Theorem 2.22), all eigenvalues of $A - BB^T X$ are in the open left half-plane $\mathbb{C}^-$ and X is stabilizing.

□

The latter theorem is important in a way that an algorithm, which provides spsd iterates $X^{(k)}$ by construction, converges, if at all, towards the unique stabilizing solution $X^{(*)}$. This fact plays an important role in the convergence proof of the novel methods proposed in Section 6.

The numerical solution of CAREs is a widespread research area. For small dimensions, various solution approaches for dense matrices are available. In the large-scale and sparse case, specially tailored methods need to be considered that make use of certain low-rank structures in the constant and quadratic term. An overview of the different methods to solve the CARE (2.20) is given in [31, Sec. 3], [24, Chap. 4], and the references therein. For more details about the CARE and further Riccati type equations the interested reader is referred to, e.g., [43, 85].

The solution strategy used within this thesis is presented in Section 4.3.

Bernoulli Equation

Considering the constant term in (2.20) to be $H = 0$, a special case of the CARE (2.20) is defined by the so-called *algebraic Bernoulli equation* (ABE)

$$A^T X + XA - XNX = 0. \tag{2.25}$$

This equation plays a certain role in various applications of control or stabilization methods for linear systems.

Being a special case of a CARE, one can apply the same ideas as in Proposition 2.20. Therefore, consider the Hamiltonian matrix Ξ_B defined as

$$\Xi_B := \begin{bmatrix} A & -N \\ 0 & -A^T \end{bmatrix} \in \mathbb{R}^{2n \times 2n}. \tag{2.26}$$

Similar to the CARE (2.20), one is interested in a stabilizing solutions X_+ of (2.25) such that $A - NX_+$ is stable. Being a homogeneous version of (2.20), the trivial solution $X = 0$ is always a solution. Nevertheless, for unstable A this is not a solution of interest. The existence of a stabilizing solution is provided by the following proposition.

Proposition 2.27 (cf. [20, Prop. 1, with $\boldsymbol{E = I_n}$])
If (A, N) is stabilizable and $\Lambda(A) \cap j\mathbb{R} = \emptyset$, then the ABE (2.25) has a unique stabilizing

positive semidefinite solution X_+. Moreover, $\operatorname{rank}(X_+) = \mu$*, where μ is the number of eigenvalues of A in $\mathbb{C}^+$.*

Considering the real-valued block upper triangular structure of (2.26), it is obvious that

$$\Lambda(\Xi_B) = \Lambda(A) \cup \Lambda(-A).$$

Hence, for the stabilizing solution X_+ it holds that

$$\Lambda(A - NX_+) = \Lambda_s(A) \cup \{\lambda = -a - \jmath b : a + \jmath b \in \Lambda_{us}(A)\} \tag{2.27}$$

with $\Lambda_s(A)$ and $\Lambda_{us}(A)$ being the stable and unstable eigenvalues of A, respectively; see, e.g., [1, eq. (19)]. In other words, the stabilizing solution X_+ mirrors all unstable eigenvalues of A at the imaginary axis $\jmath\mathbb{R}$. Further information about the numerical solution of large-scale ABEs for the low-rank case $N = BB^T$ can be found in, e.g., [1, 19–21]. How the stabilizing solution of the generalized version of the ABE (2.25) is used within this thesis is shown in Subsection 4.2.3.

In the next subsection, the specific optimal control approach that is used in this thesis is explained and the connection between the solution X of (2.20) and the aim of finding the optimal control $\boldsymbol{u}(t)$ is described.

2.3.3. Linear-Quadratic Regulator (LQR) Approach

Considering an optimal control problem, where a quadratic cost functional is meant to get minimized subject to a linear dynamical system, one speaks about a *linear-quadratic* (LQ) problem; see, e.g., [91, Chap. 3]. Besides the distinction between LTV and LTI systems, the time horizon to achieve the control goal is important to classify the considered LQ control problem. Using an infinite time horizon, "[can] by no means [...] be viewed as a trivial extension of the problem over a finite horizon" ([91, p. 39]).

In this thesis, LTI systems over an infinite time horizon are considered. This class of optimal control problems is usually referred to as *linear-quadratic regulator* (LQR) problems (cf. [91, Sec. 3.4]).

Definition 2.28 (LQR problem (cf. [91, Prob. 3.3])) *For the time-invariant system Φ (2.17), where $\boldsymbol{x}(0) = \boldsymbol{x}_0$ is given, find a control $\boldsymbol{u}(t)$ that minimizes the cost functional*

$$\mathcal{J}(\boldsymbol{u}(t), \boldsymbol{x}(t)) = \int_0^\infty \boldsymbol{x}(t)^T Q \boldsymbol{x}(t) + \boldsymbol{u}(t)^T R \boldsymbol{u}(t)\, dt. \tag{2.28a}$$

The final state is unconstrained and $Q = Q^T \succeq 0$, $R = R^T \succ 0$.

Notice that the initial time is set to $t = 0$. This can be done without loss of generality due to the time-invariance of the matrices A, B, C, Q, R; see, e.g., [91, Sec. 3.4]. Furthermore, it is important to mention that (2.28a) can only attain a finite value if

$$\lim_{t\to\infty} Q\boldsymbol{x}(t) = 0 \qquad \text{and} \qquad \lim_{t\to\infty} \boldsymbol{u}(t) = 0.$$

This means that for $t \to \infty$, the Q-weighted state and the control need to approach zero, although this might not be the desired state $\boldsymbol{x}_d(t)$ one wants to achieve. In this case one redefines the state as the difference between the *actual* state $\boldsymbol{x}_a(t)$ and the *desired* state $\boldsymbol{x}_d(t)$. If the difference $\boldsymbol{x}(t) := \boldsymbol{x}_a(t) - \boldsymbol{x}_d(t)$ goes to zero, the actual state coincides with the desired state. This approach is often called *control in the neighborhood of an equilibrium point*, see, e.g., [91, Rem. 3.11], where the desired state $\boldsymbol{x}_d(t)$ is the equilibrium point that is usually stationary but possibly unstable. Notice that for nonlinear systems this point often serves as linearization point such that one tries to eliminate small deviations from this equilibrium point.

The solution to the LQR problem in Definition 2.28 is described by the following theorem.

Theorem 2.29 (cf. [22, Thm. 2.7]) *If $Q \succeq 0$, $R \succ 0$, (A, B) is stabilizable, and (C, A) is detectable, then the LQR problem in Definition 2.28 has an unique solution given by*

$$\boldsymbol{u}_*(t) = -\underbrace{R^{-1}B^TX}_{=:K^T}\boldsymbol{x}_*(t), \tag{2.28b}$$

where $X = X^T \succeq 0$ is the unique stabilizing solution of the CARE

$$\mathcal{R}(X) = Q + A^TX + XA - XBR^{-1}B^TX = 0. \tag{2.28c}$$

Furthermore, the optimal value of the cost functional is

$$\mathcal{J}_*(\boldsymbol{x}_0) = \frac{1}{2}\boldsymbol{x}_0^TX\boldsymbol{x}_0. \tag{2.28d}$$

In the remainder of this thesis, the regularization matrix R is set, without loss of generality, to $R = I_{n_r}$ and $Q := \alpha^2 C^TC$ with $\alpha \in \mathbb{R}^+$ as output weighting. During the theoretical derivations $\alpha = 1$ is assumed to simplify the notation. Nevertheless, α plays an important role during the numerical examples.

Since in this thesis the dynamical systems of interest arise from a FEM discretization, one has to deal with generalized forms of the dynamical systems and, thus, of the arising matrix equations as introduced in the next subsection.

2.3.4. Generalized Matrix Equations

All above introduced matrix equations exist in a generalized form. Considering the dynamical system $\widehat{\Phi}(A, B, C; M)$ of the form

$$\widehat{\Phi}: \quad \begin{aligned} M\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t), && (2.29a)\\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t), && (2.29b)\end{aligned}$$

one either speaks of a *generalized state-space system* if M is non-singular, or $\widehat{\Phi}$ describes a *descriptor system* defined by a system of *differential-algebraic equations* (DAE) [83] if M is singular.

In the case of M invertible, Definition 2.12 can be extended straightforwardly to the system $\widehat{\Phi}(A, B, C; M)$ and its corresponding matrix triples $(A, B; M)$ and $(C, A; M)$. Therefore, the following two transformations from [75, eq. (1.2)] are considered. The first version performs the transformation by a *left multiplication* that yields

$$\widehat{\Phi}(A, B, C; M) \mapsto \Phi(M^{-1}A, M^{-1}B, C), \tag{2.30}$$

where the state variable $\boldsymbol{x}(t)$ remains unchanged. Another variant is a *state change of coordinates* which is done by defining $\hat{\boldsymbol{x}}(t) = M\boldsymbol{x}(t)$ such that the dynamical system in $\hat{\boldsymbol{x}}(t)$ is of the form

$$\widehat{\Phi}(A, B, C; M) \overset{\boldsymbol{x}=M^{-1}\hat{\boldsymbol{x}}}{\longmapsto} \Phi(AM^{-1}, B, CM^{-1}). \tag{2.31}$$

The choice of transformation varies depending on the problem description. In the end, one hardly ever explicitly carries out these transformations. On the one hand, both transformations are invertible such that the solution will be the same after an inverse transformation and the transformations are only used to theoretically derive certain methods based on existing approaches. The much more important fact, on the other hand, is that one might not be able or interested in building $M^{-1}A$ or AM^{-1} explicitly, since the condition numbers might grow drastically and for a large-scale sparse matrix M the products $M^{-1}A$ or AM^{-1} become dense.

Nevertheless, these transformations are very useful for clarity of the used notation as shown for two examples below.

Considering a dynamical system $\Phi(\widehat{F}, \widehat{W}, .) = \Phi(M^{-1}F, M^{-1}W, .)$, a CALE of the form (2.19) with a low-rank right-hand side $H = \widehat{W}\widehat{W}^T$ is defined via

$$\begin{aligned} \widehat{F}X + X\widehat{F}^T &= -\widehat{W}\widehat{W}^T, \\ M^{-1}FX + XF^TM^{-T} &= -M^{-1}WW^TM^{-T}. \end{aligned}$$

Multiplying the latter equation from the left with M and with M^T from the right yields the *generalized continuous-time algebraic Lyapunov* equation (GCALE)

$$FXM^T + MXF^T = -WW^T, \tag{2.32}$$

which corresponds to the generalized dynamical system $\widehat{\Phi}(F, W, .; M)$ using the *left multiplication* transformation from (2.30). More details regarding the efficient solution of GCALEs can be found in, e.g, [33, 84, 102].

The *state change of coordinates* transformation can be used to apply the LQR problem (2.28) to the generalized dynamical system (2.29). Therefore, the dynamical system $\Phi(\widehat{A}, B, \widehat{C}) = \Phi(AM^{-1}, B, CM^{-1})$ for $\hat{\boldsymbol{x}} = M\boldsymbol{x}$ as in (2.31) is considered and a CARE of the form (2.28c) is defined via

$$\begin{aligned} \widehat{C}^T\widehat{C} + \widehat{A}^TX + X\widehat{A} - XBB^TX &= 0, \\ M^{-T}C^TCM^{-1} + M^{-T}A^TX + XAM^{-1} - XBB^TX &= 0. \end{aligned}$$

Multiplying with M^T from the left and M from the right yields the *generalized continuous-time algebraic Riccati* equation (GCARE)

$$\mathcal{R}(X) = C^TC + A^TXM + M^TXA - M^TXBB^TXM = 0, \tag{2.33a}$$

corresponding to the LQR problem for the generalized dynamical system $\widehat{\Phi}(A, B, C; M)$ and the optimal control

$$\boldsymbol{u}_*(t) = -B^TX\hat{\boldsymbol{x}}(t) = -\underbrace{B^TXM}_{=:K^T}\boldsymbol{x}(t) \tag{2.33b}$$

that minimizes the cost functional

$$\begin{aligned}\mathcal{J}(\boldsymbol{u}(t), \hat{\boldsymbol{x}}(t)) &= \int_0^\infty \hat{\boldsymbol{x}}(t)^T\widehat{C}^T\widehat{C}\hat{\boldsymbol{x}}(t) + \boldsymbol{u}(t)^T\boldsymbol{u}(t)\,\mathrm{d}t\\ &= \int_0^\infty \boldsymbol{x}(t)^TM^TM^{-T}C^TCM^{-1}M\boldsymbol{x}(t) + \boldsymbol{u}(t)^T\boldsymbol{u}(t)\,\mathrm{d}t\\ &= \int_0^\infty \boldsymbol{x}(t)^TC^TC\boldsymbol{x}(t) + \boldsymbol{u}(t)^T\boldsymbol{u}(t)\,\mathrm{d}t = \mathcal{J}(\boldsymbol{u}(t), \boldsymbol{x}(t)).\end{aligned} \tag{2.33c}$$

In a straightforward way, the *generalized algebraic Bernoulli* equation (GABE) is defined via

$$A^TXM + M^TXA - M^TXNXM = 0; \tag{2.34}$$

compare, e.g., [20].

In the case of a singular matrix M, the dynamical system (2.29) has a DAE structure. The definitions of generalized Lyapunov, Riccati, and Bernoulli equation are still valid but strategies to solve these equations usually change drastically. A direct treatment to solve DAE and matrix equations with DAE structure is not within the scope of this thesis. The interested reader is referred to, e.g., [83, 127] and the references in there.

The DAE systems that arise within this thesis are indirectly projected onto generalized state-space systems like (2.29) with an invertible left-hand side matrix such that the above presented ideas and methods can be applied after a certain modification. The entire procedure is introduced in Chapter 4, where the used solution strategies for the arising matrix equations are explained in detail. Afterwards, some results of a closed loop simulation, where the calculated optimal control $\boldsymbol{u}_*(t)$ is used to stabilize the numerical realization of a flow simulation from Chapter 3, are presented in Chapter 7.

2.4. Iterative Methods

As pointed out in Subsection 2.3.2, many equations do not have analytical solutions or an explicit solution cannot be constructed using direct methods. In these cases, approximate solutions are used to approximately solve the problem. Thereby, a series of iterates $\{x_k\}$ is built up during an iterative process, where an initial solution guess x_0 is modified following a certain iteration scheme until the iterate fulfills certain conditions. Considering a general equation

$$\mathscr{F}(x_*) = b,$$

one way to characterize the quality of the approximate solution is the *error*, which is defined at step k as

$$\mathrm{err}_k := x_k - x_*.$$

Since the actual solution x_* is usually of interest and not available in advance, the *residual* is another approach to determine the quality of the solution as

$$\mathrm{res}_k := \mathscr{F}(x_k) - b.$$

Obviously, the residual and the error are equal to zero if $x_k = x_*$. However, only the residual can be computed without knowing x_*.

To analyze errors or residuals independent of the used functions and variables, one usually chooses an appropriate norm to map both quantities into the field of real numbers that can be compared. In many cases, the scaled version of these norms are of interest and can be defined via

$$\mathrm{err}_k^{\mathrm{rel}} := \frac{||x_k - x_*||}{||\tilde{x}||}, \quad \mathrm{res}_k^{\mathrm{rel}} := \frac{||\mathscr{F}(x_k) - b||}{||\tilde{x}||},$$

where $\tilde{x}$ is usually chosen from either the initial guess x_0, the actual iterate x_k, or the right hand side b. For $\tilde{x} := b$, $\mathrm{res}_k^{\mathrm{rel}}$ is called relative residual. Notice that the relative residual is an upper bound for the relative error as shown in [80, Lem 1.1.1].

The three main iterative methods, used for different classes of equations $\mathscr{F}(x) = b$ that arise within this thesis, are shortly reviewed in the following. For a more detailed analysis of iterative methods the interested reader is referred to, e.g., [113] for (sparse) linear systems and to [56, 80] for linear and nonlinear equations.

2.4.1. Newton's Method (NM)

Assume a general nonlinear function $\mathscr{F}$, whose Fréchet derivative exists. One of the most commonly used methods to solve nonlinear equations of the form

$$\mathscr{F}(x) = 0, \tag{2.35}$$

which means finding the roots of $\mathscr{F}(x)$, is Newton's method [80, 97]. The basic iteration is given for an initial iterate x_0 by

$$x_{k+1} = x_k - \mathscr{F}'(x_k)^{-1}\mathscr{F}(x_k) \tag{2.36}$$

with $\mathscr{F}'(x_k)$ the Fréchet derivative of $\mathscr{F}$ at position x_k. Instead of an explicit inversion of the Fréchet derivative in (2.36), one can solve a system that is linear in the sought update s in the form

$$\mathscr{F}'(x_k)s = -\mathscr{F}(x_k), \quad x_{k+1} = x_k + s.$$

The convergence of this method is stated in the following theorem.

Theorem 2.30 (cf. [80, Thm. 5.1.2]) *Assume there exists a solution x_* of (2.35), $\mathscr{F}'$ is Lipschitz continuous, and $\mathscr{F}'(x_*)$ is non-singular. Then there is a $\delta > 0$ such that if $x_0 \in \mathcal{B}(\delta) := \{x : ||x - x_*|| < \delta\}$, the Newton iteration (2.36) converges q-quadratically to x_*, i.e., $x_k \to x_*$ and there is a $K > 0$, independent of k, such that*

$$||x_{k+1} - x_*|| \leq K||x_k - x_*||^2,$$

compare, [80, Def. 4.1.1].

It is well known that the choice of the initial iterate x_0 remains a problem, especially for non-convex (-concave) functions $\mathscr{F}$, where the iterative process might diverge, stagnate in, or alternate around a local minimum [80, Sec. 8.1]. For more details of error analysis and convergence theory we refer to [80, Chap. 5].

2.4.2. Alternating Directions Implicit (ADI) Method

To solve a large-scale GCALE like (2.32), various methods exist in the literature as explained in Subsection 2.3.4. The focus in this thesis lies on the alternating directions implicit (ADI) method which is a powerful tool to solve large-scale GCALEs with low-rank right-hand sides.

"The ADI method was introduced in the mid-1950s by Peaceman and Rachford [101] specifically for solving equations arising from finite difference discretizations of elliptic and parabolic PDEs."[113, Sec. 4.3] Later on, this idea was applied to CALEs; see, e.g., [128]. Throughout the last decades, many new achievements were discovered that increased the efficiency of the low-rank ADI method drastically.

Most recent improvements were developed parallel to the authors work such that these improvements were not available at the beginning of his work. Hence, in this subsection the so-called *generalized low-rank Cholesky factor ADI* (G-LRCF-ADI) method is introduced. This method is the generalization of the LRCF-ADI method from [31] and is mentioned the first time in [23].

The G-LRCF-ADI method is used in Chapter 4. The above mentioned recent ADI developments are reviewed and incorporated in Chapter 6. Throughout all considerations, one assumes that the pencil (F, M) is stable such that there exists a unique solution $X = X^T \succeq 0$ of (2.32), compare Theorems 2.18 and 2.19.

Using a proper set of ADI shifts, i.e., $\{q_1, \ldots, q_\ell\} = \overline{\{q_1, \ldots, q_\ell\}}$ with $q_i \in \mathbb{C}^-$, the original G-LRCF-ADI method computes a low-rank solution factor $Z \in \mathbb{C}^{n \times \ell r}$ such that $ZZ^H \approx X \in \mathbb{R}^{n \times n}$ is the approximated solution of the Lyapunov equation (2.32); see, e.g., [23]. This solution is by construction spsd. Given proper ADI shifts as defined above, the G-LRCF-ADI method successively computes

$$\begin{aligned} V_1 &= (F + q_1 M)^{-1} W \in \mathbb{C}^{n \times r}, \\ V_\ell &= (V_{\ell-1} - (q_\ell + \overline{q}_{\ell-1})(F + q_\ell M)^{-1}(M V_{\ell-1})) \in \mathbb{C}^{n \times r}, \quad \ell \geq 2. \end{aligned} \tag{2.37a}$$

In the ℓ-th iteration, the approximate low-rank solution factor is

$$Z_\ell = \left[\sqrt{-2\,\mathrm{Re}\,(q_1)}V_1, \ldots, \sqrt{-2\,\mathrm{Re}\,(q_\ell)}V_\ell\right] \in \mathbb{C}^{n \times \ell r}. \tag{2.37b}$$

Algorithm 1 Generalized low-rank Cholesky factor ADI iteration [23, 31]

```
Input: $F, M, W$, and proper shift parameters $q_i \in \mathbb{C}^- : \ell = 1, \dots, \ell_{\max}$
Output: $Z \in \mathbb{C}^{n \times \ell r}$, such that $ZZ^H \approx X$
1: $Z = [\,]$
2: $V_1 = (F + q_1 M)^{-1} W$
3: $Z = \sqrt{-2\,\mathrm{Re}\,(q_1)} V_1$
4: for $\ell = 2, 3, \dots, \ell_{\max}$ do
5:     $V_\ell = V_{\ell-1} - (q_\ell + \overline{q_{\ell-1}})(F + q_\ell M)^{-1}(M V_{\ell-1})$
6:     $Z = \begin{bmatrix} Z & \sqrt{-2\,\mathrm{Re}\,(q_\ell)} V_\ell \end{bmatrix}$
7: end for
```

The entire process is depicted in Algorithm 1.

The crucial ingredient for fast convergence of the ADI iteration is the shift parameter set. There are various approaches to compute ADI shifts. A more involved analysis about different shift strategies is not within the scope of this thesis. The interested reader is referred to [30, 84, 104, 128] for more details. In Chapter 4, some heuristic shifts based on [104] are used that are precomputed and that only depend on the pencil (F, M). Later on in Chapter 6, a variant of the novel shift approach in [30] is used to compute ADI shifts adaptively during the iteration that depend on the pencil (F, M) as well as the right-hand side W and the previous iterates $V_{\ell-1}$.

2.4.3. Krylov Subspace Methods

A widely used class of iterative methods to solve a linear system of equations of the form

$$A\boldsymbol{x} = \boldsymbol{b}, \quad \text{with} \quad A \in \mathbb{R}^{n \times n},\ \boldsymbol{x},\ \boldsymbol{b} \in \mathbb{R}^n, \tag{2.38}$$

are *Krylov subspace* methods. These methods are projection based methods [113, Chap. 5] that use the m-dimensional *Krylov subspace*

$$\mathscr{K}_m(A, \boldsymbol{v}) := \operatorname{span}\left\{\boldsymbol{v}, A\boldsymbol{v}, \dots, A^{m-1}\boldsymbol{v}\right\}. \tag{2.39}$$

Hence, for a starting vector $\boldsymbol{x}_0 \in \mathbb{R}^n$ there is an approximate solution $\boldsymbol{x}_m \in \boldsymbol{x}_0 + \mathscr{K}_m$ such that

$$\boldsymbol{b} - A\boldsymbol{x}_m \perp \mathscr{L}_m. \tag{2.40}$$

This means the residual $\boldsymbol{b} - A\boldsymbol{x}_m$ is orthogonal to $\mathscr{L}_m$, which defines another m-dimensional subspace as explained in more details in [113, Chap. 6]. By successively increasing the dimension m, the iterate $\boldsymbol{x}_m$ converges towards the solution $\boldsymbol{x}$ in at most n steps. In other words, in each step one includes a new search direction to improve the iterate $\boldsymbol{x}_m$ such that this iterate is the best approximation of the solution regarding the subspace $\mathscr{L}_m$. After n steps, the subspaces $\mathscr{K}_m$ and $\mathscr{L}_m$ span the entire solution domain of dimension n and $\boldsymbol{x}_m$ is the best approximation that can be obtained.

In general, the Krylov subspace methods converge within a lot less steps towards the desired tolerance or one can apply certain acceleration techniques, such as preconditioning [113, Chap. 9], to speed up convergence. The problem specific preconditioning technique used in this thesis is presented in Section 5.2.

Depending on the definition of $\mathscr{L}_m$, different Krylov subspace methods for different classes of matrices can be defined. Within this thesis, the *generalized minimal residual* (GMRES) method, [114], is considered. GMRES takes $\mathscr{L}_m = A\mathscr{K}_m$ and $\boldsymbol{v} = \frac{\boldsymbol{b}-A\boldsymbol{x}_0}{||\boldsymbol{b}-A\boldsymbol{x}_0||_2}$ to minimize the residual norm $||\boldsymbol{b} - A\boldsymbol{x}_m||_2$ over all vectors in $\boldsymbol{x}_0 + \mathscr{K}_m$ for nonsymmetric, non-singular matrices A. For all numerical examples, the MATLAB function `gmres(A,b)` is used. For further details see, e.g., [113, Sec. 6.5], [56, Sec. 4.1.1], and [90, Sec. 2.5.5].

Another well known Krylov subspace method is the *conjugate gradient* (CG) method [72] that uses $\mathscr{L}_m = \mathscr{K}_m$ and $\boldsymbol{v} = \boldsymbol{b} - A\boldsymbol{x}_0$ and can be used for symmetric, positive-definite (spd) matrices A. This method is mentioned in Subsection 5.2.2 but is not further used within this thesis.

For more examples of Krylov subspace methods, the interested reader is referred to the literature cited within this subsection.

As a historic fact it is interesting to mention that "in his 1931 paper [82], Krylov was not thinking in terms of projection processes, and he was not interested in solving a linear system"[90, p. 19].

3

Scenarios for Multi-Field Flow Problems

Contents

In this chapter, the main scenarios are introduced that serve as examples for multi-field flow problems. Thereby, the complexity of the considered PDEs increases which leads to more demanding numerical treatments. All models are related to transport processes for either scalar or vector fields.

For all examples, a fixed, connected, and bounded domain $\Omega \subset \mathbb{R}^d$ with $d \in \{2, 3\}$ as well as an infinite time horizon form the defining space time cylinder $Q_\infty := \Omega \times [0, \infty)$ for the PDEs. The boundary $\Gamma := \partial\Omega \subset \mathbb{R}^{d-1}$ is partitioned as $\Gamma = \Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{wall}}$, where certain boundary conditions (BC) are applied. Notice that all considerations are formulated in Cartesian coordinates.

The chapter is organized as follows. First, PDEs that describe passive transports for a scalar variable are introduced and a first test domain is described. Afterwards, incompressible flows, describing the physical behavior of a vector field, are defined and two describing PDEs, as well as the second test domain, are introduced in Section 3.2. Following, in Section 3.3 a coupled flow problem is proposed together with the defining PDEs and the third test domain. To fit the LQR scheme from Subsection 2.3.3, nonlinear PDEs are linearized as described in Section 3.4. Afterwards, the linear PDEs are discretized in Section 3.5. The properties of the arising large-scale matrix pencils are examined in detail in Section 3.6.

3.1. Scalar Transport Equations

Two fundamental processes of scalar transportation are diffusion and convection. Thereby, the change in the distribution of a state, e.g., temperature or concentration, within the domain is described. In many applications, a reactive process is considered additionally. All these processes are introduced in the following and by combining them, one ends up with the first scenario that plays a role in this thesis. Every state depends on $(t, \vec{x}) \in Q_\infty$. BC and initial conditions (IC) are omitted for the first general processes.

3.1.1. Diffusion Process

As stated in [117, p. 15], diffusion takes place in areas with different state levels and is a direct consequence of the interaction between micro particles that are in contact. It is a natural process that seeks for an equilibrium in the state distribution. The diffusion process was examined in detail for temperature distributions by Fourier in 1822 and can be generalized straight forward for certain other state distributions.

The main result of the *Fourier law* is that the change $\vec{q}(t, \vec{x})$ in the state level is proportional to the gradient of the state $a(t, \vec{x})$, i.e.,

$$\vec{q}(t, \vec{x}) = -k_D \nabla a(t, \vec{x}).$$

Following the derivation in [117, Sec. 2.1], this proportionality needs to fulfill the energy conservation law

$$\frac{\mathrm{d}}{\mathrm{d}t} a(t, \vec{x}) = -\operatorname{div}(\vec{q}(t, \vec{x})) + f(t, \vec{x}).$$

Hence, the diffusion process for a state $a(t, \vec{x}) \in \mathbb{R}$ and a source term $f(t, \vec{x}) \in \mathbb{R}$ is described by

$$\frac{\mathrm{d}}{\mathrm{d}t} a(t, \vec{x}) = \operatorname{div}(k_D(\vec{x}) \nabla a(t, \vec{x})) + f(t, \vec{x}), \qquad \text{in } Q_\infty, \tag{3.1}$$

where $k_D(\vec{x}) \in \mathbb{R}$ is the proportionality constant. If $k_D(\vec{x})$ is constant over Ω, (3.1) simplifies to

$$\frac{\mathrm{d}}{\mathrm{d}t}a(t,\vec{x}) = k_D \Delta a(t,\vec{x}) + f(t,\vec{x}), \qquad \text{in } Q_\infty. \tag{3.2}$$

If the state $a(t,\vec{x})$ is time-invariant, (3.1) and (3.2) can be simplified to the stationary diffusion equation, also known as Poisson equation, which "is the simplest and the most famous elliptic PDE"[56, Chap. 1].

3.1.2. Convection Process

Convection, in its simplest form, is the result of the movement of the medium in Ω with a given time-invariant velocity $\vec{w}(\vec{x}) = [w_{x_1}(\vec{x}), \dots, w_{x_d}(\vec{x})]^T \in \mathbb{R}^d$. The change of the state is, therefore, dependent on the velocity of the motion. To determine the change of the state $a(t,\vec{x})$ with respect to time, similar to (3.1), one needs to consider the complete derivative with respect to time [117, p. 17]. This leads to the "particle-acceleration" [135, Sec. 1-3.2] operator defined as

$$\frac{\mathrm{d}}{\mathrm{d}t} = \frac{\partial}{\partial t} + \vec{w}(\vec{x}) \cdot \nabla \tag{3.3}$$

with the convection operator

$$(\vec{w}(\vec{x}) \cdot \nabla) = \left(\sum_{i=1}^{d} w_{x_i} \frac{\partial}{\partial x_i} \right); \tag{3.4}$$

see, e.g., [56, p. 4]. To this end, the time-varying convection equation reads

$$\frac{\partial}{\partial t}a(t,\vec{x}) = -(\vec{w}(\vec{x}) \cdot \nabla)a(t,\vec{x}) + f(t,\vec{x}), \qquad \text{in } Q_\infty. \tag{3.5}$$

As stated in [56, p. 114], for a time-invariant state and right hand side, (3.5) simplifies to the hyperbolic PDE

$$(\vec{w}(\vec{x}) \cdot \nabla)a(\vec{x}) = f(\vec{x}), \qquad \text{on } \Omega.$$

3.1.3. Reaction Process

A reaction process does not represent a spatial transport, but the change of the state due to the state itself, described by the proportional change of the state in time by

$$\frac{\mathrm{d}}{\mathrm{d}t}a(t,\vec{x}) = k_R\, a(t,\vec{x}), \qquad \text{in } Q_\infty \tag{3.6}$$

with a proportionality factor $k_R \in \mathbb{R}$. It is a simplified model for growth processes, which do not model the reason for the growth, as it is often needed in chemical and biological models. Properly speaking, (3.6) is not a PDE but an ODE since only time dependent changes are considered.

In the following, all three process are combined.

3.1.4. Combined Transport Process (CTP)

A mixture of a diffusion, convection, and reaction process yields the *combined transport process (CTP)* which is the first test scenario of this thesis and is defined by

$$\frac{\partial}{\partial t}a(t,\vec{x}) - k_D\Delta a(t,\vec{x}) + (\vec{w}(\vec{x})\cdot\nabla)a(t,\vec{x}) - k_R\, a(t,\vec{x}) = f(t,\vec{x}), \qquad \text{in } Q_\infty. \tag{3.7}$$

The CTP model describes the spread of a scalar field property within the domain Ω. The BC are not further described at this point. Examples for such scalar fields are, e.g., temperature or concentration. The specific parameters k_D, k_R, $\vec{w}(\vec{x})$, and $f(t,\vec{x})$ are specified in the numerical examples. Notice that this parabolic PDE also serves as test example in, e.g., [59, 94] with Ω as defined next.

3.1.5. Test Domain I: Unit Cube Domain (UCD)

The first test domain is the standard unit cube domain $\Omega_{\text{UCD}} = (0,1)\times(0,1)\subset\mathbb{R}^2$ in 2D or $\Omega_{\text{UCD}} = (0,1)\times(0,1)\times(0,1)\subset\mathbb{R}^3$ in 3D. This purely academic and rather simple domain is incorporated in this thesis since it served as an easy implementable domain to test and develop the methods in Chapter 6 as published in [26]. Notice that the same domain is used in [59, 94].

To describe transport problems for vector fields, more involved PDEs need to be considered. One possible approach are fluid flow problems as described in the next section.

3.2. Incompressible Flows

The following scenarios are mainly concerned with describing the physical behavior of a fluid under certain BC and IC.

Definition 3.1 (Fluid, cf. [119, p. 2]) *A fluid is a material that deforms continually upon the application of surface forces. A fluid does not have a preferred shape and different elements of a homogeneous fluid may be rearranged freely without affecting the macroscopic properties of the fluid, i.e., the fluids are mobile. A fluid offers a resistance to attempts to produce relative motions of its different elements, i.e., a deformation, and this resistant vanishes with the rate of deformation. Fluids, unlike solids, cannot support a tension or negative pressure.*

The property incompressible refers to a fluid whose density does not change in time or space and is described by the constant parameter $\rho\in\mathbb{R}^+$.

Various slow moving gases, most liquids, and some melted solids can be classified as incompressible fluids. Excluded are non-Newtonian fluids [57, p. 15], such as blood or starch mixed with water, whose viscosity $\eta\in\mathbb{R}^+$ changes depending on shear forces. A pictorial definition of viscosity is as follows.

Definition 3.2 (Viscosity, cf. [57, p. 14]) *The property of a fluid that measures its resistance to change of shape is called the viscosity.*

As mentioned above, liquids and gases are considered as fluids. “To distinguish between a liquid and a gas, we also noted that although both will occupy the container in which they are placed, a liquid presents a free surface if it does not completely fill the container. A gas will always fill the volume of the container in which it is placed.”[64, p. 23]

Describing the physical behavior of fluids is a highly difficult and wide research area by itself. In this thesis, the focus lies on only two simplified models that are introduced in the next two subsections. The interested reader is referred to the following literature for more detailed studies. In [119], an overview regarding theoretical fluid dynamics is given. Besides the description of channel flow, water waves, and lunar tides, many essential equations such as Euler, Stokes, Oseen, and Navier-Stokes equations are introduced. A more detailed introduction to turbulent flow behavior can be found in [135]. The books [57, 64] give a widespread introduction to fluid mechanics including many problem tasks that are helpful for studying the subject of fluid mechanics. A more detailed numerical view point of fluid dynamics is provided by the book [134].

In what follows, the space is always limited to the 2-dimensional case. This simplification can be understood as a projection of an 3-dimensional flow along the third space dimension if the flow does not change regarding this dimension. For example, considering an elliptically shaped long pillar of a bridge inside a river, the velocity of the fluid does not change significantly along the height of this pillar if the distance to the ground or surface is sufficiently large. Hence, one can consider a two dimensional flow around an elliptical shaped obstacle as depicted on the title page.

The following two scenarios describe the velocity field $\vec{v}(t,\vec{x}) \in \mathbb{R}^2$ and the scalar pressure $\chi(t,\vec{x}) \in \mathbb{R}^+$ of an incompressible fluid defined for $\vec{x} \in \Omega \subset \mathbb{R}^2$ and $t \in [0,\infty)$.

3.2.1. Navier–Stokes Equations (NSE)

The first set of equations to describe incompressible flows are the *Navier-Stokes equations* (NSE) [57, eqs. (8.1),(8.3)] and build the second scenario in this thesis. Describing the motion of the fluid as combination of pressure and viscous forces, the NSE cover the conservation of mass, momentum, and energy within the system, compare [57]. Following the notation in [15, Sec. 1], the NSE can be written in dimensionless form as

$$\left.\begin{aligned} \frac{\partial}{\partial t}\vec{v}(t,\vec{x}) - \frac{1}{\mathrm{Re}}\Delta\vec{v}(t,\vec{x}) + (\vec{v}(t,\vec{x})\cdot\nabla)\vec{v}(t,\vec{x}) + \nabla\chi(t,\vec{x}) &= \vec{f}(\vec{x}), \qquad (3.8a)\\ \operatorname{div}\vec{v}(t,\vec{x}) &= 0, \qquad (3.8b)\end{aligned}\right\} \text{ in } Q_\infty.$$

The dynamical part (3.8a) consists, from the left to the right, of the time derivative, the scaled diffusion, the convection, the pressure gradient, and some time-invariant external forces. The left side of (3.8b), “is physically the time rate of change of the volume of a moving fluid element, per unit volume”[134, p. 23]. Hence, the mass conservation is assured by setting $\operatorname{div}\vec{v}(t,\vec{x}) = 0$, which is the so-called divergence-free condition. The key for the dimensionless form is the scaling factor

$$\mathrm{Re} := \frac{\rho \cdot v_{ref} \cdot d_{ref}}{\eta} \in \mathbb{R}^+, \tag{3.8c}$$

called Reynolds number; see, e.g., [57, Sec. 10.2.4]. The Reynolds number describes the ratio of inertial and viscous forces within the fluid or, in other words, "may be interpreted as the magnitude of the ratio of the acceleration of a fluid in steady flow, [...] to the viscous force per unit mass [...]" [57, p. 453]. Thereby, $v_{ref} \in \mathbb{R}^+$ is a reference velocity and $d_{ref} \in \mathbb{R}^+$ a reference length.

On the boundary Γ, the velocity $\vec{v}(t, \vec{x})$ is described as

$$\vec{v}(t,\vec{x}) = \vec{g}_{\mathrm{in}}(\vec{x}), \qquad \text{on } \Gamma_{\mathrm{in}}, \tag{3.8d}$$

$$\vec{v}(t,\vec{x}) = 0, \qquad \text{on } \Gamma_{\mathrm{wall}}, \tag{3.8e}$$

$$-\frac{1}{\mathrm{Re}}\nabla\vec{v}(t,\vec{x})\,\vec{n}(\vec{x}) + \chi(t,\vec{x})\vec{n}(\vec{x}) = 0, \qquad \text{on } \Gamma_{\mathrm{out}} \tag{3.8f}$$

with $\vec{n}(\vec{x})$ the outward normal to Γ_{out}. The latter condition is the so-called *do-nothing* condition [47, 73] that ensures that the fluid leaves the domain without hindrance such that the length of the channel is not significantly influencing the correct physical behavior, as shown in [73]. The no-slip condition (3.8e) on Γ_{wall} states that the fluid is in rest at the walls due to adhesion or simple friction.

The initial condition of the fluid is given by

$$\vec{v}(0,\vec{x}) = 0, \quad \text{in } \Omega \tag{3.8g}$$

that describes that the fluid is in rest for $t = 0$. The parameters v_{ref}, d_{ref}, as well as the inflow function $\vec{g}_{in}$ are specified in the numerical examples later on. For better readability, the parameters t, $\vec{x}$ are skipped hereafter.

The physical meaning of diffusion and convection is more complicated for vector fields then for scalar fields. The diffusion process $\Delta\vec{v}$ can be seen as an impulsive shearing motion within the fluid due to its own movement, as exemplary described in [57, Sec. 6.5.9].

To define convection, one applies (3.4) to the vector field $\vec{v}$ in each component, such that the nonlinear term in (3.8a) is defined as

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

These "convective accelerations are nonlinear in character and present such vexing analytical problems as failure of the superposition principle; [e.g.,] non-unique solutions, even in steady laminar flow; [...] Note that these nonlinear terms are accelerations, not viscous stresses. It is ironic that the main obstacle in viscous-flow analysis is an inviscid term; the viscous stresses themselves are linear if the viscosity is assumed constant" [135, p. 18].

The incompressible NSE (3.8) define the behavior of fluids with moderate Reynolds numbers such that the impact of convection and diffusion is noticeable in a similar order of magnitude. For fluids with high density, low viscosity, or under high velocity, one cannot use these equations anymore and more complicated flow models need to be used such that turbulences are considered as well.

In the limit, as $\mathrm{Re} \to \infty$, the diffusion term vanishes and one ends up with the Euler equations [57, Sec. 4.3] that do not play a further role in this thesis. In contrast, if the Reynolds number is small, the diffusion term dominates the entire equation such that the (particle) acceleration terms from (3.3) can be neglected [57, Sec. 6.5.7]. For this configuration, one speaks about the (creeping) Stokes equations that is the second flow model considered in this thesis and described in the next subsection.

3.2.2. Stokes Equations

During the early phase of our research regarding feedback stabilization of incompressible flow problems, various technical and conceptional problems occurred. Most of these problems were caused by the non-linearity and the instability of the NSE. Hence, we considered the linear Stokes equations defined by

$$\frac{\partial}{\partial t}\vec{v}(t,\vec{x}) - \frac{1}{\mathrm{Re}}\Delta\vec{v}(t,\vec{x}) + \nabla\chi(t,\vec{x}) = \vec{f}(\vec{x}), \qquad \text{in } Q_\infty; \tag{3.9a}$$

$$\operatorname{div}\vec{v}(t,\vec{x}) = 0, \tag{3.9b}$$

compare [35]. Following the derivation of the Stokes equations in [57, Sec. 6.5.7], no time derivative and no right-hand side occur due to the dominant diffusion term. In contrast, it is pointed out in [48, Sec. 2] that different scaling approaches yield various resulting equations with or without any time derivatives involved. Since a time dependent control approach is considered in this thesis, the time dependent Stokes equations (3.9) are considered and the scaling needs to be done concerning length and velocity as well, such that $\nu := \frac{\eta}{\rho}$, as used in [35], is replaced by $\frac{1}{\mathrm{Re}}$. In the end, the third scenario used in this thesis, are the Stokes equations (3.9) that are equivalent to the NSE (3.8) without the nonlinear convection term. Thus, the same BC and IC as (3.8d)–(3.8g) are used. Notice that $v_{ref} = d_{ref} = 1$ in [35] such that $\nu = \frac{1}{\mathrm{Re}}$ and (3.9) is equivalent to the formulations in [35].

The next subsection describes the geometry of the second test domain for Ω.

3.2.3. Test Domain II: Kármán Vortex Street (KVS)

The second test domain is the so-called *Kármán vortex street* (often also referred to as *Kármán vortex shedding*) that is abbreviated in the following with KVS. It describes the flow around a cylindrical obstacle with no-slip conditions on the obstacles surface, where alternating vortexes occur behind the obstacle that show a periodic behavior. This well known phenomena can be observed in nature in many examples. Starting from bridge

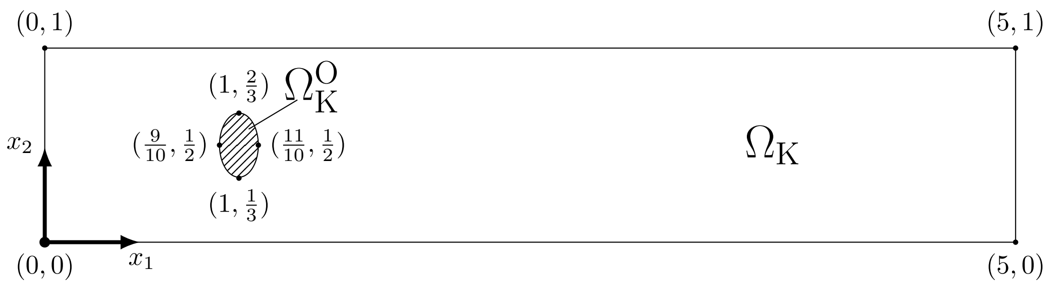


Figure 3.1.: Kármán vortex street (KVS).

pillars that generate turbulences in a river up to high mountain peeks, where satellite pictures show periodic vertexes in the passing clouds[1].

The more academic test domain that is used for all numerical examples regarding the KVS domain is depicted in Figure 3.1. It is identical to the domain used in [15, 35] and describes the flow from the left to the right through a rectangular channel $\Omega_{\mathrm{K}} \subset \mathbb{R}^2$ of width $d_{in} = 1$ in x_2 direction and length $5 \cdot d_{in}$ in x_1 direction. The obstacle $\Omega_{\mathrm{K}}^{\mathrm{O}}$ is of elliptic shape with dimensions $\frac{1}{5}d_{in}$ in x_1 and $\frac{1}{3}d_{in}$ in x_2 direction centered at $(1, 0.5)$.

The NSE (3.8) and the Stokes equations (3.9) are solved over $\Omega_{\mathrm{KVS}} := \Omega_{\mathrm{K}} \backslash \Omega_{\mathrm{K}}^{\mathrm{O}}$.

3.3. Coupled Flow Model (CFM)

In this section, the above introduced models for incompressible flows are coupled with a special case of the CTP equations (3.7) as introduced in the next subsection. The main idea is to imitate a simplified version of a reactor. Thereby, a fluid, described by an instationary incompressible flow model, e.g., NSE, acts as carrier medium which transports a certain reactive substance. This substance reacts in a certain way in a specific part of the reactor. The reaction process is not part of the dynamic model but is represented by the BC of a special reactive subdomain. One way to describe this transport process is the diffusion-convection equation (DCE) as introduced next.

3.3.1. Diffusion-Convection Equation (DCE)

Combining a diffusion with a convection process, the distribution of the concentration $c(t,\vec{x}) \in \mathbb{R}^+$ of a reactive substance is described by the DCE

$$\frac{\partial}{\partial t}c(t,\vec{x}) - \frac{1}{\mathrm{ReSc}}\Delta c(t,\vec{x}) + (\vec{v}(t,\vec{x}) \cdot \nabla)c(t,\vec{x}) = 0, \qquad \text{in } Q_\infty \tag{3.10a}$$

with the *Schmidt number* [9, p. 79]

$$\mathrm{Sc} := \frac{\nu}{D} = \frac{\eta}{\rho \cdot D} \in \mathbb{R}^+. \tag{3.10b}$$

[1]https://en.wikipedia.org/wiki/K%C3%A1rm%C3%A1n_vortex_street#/media/File:Vortex-street-1.jpg

Following the descriptions in [14, 36, 132], the BC and IC are defined as

$$c(t,\vec{x}) = h_{\text{in}}(\vec{x}), \qquad \text{on } \Gamma_{\text{in}}, \tag{3.10c}$$

$$\frac{\partial c(t,\vec{x})}{\partial \vec{n}(\vec{x})} = 0, \qquad \text{on } \Gamma_{\text{wall}} \cup \Gamma_{\text{out}}, \tag{3.10d}$$

$$c(t,\vec{x}) = 0, \qquad \text{on } \Gamma_r, \tag{3.10e}$$

$$c(0,\vec{x}) = 0, \qquad \text{in } \Omega. \tag{3.10f}$$

Thereby, Γ_{r} describes the boundary of the reactive subdomain Ω_{R}. For the simplified reaction process on Γ_{r} one assumes that the concentration of the reactive substance instantly annihilates at contact with Γ_{r}. This is realized by the homogeneous Dirichlet BC (3.10e).

Additionally, the convection defining velocity field $\vec{v}(t,\vec{x}) \in \mathbb{R}^2$ results from the NSE (3.8) such that both systems are coupled in one direction.

The equations (3.10) are a special case of the CTP model (3.7) with $k_D = \frac{1}{\text{ReSc}}$, a time-invariant convection $\vec{w}(\vec{x}) = \vec{v}(t,\vec{x})$, $k_R = 0$, and $f = 0$. Notice that (3.10) can also be straightforwardly defined for the distribution of a temperature $\theta(t,\vec{x}) \in \mathbb{R}^+$, which is not further considered in this thesis.

To use this model as description of the dynamics within a reactor, it is important that neither concentration nor temperature is changing the volume of the carrier fluid described by the velocity $\vec{v}(t,\vec{x})$ to not violate the divergence-free condition (3.8b). The coupled model of (3.8) and (3.10) serves as our fourth scenario and is abbreviated by CFM in the following.

3.3.2. Test Domain III: Reactor Model (RM)

The third test domain describes a simplified reactor model as depicted in Figure 3.2, where the coupled flow problem CFM is considered. On both sides of the rectangular reactor chamber of dimension $5 \cdot d_{in}$ in x_1 and $8 \cdot d_{in}$ in x_2 direction, one finds a rectangular channel that serves as inflow and outflow area, respectively. Both channels are of diameter $1 \cdot d_{in}$ and length $3 \cdot d_{in}$ and are attached in the middle of the left and right chamber boundary. Inside of the reactor domain Ω_{R}, there is a square obstacle $\Omega_{\text{R}}^{\text{O}}$ of size $1.5 \cdot d_{in} \times 1.5 \cdot d_{in}$ with the center at $(6.25, 3.5)$. The idea behind this configuration is the flow of a carrier medium through the reactor that enters through the left inflow channel, surrounds the obstacle, and leaves the domain via the outflow channel on the right. Thereby, the flow is described by (3.8) and the domain can be seen as a more complicated version of the KVS. Additionally, the spread of a concentration, described by the DCE (3.10), is considered that interacts in a certain way on the surface of the obstacle. Although this rather academic configuration is a highly simplified model of a reactor, it is sufficient to explore various interactions and to examine optimal control approaches. The reactor model is abbreviated by RM in the following and the CFM model is solved over $\Omega_{\text{RM}} := \Omega_{\text{R}} \backslash \Omega_{\text{R}}^{\text{O}}$.

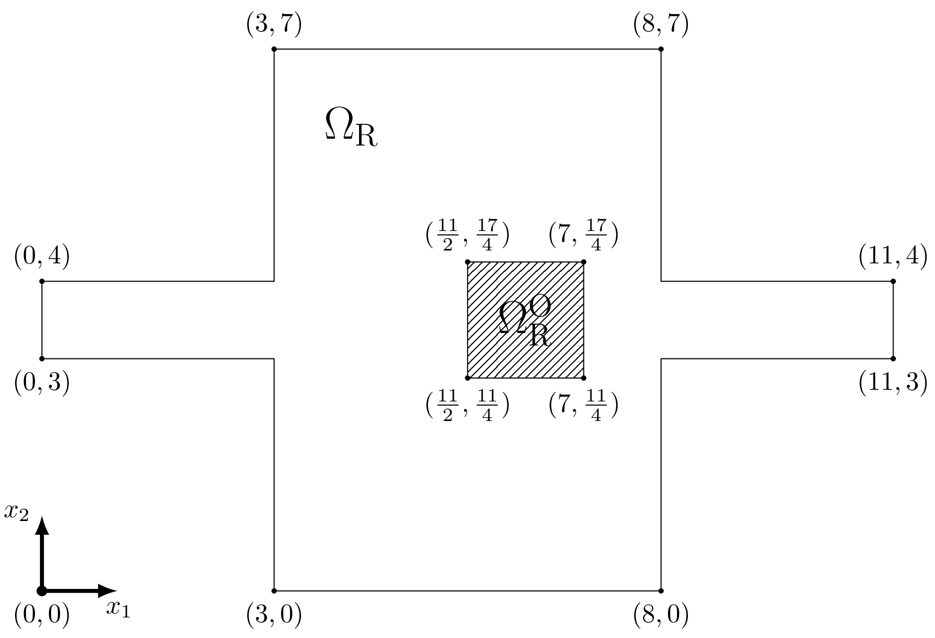


Figure 3.2.: Reactor model (RM).

3.4. Linearization

The NSE and the CFM scenarios are based on nonlinear PDEs. These models need to be linearized to apply the LQR approach from Subsection 2.3.3 as described for the NSE scenario in [15] and for the CFM scenario in [14]. Both linearizations are reviewed in more detail in this subsection. Thereby, the main idea of the linearization is to linearize around a stationary but possibly unstable trajectory with certain special properties.

Although it is not explained further in this section for the linear PDEs, it is important to mention that in order to apply the LQR approach one always needs to linearize around such a trajectory since the LQR approach forces

$$\lim_{t \to \infty} \boldsymbol{x}(t) = 0,$$

which is in general not the desired behavior, as mentioned in Subsection 2.3.3. However, for linear PDEs the notation does not change and one only needs to adapt the BC and IC. The consequences and the correct usage of the LQR approach is pointed out in Chapter 7.

NSE: Let us first consider the NSE scenario from Subsection 3.2.1. The stationary NSE is defined by the solution $(\vec{w}(\vec{x}), \chi_s(\vec{x}))$ and the external force $\vec{f}(\vec{x})$ as

$$\left.\begin{aligned} -\frac{1}{\text{Re}}\Delta\vec{w} + (\vec{w}\cdot\nabla)\vec{w} + \nabla\chi_s &= \vec{f}, \quad &(3.11a)\\ \nabla\cdot\vec{w} &= 0, \quad &(3.11b)\end{aligned}\right\} \text{on } \Omega$$

with the same BC and IC as (3.8d)–(3.8g). After defining the differences

$$\vec{z}(t,\vec{x}) := \vec{v}(t,\vec{x}) - \vec{w}(\vec{x}), \tag{3.12a}$$
$$p(t,\vec{x}) := \chi(t,\vec{x}) - \chi_s(\vec{x}), \tag{3.12b}$$

the NSE (3.8) reads

$$\begin{aligned}\frac{\partial(\vec{w}+\vec{z})}{\partial t} - \frac{1}{\mathrm{Re}}\Delta(\vec{w}+\vec{z}) + ((\vec{w}+\vec{z})\cdot\nabla)(\vec{w}+\vec{z}) + \nabla(\chi_s+p) &= \vec{f},\\ \nabla\cdot(\vec{w}+\vec{z}) &= 0,\end{aligned}$$

which can be written as

$$\begin{aligned}&\frac{\partial\vec{w}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{w} + (\vec{w}\cdot\nabla)\vec{w} + \nabla\chi_s - \vec{f}\\ &+\frac{\partial\vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{w}\cdot\nabla)\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + (\vec{z}\cdot\nabla)\vec{z} + \nabla p = 0,\end{aligned} \tag{3.13a}$$
$$\nabla\cdot\vec{z} = -\nabla\cdot\vec{w}. \tag{3.13b}$$

The first line in (3.13a), as well as the right-hand side in (3.13b), vanish due to (3.11). Furthermore, the quadratic term $(\vec{z}\cdot\nabla)\vec{z}$ is approximately zero, assuming small deviations $\vec{z}$, as it is pointed out in the LQR approach in Subsection 2.3.3. Summarizing, one ends up with the linearized NSE

$$\frac{\partial\vec{z}}{\partial t} - \frac{1}{\mathrm{Re}}\Delta\vec{z} + (\vec{w}\cdot\nabla)\vec{z} + (\vec{z}\cdot\nabla)\vec{w} + \nabla p = 0, \quad \text{in } Q_\infty \tag{3.14a}$$
$$\nabla\cdot\vec{z} = 0, \tag{3.14b}$$

and the BC and IC

$$\vec{z}(t,\vec{x}) = 0, \qquad \text{on } \Gamma_{\text{in}}\cup\Gamma_{\text{wall}}, \tag{3.14c}$$
$$-\frac{1}{\mathrm{Re}}\nabla\vec{z}(t,\vec{x})\,\vec{n}(\vec{x}) + p(t,\vec{x})\vec{n}(\vec{x}) = 0, \qquad \text{on } \Gamma_{\text{out}}, \tag{3.14d}$$
$$\vec{z}(0,\vec{x}) = 0, \qquad \text{in } \Omega. \tag{3.14e}$$

CFM: Secondly, the CFM scenario in Section 3.3 is considered. The DCE (3.10) itself is linear in $c(t,\vec{x})$ and in the convective field $\vec{v}(t,\vec{x})$. Since the NSE (3.8) is coupled with the DCE (3.10), one needs to consider the linearized velocity (3.12a) such that the DCE (3.10) reads

$$\frac{\partial c}{\partial t} - \frac{1}{\mathrm{ReSc}}\Delta c + ((\vec{w}+\vec{z})\cdot\nabla)c = \frac{\partial c}{\partial t} - \frac{1}{\mathrm{ReSc}}\Delta c + (\vec{w}\cdot\nabla)c + (\vec{z}\cdot\nabla)c = 0.$$

In analogy to the NSE case, one defines the linearized concentration as

$$c^{(\vec{z})} := c - c^{(\vec{w})}, \tag{3.15}$$

such that the DCE (3.10) can be written as

$$\frac{\partial(c^{(\vec{w})}+c^{(\vec{z})})}{\partial t}-\frac{1}{\mathrm{ReSc}}\Delta(c^{(\vec{w})}+c^{(\vec{z})})+(\vec{w}\cdot\nabla)(c^{(\vec{w})}+c^{(\vec{z})})+(\vec{z}\cdot\nabla)(c^{(\vec{w})}+c^{(\vec{z})})=0. \tag{3.16}$$

Thereby, $c^{(\vec{w})}$ is the stationary concentration field depending on the stationary velocity $\vec{w}$ that fulfills the stationary DCE

$$-\frac{1}{\mathrm{ReSc}}\Delta c^{(\vec{w})}+(\vec{w}\cdot\nabla)c^{(\vec{w})}=0 \tag{3.17}$$

with the same BC and IC (3.10c)–(3.10f). Similar to the NSE case, one can assume $(\vec{z}\cdot\nabla)c^{(\vec{z})}$ to vanish and the remaining terms of (3.16) form the linearized DCE

$$\frac{\partial c^{(\vec{z})}}{\partial t}-\frac{1}{\mathrm{ReSc}}\Delta c^{(\vec{z})}+(\vec{w}\cdot\nabla)c^{(\vec{z})}+(\vec{z}\cdot\nabla)c^{(\vec{w})}=0 \tag{3.18a}$$

with the BC and IC

$$c^{(\vec{z})}(t,\vec{x})=0, \qquad \text{on } \Gamma_{\mathrm{in}}\cup\Gamma_{\mathrm{r}}, \tag{3.18b}$$

$$\frac{\partial c^{(\vec{z})}(t,\vec{x})}{\partial\vec{n}(\vec{x})}=0, \qquad \text{on } \Gamma_{\mathrm{wall}}\cup\Gamma_{\mathrm{out}}, \tag{3.18c}$$

$$c^{(\vec{z})}(0,\vec{x})=0, \qquad \text{in } \Omega. \tag{3.18d}$$

3.5. Discretization

All considered and introduced scenarios are PDEs with certain BC and IC. As introduced in Section 2.3, the main contribution of this thesis is the solution of an optimal control problem for dynamical systems of the form $\widehat{\Phi}$ as in (2.29), where the dynamical part is a matrix-valued ODE depending on the time t.

To transform the PDEs into $\widehat{\Phi}$, the so-called *method of lines* [118] is considered to semi-discretize the PDEs as pointed out in, e.g., [14, 15, 35, 36]. Thereby, the domain Ω is fully discretized and the time t is kept continuous. As introduced in Subsection 2.1, finite elements are considered with different problem depending ansatz functions. The discretized state variables are denoted by boldface variables.

The right choice of the finite element space, as well as their correct use, is not part of this thesis. An extensive introduction of PDE discretizations via FEMs can be found in [6]. More details about the use of FEMs for, especially, scalar and vector-valued transport problems can be found in [56]. In this section, the notation for the discretized systems is fixed and special structures of the arising matrices are pointed out. All scenarios are written in the form (2.29), where the input matrix $B\in\mathbb{R}^{n\times n_r}$, the control $\boldsymbol{u}(t)\in\mathbb{R}^{n_r}$, the output $\boldsymbol{y}\in\mathbb{R}^{n_a}$, and the output matrix $C\in\mathbb{R}^{n_a\times n}$ depend on the control problem configuration. Some more details about the input operator B that describes the external influence to modify the system are given in Subsection 4.1.3. The specific output operators C, which specify the states that contribute to the cost functional (2.28a), are defined during the numerical examples in Section 4.4 for the Stokes/NSE and CFM scenarios and in Section 6.4 for the CTP scenario. In all examples, $n_a + n_r \ll n$ which means that only a few control inputs interact with the system and, additionally, only a few outputs can be observed.

CTP: As described in [26], the CTP model (3.7) is discretized using piecewise constant ansatz functions $\mathcal{P}_1$. Thus, $\boldsymbol{a}(t) \in \mathbb{R}^n$ is the discretized version of the field variable $a(t, \vec{x}) \in \mathbb{R}$, which yields the discretized system

$$M_{\boldsymbol{a}} \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{a}(t) = A_{\boldsymbol{a}} \boldsymbol{a}(t) + B_{\boldsymbol{a}} \boldsymbol{u}_{\boldsymbol{a}}(t), \tag{3.19a}$$

$$\boldsymbol{y}_{\boldsymbol{a}}(t) = C_{\boldsymbol{a}} \boldsymbol{a}(t) \tag{3.19b}$$

with

$$A_{\boldsymbol{a}} = -k_D S_{\boldsymbol{a}} - K_{\boldsymbol{a}}(\vec{w}) + k_R M_{\boldsymbol{a}} \in \mathbb{R}^{n \times n}. \tag{3.19c}$$

Thereby, $M_{\boldsymbol{a}}$ is the mass matrix regarding the scalar variable $a(t, \vec{x})$ and $S_{\boldsymbol{a}}$ applied to $\boldsymbol{a}(t) \in \mathbb{R}^n$ represents the discrete version of the scalar Laplacian $-\Delta a$, which is usually called stiffness matrix. $K_{\boldsymbol{a}}(\vec{w})\, \boldsymbol{a}(t)$ describes the discretized version of the convection process $(\vec{w} \cdot \nabla) a$ that depends on the stationary velocity $\vec{w}$. All matrices are defined over $\mathbb{R}^{n \times n}$. The associated domain UCD is discretized by a uniform triangulation containing squares of dimension $h \times h$ in 2D and cubes of dimension $h \times h \times h$ in 3D. Each square in 2D is divided into two triangles and each cube in 3D is divided into six tetrahedra. Considering homogeneous Dirichlet BC, the spatial dimension is $n = (h^{-1} - 1)^d$.

Stokes and NSE: To discretize the Stokes and NSE, the $\mathcal{P}_2$–$\mathcal{P}_1$ *Taylor–Hood* element [78], as introduced in Section 2.1, is used. Thereby, the velocity space is of dimension $n_{\boldsymbol{z}}$ and the pressure space is of dimension $n_{\boldsymbol{p}}$ such that $\boldsymbol{z}(t) \in \mathbb{R}^{n_{\boldsymbol{z}}}$ denotes the discretized velocity and $\boldsymbol{p}(t) \in \mathbb{R}^{n_{\boldsymbol{p}}}$ the discretized pressure. Both the Stokes system (3.9) and the linearized NSE (3.14) can be written as

$$M_{\boldsymbol{z}} \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{z}(t) = A_{\boldsymbol{z}} \boldsymbol{z}(t) + G \boldsymbol{p}(t) + B_{\boldsymbol{z}} \boldsymbol{u}_{\boldsymbol{z}}(t), \tag{3.20a}$$

$$0 = G^T \boldsymbol{z}(t), \tag{3.20b}$$

$$\boldsymbol{y}_{\boldsymbol{z}}(t) = C_{\boldsymbol{z}} \boldsymbol{z}(t), \tag{3.20c}$$

(cf. [15, 35]). The only different occurs in the system matrix $A_{\boldsymbol{z}} \in \mathbb{R}^{n_{\boldsymbol{z}} \times n_{\boldsymbol{z}}}$ that can be split as follows:

$$\text{Stokes: } A_{\boldsymbol{z}} := -\frac{1}{\mathrm{Re}} S_{\boldsymbol{z}}, \qquad \text{NSE: } A_{\boldsymbol{z}} = -\left(\frac{1}{\mathrm{Re}} S_{\boldsymbol{z}} + K_{\boldsymbol{z}}(\vec{w}) + R_{\boldsymbol{z}}(\vec{w}) \right). \tag{3.20d}$$

In more detail, $M_{\boldsymbol{z}}$ represents the mass matrix on the velocity space and $S_{\boldsymbol{z}}$ the velocity stiffness matrix such that $-S_{\boldsymbol{z}} \boldsymbol{z}$ is the discrete counterpart of the vector-valued Laplacian

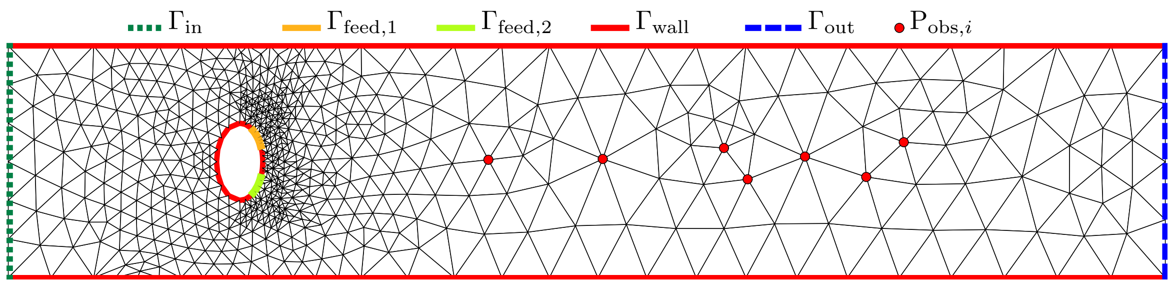


Figure 3.3.: Coarsest triangulation of KVS with BC and observation points (Level 1 in Table 4.3a).

$\Delta\vec{z}$. Furthermore, $K_{\boldsymbol{z}}(\vec{w})\boldsymbol{z}$ is the discrete convection $(\vec{w}\cdot\nabla)\vec{z}$ and $R_{\boldsymbol{z}}(\vec{w})\boldsymbol{z}$ the discrete reaction process $(\vec{z}\cdot\nabla)\vec{w}$, both are dependent on the stationary velocity field $\vec{w}$. All these matrices are defined over $\mathbb{R}^{n_{\boldsymbol{z}}\times n_{\boldsymbol{z}}}$. Additionally, $-G\boldsymbol{p}$ describes the gradient ∇p and, in its transposed version applied to the velocity, also defines the divergence operator $\nabla\cdot\vec{z}$ as $G^T\boldsymbol{z}$ with $G\in\mathbb{R}^{n_{\boldsymbol{z}}\times n_{\boldsymbol{p}}}$.

For inf-sup stable finite elements used in an inflow-outflow configuration, such as $\mathcal{P}_2$–$\mathcal{P}_1$ elements, it holds that $n_{\boldsymbol{z}} > n_{\boldsymbol{p}}$. Furthermore, the discretized gradient G has full rank, i.e., $\operatorname{rank}(G) = n_{\boldsymbol{p}}$ [56, Sec. 5.3]. It is important to point out that the velocity finite element space is reduced by the number of Dirichlet boundary nodes, since the linearized problem has homogeneous Dirichlet BC, such that the corresponding nodes can be eliminated and $n_{\boldsymbol{z}}$ denotes the remaining dimension. The coarsest grid for the KVS, where the Stokes and NSE are solved, is depicted in Figure 3.3. More details on the used dimensions are depicted in Section 4.4.

CFM: The concentration $c^{(\vec{z})}$ in the CFM model is discretized by $\mathcal{P}_1$ finite elements that form a finite element space of dimension $n_{\boldsymbol{c}}$ for the concentration. The linearized DCE (3.18) in its discretized version reads

$$M_{\boldsymbol{c}}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{c}(t) = A_{\boldsymbol{c}}\boldsymbol{c}(t) - R_{\boldsymbol{z}}(c^{(\vec{w})})\boldsymbol{z}(t) + B_{\boldsymbol{c}}\boldsymbol{u}_{\boldsymbol{c}}(t), \tag{3.21a}$$

$$\boldsymbol{y}_{\boldsymbol{c}}(t) = C_{\boldsymbol{c}}\boldsymbol{c}(t), \tag{3.21b}$$

where the system matrix $A_{\boldsymbol{c}}$ can be partitioned as

$$A_{\boldsymbol{c}} = -\frac{1}{\mathrm{ReSc}}S_{\boldsymbol{c}} - K_{\boldsymbol{c}}(\vec{w}). \tag{3.21c}$$

This yields the concentration mass matrix $M_{\boldsymbol{c}}$, the concentration stiffness matrix $S_{\boldsymbol{c}}$, and the convection matrix $K_{\boldsymbol{c}}(\vec{w})$ depending on the stationary velocity $\vec{w}$, similar to the matrix definitions in the CTP scenario, but defined over $\mathbb{R}^{n_{\boldsymbol{c}}\times n_{\boldsymbol{c}}}$. The coupling term $(\vec{z}\cdot\nabla)c^{(\vec{w})}$ leads to a reaction type term $R_{\boldsymbol{z}}(c^{(\vec{w})})\boldsymbol{z}$ that depends on the stationary concentration $c^{(\vec{w})}$ and is applied to the velocity field $\vec{z}(t)$ with $R_{\boldsymbol{z}}(c^{(\vec{w})}) \in \mathbb{R}^{n_{\boldsymbol{c}}\times n_{\boldsymbol{z}}}$. Similar to the Stokes and NSE, all nodes related to the homogeneous Dirichlet BC can be removed such that $n_{\boldsymbol{c}}$ is the purified dimension. The initial triangulation of the RM

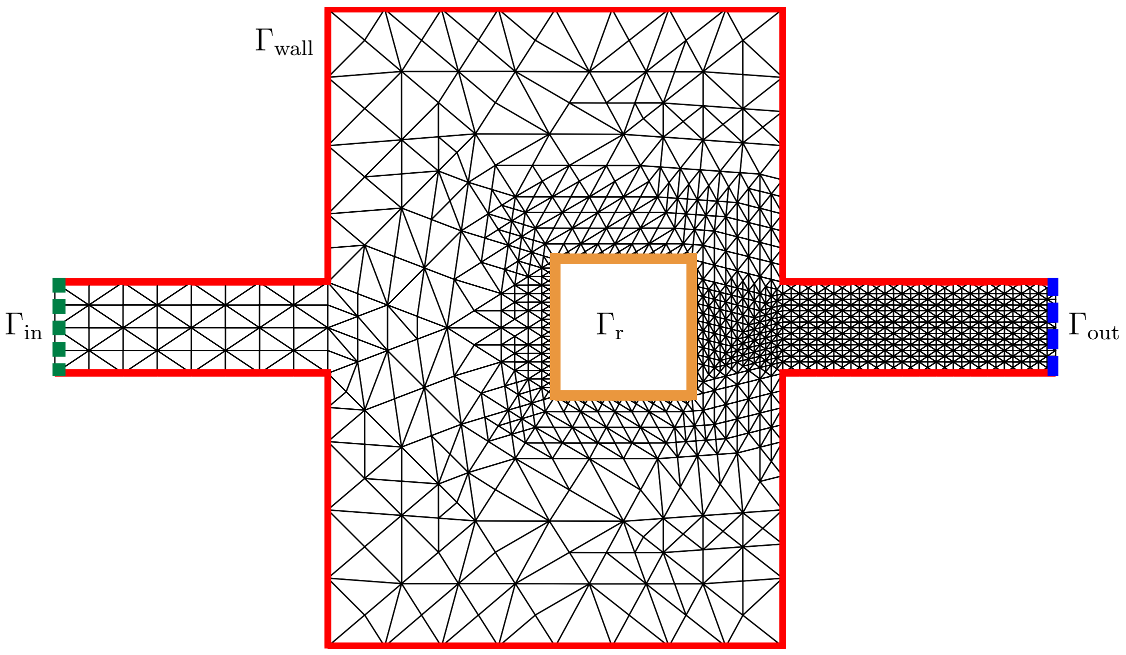


Figure 3.4.: Coarsest triangulation of RM with BC (Level 1 in Table 4.3b).

domain is depicted in Figure 3.4. The used finite element dimensions depend on the refinement levels and are explained in Section 4.4.

For the CFM model, the systems (3.20) and (3.21) are considered together. Both systems are coupled via $R_{\boldsymbol{z}}(c^{(\vec{w})})\boldsymbol{z}(t)$ in (3.21a). In the coupled CFM formulation, the control input is only considered on the velocity space via $B_{\boldsymbol{z}}\boldsymbol{u}_{\boldsymbol{z}}(t)$ in (3.20a) such that $B_{\boldsymbol{c}} = 0$ in (3.21a). Furthermore, the observation only takes place on the concentration space via (3.21b) such that $C_{\boldsymbol{z}} = 0$ in (3.20c).

Pressure Space: Notice that for certain preconditioning methods, which are explained in Section 5.2, some matrices from the NSE model are assembled on the pressure space as well using the $\mathcal{P}_1$ finite elements, which leads to $M_{\boldsymbol{p}}$, $S_{\boldsymbol{p}}$, and $K_{\boldsymbol{p}}(\vec{w})$ all being defined over $\mathbb{R}^{n_{\boldsymbol{p}} \times n_{\boldsymbol{p}}}$. These pressure space matrices are assembled over the same finite element space as used for the DCE but no nodes are removed due to homogeneous Dirichlet BC. This leads to slightly different properties, as explained in the next section.

All discretizations for Stokes, NSE, CFM, and the pressure space are performed in the finite element flow solver NAVIER [12].

3.6. System Properties

The above proposed matrix systems arise from discretizing the different scenarios. All these systems share special properties and fit into certain structures. In this section, these properties are reviewed. Afterwards, two different structures are introduced and their usage is explained.

Using the above explained finite element spaces, the mass matrices of all systems are by definition spd. As shown in [44], the stiffness matrices, however, are spsd, which means the matrix has at least one zero eigenvalue. This can have negative effects on certain computations and should be circumvented. Fortunately, by removing the nodes related to the homogeneous Dirichlet BC as for the velocity and concentration spaces, the stiffness matrices become spd as well. Furthermore, the pressure space stiffness matrix $S_{\boldsymbol{p}}$ artificially becomes spd by pinning a boundary node as suggested in [44]. The convection and reaction matrices are in general not symmetric, especially $R_{\boldsymbol{z}}(c^{(\vec{w})})$ which is rectangular.

All matrices connected to the velocity space are partitioned due to the two spatial dimensions. Thereby, the mass, stiffness, and convection matrices are block diagonal matrices, e.g., for the mass matrix, of the form

$$M = \begin{bmatrix} M_{v_{x_1},v_{x_1}} & 0 \\ 0 & M_{v_{x_2},v_{x_2}} \end{bmatrix},$$

which decouples the velocity components of the different space directions. Only the reaction matrix $R_{\boldsymbol{z}}(\vec{w})$ couples all components and is of the form

$$R_{\boldsymbol{z}}(\vec{w}) = \begin{bmatrix} R_{v_{x_1},v_{x_1}}(w_{x_1}) & R_{v_{x_1},v_{x_2}}(w_{x_2}) \\ R_{v_{x_2},v_{x_1}}(w_{x_1}) & R_{v_{x_2},v_{x_2}}(w_{x_2}) \end{bmatrix}.$$

The rectangular coupling matrices G, $R_{\boldsymbol{z}}(c^{(\vec{w})})$ are of the forms

$$G = \begin{bmatrix} G_{p,v_{x_1}} \\ G_{p,v_{x_2}} \end{bmatrix}, \qquad R_{\boldsymbol{z}}(c^{(\vec{w})}) = \begin{bmatrix} R_{v_{x_1},c^{(\vec{z})}}(c^{(\vec{w})}) & R_{v_{x_2},c^{(\vec{z})}}(c^{(\vec{w})}) \end{bmatrix}.$$

Some of these properties are used in more detail in the remainder of this thesis.

Using the above introduced matrix systems, the first structure defines a generalized dynamical system $\widehat{\boldsymbol{\Phi}}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}; \boldsymbol{M})$ as introduced in Subsection 2.3.4. Despite the block structure of the involved matrices, the overall dimension of the system is denoted by N. For models involving the Stokes equations or the NSE, the system matrix $\boldsymbol{A}$ is an indefinite saddle point system (SPS) with no zero eigenvalues; see, e.g., [15, 35, 42]. Additionally, the left-hand side matrix $\boldsymbol{M}$, which is often called system mass matrix, is singular. Depending on the number of inputs n_r and number of outputs n_a such systems are called single-input-single-output (SISO) or multiple-input-multiple-output (MIMO) descriptor system. A well known fact in control theory is that systems with larger number of in- or outputs are getting more complicated to solve; see, e.g., [38–40]. In Table 3.1a, the detailed block structure of $\widehat{\boldsymbol{\Phi}}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}; \boldsymbol{M})$ for all considered scenarios is depicted.

If the system mass matrix $\boldsymbol{M}$ is singular, specially tailored methods for DAE need to be considered. As mentioned above, this thesis does not explicitly investigate general-purpose DAE methods, but uses certain techniques that circumvent the DAE character of the system. All arising matrix systems can be written as

$$\widehat{\boldsymbol{\Phi}}_{\mathrm{DAE}}: \qquad \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} \widehat{\boldsymbol{x}}(t) \\ \boldsymbol{p}(t) \end{bmatrix} = \begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{x}}(t) \\ \boldsymbol{p}(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u}(t), \tag{3.22a}$$

$$\boldsymbol{y}(t) = C\,\widehat{\boldsymbol{x}}(t) \tag{3.22b}$$

with $M = M^T \succ 0 \in \mathbb{R}^{n\times n}$, $\widehat{\boldsymbol{x}}(t) \in \mathbb{R}^n$, $\boldsymbol{p}(t) \in \mathbb{R}^m$, $\widehat{G} \in \mathbb{R}^{n\times m}$, and $\operatorname{rank}\left(\widehat{G}\right) = m$. Thereby, the first block row in (3.22a) describes the differential part and the second block row in (3.22a) the algebraic part of the dynamical system $\widehat{\boldsymbol{\Phi}}_{\mathrm{DAE}}$. Notice that the input $\boldsymbol{u}(t)$ and output $\boldsymbol{y}(t)$ are identical to the generalized dynamical system $\widehat{\boldsymbol{\Phi}}$. Systems with this special block structure are DAE of differential index two; see, e.g., [50, 133]. As denoted in Subsection 2.2.3, the matrix pencil

$$\left(\begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}; \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right) \tag{3.23}$$

has $n - m$ finite eigenvalues $\lambda_i \in \mathbb{C}$ and $2m$ infinite eigenvalues $\lambda_\infty = \infty$, compare [15, 35]. Hence, the dynamical part of the system is of dimension $n - m$ and does not coincide with the differential part. The detailed block definitions of the structure (3.22) for the different scenarios is depicted in the lower part of Table 3.1b. Thereby, the CTP scenario is not mentioned explicitly since its mass matrix is not singular and the entire system only contains the dynamical part. This means the CTP system is not of DAE character. Nevertheless, the CTP scenario can be seen as a special case of (3.22) with $m = 0$ and all following results can be applied in a simplified version.

Based on the block structure (3.22), the feedback stabilization approach for index-2 DAE systems is described in detail in the next chapter.

Table 3.1.: Overview of specially structured system formulations.

(a) Detailed block structure of MIMO descriptor system (2.29).

$$\widehat{\boldsymbol{\Phi}}: \qquad \begin{aligned} \boldsymbol{M}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t) \end{aligned}$$

	$\boldsymbol{M} \in \mathbb{R}^{N\times N}$	$\boldsymbol{A} \in \mathbb{R}^{N\times N}$	$\boldsymbol{B} \in \mathbb{R}^{N\times n_r}$	$\boldsymbol{C} \in \mathbb{R}^{n_a\times N}$	$\boldsymbol{x} \in \mathbb{R}^{N}$	$\boldsymbol{u} \in \mathbb{R}^{n_r}$	$\boldsymbol{y} \in \mathbb{R}^{n_a}$	N
CTP	$M_{\boldsymbol{a}}$	$A_{\boldsymbol{a}}$	$B_{\boldsymbol{a}}$	$C_{\boldsymbol{a}}$	$\boldsymbol{a}(t)$	$\boldsymbol{u}_{\boldsymbol{a}}(t)$	$\boldsymbol{y}_{\boldsymbol{a}}(t)$	n
NSE	$\begin{bmatrix} M_{\boldsymbol{z}} & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} A_{\boldsymbol{z}} & G \\ G^T & 0 \end{bmatrix}$	$\begin{bmatrix} B_{\boldsymbol{z}} \\ 0 \end{bmatrix}$	$\begin{bmatrix} C_{\boldsymbol{z}} & 0 \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{p}(t) \end{bmatrix}$	$\boldsymbol{u}_{\boldsymbol{z}}(t)$	$\boldsymbol{y}_{\boldsymbol{z}(t)}$	$n_{\boldsymbol{z}} + n_{\boldsymbol{p}}$
CFM	$\begin{bmatrix} M_{\boldsymbol{z}} & 0 & 0 \\ 0 & M_{\boldsymbol{c}} & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} A_{\boldsymbol{z}} & 0 & G \\ -R_{\boldsymbol{z}}(c^{(\vec{w})}) & A_{\boldsymbol{c}} & 0 \\ G^T & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} B_{\boldsymbol{z}} \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 & C_{\boldsymbol{c}} & 0 \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{c}(t) \\ \boldsymbol{p}(t) \end{bmatrix}$	$\boldsymbol{u}_{\boldsymbol{z}}(t)$	$\boldsymbol{y}_{\boldsymbol{c}}(t)$	$n_{\boldsymbol{z}} + n_{\boldsymbol{c}} + n_{\boldsymbol{p}}$

(b) Detailed block structure for DAE system (3.22).

$$\widehat{\boldsymbol{\Phi}}_{\mathrm{DAE}}: \qquad \begin{aligned} \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} \widehat{\boldsymbol{x}}(t) \\ \boldsymbol{p}(t) \end{bmatrix} &= \begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{x}}(t) \\ \boldsymbol{p}(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= C\,\widehat{\boldsymbol{x}}(t) \end{aligned}$$

	$M \in \mathbb{R}^{n\times n}$	$A \in \mathbb{R}^{n\times n}$	$\widehat{G} \in \mathbb{R}^{n\times m}$	$B \in \mathbb{R}^{n\times n_r}$	$C \in \mathbb{R}^{n_a\times n}$	$\widehat{\boldsymbol{x}} \in \mathbb{R}^{n}$	n	m
NSE	$M_{\boldsymbol{z}}$	$A_{\boldsymbol{z}}$	G	$B_{\boldsymbol{z}}$	$C_{\boldsymbol{z}}$	$\boldsymbol{z}(t)$	$n_{\boldsymbol{z}}$	$n_{\boldsymbol{p}}$
CFM	$\begin{bmatrix} M_{\boldsymbol{z}} & 0 \\ 0 & M_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} A_{\boldsymbol{z}} & 0 \\ -R_{\boldsymbol{z}}(c^{(\vec{w})}) & A_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} G \\ 0 \end{bmatrix}$	$\begin{bmatrix} B_{\boldsymbol{z}} \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 & C_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{z}(t) \\ \boldsymbol{c}(t) \end{bmatrix}$	$n_{\boldsymbol{z}} + n_{\boldsymbol{c}}$	$n_{\boldsymbol{p}}$

4 Feedback Stabilization for Index-2 DAE Systems

Contents

The fourth chapter is one of the main chapters of this thesis. The goal is to apply the LQR approach from Subsection 2.3.3 to the scenarios introduced in Chapter 3. Thereby, the first major contribution is a framework that can handle the arising matrix systems with index-2 DAE character in an efficient and numerically practicable way that exploits the given structure of the system.

Throughout all considerations, a small number of inputs n_r and outputs n_a is considered such that for all examples $n_a + n_r \ll n$. This means that the LQR approach minimizes a cost functional that considers only a few observations $\boldsymbol{y}(t) \in \mathbb{R}^{n_a}$ and, furthermore, can influence the system only via a few inputs $\boldsymbol{u}(t) \in \mathbb{R}^{n_r}$. In practical applications, the amount of observation and interaction points is often limited such that these assumptions are fulfilled. It is, for example, not possible to observe the velocity and the pressure in each point of a general domain. Due to this assumption, low-rank coefficients arise in the linear and quadratic matrix equations and, therefore, specially tailored low-rank methods can be applied to numerically solve these equations.

For systems with either many inputs or many outputs, special “terminal reductions” techniques, as described in [40], need to be considered. In the case of many inputs and many outputs, no efficient numerical tools are available to handle the arising dense large-scale systems.

To avoid expensive general-purpose DAE methods, an indirect projection method is applied that was introduced in [71]. This method exploits the existing structure of the system and is adapted to the scenarios of Chapter 3. The defining PDEs are first linearized and then discretized such that one ends up with the descriptor system (3.22).

The chapter is organized as follows. In the first section, the LQR approach from Subsection 2.3.3 is modified to fit the descriptor system (3.22). As it turns out, the main computational work is the solution of a GCARE of the form (2.33a). One specific solution strategy, namely the *Kleinman–Newton* method (KNM), is adapted to this problem definition in Section 4.2. Notice that in some literature this method is referred to as Newton–Kleinman method. To be consistent with previous publications of the author, the term KNM is used in this thesis. Subsequent to the KNM, large-scale GCALEs of the form (2.32) arise, which are dealt with in Subsection 4.2.2. The resulting nested iteration method is examined in Section 4.3. To this end, numerical examples show the applicability of this method in Section 4.4.

In summary, this chapter combines various results regarding feedback stabilization of multi-field flow problems from [14, 15, 35] to a generalized framework. A distinction between the different scenarios is only pointed out if significant differences occur.

4.1. Riccati Approach for Index-2 DAE Systems

To apply a Riccati-based feedback stabilization approach to index-2 DAE systems, we follow the analytical results in various publications by Raymond, e.g., [105–109]. Raymond describes a method for feedback boundary stabilization for Navier-Stokes equations of the form (3.8) in 2D in [106] that is extended to the 3D case in [107]. The main idea is to linearize the NSE around a stationary but possibly unstable solution as described in Section 3.4. Raymond shows that the linear feedback derived for the linearized NSE can stabilize the original NSE with an exponentially fast decay if the deviation is small enough and the solution of the nonlinear NSE is in the neighborhood of an equilibrium (compare Subsection 2.3.3). To this end, Raymond applies the so-called *Leray* projector (or *Helmholtz* projector) to ensure that the convergence-free condition (3.8b) is automatically fulfilled and a standard LQR approach can be applied to the projected evolution equation [106, Sec. 4].

Our main goal is to mimic this approach in a simplified version for a finite dimensional representation which is numerically applicable. The central question that arises is how to construct a projection onto the correct subspace such that the solenoidal condition (3.8b) is automatically fulfilled. In [13], Bänsch/Benner suggest the use of the discrete projector from [71], which is suggested there to apply balanced truncation techniques to Stokes-type descriptor systems. In [15, 35], the equivalence of the Leray projector on operator level and the projector from [71] on matrix level is shown. Although this approach was conceived for linearized NSE, it can be extended straightforwardly to the CFM model as described in [14]. Following these observations, a discrete version of the Leray projector, which can also handle general coupled flow problems of the form (3.22), is derived in detail in the following subsection, which is a more detailed description of [15, Sec. 2.3]. All statements are restricted to the 2-dimensional case.

4.1.1. Leray Projection as Discretized Projector

The analytic approach by Raymond is based on the projection of the velocity field $\vec{v}(t,\vec{x})$ from the 2-dimensional space $(L^2(\Omega))^2$ onto the space of divergence-free functions, such that the solenoidal condition (3.8b) is fulfilled by construction, with vanishing normal components on the boundary Γ.

Leray Projector: To define a projector as described above, consider the *Helmholtz–Leray* decomposition [60, Sec. II.3]

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

with

$$\begin{aligned} H(\operatorname{div},0) &:= \{\vec{v} \in (L^2(\Omega))^2 : \operatorname{div}\vec{v} = 0, \vec{v}\cdot\vec{n}_{|\Gamma} = 0\},\\ H(\operatorname{div},0)^{\perp} &:= \{\nabla p : p \in (H^1(\Omega))^2\}. \end{aligned}$$

Using this decomposition, the velocity field $\vec{v} \in (L^2(\Omega))^2$ can be split into the divergence-free part $\vec{v}_{\text{div},0} \in H(\operatorname{div},0)$ and the curl-free part $\nabla p \in H(\operatorname{div},0)^{\perp}$ that fulfill

$$\begin{aligned} \vec{v}_{\text{div},0} + \nabla p &= \vec{v}, && \text{on} \quad \Omega,\\ \operatorname{div}\vec{v}_{\text{div},0} &= 0, && \\ \vec{v}_{\text{div},0}\cdot\vec{n} &= 0, && \text{on} \quad \Gamma. \end{aligned} \tag{4.1}$$

When defining an operator $\mathscr{P} : (L^2(\Omega))^2 \to H(\mathrm{div}, 0)$ that uses the system (4.1) to map $\vec{v} \in (L^2(\Omega))^2$ onto $\vec{v}_{\mathrm{div},0} \in H(\mathrm{div}, 0)$, one ends up with "the Leray projector (for the corresponding boundary conditions)" [60, p. 38]. Notice that "contrary to the usual Helmholtz decomposition [as described in, e.g., [54, 62]], the Helmholtz–Leray decomposition of $\vec{v}$ is unique (up to an additive constant for p)" [60, p. 37]. It can be shown that $\mathscr{P}$ is orthogonal (self-adjoint) [106, Sec. 2.2.] and by its construction it holds that $\operatorname{null}(\mathscr{P}) = H(\mathrm{div}, 0)^{\perp}$ and $\operatorname{range}(\mathscr{P}) = H(\mathrm{div}, 0)$.

Discrete Projection: To determine a discrete equivalent to the Leray projector, the system (4.1) needs to be discretized. Using the matrices from Section 3.5, (4.1) can be written as

$$M_{\boldsymbol{z}}\boldsymbol{v}_{\mathrm{div},0} + G\boldsymbol{p} = M_{\boldsymbol{z}}\boldsymbol{v}, \tag{4.2a}$$

$$G^T\boldsymbol{v}_{\mathrm{div},0} = 0 \tag{4.2b}$$

with $\boldsymbol{v} \in \mathbb{R}^{n_{\boldsymbol{z}}}$ the discretized velocity and $\boldsymbol{v}_{\mathrm{div},0} \in \mathbb{R}^{n_{\boldsymbol{z}}}$ the discretized divergence-free velocity. Notice that $M_{\boldsymbol{z}} = M_{\boldsymbol{z}}^T \succ 0$. This system fulfills the boundary condition $\vec{v} \cdot \vec{n}_{\Gamma} = 0$ on Γ by construction, compare Section 3.5. By multiplying (4.2a) with $G^T M_{\boldsymbol{z}}^{-1}$ from the left and using (4.2b), $\boldsymbol{p}$ can be written explicitly as

$$\boldsymbol{p} = (G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T \boldsymbol{v}$$

such that (4.2a) can be transformed into

$$\begin{aligned} M_{\boldsymbol{z}}\boldsymbol{v}_{\mathrm{div},0} + G(G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T \boldsymbol{v} &= M_{\boldsymbol{z}}\boldsymbol{v}, \\ \boldsymbol{v}_{\mathrm{div},0} &= M_{\boldsymbol{z}}^{-1}(M_{\boldsymbol{z}}\boldsymbol{v} - G(G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T \boldsymbol{v}) \\ &= (I_{n_{\boldsymbol{z}}} - M_{\boldsymbol{z}}^{-1} G (G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T)\boldsymbol{v} \\ &= \mathit{\Pi}^T \boldsymbol{v} \end{aligned}$$

with

$$\mathit{\Pi} := I_{n_{\boldsymbol{z}}} - G(G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T M_{\boldsymbol{z}}^{-1} \in \mathbb{R}^{n_{\boldsymbol{z}} \times n_{\boldsymbol{z}}}. \tag{4.3}$$

Thus, $\mathit{\Pi}^T$ seems to be a candidate for the discrete version of the Leray projector $\mathscr{P}$. The projection matrix $\mathit{\Pi}$, as defined in (4.3), has been derived in [71] in the context of balanced truncation model order reduction for Stokes-type systems and various properties of $\mathit{\Pi}$ have been stated there. Since, in fact, $\mathit{\Pi}^T$ seems to be the discrete Leray projection, its properties are examined. It is easy to verify that $\left(\mathit{\Pi}^T\right)^2 = \mathit{\Pi}^T$. Furthermore, it can be shown that

$$\operatorname{null}\left(\mathit{\Pi}^T\right) = \operatorname{range}\left(M_{\boldsymbol{z}}^{-1} G\right) \quad \text{and} \quad \operatorname{range}\left(\mathit{\Pi}^T\right) = \operatorname{null}\left(G^T\right), \tag{4.4}$$

which represent the discretized versions of the curl-free components $H(\mathrm{div}, 0)^{\perp}$ and of the divergence-free components $H(\mathrm{div}, 0)$, respectively. In [71], it is stated that $\mathit{\Pi}$ is an oblique projector since $\mathit{\Pi} \neq \mathit{\Pi}^T$. This contradicts the property of the Leray projector

$\mathscr{P}$ being orthogonal. However, the discrete equivalent of the $(L^2(\Omega))^2$ inner product is the $M_{\boldsymbol{z}}$-inner product, defined for $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{R}^{n_{bz}}$ as

$$< \boldsymbol{v}_1, \boldsymbol{v}_2 >_{M_{\boldsymbol{z}}} := (M_{\boldsymbol{z}}\boldsymbol{v}_1, \boldsymbol{v}_2) = \boldsymbol{v}_1^T M_{\boldsymbol{z}} \boldsymbol{v}_2 \in \mathbb{R}. \tag{4.5}$$

Using this definition, it can be shown that

$$\begin{aligned} < \Pi^T \boldsymbol{v}_1, \boldsymbol{v}_2 >_{M_{\boldsymbol{z}}} &= \boldsymbol{v}_1^T (I_{n_{\boldsymbol{z}}} - G(G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T M_{\boldsymbol{z}}^{-1}) M_{\boldsymbol{z}} \boldsymbol{v}_2 \\ &= \boldsymbol{v}_1^T M_{\boldsymbol{z}} (I_{n_{\boldsymbol{z}}} - M_{\boldsymbol{z}}^{-1} G (G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T) \boldsymbol{v}_2 \\ &= < \boldsymbol{v}_1, \Pi^T \boldsymbol{v}_2 >_{M_{\boldsymbol{z}}} . \end{aligned}$$

This verifies that Π^T is orthogonal with respect to the M-inner product and that

$$M_{\boldsymbol{z}} \Pi^T = \Pi M_{\boldsymbol{z}}, \tag{4.6}$$

as stated in [71, p. 1041].

Taking into account all these properties of Π^T and the fact that the projector onto a specific subspace is unique, it follows that Π^T is the discrete version of the Leray projector $\mathscr{P}$ and it holds that

$$\mathscr{P}(\vec{v}) = \vec{v}_{\text{div},0} \quad \Leftrightarrow \quad \Pi^T \boldsymbol{v} = \boldsymbol{v}_{\text{div},0}. \tag{4.7}$$

The discrete projector Π^T is a large-scale dense matrix in $\mathbb{R}^{n_{\boldsymbol{z}} \times n_{\boldsymbol{z}}}$ that cannot be formed explicitly and one tries to avoid its usage whenever possible. Nevertheless, there are certain situation, where the projection $\boldsymbol{v} \mapsto \Pi^T \boldsymbol{v} = \boldsymbol{v}_{\text{div},0}$ or a projection of the form $\boldsymbol{f} \mapsto \Pi \boldsymbol{f} = \widetilde{\boldsymbol{f}}$ needs to be performed. Both can be performed efficiently as described in the next lemma.

Lemma 4.1 *For $\boldsymbol{v}, \boldsymbol{f} \in \mathbb{R}^{n_{\boldsymbol{z}}}$ and (4.3), the following equivalences hold.*

$$\boldsymbol{v} \mapsto \Pi^T \boldsymbol{v} = \boldsymbol{v}_{\text{div},0} \quad \Leftrightarrow \quad \begin{bmatrix} M_{\boldsymbol{z}} & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{\text{div},0} \\ * \end{bmatrix} = \begin{bmatrix} M_{\boldsymbol{z}} \boldsymbol{v} \\ 0 \end{bmatrix}. \tag{4.8}$$

$$\boldsymbol{f} \mapsto \Pi \boldsymbol{f} = \widetilde{\boldsymbol{f}} \quad \Leftrightarrow \quad \begin{aligned} \begin{bmatrix} M_{\boldsymbol{z}} & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{v}} \\ * \end{bmatrix} &= \begin{bmatrix} \boldsymbol{f} \\ 0 \end{bmatrix}, \\ \widetilde{\boldsymbol{f}} &= M_{\boldsymbol{z}} \widetilde{\boldsymbol{v}}. \end{aligned} \tag{4.9}$$

This means, the multiplications with the dense projectors Π and Π^T from the left can be performed implicitly by solving a $(n_{\boldsymbol{z}} + n_{\boldsymbol{p}})$-dimensional sparse system and by performing one multiplication with $M_{\boldsymbol{z}}$. Notice that the lower part of the solution is denoted by $*$ as a placeholder for a term not being used.

Proof. The equivalence in (4.8) follows by construction from the definition of Π^T in (4.3). Hence, a given velocity $\boldsymbol{v}$ is projected onto the (discretely) divergence-free velocity

$\boldsymbol{v}_{\text{div},0}$. To show the equivalence in (4.9), which does not have a special physical meaning, consider

$$\begin{aligned} \widetilde{\boldsymbol{f}} = \Pi \boldsymbol{f} = \Pi M_{\boldsymbol{z}} M_{\boldsymbol{z}}^{-1} \boldsymbol{f} = M_{\boldsymbol{z}} \Pi^T M_{\boldsymbol{z}}^{-1} \boldsymbol{f}, \\ \Leftrightarrow \quad M_{\boldsymbol{z}}^{-1} \widetilde{\boldsymbol{f}} =: \widetilde{\boldsymbol{v}} = \Pi^T (M_{\boldsymbol{z}}^{-1} \boldsymbol{f}), \end{aligned}$$

where (4.8) can be applied. Hence, $\widetilde{\boldsymbol{f}} = M_{\boldsymbol{z}} \widetilde{\boldsymbol{v}}$ can be computed.

□

As shown in [71, Sec. 3], null (Π) is an $n_{\boldsymbol{p}}$-dimensional subspace and Π can be decomposed into

$$\Pi = \Theta_l \Theta_r^T \quad \text{with} \quad \Theta_l^T \Theta_r = I_{n_{\boldsymbol{z}}}, \; \Theta_l, \Theta_r \in \mathbb{R}^{n_{\boldsymbol{z}} \times (n_{\boldsymbol{z}} - n_{\boldsymbol{p}})}.$$

Such a decomposition exists for any projector; see, e.g., [113, Sec. 1.12.2].

The statements from this subsection are adapted to be used for generalized index-2 DAE in the following.

Generalized Projection: For generalized systems of the form (3.22), one can straightforwardly define the generalized projector

$$\widehat{\Pi} := I_N - \widehat{G}(\widehat{G}^T M^{-1} \widehat{G})^{-1} \widehat{G}^T M^{-1}. \tag{4.10}$$

Thereby, only the structural information of (3.22) is considered and the derivation from above is adopted. For the different scenarios one ends up with the following versions:

- The CTP scenario (3.19) does not have any algebraic constraints such that $m = 0$ and $\widehat{G}$ is an empty matrix that yields

$$\widehat{\Pi}_{CTP} := I_n \quad \text{and} \quad \widehat{\Pi}_{CTP}^T \boldsymbol{a} = \boldsymbol{a}. \tag{4.11a}$$

- The discretized flow model (3.20) is the above described system for which this projection framework has been described in [15, 35] such that

$$\widehat{\Pi}_{NSE} := \Pi \quad \text{and} \quad \widehat{\Pi}_{NSE}^T \boldsymbol{z} = \boldsymbol{z}_{\text{div},0}. \tag{4.11b}$$

- Following the specifications in [14], the projector for the CFM scenario can be written as

$$\begin{aligned} \widehat{\Pi}_{CFM} &:= \begin{bmatrix} I_{n_{\boldsymbol{z}}} & 0 \\ 0 & I_{n_{\boldsymbol{c}}} \end{bmatrix} - \begin{bmatrix} G \\ 0 \end{bmatrix} \left(\begin{bmatrix} G^T & 0 \end{bmatrix} \begin{bmatrix} M_{\boldsymbol{z}}^{-1} & 0 \\ 0 & M_{\boldsymbol{c}}^{-1} \end{bmatrix} \begin{bmatrix} G \\ 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} G^T & 0 \end{bmatrix} \begin{bmatrix} M_{\boldsymbol{z}}^{-1} & 0 \\ 0 & M_{\boldsymbol{c}}^{-1} \end{bmatrix} \\ &= \begin{bmatrix} I_{n_{\boldsymbol{z}}} - G(G^T M_{\boldsymbol{z}}^{-1} G)^{-1} G^T M_{\boldsymbol{z}}^{-1} & 0 \\ 0 & I_{n_{\boldsymbol{c}}} \end{bmatrix} \\ &= \begin{bmatrix} \Pi & 0 \\ 0 & I_{n_{\boldsymbol{c}}} \end{bmatrix}, \end{aligned} \tag{4.11c}$$

such that

$$\widehat{\Pi}_{CFM}^T \widehat{\boldsymbol{x}} = \begin{bmatrix} \Pi & 0 \\ 0 & I_{n_{\boldsymbol{c}}} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{c} \end{bmatrix} = \begin{bmatrix} \Pi^T \boldsymbol{z} \\ \boldsymbol{c} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_{\mathrm{div},0} \\ \boldsymbol{c} \end{bmatrix}, \tag{4.11d}$$

which shows that only the velocity part is projected, equivalent to the Stokes and NSE cases, but with the concentration remaining unchanged.

The next subsection describes the generalized framework to apply the LQR approach from Subsection 2.3.3 to systems of the form (3.22) using the projector (4.10).

4.1.2. Projected LQR Problem

The main idea in [71] is to project a dynamical system with algebraic constraints onto a generalized dynamical system with an invertible left-hand side, where the solution automatically fulfills the algebraic constraints. Hence, the DAE character of (3.22) is circumvented and various methods for generalized state-space systems can be applied. Following the transformation steps in [71, Sec. 3], adopted to the generalized system $\widehat{\Phi}_{\mathrm{DAE}}$, and using the generalized projector (4.10), the system (3.22) reduces to

$$\widehat{\Pi} M \widehat{\Pi}^T \frac{\mathrm{d}}{\mathrm{d}t} \widehat{\boldsymbol{x}}(t) = \widehat{\Pi} A \widehat{\Pi}^T \widehat{\boldsymbol{x}}(t) + \widehat{\Pi} B \boldsymbol{u}(t), \tag{4.12a}$$

$$\boldsymbol{y}(t) = C \widehat{\Pi}^T \widehat{\boldsymbol{x}}(t) \tag{4.12b}$$

with $\widehat{\Pi}^T \widehat{\boldsymbol{x}}(t) = \widehat{\boldsymbol{x}}(t)$. Thereby, $\widehat{\Pi}^T$ ensures that the solution $\widehat{\boldsymbol{x}}(t)$ fulfills the algebraic constraints $\widehat{G}^T \widehat{\boldsymbol{x}}(t) = 0$ and "simultaneously resides in the correct solution manifold, the so-called *hidden manifold* [133]" [35, p. 3], defined by

$$\begin{aligned} 0 &= \widehat{G}^T M^{-1} A \widehat{\boldsymbol{x}}(t) + \widehat{G}^T M^{-1} \widehat{G} \boldsymbol{p}(t) + \widehat{G}^T M^{-1} B \boldsymbol{u}(t) \\ &= G^T M_{\boldsymbol{z}}^{-1} A_{\boldsymbol{z}} \boldsymbol{z}(t) + G^T M_{\boldsymbol{z}}^{-1} G \boldsymbol{p}(t) + G^T M_{\boldsymbol{z}}^{-1} B_{\boldsymbol{z}} \boldsymbol{u}_{\boldsymbol{z}}(t). \end{aligned} \tag{4.13}$$

Notice that the hidden manifold is correctly restricted to the velocity and pressure spaces. "If required, $\boldsymbol{p}(t)$ can be computed from (4.13)."[71, p. 1042] Although the system (4.12) has no explicit algebraic constraints anymore, the left-hand side $\widehat{\Pi} M \widehat{\Pi}^T$ has an $n_{\boldsymbol{p}}$-dimensional null-space and cannot be inverted. This can be circumvented by (formally) considering the decomposition

$$\widehat{\Pi} = \widehat{\Theta}_l \widehat{\Theta}_r^T \quad \text{such that} \quad \widehat{\Theta}_l^T \widehat{\Theta}_r = I_{\widetilde{n}} \tag{4.14}$$

with $\widehat{\Theta}_l, \widehat{\Theta}_r \in \mathbb{R}^{n \times \widetilde{n}}$ and the reduced dimension $\widetilde{n} := n - m$. Such a decomposition exists, since (3.22) has $\widetilde{n}$ finite eigenvalues [50, 113]. Using this decomposition, the descriptor system (4.12) can be expressed in terms of $\widetilde{\boldsymbol{x}}(t) = \widehat{\Theta}_l^T \widehat{\boldsymbol{x}}(t) \in \mathbb{R}^{\widetilde{n}}$ as

$$\mathcal{M} \frac{\mathrm{d}}{\mathrm{d}t} \widetilde{\boldsymbol{x}}(t) = \mathcal{A} \widetilde{\boldsymbol{x}}(t) + \mathcal{B} \boldsymbol{u}(t), \tag{4.15a}$$

$$\boldsymbol{y}(t) = \mathcal{C} \widetilde{\boldsymbol{x}}(t) \tag{4.15b}$$

with

$$\mathcal{M} := \widehat{\Theta}_r^T M \widehat{\Theta}_r, \quad \mathcal{A} := \widehat{\Theta}_r^T A \widehat{\Theta}_r, \quad \mathcal{B} := \widehat{\Theta}_r^T B, \quad \mathcal{C} := C \widehat{\Theta}_r. \tag{4.15c}$$

The system (4.15) is a generalized dynamical system of the form $\widehat{\Phi}(\mathcal{A}, \mathcal{B}, \mathcal{C}; \mathcal{M})$ with $\mathcal{M} = \mathcal{M}^T \in \mathbb{R}^{\tilde{n} \times \tilde{n}}$ as introduced in Subsection 2.3.4. Furthermore, the $\tilde{n}$ eigenvalues of the pencil $(\mathcal{A}, \mathcal{M})$ are by construction identical to the finite $n - m$ eigenvalues of the pencil (3.23) as stated in [71]. Following the statements in Subsection 2.3.4, the projected LQR problem can be defined as follows:
Minimize the cost functional

$$\mathcal{J}(\widetilde{\boldsymbol{x}}(t), \boldsymbol{u}(t)) = \int_0^\infty \widetilde{\boldsymbol{x}}(t)^T \mathcal{C}^T \mathcal{C} \widetilde{\boldsymbol{x}}(t) + \boldsymbol{u}(t)^T \boldsymbol{u}(t) \,\mathrm{d}t \tag{4.16a}$$

subject to (4.15). As described in Subsection 2.3.4, the solution to this LQR problem is specified by the control law

$$\boldsymbol{u}_*(t) = -\underbrace{\mathcal{B}^T \mathcal{X} \mathcal{M}}_{=:\mathcal{K}^T} \widetilde{\boldsymbol{x}}(t) \in \mathbb{R}^{n_r}, \tag{4.16b}$$

where $\mathcal{X} = \mathcal{X}^T \succeq 0 \in \mathbb{R}^{\tilde{n} \times \tilde{n}}$ is the unique stabilizing solution of the GCARE

$$\mathcal{R}(\mathcal{X}) = \mathcal{C}^T \mathcal{C} + \mathcal{A}^T \mathcal{X} \mathcal{M} + \mathcal{M} \mathcal{X} \mathcal{A} - \mathcal{M} \mathcal{X} \mathcal{B} \mathcal{B}^T \mathcal{X} \mathcal{M}. \tag{4.16c}$$

The main obstacle in solving the LQR problem (4.16) is to determine the solution of the projected GCARE (4.16c) as explained in the next subsection.

An overview of the structure of the involved projected matrices in (4.15) for the different scenarios is given in Table 4.1a.

***Remark* 4.2** *It is important to point out again that the projector $\widehat{\Pi}$ is a dense matrix of dimension $n \times n$, with n the dimension of the finite element space, such that its explicit assembling or the computation of its decomposition might not be feasible in the large-scale case. This means neither the projector, nor its decomposition, nor the projected matrices can be built or stored explicitly for usual (FEM) discretizations. Hence, this strategy is only a theoretical approach and the projection is never performed explicitly as described in detail in [71]. Nevertheless, the following statements are correctly defined for the projected matrices. Later on, in Subsection 4.2.2, it is shown how the projection is done implicitly by using the original sparse matrices that arise from the discretization in Section 3.5 and by solving certain SPSs.*

Before the solution strategy for the GCARE (4.16c) is presented in Section 4.2, the incorporation of the control input is addressed in the next subsection.

4.1.3. Projected Boundary Control Input

In the CTP scenario, the right-hand side $f(t, \vec{x})$ in (3.7) is defined over Ω as external force that influences our PDE. This means, $f(t, \vec{x})$ describes a distributed control input,

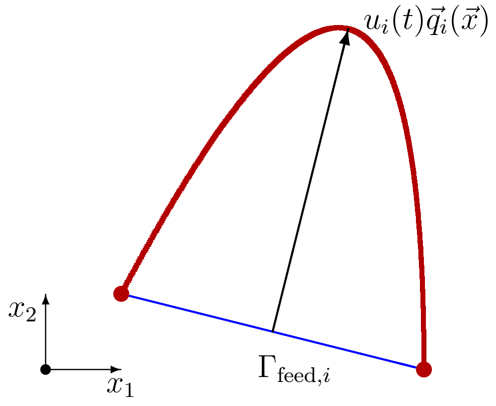


Figure 4.1.: The parabolic in-/outflow profile $\vec{q}_i(\vec{x}) \in \mathbb{R}^2$ at the control boundary part $\Gamma_{\text{feed},i}$ with the intensity controlled by $u_i(t) \in \mathbb{R}$.

where the controller interacts with all particles in the domain Ω or a special control subdomain $\Omega_{\text{feed}} \subset \Omega$. This can be easily assembled within the corresponding finite element space as $B_{\boldsymbol{a}}\boldsymbol{u}_{\boldsymbol{a}}(t)$, where all spatial information is included in $B_{\boldsymbol{a}}$ and the time-varying information defines the control input $\boldsymbol{u}_{\boldsymbol{a}}(t)$; compare, e.g., [26, Sec. 6]. This directly fits the scheme of the generalized dynamical system (2.29).

The main scenarios in this thesis that include an instationary flow process are considered to be stabilized using a boundary control input. In contrast to distributed control, as used for the CTP scenario, the boundary control approach is technically more practicable for any sorts of fluid flow. A common choice for the boundary control are Dirichlet BC. As described in Section 3.5, the control acts only on the velocity space such that for the Stokes/NSE and CFM scenario, one needs to consider non-homogeneous Dirichlet BC for the velocity as control input on some special boundary parts $\Gamma_{\text{feed},i}$ with

$$\bigcup_{i=1}^{n_r} \Gamma_{\text{feed},i} = \Gamma_{\text{feed}} \subset \Gamma_{\text{wall}}. \tag{4.17}$$

Unfortunately, this conflicts with the homogeneous Dirichlet BC that are required to apply the projection ideas from Subsection 4.1.1. A way to circumvent this drawback is described in [15, Sec. 2.4] and is revisited in more detail in this subsection.

The main idea is to construct an operator that distributes the boundary control into the interior of Ω as described in [108] that, additionally, fulfills the solenoidal condition (3.8b). This construction is necessary, since Raymond works on the space $H(\operatorname{div}, 0)$, where all functions have vanishing normal components on the boundary, to define the feedback stabilization as in [106]. In our considerations, only control components in normal direction are used and tangential components are assumed to be zero. Raymond shows in [106] that such a control operator can be used to stabilize two-dimensional flow problems described by the NSE (3.8).

The incorporation of such a normal boundary control for a system of the form (4.15), where $\widetilde{\boldsymbol{x}}(t)$ has zero normal components in a discrete sense, can be done in a simplified but numerically feasible way as follows.

Following the statements in [15, Sec. 2.4], one considers n_r different inputs $u_i(t)$ that operate on different parts of the control boundary Γ_{feed}, i.e., the non-homogeneous Dirichlet inputs can be written as

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

where $\vec{q}_i$ has support on $\Gamma_{\text{feed},i}$ as defined in (4.17). To match the bordering no-slip BC on Γ_{wall}, a parabolic in-/outflow $\vec{q}_i$ is considered, whose intensity is controlled by $u_i(t)$; see Figure 4.1.

The non-zero part $\Theta_r^T B_{\boldsymbol{z}}$ of the control operator $\mathcal{B}$ in (4.15) is the projected version of the velocity input operator $B_{\boldsymbol{z}}$ that has n_r columns, each of which is computed in the following way.

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

1. *Solve the discretized version of the linearized Navier–Stokes equations (3.14) with homogeneous Dirichlet boundary condition except for $\vec{g}_{\text{feed}}$ on $\Gamma_{\text{feed},i}$, where the Dirichlet condition $1 \cdot \vec{q}_i(\vec{x})$ is imposed. Denote the resulting velocity field by $\boldsymbol{v}_i \in \mathbb{R}^{\widetilde{n}_{\boldsymbol{z}}}$.*

2. *Apply the discrete version of the projection (4.1) onto $\boldsymbol{v}_i$ that results in $\widetilde{\boldsymbol{v}}_i(t)$.*

3. *Multiply $\widetilde{\boldsymbol{v}}_i$ by the discrete linearized Navier–Stokes operator from step 1 to get $\widehat{\boldsymbol{v}}_i$.*

4. *Repeating step 2 for $\widehat{\boldsymbol{v}}_i$ and removing all boundary knots, one ends up with the i-th column $\boldsymbol{b}_i \in \mathbb{R}^{n_{\boldsymbol{z}}}$ of $B_{\boldsymbol{z}}$.*

To follow these steps, it is important to mention that one needs the original matrices from the finite element discretization containing all boundary knots defined over $\mathbb{R}^{\widetilde{n}_{\boldsymbol{z}} \times \widetilde{n}_{\boldsymbol{z}}}$ with $\widetilde{n}_{\boldsymbol{z}} > n_{\boldsymbol{z}}$. The specific Dirichlet conditions are incorporated by setting the rows, corresponding to the Dirichlet knots, in each matrix to zero, except the diagonal entry that is set to one. Afterwards, the Dirichlet conditions are explicitly set in the right-hand side vector, which implies the correct Dirichlet conditions in the solution vector. The entire procedure needs to be completed only once during the matrix assembling phase.

The input operator $B_{\boldsymbol{z}}$ mimics the definition of the input operator for the indefinite dimensional space in [108] but leaves aside the following difficulties. First, the BC in (4.1) define only the normal components of $\vec{v}_{\text{div},0}$ as zero, whereas our approach uses homogeneous Dirichlet BC. Secondly, the Leray projector projects onto $H(\text{div},0)$, which is a subspace of $(L^2(\Omega))^2$. However, these functions might not be smooth enough to apply the linearized NSE operator.

Although this strategy is not the exact numerical realization of the approach from Raymond, it shows good results in the numerical experiments and is, therefore, used in the remainder of this thesis. To the author's knowledge, there is no better numerical realization of this input operator available at this time using standard finite elements. A different approach using specially tailored finite elements has been investigated in [34].

Table 4.1.: Matrix structures involved in Kleinman–Newton-ADI for index-2 DAE systems.

(a) Overview of projected matrices in (4.15).

$$\widehat{\boldsymbol{\Phi}}: \qquad \begin{aligned} \mathcal{M}\frac{\mathrm{d}}{\mathrm{d}t}\widetilde{\boldsymbol{x}}(t) &= \mathcal{A}\widetilde{\boldsymbol{x}}(t) + \mathcal{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \mathcal{C}\widetilde{\boldsymbol{x}}(t) \end{aligned}$$

	$\mathcal{M} \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}}$	$\mathcal{A} \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}}$	$\mathcal{B} \in \mathbb{R}^{\widetilde{n}\times n_r}$	$\mathcal{C} \in \mathbb{R}^{n_a\times\widetilde{n}}$	$\widehat{\Theta}_{\{l,r\}} \in \mathbb{R}^{n\times\widetilde{n}}$	$\widetilde{\boldsymbol{x}} \in \mathbb{R}^{\widetilde{n}}$	$\widetilde{n}$
CTP	$M_{\boldsymbol{a}}$	$A_{\boldsymbol{a}}$	$B_{\boldsymbol{a}}$	$C_{\boldsymbol{a}}$	I_n	$\boldsymbol{a}(t)$	n
NSE	$\Theta_r^T M_{\boldsymbol{z}} \Theta_r$	$\Theta_r^T A_{\boldsymbol{z}} \Theta_r$	$\Theta_r^T B_{\boldsymbol{z}}$	$C_{\boldsymbol{z}} \Theta_r$	$\Theta_{\{l,r\}}$	$\Theta_l^T \boldsymbol{z}(t)$	$n_{\boldsymbol{z}} - n_{\boldsymbol{p}}$
CFM	$\begin{bmatrix} \Theta_r^T M_{\boldsymbol{z}} \Theta_r & 0 \\ 0 & M_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} \Theta_r^T A_{\boldsymbol{z}} \Theta_r & 0 \\ -R_{\boldsymbol{z}}(c^{(\vec{w})})\Theta_r & A_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} \Theta_r^T B_{\boldsymbol{z}} \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 & C_{\boldsymbol{c}} \end{bmatrix}$	$\begin{bmatrix} \Theta_{\{l,r\}} & 0 \\ 0 & I_{n_{\boldsymbol{c}}} \end{bmatrix}$	$\begin{bmatrix} \Theta_l^T \boldsymbol{z}(t) \\ \boldsymbol{c}(t) \end{bmatrix}$	$n_{\boldsymbol{z}} - n_{\boldsymbol{p}} + n_{\boldsymbol{c}}$

(b) Block structures in Algorithm 3 using original sparse matrices.

	$\begin{bmatrix} A^T - K^{(k)}B^T + q_\ell M & \widehat{G} \\ \widehat{G} & 0 \end{bmatrix}$	$W^{(k)}$	$\widetilde{V}_\ell$	$K_\ell^{(k+1)}$	$\boldsymbol{u}(t)$
CTP	$A_{\boldsymbol{a}}^T - K^{(k)}B_{\boldsymbol{a}}^T + q_\ell M_{\boldsymbol{a}}$	$\begin{bmatrix} C_{\boldsymbol{a}}^T & K^{(k)} \end{bmatrix}$	$\widetilde{V}_\ell$	$K_{\ell-1}^{(k+1)} + M_{\boldsymbol{a}}\widetilde{V}_\ell\widetilde{V}_\ell^H B_{\boldsymbol{a}}$	$-K^T\boldsymbol{a}(t)$
NSE	$\begin{bmatrix} A_{\boldsymbol{z}}^T - K^{(k)}B_{\boldsymbol{z}}^T + q_\ell M_{\boldsymbol{z}} & G \\ G^T & 0 \end{bmatrix}$	$\begin{bmatrix} C_{\boldsymbol{z}}^T & K^{(k)} \end{bmatrix}$	$\widetilde{V}_\ell$	$K_{\ell-1}^{(k+1)} + M_{\boldsymbol{z}}\widetilde{V}_\ell\widetilde{V}_\ell^H B_{\boldsymbol{z}}$	$-K^T\boldsymbol{z}(t)$
CFM	$\begin{bmatrix} A_{\boldsymbol{z}}^T - K_{\boldsymbol{z}}^{(k)}B_{\boldsymbol{z}}^T + q_\ell M_{\boldsymbol{z}} & -R_{\boldsymbol{z}}(c^{(\vec{w})})^T & G \\ -K_{\boldsymbol{c}}^{(k)}B_{\boldsymbol{z}}^T & A_{\boldsymbol{z}}^T + q_\ell M_{\boldsymbol{c}} & 0 \\ G^T & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & K_{\boldsymbol{z}}^{(k)} \\ C_{\boldsymbol{c}}^T & K_{\boldsymbol{c}}^{(k)} \end{bmatrix}$	$\begin{bmatrix} \widetilde{V}_{\boldsymbol{z},\ell} \\ \widetilde{V}_{\boldsymbol{c},\ell} \end{bmatrix}$	$\begin{bmatrix} K_{\boldsymbol{z},\ell-1}^{(k+1)} + M_{\boldsymbol{z}}\widetilde{V}_{\boldsymbol{z},\ell}\widetilde{V}_{\boldsymbol{z},\ell}^H B_{\boldsymbol{z}} \\ K_{\boldsymbol{c},\ell-1}^{(k+1)} + M_{\boldsymbol{c}}\widetilde{V}_{\boldsymbol{c},\ell}\widetilde{V}_{\boldsymbol{z},\ell}^H B_{\boldsymbol{z}} \end{bmatrix}$	$-\big(K_{\boldsymbol{z}}^T\boldsymbol{z}(t) + K_{\boldsymbol{c}}^T\boldsymbol{c}(t)\big)$

4.2. Solving the Generalized Algebraic Riccati Equation

The GCARE (4.16c) is a quadratic equation in $\mathcal{X} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$ as introduced in Subsection 2.3.2 for standard state-space systems and in Subsection 2.3.4 for generalized state-space systems. The Riccati operator $\mathcal{R} : \mathbb{R}^{\tilde{n}\times\tilde{n}} \to \mathbb{R}^{\tilde{n}\times\tilde{n}}$ is twice Fréchet differentiable and its derivatives can be defined as

$$\mathcal{R}'(\mathcal{X})\mathcal{N} := (\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X})^T\mathcal{N}\mathcal{M} + \mathcal{M}\mathcal{N}(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}), \tag{4.18a}$$

$$\mathcal{R}''(\mathcal{X})(\mathcal{N}_1, \mathcal{N}_2) := -\mathcal{M}\mathcal{N}_1\mathcal{B}\mathcal{B}^T\mathcal{N}_2\mathcal{M} - \mathcal{M}\mathcal{N}_2\mathcal{B}\mathcal{B}^T\mathcal{N}_1\mathcal{M}; \tag{4.18b}$$

see, e.g., [5, 85]. Due to the quadratic character of $\mathcal{R}(\mathcal{X})$, the Fréchet derivative of order two is independent of $\mathcal{X}$ and one can reformulate the Riccati operator applied to $\widetilde{\mathcal{X}}$ exactly by using the Taylor series

$$\mathcal{R}(\widetilde{\mathcal{X}}) = \mathcal{R}(\mathcal{X}) + \mathcal{R}'(\mathcal{X})(\widetilde{\mathcal{X}} - \mathcal{X}) + \frac{1}{2}\mathcal{R}''(\mathcal{X})(\widetilde{\mathcal{X}} - \mathcal{X}, \widetilde{\mathcal{X}} - \mathcal{X}). \tag{4.19}$$

One way to solve (4.16c) iteratively is to apply Newton's method from Subsection 2.4.1 specially tailored to the structure of (4.16c) as depicted in the following.

4.2.1. Kleinman–Newton Method (KNM)

Applying Newton's method to the GCARE (4.16c) yields in the k+1-st step

$$\mathcal{R}'(\mathcal{X}^{(k)})\mathcal{S}^{(k)} = -\mathcal{R}(\mathcal{X}^{(k)}), \tag{4.20a}$$

where (4.20a) is a GCALE in $\mathcal{S}^{(k)}$ as defined in (2.19). Hence, the increment $\mathcal{S}^{(k)} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$ updates the solution via

$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} + \mathcal{S}^{(k)}. \tag{4.20b}$$

This standard formulation is preferred if direct solution methods are used to solve (4.20a); see, e.g., [25, p. 101]. In [81], Kleinman introduced a reformulation of (4.20) that directly iterates on the solution, i.e.,

$$\mathcal{R}'(\mathcal{X}^{(k)})\mathcal{X}^{(k+1)} = \mathcal{R}'(\mathcal{X}^{(k)})\mathcal{X}^{(k)} - \mathcal{R}(\mathcal{X}^{(k)}).$$

Using (4.18a), this can be written as

$$\begin{aligned}
&\left(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M}\right)^T \mathcal{X}^{(k+1)}\mathcal{M} + \mathcal{M}\mathcal{X}^{(k+1)}\left(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M}\right)\\
&= \left(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M}\right)^T \mathcal{X}^{(k)}\mathcal{M} + \mathcal{M}\mathcal{X}^{(k)}\left(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M}\right)\\
&\quad - (\mathcal{C}^T\mathcal{C} + \mathcal{A}^T\mathcal{X}^{(k)}\mathcal{M} + \mathcal{M}\mathcal{X}^{(k)}\mathcal{A} - \mathcal{M}\mathcal{X}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M})\\
&= -\mathcal{C}^T\mathcal{C} - \mathcal{M}\mathcal{X}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}\mathcal{M}.
\end{aligned}$$

Algorithm 2 Kleinman–Newton method [81]

```
Input: $\mathcal{A}$, $\mathcal{M}$, $\mathcal{B}$, $\mathcal{C}$, $tol_{\text{Newton}}$, and initial stabilizing feedback $\mathcal{K}^{(0)}$
Output: unique stabilizing solution $\mathcal{X}^{(*)}$ of CARE (4.16c)
1: Set $k = 0$.
2: while $||\mathcal{R}(\mathcal{X}^{(k)})|| > tol_{\text{Newton}}$ do
3:     Set $\mathcal{A}^{(k)} = \mathcal{A} - \mathcal{B}\left(\mathcal{K}^{(k)}\right)^T$, $\mathcal{W}^{(k)} = \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix}$.
4:     Compute $\mathcal{X}^{(k+1)}$ that solves:
           $\left(\mathcal{A}^{(k)}\right)^T \mathcal{X}^{(k+1)}\mathcal{M} + \mathcal{M}\mathcal{X}^{(k+1)}\mathcal{A}^{(k)} = -\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T.$
5:     $k = k + 1$
6:     $\mathcal{K}^{(k)} = \mathcal{M}\mathcal{X}^{(k)}\mathcal{B}$
7: end while
8: $\mathcal{X}^{(*)} = \mathcal{X}^{(k)}$
```

Using the *feedback matrix* $\mathcal{K}^{(k)} := \mathcal{M}\mathcal{X}^{(k)}\mathcal{B} \in \mathbb{R}^{\widetilde{n}\times n_r}$ as defined in (2.33), which is often also called *feedback gain* or simply *gain matrix*, this equation can be written more compactly as

$$\left(\mathcal{A}^{(k)}\right)^T \mathcal{X}^{(k+1)}\mathcal{M} + \mathcal{M}\mathcal{X}^{(k+1)}\mathcal{A}^{(k)} = -\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T. \tag{4.21}$$

Thereby, the closed-loop matrix $\mathcal{A}^{(k)} := \mathcal{A} - \mathcal{B}\left(\mathcal{K}^{(k)}\right)^T$ and the right-hand side factor $\mathcal{W}^{(k)} := \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix} \in \mathbb{R}^{\widetilde{n}\times(n_a+n_r)}$ define a GCALE of the form (2.32) with a low-rank right-hand side, since $n_a + n_r \ll \widetilde{n}$ in all considered examples. If $\Lambda(\mathcal{A},\mathcal{M}) \subset \mathbb{C}^-$, i.e., the matrix pencil is stable, then $\mathcal{K}^{(0)} = 0$, as discussed in detail in Subsection 4.2.3. Hence, the size of $\mathcal{W}^{(k)}$ can be different. To simplify notation, this variation in size is not reflected further. The GCALE (4.21) has to be solved in every Newton step.

It can be shown that if $(\mathcal{A},\mathcal{B};\mathcal{M})$ is stabilizable and $(\mathcal{C},\mathcal{A};\mathcal{M})$ is detectable, then the iteration over (4.21) converges q-quadratically towards the unique spsd stabilizing solution of (4.16c) considering an initial iterate $\mathcal{X}^{(0)}$ or an initial feedback $\mathcal{K}^{(0)}$ that is stabilizing. Furthermore, the sequence of iterates satisfies

$$\mathcal{X}^{(1)} \succeq \mathcal{X}^{(2)} \succeq \cdots \succeq \mathcal{X}^{(*)} \succeq 0; \tag{4.22}$$

see, e.g., [81] using $\mathcal{K}^{(0)}$, [85, Thm. 9.2.1] using $\mathcal{X}^{(0)}$, or [31, Thm. 1] as a summary of both cases. The entire process is known as the *Kleinman–Newton* method (KNM) and is depicted in Algorithm 2. Thereby, most computational work is done by solving (4.21). An efficient way to solve this GCALE, exploiting the low-rank structure of the right-hand side, is the low-rank ADI iteration as introduced in Subsection 2.4.2. The next subsection introduces the necessary adaptions of the low-rank ADI method to handle the projected GCALE (4.21).

4.2.2. ADI Method Applied to Projected GCALEs

Similar to the Kleinman–Newton iteration, one can straightforwardly write the G-LRCF-ADI iteration as depicted in Algorithm 1 for the GCALE (4.21) using the projected and dense matrices. The main computational work in Algorithm 1 is done in Lines 2 and 5 by solving with $\left((\mathcal{A}^{(k)})^T + q_\ell \mathcal{M}\right)$ for either $\mathcal{W}^{(k)} = \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix}$ for $\ell = 1$ or $\mathcal{M}\mathcal{V}_{\ell-1}$ for $\ell > 1$ as right-hand side.

Before discussing the strategy of how to solve these projected linear systems without using the projection explicitly, some important observations from [71, Sec. 4] need to be recalled in a generalized form. Therefore, consider a fixed k in (4.21) to skip the Newton's method index for now.

The projected GCALE (4.21) is of the form [71, eq. 4.1b] that uses the decomposition (4.14). Its solution $\mathcal{X}$ can be transformed into $X = \widehat{\Theta}_r \mathcal{X} \widehat{\Theta}_r^T$ which is the solution of a $\widehat{\Pi}$ projected GCALE like [71, eq. 4.3b] corresponding to (4.12). As stated in [71, eq. 4.2], X is invariant regarding multiplication with $\widehat{\Pi}^T$ from the left and $\widehat{\Pi}$ from the right, such that $X = \widehat{\Pi}^T X \widehat{\Pi}$. Assuming the low-rank decompositions $X = ZZ^H$ and $\mathcal{X} = \mathcal{Z}\mathcal{Z}^H$ it holds that

$$\widehat{\Pi}^T Z = Z \quad \Rightarrow \quad X = ZZ^H = \widehat{\Pi}^T Z Z^H \widehat{\Pi}, \tag{4.23a}$$

$$\widehat{\Theta}_r \mathcal{Z} = Z \quad \Rightarrow \quad X = ZZ^H = \widehat{\Theta}_r \mathcal{Z}\mathcal{Z}^H \widehat{\Theta}_r^T. \tag{4.23b}$$

This means that the low-rank factor Z is $\widehat{\Pi}^T$-invariant, hence, $\operatorname{span}\{Z\} \subset \operatorname{range}\left(\widehat{\Pi}^T\right)$. Using (4.15c) and these implications, the projected feedback matrix $\mathcal{K}$ can be written as

$$\mathcal{K} = \mathcal{M}\mathcal{X}\mathcal{B} = \widehat{\Theta}_r^T M \widehat{\Theta}_r \mathcal{X} \widehat{\Theta}_r^T B = \widehat{\Theta}_r^T M X B := \widehat{\Theta}_r^T K \tag{4.24}$$

with the feedback K that corresponds to the original DAE (3.22).

This enables one to write the projected linear systems that need to be solved within Algorithm 1 in the form

$$\left((\mathcal{A}^{(k)})^T + q_\ell \mathcal{M}\right) \mathcal{V}_\ell = \mathcal{Y}_{\ell-1} \tag{4.25}$$

with the projected right-hand side

$$\mathcal{Y}_{\ell-1} := \begin{cases} \mathcal{W}^{(k)} = \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix} = \widehat{\Theta}_r^T \begin{bmatrix} C^T & K^{(k)}\end{bmatrix} =: \widehat{\Theta}_r^T W^{(k)} = \widehat{\Theta}_r^T Y_0, & \ell = 1, \\ \mathcal{M}\mathcal{V}_{\ell-1} = \widehat{\Theta}_r^T M \widehat{\Theta}_r \mathcal{V}_{\ell-1} = \widehat{\Theta}_r^T M V_{\ell-1} = \widehat{\Theta}_r^T Y_{\ell-1}, & \ell \geq 2 \end{cases}$$
$$= \widehat{\Theta}_r^T Y_{\ell-1}. \tag{4.26}$$

Notice that $\widehat{\Theta}_r \mathcal{Z} = Z$ consists of scaled blocks

$$\widetilde{V}_i = \sqrt{-2\operatorname{Re}(q_i)} V_i, \quad \forall i = 1, \ldots, \ell, \tag{4.27}$$

as defined in (2.37b). Applying (4.23) for each block V_i, it holds that

$$\widehat{\Theta}_r \mathcal{V}_i = V_i = \widehat{\Pi}^T V_i, \quad \forall\, i = 1, \ldots, \ell, \tag{4.28}$$

which means that V_i is $\widehat{\Pi}^T$-invariant and $\operatorname{span}\{V_i\} \subset \operatorname{range}\left(\widehat{\Pi}^T\right), \ \forall\ i = 1, \ldots, \ell$. Hence, the system (4.25) can be written as

$$\begin{aligned}\left(\widehat{\Theta}_r^T A^T \widehat{\Theta}_r - \widehat{\Theta}_r^T K^{(k)} B^T \widehat{\Theta}_r + q_\ell\, \widehat{\Theta}_r^T M \widehat{\Theta}_r\right) \mathcal{V}_\ell &= \widehat{\Theta}_r^T Y_{\ell-1},\\ \widehat{\Theta}_r^T \left(A^T - K^{(k)} B^T + q_\ell\, M\right) \widehat{\Theta}_r \mathcal{V}_\ell &= \widehat{\Theta}_r^T Y_{\ell-1}.\end{aligned}$$

Multiplying this from the left by $\widehat{\Theta}_l$ and using (4.28) yields

$$\widehat{\Pi}\left(A^T - K^{(k)} B^T + q_\ell\, M\right) \widehat{\Pi}^T V_\ell = \widehat{\Pi} Y_{\ell-1}. \tag{4.29}$$

Since $V_\ell = \widehat{\Pi}^T V_\ell$, one can use [71, Lem. 5.2] to compute the solution of (4.29) by solving the SPS

$$\begin{bmatrix} A^T - K^{(k)} B^T + q_\ell\, M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_\ell \\ * \end{bmatrix} = \begin{bmatrix} Y_{\ell-1} \\ 0 \end{bmatrix}. \tag{4.30}$$

Notice that the system (4.30) might be complex due to the possible complex shift q_ℓ.

To this end, the G-LRCF-ADI method can be applied to solve projected Lyapunov equations of the form (4.21) without any explicit projection as stated in, e.g., [15, 35, 71] for systems of the form (3.20) and in [14, 36] for systems that couple (3.20) and (3.21). Thereby, only operations involving the original sparse matrices are needed and the solution also fulfills the solenoidal condition (3.8b) on the discrete level by construction. Furthermore, the feedback matrix can be defined via (4.24) using only the original sparse matrices.

The crucial step for an efficient algorithm is the solution of the complex-valued SPS (4.30) that needs to be solved in every ADI step for a varying shift q_ℓ. As it is pointed out in [15, 35, Sec. 3.1], the upper left block of the SPS (4.30) is, in general, dense due to the low-rank product $K^{(k)} B^T$ such that the main part of (4.30) is also dense. Using [115, Def. 4.2], one can write (4.30) as a *sparse plus low-rank* (splr) system

$$\Bigg(\underbrace{\begin{bmatrix} A^T + q_\ell\, M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}}_{=:\boldsymbol{F}_\ell} - \underbrace{\begin{bmatrix} K^{(k)} \\ 0 \end{bmatrix}}_{=:\boldsymbol{K}^{(k)}} \underbrace{\begin{bmatrix} B^T & 0 \end{bmatrix}}_{=\boldsymbol{B}^T}\Bigg) \underbrace{\begin{bmatrix} V_\ell \\ * \end{bmatrix}}_{=:\boldsymbol{V}_\ell} = \underbrace{\begin{bmatrix} Y_{\ell-1} \\ 0 \end{bmatrix}}_{=:\boldsymbol{Y}_{\ell-1}}$$

that can be illustrated in a more compact form as

$$\left(\boldsymbol{F}_\ell - \boldsymbol{K}^{(k)} \boldsymbol{B}^T\right) \boldsymbol{V}_\ell = \boldsymbol{Y}_{\ell-1}. \tag{4.31}$$

To evaluate (4.31) one can use the *Sherman–Morrison–Woodbury* formula, see, e.g., [63], such that

$$\left(\boldsymbol{F}_\ell - \boldsymbol{K}^{(k)} \boldsymbol{B}^T\right)^{-1} = \left(I_N + \boldsymbol{F}_\ell^{-1} \boldsymbol{K}^{(k)} \left(I_{n_r} - \boldsymbol{B}^T \boldsymbol{F}_\ell^{-1} \boldsymbol{K}^{(k)}\right)^{-1} \boldsymbol{B}^T\right) \boldsymbol{F}_\ell^{-1}.$$

This means one only needs to solve with the sparse matrix $\boldsymbol{F}_\ell \in \mathbb{C}^{N \times N}$ and the small dense matrix $\left(I_{n_r} - \boldsymbol{B}^T \boldsymbol{F}_\ell^{-1} \boldsymbol{K}^{(k)}\right) \in \mathbb{C}^{n_r \times n_r}$. The blue marked additional solves with $\boldsymbol{F}_\ell$

for the right-hand side $\boldsymbol{K}^{(k)}$ can be achieved by adding the n_r columns of $\boldsymbol{K}^{(k)}$ to the right hand side $\boldsymbol{Y}_{\ell-1}$ such that one needs to solve the SPS

$$\begin{bmatrix} A^T + q_\ell M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} \widehat{V}_\ell \\ * \end{bmatrix} = \begin{bmatrix} \widehat{Y}_{\ell-1}^{(k)} \\ 0 \end{bmatrix} \tag{4.32a}$$

with the $N \times (n_a + 2n_r)$ dimensional low-rank matrices

$$\widehat{\boldsymbol{V}}_\ell = \begin{bmatrix} \widehat{V}_\ell \\ * \end{bmatrix} := \begin{bmatrix} \widehat{V}_\ell^Y & \widehat{V}_\ell^K \\ * & * \end{bmatrix} \quad \text{and} \quad \widehat{\boldsymbol{Y}}_{\ell-1}^{(k)} = \begin{bmatrix} \widehat{Y}_{\ell-1}^{(k)} \\ 0 \end{bmatrix} := \begin{bmatrix} Y_{\ell-1} & K^{(k)} \\ 0 & 0 \end{bmatrix}.$$

The original solution V_ℓ of (4.30) can be reconstructed via

$$V_\ell = (I_n + \widehat{V}_\ell^K (I_{n_r} - B^T \widehat{V}_\ell^K)^{-1} B^T) \widehat{V}_\ell^Y. \tag{4.32b}$$

To this end, in every ADI step the sparse SPS (4.32a) needs to be solved. This problem is examined in detail in Chapter 5. Beforehand, in Section 4.4, the *backslash* operator from MATLAB is used, which is a highly efficient sparse direct solver examined in Section 5.1.

Before the G-LRCF-ADI method for projected equations can be combined with the Kleinman–Newton iteration, a remaining problem needs to be addressed.

4.2.3. Initial Feedback

The open problem is the requirement of a stable pencil $(\mathcal{A}^{(k)}, \mathcal{M})$, $\forall k \geq 0$ as stated in Subsection 2.4.2. Using the linearized NSE as in the NSE or CFM scenario, $\Lambda(\mathcal{A}, \mathcal{M})$, which is equivalent to the finite subset of the spectrum of (3.23), can have unstable eigenvalues $\lambda_{us} \in \mathbb{C}^+$; see, e.g., [15, Sec. 3.3]. In this case, before using the G-LRCF-ADI, one needs to determine an initial feedback $K^{(0)} \in \mathbb{R}^{n \times n_r}$ such that

$$\Lambda\left(\begin{bmatrix} A - B\left(K^{(0)}\right)^T & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right) \subset \mathbb{C}^-. \tag{4.33}$$

One way to construct such an initial feedback is based on the ideas in [70] and is depicted in the following by mainly using the statements and notations of [15, Sec. 2.7] adapted to the generalized index-2 DAE systems of the form (3.22).

Most of the $n - m$ finite eigenvalues of (3.23) are stable and only n_{us} eigenvalues are unstable with $n_{us} \ll n$. These unstable finite eigenvalues $\lambda_{us}^{(i)} \in \mathbb{C}^+$ need to be computed together with their corresponding left and right eigenvectors $\boldsymbol{\omega}^{(i)}, \boldsymbol{\psi}^{(i)} \in \mathbb{C}^N$ for $i = 1, \ldots, n_{us}$, i.e., the generalized EVPs

$$\begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \boldsymbol{\psi}^{(i)} = \lambda_{us}^{(i)} \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{\psi}^{(i)} \quad \text{and} \quad \begin{bmatrix} A^T & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \boldsymbol{\omega}^{(i)} = \overline{\lambda}_{us}^{(i)} \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{\omega}^{(i)}$$

need to be solved, as explained in more detail in Subsection 2.2.3, to define the eigentriplet $(\lambda_{us}^{(i)}, \boldsymbol{\psi}^{(i)}, \boldsymbol{\omega}^{(i)})$.

Notice that these unstable eigentriplets are either real or occur as a complex conjugated pair. In the following, the matrix pencil (3.22a) is projected onto the space spanned by the eigenvectors of the unstable eigenvalues. As shown next, there exists a real-valued projection basis, although the unstable eigenvalues and their corresponding eigenvectors might be complex. Considering the i-th eigentriplet $(\lambda_{us}^{(i)}, \boldsymbol{\psi}^{(i)}, \boldsymbol{\omega}^{(i)})$ and the i+1-st eigentriplet $(\overline{\lambda_{us}^{(i)}}, \overline{\boldsymbol{\psi}^{(i)}}, \overline{\boldsymbol{\omega}^{(i)}})$, it holds for $\alpha_1, \alpha_2 \in \mathbb{C}$ that

$$\begin{aligned}
\operatorname{span}\left\{\boldsymbol{\psi}^{(i)}, \overline{\boldsymbol{\psi}^{(i)}}\right\} &= \alpha_1 \boldsymbol{\psi}^{(i)} + \alpha_2 \overline{\boldsymbol{\psi}^{(i)}} \\
&= \alpha_1 \left(\operatorname{Re}\left(\boldsymbol{\psi}^{(i)}\right) + \jmath \operatorname{Im}\left(\boldsymbol{\psi}^{(i)}\right)\right) + \alpha_2 \left(\operatorname{Re}\left(\boldsymbol{\psi}^{(i)}\right) - \jmath \operatorname{Im}\left(\boldsymbol{\psi}^{(i)}\right)\right) \\
&= (\alpha_1 + \alpha_2) \operatorname{Re}\left(\boldsymbol{\psi}^{(i)}\right) + (\alpha_1 \jmath - \alpha_2 \jmath) \operatorname{Im}\left(\boldsymbol{\psi}^{(i)}\right) \\
&= \operatorname{span}\left\{\operatorname{Re}\left(\boldsymbol{\psi}^{(i)}\right), \operatorname{Im}\left(\boldsymbol{\psi}^{(i)}\right)\right\}.
\end{aligned}$$

Thus, by replacing each complex conjugat pair of left or right eigenvectors, e.g., $(\boldsymbol{\psi}^{(i)}, \boldsymbol{\psi}^{(i+1)} := \overline{\boldsymbol{\psi}^{(i)}})$, by the real-valued pair $(\boldsymbol{\psi}^{(i)} := \operatorname{Re}\left(\boldsymbol{\psi}^{(i)}\right), \boldsymbol{\psi}^{(i+1)} := \operatorname{Im}\left(\boldsymbol{\psi}^{(i)}\right))$, one can build a real-valued projection basis defined by the left and right eigenvectors via

$$H_L := \left[\boldsymbol{\omega}^{(1)}, \dots, \boldsymbol{\omega}^{(n_{us})}\right] \in \mathbb{R}^{N \times n_{us}} \quad \text{and} \quad H_R := \left[\boldsymbol{\psi}^{(1)}, \dots, \boldsymbol{\psi}^{(n_{us})}\right] \in \mathbb{R}^{N \times n_{us}}.$$

These projectors are used to build the matrices

$$\widetilde{\boldsymbol{M}} := H_L^T \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} H_R, \quad \widetilde{\boldsymbol{A}} := H_L^T \begin{bmatrix} A & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} H_R, \quad \widetilde{\boldsymbol{B}} := H_L^T \begin{bmatrix} B \\ 0 \end{bmatrix},$$

which define the n_{us}-dimensional GABE

$$\widetilde{\boldsymbol{A}}^T X^{(0)} \widetilde{\boldsymbol{M}} + \widetilde{\boldsymbol{M}}^T X^{(0)} \widetilde{\boldsymbol{A}} - \widetilde{\boldsymbol{M}}^T X^{(0)} \widetilde{\boldsymbol{B}} \widetilde{\boldsymbol{B}}^T X^{(0)} \widetilde{\boldsymbol{M}} = 0, \tag{4.34}$$

whose unique stabilizing solution can be used to define the initial feedback $K^{(0)} \in \mathbb{R}^{n \times n_r}$ as first n rows of

$$\begin{bmatrix} K^{(0)} \\ 0 \end{bmatrix} := \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} H_L X^{(0)} H_L^T \begin{bmatrix} B \\ 0 \end{bmatrix}.$$

The GABE (4.34) is solved using the generalized Newton iteration for the sign-function described in [20]. The resulting closed-loop pencil (4.33) is stable and its spectrum can be defined via (2.27).

Notice that some algorithms compute the eigenvectors in such a way that $\widetilde{\boldsymbol{M}} = I_{n_{us}}$, which means H_L and H_R are bi-orthogonal with respect to the $\boldsymbol{M}$-inner product. In this case, one has to solve an ordinary Bernoulli equation like (2.25). The initial feedback $K^{(0)}$ ensures that the convergence requirements of the Kleinman–Newton method in Subsection 4.2.1 are fulfilled and the GCALE (4.21) has a unique stabilizing solution for $k = 0$ such that all further $\mathcal{A}^{(k)}$ are stable as well.

Notice that in [1], the authors use the unique stabilizing solution of the GABE to stabilize linearized NSE via boundary control. However, such a Bernoulli feedback is computed by penalizing the divergence-free condition without the use of any cost functional. Hence, one cannot improve the quality of the solution and the divergence-free condition is disturb. Furthermore, the GABE method is not able to improve the stability of already stable eigenvalues [1, Sec. 6.6.3]. This is necessary in the practical closed loop simulation as it is shown in the closed loop simulation in Chapter 7.

4.3. Kleinman–Newton-ADI (KN-ADI) for Index-2 DAE Systems

After computing the initial feedback $K^{(0)}$ for unstable pencils (3.23) and assembling the correct input operator $B_{\boldsymbol{z}}$, one can combine the KNM from Subsection 4.2.1 with the G-LRCF-ADI method from Subsection 4.2.2 to form the Kleinman–Newton-ADI (KN-ADI) method for index-2 DAE systems as described in detail in this section. First, a feedback accumulation strategy is shown in the following subsection. Afterwards, different stopping criteria are discussed in Subsection 4.3.2 and a convergence proof is given in Subsection 4.3.3. At the end of this section the complete algorithm is depicted that is used for the numerical examples in Section 4.4.

4.3.1. Feedback Accumulation

In many applications, such as those considered in this thesis, one is not interested in explicitly forming the solution factor Z or even the solution X. One rather seeks a tool to compute the feedback matrix $K \in \mathbb{R}^{n \times n_r}$. This matrix is of low rank and is much cheaper to store than the original solution factor $Z \in \mathbb{C}^{n \times \ell(n_a+n_r)}$. Additionally, the right-hand side in Newton's method in Algorithm 2 requires only K such that the solution X is only used to update the feedback K at the end of each Newton step. Using the G-LRCF-ADI method to solve the Newton step (4.21) yields a low-rank representation of the solution $X = ZZ^H$ as defined in (2.37b) and (4.23). Hence, the feedback $K^{(k+1)}$ in the k-th Newton step can be accumulated in the ℓ-th ADI step using (4.27) via

$$\begin{aligned} K_\ell^{(k+1)} &= MZ_\ell Z_\ell^H B = M\begin{bmatrix}\widetilde{V}_1 & \dots & \widetilde{V}_\ell\end{bmatrix}\left(\begin{bmatrix}\widetilde{V}_1^H\\ \vdots \\ \widetilde{V}_\ell^H\end{bmatrix} B\right) = M\sum_{i=1}^{\ell}\widetilde{V}_i\left(\widetilde{V}_i^H B\right) \\ &= K_{\ell-1}^{(k+1)} + M\widetilde{V}_\ell\left(\widetilde{V}_\ell^H B\right), \ \forall \ell \geq 1 \end{aligned} \tag{4.35}$$

with $K_0^{(k+1)} = 0$.

Depending on the considered scenario, this iterative assembling can be written in more detail as depicted in Table 4.1b; compare, e.g., [14, 15, 31, 35].

4.3.2. Stopping Criteria

Although the Newton step (4.21) can be solved efficiently and the memory requirements can be reduced drastically by using the feedback accumulation from above, one drawback in using the Kleinman–Newton-ADI iteration for projected Riccati equations is the choice of suitable stopping criteria for the inner (ADI) and outer (Kleinman–Newton) iteration. As pointed out in Subsection 2.4, a common way to terminate iterative methods is the residual. To compute the residuals of (4.21) and (4.16c), the application of the projector $\widehat{\Pi}$ is necessary. This can be achieved by using the computations in (4.8) and (4.9) as shown next.

Projected Residual The Euclidean norm of large-scale, quadratic matrices can be computed using the so-called *power iteration*; see, e.g., [63, Sec. 7.3.1]. This method only involves matrix-vector products and, eventually, converges towards the eigenvalue with the largest magnitude. For symmetric matrices its absolute value is the Euclidean norm. Certain subspace acceleration techniques such as the Lanczos method [63, Alg. 9.2.1] (for symmetric matrices) or the Arnoldi method [63, Sec. 9.4] (for general matrices) are useful if the eigenvalues are not sufficiently separated, which slows down the convergence rate of the power iteration, see, e.g., [63, Sec. 7.3.1].

The Lyapunov and Riccati residual are both symmetric for symmetric solutions X, meaning that the Lanczos method is the method of choice. Using the above reformulations, the Lyapunov equation (4.21) and the Riccati equation (4.16c), depending on the low-rank solution ZZ^H, can be written as

$$\mathcal{L}\left(ZZ^H\right) = \widehat{\Pi}\left((A^T - KB^T)ZZ^H M + MZZ^H(A - BK^T) + WW^T\right)\widehat{\Pi}^T, \qquad (4.36a)$$

$$\mathcal{R}\left(ZZ^H\right) = \widehat{\Pi}\left(C^T C + A^T ZZ^H M + MZZ^T A - MZZ^H BB^T ZZ^H M\right)\widehat{\Pi}^T. \qquad (4.36b)$$

Both are of the general form

$$\mathcal{G}\left(ZZ^H\right) = \widehat{\Pi}\widetilde{\mathcal{G}}\left(ZZ^H\right)\widehat{\Pi}^T, \qquad (4.37)$$

where $\widetilde{\mathcal{G}}\left(ZZ^H\right) \in \mathbb{R}^{n\times n}$ is a projector free, symmetric matrix depending on ZZ^H. As mentioned above, the power, Lanczos, and Arnoldi iteration basically only involve multiplications with vectors from the right, which means for a given vector $\boldsymbol{s}_0 \in \mathbb{R}^n$ one has to perform

$$\boldsymbol{s}_1 = \widehat{\Pi}^T\boldsymbol{s}_0, \quad \boldsymbol{s}_2 = \widetilde{\mathcal{G}}\left(ZZ^H\right)\boldsymbol{s}_1, \quad \boldsymbol{s}_3 = \widehat{\Pi}\boldsymbol{s}_2 \qquad (4.38)$$

in each iteration step to apply any of the above mentioned methods. Besides the standard iteration step $\boldsymbol{s}_2 = \widetilde{\mathcal{G}}\left(ZZ^H\right)\boldsymbol{s}_1$, one needs to perform the projections (4.8) and (4.9) in each iteration step. Notice that the Arnoldi and Lanczos iterations converge usually within five to ten iteration steps to the required accuracy. Although this computation only involves the original sparse matrices, its use in every ADI step increases the computation costs drastically, especially in the large-scale case. An efficient way to overcome this drawback is pointed out next.

Relative Change In [31, Sec. 4.2], it is shown that the relative change in the solution factor Z is a suitable stopping criterion since "the norms $||\widetilde{V}_\ell||_F$ tend to decay quite evenly"[31, p. 765]. The square of the norm of this relative change can be defined via

$$\frac{||\widetilde{V}_\ell||_F^2}{||Z_\ell||_F^2} = \frac{||\widetilde{V}_\ell||_F^2}{||Z_{\ell-1}||_F^2 + ||\widetilde{V}_\ell||_F^2}$$

whose accumulation is inexpensive since $\widetilde{V}_\ell$ is of low rank in general.

As mentioned above, in the applications of this thesis one is only interested in computing the feedback $K \in \mathbb{R}^{n \times n_r}$. "Therefore, it seems to be reasonable to stop the iteration as soon as the changes in the matrices $K^{(k)}$ become small or more precisely

$$\frac{||K^{(k)} - K^{(k-1)}||_F}{||K^{(k)}||_F} \leq \varepsilon. \tag{4.39}$$

Here, ε is a tiny, positive constant (e.g., $\varepsilon = \widetilde{n} \cdot \varepsilon_{\text{mach}}$). This criterion is very inexpensive, because $K \in \mathbb{R}^{n \times n_r}$ and $n_r \ll n$."[31, Sec. 5.4] This approach uses the feedback accumulation from Subsection 4.3.1 to circumvent the above mentioned drawback.

The nested KN-ADI iteration for index-2 DAE systems is depicted in Algorithm 3. Thereby, the feedback K is accumulated directly, as described in (4.35). Furthermore, the relative change defined in (4.39) is used as stopping criterion for the inner and outer iteration. Notice that the output matrix C is weighted by a factor $\alpha \in \mathbb{R}^+$ as described at the end of Subsection 2.3.3 that plays an important role in the numerical examples in Section 4.4. The specific block structure for the different scenarios is depicted in detail in Table 4.1b. More details about the convergence of Algorithm 3 are depicted next.

4.3.3. Convergence of KN-ADI for Index-2 DAE Systems

In this subsection, some special properties of the index-2 DAE system (3.22) and a convergence proof for the KNM applied to (4.16c) are given. Due to the DAE structure of (3.22), some parts of Defenition 2.12 need to be extended.

Definition 4.3 (System properties of index-2 DAE $\widehat{\boldsymbol{\Phi}}_{\mathbf{DAE}}$) *For the index-2 DAE (3.22), the following definitions hold.*

*(a) A matrix pencil $(\boldsymbol{A}, \boldsymbol{M})$ in (3.22) is called **stable** if it is regular and all the finite eigenvalues of $(\boldsymbol{A}, \boldsymbol{M})$ lie in the open left half-plane (cf. [123, Def. 3.8]).*

*(b) We call a triple $(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{M})$ **stabilizable** if there is a matrix $K^{(0)} \in \mathbb{R}^{n \times n_r}$ that yields (4.33). A triple $(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{M})$ is called **detectable** if and only if $(\boldsymbol{A}^T, \boldsymbol{C}^T; \boldsymbol{M})$ is stabilizable. Notice that we are only interested in stabilizing trajectories. Therefore, this definition of stabilizable is equivalent to the definition of "behaviorally stabilizable" as defined in [127, Def. 2.2.3 a].*

The GARE (4.16c) is defined over the projected space $\mathbb{R}^{\widetilde{n} \times \widetilde{n}}$. The following lemma connects the projected system $\widehat{\Phi}(\mathcal{A}, \mathcal{B}, \mathcal{C}; \mathcal{M})$ with the original system $\widehat{\Phi}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}; \boldsymbol{M})$ regarding stabilizability and detectability.

Algorithm 3 Kleinman–Newton-ADI (KN-ADI) method for index-2 DAE systems

```
Input: $M, A, \widehat{G}, B, C$, initial feedback $K^{(0)}$, ADI shift parameters
       $q_i \in \mathbb{C}^- : i = 1, \ldots, n_{\mathrm{ADI}}$, $tol_{\mathrm{ADI}}$, $tol_{\mathrm{Newton}}$, and $\alpha \in \mathbb{R}^+$
Output: feedback matrix $K$
 1: for $k = 0, 1, \ldots, k_{\max}$ do
 2:     Set $W^{(k)} = \begin{bmatrix} \alpha C^T & K^{(k)} \end{bmatrix}$.
 3:     Get $V_1$ by solving
            $$\begin{bmatrix} A^T - K^{(k)} B^T + q_1 M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ * \end{bmatrix} = \begin{bmatrix} W^{(k)} \\ 0 \end{bmatrix}.$$
 4:     $K_1^{(k+1)} = -2\,\mathrm{Re}\,(q_1)\, M V_1 V_1^H B$
 5:     for $\ell = 2, 3, \ldots, \ell_{\max}$ do
 6:         Get $V_\ell$ by solving
                $$\begin{bmatrix} A^T - K^{(k)} B^T + q_\ell M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_\ell \\ * \end{bmatrix} = \begin{bmatrix} M V_{\ell-1} \\ 0 \end{bmatrix}.$$
 7:         $\widetilde{V}_\ell = \sqrt{-2\,\mathrm{Re}\,(q_\ell)}\,(V_{\ell-1} - (q_\ell + \overline{q_{\ell-1}}) V_\ell)$
 8:         $K_\ell^{(k+1)} = K_{\ell-1}^{(k+1)} + M \widetilde{V}_\ell \widetilde{V}_\ell^H B$
 9:         if $\left( \frac{\|K_\ell^{(k+1)} - K_{\ell-1}^{(k+1)}\|_F}{\|K_\ell^{(k+1)}\|_F} < tol_{\mathrm{ADI}} \right)$ then
10:             break
11:         end if
12:     end for
13:     $K^{(k+1)} = K_\ell^{(k+1)}$
14:     if $\left( \frac{\|K^{(k+1)} - K^{(k)}\|_F}{\|K^{(k+1)}\|_F} < tol_{\mathrm{Newton}} \right)$ then
15:         break
16:     end if
17: end for
18: $K = K^{(k)}$
```

Lemma 4.4 *The matrix triple* $(\mathcal{A}, \mathcal{B}; \mathcal{M})$ *is stabilizable (*$(\mathcal{C}, \mathcal{A}; \mathcal{M})$ *is detectable) if and only if* $(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{M})$ *is stabilizable (*$(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{M})$ *is detectable).*

Proof. If $(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{M})$ is stabilizable, there exists a $K^{(0)}$ such that

$$\left(\begin{bmatrix} A - B\left(K^{(0)}\right)^T & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right) =: \left(\boldsymbol{A} - \boldsymbol{B}\left(\boldsymbol{K}^{(0)}\right)^T, \boldsymbol{M} \right)$$

is stable. As stated in Subsection 4.1.2, the finite eigenvalues of $(\boldsymbol{A}, \boldsymbol{M})$ are equivalent to the eigenvalues of $(\mathcal{A}, \mathcal{M})$ and, therefore, the finite eigenvalues of $\left(\boldsymbol{A} - \boldsymbol{B}\left(\boldsymbol{K}^{(0)}\right)^T, \boldsymbol{M} \right)$ are equivalent to the eigenvalues of

$$\left(\widehat{\Theta}_r^T \left(A - B\left(K^{(0)}\right)^T \right) \widehat{\Theta}_r,\ \widehat{\Theta}_r^T M \widehat{\Theta}_r \right).$$

Using the definitions (4.15c) and (4.24) yields

$$\widehat{\Theta}_r^T \left(A - B\left(K^{(0)}\right)^T\right)\widehat{\Theta}_r = \mathcal{A} - \mathcal{B}\left(\mathcal{K}^{(0)}\right)^T$$

that shows that $(\mathcal{A}, \mathcal{B}; \mathcal{M})$ is stabilizable. The case for detectability follows by definition.

□

Using these results, the convergence proof for the KNM applied to (4.16c) is stated in the following theorem. This theorem is the main result of this chapter.

Theorem 4.5 *Assume $(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{M})$ is stabilizable and $(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{M})$ is detectable. Then, there exists a maximal symmetric solution $X^{(*)} = \widehat{\Theta}_r \mathcal{X}^{(*)} \widehat{\Theta}_r^T$ with $\mathcal{R}(\mathcal{X}^{(*)}) = 0$ for which*

$$\left(\begin{bmatrix} A - BB^T X^{(*)} M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}\right) \tag{4.40}$$

is stable. Furthermore, for the sequence $\left\{X^{(k)}\right\}_{k=0}^{\infty}$ defined by $X^{(k)} := \widehat{\Theta}_r \mathcal{X}^{(k)} \widehat{\Theta}_r^T$, (4.21), and a symmetric matrix $X^{(0)}$ for which $\left(\boldsymbol{A} - \boldsymbol{B}\left(\boldsymbol{K}^{(0)}\right)^T, \boldsymbol{M}\right)$ is stable, it holds that, for $k \geq 1$,

$$X^{(1)} \succeq X^{(2)} \succeq \cdots \succeq X^{(k)} \succeq 0. \tag{4.41a}$$

Moreover,

$$\lim_{k \to \infty} X^{(k)} = X^{(*)} \tag{4.41b}$$

and, furthermore, there is a constant $0 < \widetilde{\kappa} < \infty$ such that, for $k \geq 1$,

$$||X^{(k+1)} - X^{(*)}||_F \leq \widetilde{\kappa} ||X^{(k)} - X^{(*)}||_F^2. \tag{4.42}$$

Proof. From $(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{M})$ being stabilizable and $(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{M})$ being detectable it follows by Lemma 4.4 that $(\mathcal{A}, \mathcal{B}; \mathcal{M})$ is stabilizable and $(\mathcal{C}, \mathcal{A}; \mathcal{M})$ is detectable. Hence, the GARE (4.16c) has a maximal symmetric solution $\mathcal{X}^{(*)} \succeq 0$ that, on the one hand, stabilizes $(\mathcal{A} - \mathcal{B}\mathcal{B}^T \mathcal{X}^{(*)} \mathcal{M}; \mathcal{M})$ and, on the other hand, defines $X^{(*)} = \widehat{\Theta}_r \mathcal{X}^{(*)} \widehat{\Theta}_r^T$ that stabilizes (4.40) due to Theorem 2.25.

Defining the initial feedback $\mathcal{K}^{(0)} := \widehat{\Theta}_r^T K^{(0)} := \widehat{\Theta}_r^T M X^{(0)} B$ as in (4.24), all requirements of [31, Thm. 1] are fulfilled such that "$\left\{\mathcal{X}^{(k)}\right\}_{k=0}^{\infty}$ is a non-increasing sequence, satisfying

$$\mathcal{X}^{(k)} \succeq \mathcal{X}^{(k+1)} \succeq 0 \tag{4.43}$$

for all $k \geq 1$. Moreover, $\mathcal{X}^{(*)} = \lim_{k \to \infty} \mathcal{X}^{(*)}$ exists and is the unique stabilizing solution of the GARE (4.16c)"[31, Thm. 1(b)]. Multiplying (4.43) from the left by $\widehat{\Theta}_r$, from the right by $\widehat{\Theta}_r^T$, and defining $X^{(k)} := \widehat{\Theta}_r \mathcal{X}^{(k)} \widehat{\Theta}_r^T$ for all $k \geq 0$ yields

$$\begin{aligned} \widehat{\Theta}_r \mathcal{X}^{(k)} \widehat{\Theta}_r^T &\succeq \widehat{\Theta}_r \mathcal{X}^{(k+1)} \widehat{\Theta}_r^T \succeq 0, \\ X^{(k)} &\succeq X^{(k+1)} \succeq 0, \end{aligned}$$

that in the limit $k \to \infty$ implies $X^{(*)} := \lim_{k\to\infty} X^{(k)}$ is symmetric and stabilizes (4.40).

To this end, [31, Thm. 1(c)] yields

$$||\mathcal{X}^{(k+1)} - \mathcal{X}^{(*)}||_F \leq \kappa||\mathcal{X}^{(k)} - \mathcal{X}^{(*)}||_F^2, \quad k \geq 1,\ 0 < \kappa < \infty.$$

Multiplying this with $||\widehat{\Theta}_r||_F < \infty$ from the left and $||\widehat{\Theta}_r^T||_F < \infty$ from the right yields

$$||\widehat{\Theta}_r||_F||\mathcal{X}^{(k+1)} - \mathcal{X}^{(*)}||_F||\widehat{\Theta}_r^T||_F \leq \kappa||\widehat{\Theta}_r||_F^2||\mathcal{X}^{(k)} - \mathcal{X}^{(*)}||_F^2. \tag{4.44}$$

Since the Frobenius norm is a sub-multiplicative norm, the left-hand side of (4.44) can be written as

$$||X^{(k+1)} - X^{(*)}||_F = ||\widehat{\Theta}_r(\mathcal{X}^{(k+1)} - \mathcal{X}^{(*)})\widehat{\Theta}_r^T||_F \leq ||\widehat{\Theta}_r||_F||\mathcal{X}^{(k+1)} - \mathcal{X}^{(*)}||_F||\widehat{\Theta}_r^T||_F.$$

Inserting $I_{\widetilde{n}} = \widehat{\Theta}_l^T\widehat{\Theta}_r$ on the left and $I_{\widetilde{n}} = \widehat{\Theta}_r^T\widehat{\Theta}_l$ on the right of the most right term in (4.44) yields

$$\begin{aligned}\kappa||\widehat{\Theta}_r||_F^2||\widehat{\Theta}_l^T\widehat{\Theta}_r(\mathcal{X}^{(k)} - \mathcal{X}^{(*)})\widehat{\Theta}_r^T\widehat{\Theta}_l||_F^2 &= \kappa||\widehat{\Theta}_r||_F^2||\widehat{\Theta}_l^T(X^{(k)} - X^{(*)})\widehat{\Theta}_l||_F^2 \\ &\leq \underbrace{\kappa||\widehat{\Theta}_r||_F^2||\widehat{\Theta}_l||_F^4}_{=:\widetilde{\kappa}<\infty}||X^{(k)} - X^{(*)}||_F^2\end{aligned}$$

such that one finally ends up with (4.42).

□

***Remark* 4.6** *In [41], Benner and Stykel introduce a similar approach to solve DAE-based GAREs. The main difference between the approach in [41] and the approach examined in this thesis, is the definition of the projector. We use the projector Π^T that projects the solution onto the hidden manifold (4.13). As mentioned above, Π^T is orthogonal with respect to the $M_{\mathbf{z}}$-inner product. In [41], the so-called spectral projectors are used, which project onto the right and left deflating subspaces. Thereby, the right deflating subspace belongs to the finite eigenvalues of the pencil and the left deflating subspace belongs to the infinite eigenvalues. Notice that the spectral projectors are orthogonal in Euclidean inner product. Since the dynamics of the system are determined solely by the finite spectrum, both projection methods project onto the same subspace using different topologies.*

In more detail, the projection idea in [41] can be applied to general DAE defined by a regular pencil as described in [41, Sec. 2]. But "the projectors [...] are required in explicit form [and the] computation of these projectors is, in general, very expensive"[41, p. 590]. In contrast, the projector Π^T is specially designed for the index-2 DAE system arising for fluid flow problems. Thus, the specific structure of the matrices is used to define Π^T and makes it possible to apply the projector efficiently as introduced in Lemma 4.1. Furthermore, the explicit application of the projector Π^T is never used in Algorithm 3 unless one is interested in computing the projected residual explicitly. Notice that "this structure can be exploited to construct the projectors [...] explicitly and cheaply" as well by the method defined in [41]. A comparison of both methods regarding their overall computation costs should be part of future research.

As mentioned above, in many applications one is only interested in the feedback that stabilizes the arising matrix pencil.

Conclusion 4.7 *Under the assumptions of Theorem 4.5, there exists a feedback matrix $\mathcal{K}^{(*)} = \mathcal{M}\mathcal{X}^{(*)}\mathcal{M} = \widehat{\Theta}_r^T K^{(*)}$ such that*

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

is stable and $\boldsymbol{u}_*(t) := -K^{(*)}\widehat{\boldsymbol{x}}(t)$ is minimizing the cost functional (4.16a).

The following section verifies the above results through various numerical examples using the scenarios Stokes/NSE and CFM from Chapter 3. The scenario CTP is not further considered in this numerical tests since the LQR approach for non DAE structured systems is dealt with in various publications in the last decades.

4.4. Numerical Experiments for Feedback Stabilization

The first numerical section of this thesis shows experiments which validate the theoretical results from Chapter 4. All computations are executed within MATLAB version 8.0.0.783 (R2012b) on a 64-bit compute server with an Intel® Xeon® X5650 @2.67GHz processor. This processor has 2 CPUs, in total 12 Cores (6 Cores per CPU), and 48 GB main memory available. More details regarding the used compute server are listed in Appendix A.

The numerical tests from [14, 15, 35] are repeated and extended, since the algorithm has been modified slightly throughout its development time. Hence, all experiments are carried out with the same index-2 DAE structure exploiting algorithm as depicted in Algorithm 3. Most results are identical to the results in [14, 15, 35]. Various other aspects are presented in addition to show unpublished viewpoints that motivate the investigations in Chapter 6.

The core scenario for these numerical tests is the NSE scenario from Subsection 3.2.1. On the one hand, the Stokes system from Subsection 3.2.2 can be seen as a simplified version of the NSE. On the other hand, the CFM system in Section 3.3 introduces an additional block structure that straightforwardly fits into the NSE scheme. Hence, after presenting the NSE results, some selected experiments show the influence of these structure variations.

4.4.1. Parameter Setup for Numerical Examples - Part I

At the beginning of this subsection, the output matrices C for the KVS and the RM are specified. In the KVS setting in Section 3.2, one is interested in avoiding alternating vortexes behind the obstacle $\Omega_{\mathrm{K}}^{\mathrm{O}}$ as described in Subsection 3.2.3. Therefore, the vertical velocity component v_{x_2} is measured on $n_{\boldsymbol{a}} = 7$ observation points $\mathrm{P}_{\mathrm{obs},i}$ as depicted in Figure 3.3. Hence, we define $C_{\boldsymbol{z}} = \{c_{i,j}\}_{i,j=1}^{n_{\boldsymbol{a}},n_{\boldsymbol{z}}} \in \mathbb{R}^{n_{\boldsymbol{a}} \times n_{\boldsymbol{z}}}$ via

$$c_{i,j} = \begin{cases} 1, & \text{if } j \text{ is the index of the } x_2 \text{ component of } \mathrm{P}_{\mathrm{obs},i}, \\ 0, & \text{otherwise.} \end{cases}$$

The output of the CFM scenario on the RM in Section 3.3 has been described in [14]. In detail, one defines

$$q := \int_{\Gamma_{\mathrm{r}}} \partial_{\vec{n}} c^{(\vec{w})} \, \mathrm{ds} \tag{4.45}$$

as the total flux of the stationary concentration $c^{(\vec{w})}$ through the obstacle boundary Γ_{r}. Hence, one can define the cost functional

$$\mathcal{J}(c, \boldsymbol{u_c}(t)) := \frac{1}{2} \int_0^\infty \alpha \left| \int_{\Gamma_{\mathrm{r}}} \partial_{\vec{n}} c \, \mathrm{ds} \; - q \right|^2 + |\boldsymbol{u_c}(t)|^2 \, \mathrm{d}t, \tag{4.46}$$

which measures the difference of the actual flux of c through Γ_{r} and q, as well as the control costs $\boldsymbol{u_c}$, in the square of the Euclidean norm. Including (4.45) in (4.46), one obtains

$$\int_{\Gamma_{\mathrm{r}}} \partial_{\vec{n}} c \, \mathrm{ds} \; - q = \int_{\Gamma_{\mathrm{r}}} \partial_{\vec{n}} (c - c^{\vec{w}}) \, \mathrm{ds} = \int_{\Gamma_{\mathrm{r}}} \partial_{\vec{n}} c^{(\vec{z})} \, \mathrm{ds}.$$

Discretizing the latter equation within NAVIER, one ends up with the output equation

$$C_{\boldsymbol{c}} \boldsymbol{c}(t) = \boldsymbol{y_c}(t)$$

as in (3.21b). Notice that we split the obstacle boundary into $n_{\boldsymbol{a}} = 4$ parts such that

$$\Gamma_{\mathrm{r}} = \bigcup_{i=1}^{n_{\boldsymbol{a}}} \Gamma_{\mathrm{r},i}.$$

In detail, each $\Gamma_{\mathrm{r},i}$ is defined over one edge of the obstacle $\Omega_{\mathrm{R}}^{\mathrm{O}}$. Thus, each edge $\Gamma_{\mathrm{r},i}$ is represented in the i-th row of $C_{\boldsymbol{c}} \in \mathbb{R}^{n_{\boldsymbol{a}} \times n_{\boldsymbol{c}}}$.

Before stating the results of the numerical experiments, various parameters for the experiments need to be set. At first, the Reynolds and the Schmidt number within the PDEs (3.8a), (3.9a), and (3.10a) need to be defined. Both numbers reciprocally scale the diffusion part, such that a higher value decreases the influence of the diffusion part. As a consequence, the convective term becomes more influential and one speaks about a convection dominated flow problem. For the NSE and Stokes scenario, a Reynolds number of $\mathrm{Re} \in \{100, 200, 300, 400, 500\}$ is chosen. Using an even higher Reynolds number would yield numerical instabilities for the initial triangulations within the flow solver NAVIER. Therefore, the Reynolds number is restricted to that range. For the

Table 4.2.: **CFM scenario:** Parameter sets.

Set	Re	Sc
I	1	1
II	1	10
III	10	1
IV	1	100
V	10	10

Table 4.3.: Refinement levels for finite element discretizations of KVS (Figure 3.3) and RM (Figure 3.4).

(a) Dimensions of discretized velocity and pressure space for KVS.

Level	$n_{\boldsymbol{z}}$	$n_{\boldsymbol{p}}$
1	4 796	672
2	12 292	1 650
3	28 914	3 784
4	64 634	8 318
5	140 110	17 878
6	296 888	37 601

(b) Dimensions of discretized velocity, concentration, and pressure space for RM.

Level	$n_{\boldsymbol{z}}$	$n_{\boldsymbol{c}}$	$n_{\boldsymbol{p}}$
1	10 279	1 187	1 276
2	22 750	2 610	2 707
3	48 352	5 466	5 643
4	101 271	11 423	11 618

CFM scenario, five different combinations of Re and Sc are considered and are depicted in Table 4.2. Due to the geometry of the RM, especially the inward corners, no parameter sets with Re $= 100$ or ReSc > 100 can be considered for the used starting triangulations without additional stabilization or refinement techniques.

The second parameter that drastically influences the numerical experiments is the level of refinement. As introduced in Section 2.1, the KVS and the RM are discretized by triangles. Each triangle has three $\mathcal{P}_1$ and six $\mathcal{P}_2$ nodes. Using the bisection refinement [11], various levels of refinement can be generated, which yields different dimensions for the discretized solutions, as depicted in Table 4.3. In detail, Table 4.3a shows the dimensions for the velocity and pressure space for the KVS. For the RM, four different levels are considered as depicted in Table 4.3b. The triangulations for Level 1 are depicted in Figure 3.3 for the KVS and in Figure 3.4 for the RM. The numbers for the velocity and concentration space represent the inner nodes that are not related to Dirichlet BC, as explained in Section 3.5.

All computations are performed using heuristic ADI shifts as described in detail in [104]. This heuristic approach computes a small number of Ritz values as approximations of the eigenvalues of the pencil with small and large magnitude. Therefore, a truncated Arnoldi process [113] is carried out. In detail, the Arnoldi process using $\boldsymbol{M}^{-1}\boldsymbol{A}$ yields Ritz values with a large magnitude and the Arnoldi process using $\boldsymbol{A}^{-1}\boldsymbol{M}$ yields Ritz values with a small magnitude. Due to the $2n_{\boldsymbol{p}}$ infinite eigenvalues of the pencil $(\boldsymbol{A}, \boldsymbol{M})$,

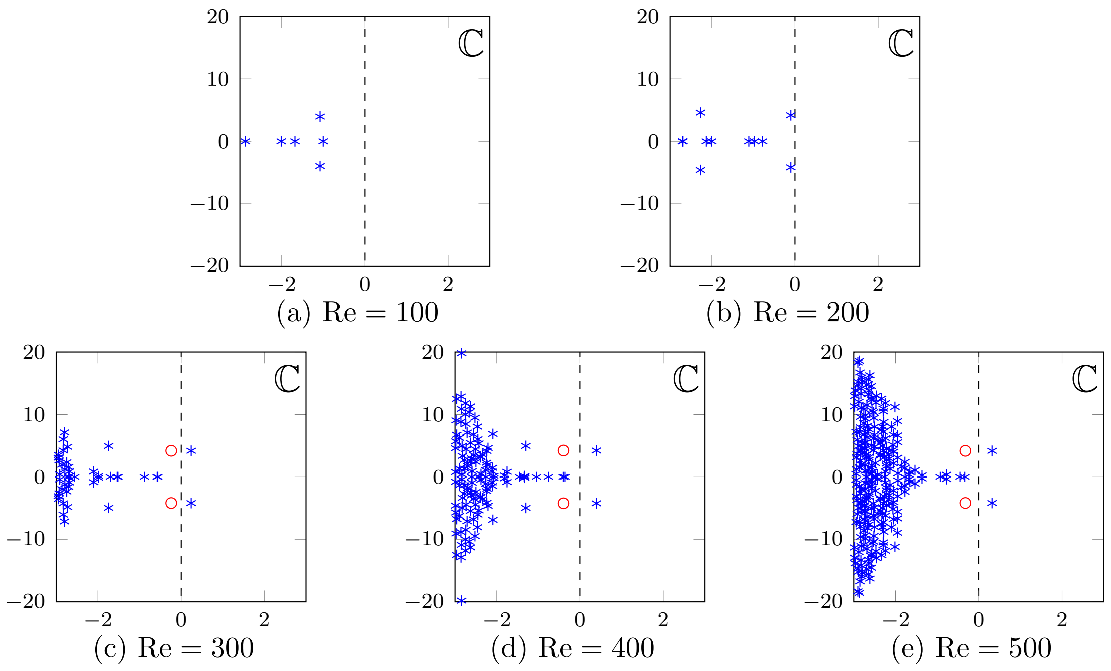


Figure 4.2.: **NSE scenario:** Eigenvalues of the matrix pencil (3.23) that are close to $j\mathbb{R}$ for different Reynolds numbers
(original: ($*$), initial feedback: ($\circ$), refinement: Level 1).

the Arnoldi process for $\boldsymbol{M}^{-1}\boldsymbol{A}$ is not defined properly. Therefore, the pencil is shifted via the results in Theorem 2.9 with $\delta_1 = 1$, $\delta_2 = -0.02$, and $\gamma = 0$. This shifting process maps all infinite eigenvalues onto -50, while all finite eigenvalues remain unchanged. Using an all-ones vector as starting vector, 20 large and 20 small magnitude Ritz values, 15 proper heuristic ADI shifts $\{q_i\}_{i=1}^{15} = \overline{\{q_i\}_{i=1}^{15}} \subset \mathbb{C}^-$ are computed. The choice of ADI shift parameters is a difficult topic by itself and is not within the scope of this thesis. More details regarding these heuristic ADI shifts can be found in [84, 104, 115].

Furthermore, the stopping criteria for the inner ADI loop and the outer Newton loop are set. As discussed in Subsection 4.3.2, the relative change (4.35) of the desired feedback matrix K is used. For the outer Newton iteration, the desired tolerance is set to $tol_{\text{Newton}} := 10^{-8}$. Unless otherwise stated, the tolerance for the relative change of K in the inner ADI iteration is set to $tol_{\text{ADI}} := 10^{-7}$. In addition, all linear systems are solved by the sparse direct solver in MATLAB, represented by the backslash operator.

The final parameter, which is used to modify the underlying LQR setup, is the output weighting $\alpha \in \mathbb{R}$ as defined in Theorem (2.29). In general, this output weighting is set to $\alpha = 1.0$, as used in the theoretical derivations. Later on, $\alpha \in \{10^{-2}, 10^{-1}, 10^{1}, 10^{2}\}$ is used to present the influence of the parameters. The modification of this parameter is covered by the theory. One can redefine $\boldsymbol{C} := \sqrt{\alpha}\,\boldsymbol{C}$ in Theorem 4.5, which does not violate the assumption of $(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{M})$ being detectable.

Before stating convergence results of the KN-ADI method for the NSE scenario, the influence of the Reynolds number is depicted in Figure 4.2. Thereby, the eigenvalues of

Table 4.4.: **NSE scenario:** Number of Newton steps (#Newt) and ADI steps (#ADI) during KN-ADI process ($tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$).

(a) Influence of output weighting α during the KN-ADI process (refinement: Level 1).

Re / α	100		200		300		400		500	
	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI
10^{-2}	3	288	4	732	5	374	5	435	4	427
10^{-1}	3	288	5	747	5	404	5	654	6	948
10^{0}	5	550	8	1268	8	922	9	1444	9	2079
10^{1}	8	879	12	1882	13	1773	14	2484	14	3328
10^{2}	14	1633	18	2566	19	2940	20	3824	20	4923

(b) Influence of refinement levels during the KN-ADI process ($\alpha = 1$).

Re	100		200		300		400		500	
	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI
Level 1	5	550	8	1268	8	922	9	1444	9	2079
Level 2	4	345	8	1039	9	1197	9	1281	10	1680
Level 3	4	345	9	1218	10	1980	11	2498	10	3040
Level 4	4	267	9	1098	10	1809	11	2423	11	3283
Level 5	5	419	10	1069	11	1939	12	2515	11	3093
Level 6	6	638	10	872	12	1690	13	2459	14	3237

the matrix pencil (3.23) are denoted by ($*$); compare [15, Sec. 3.3]. It turns out that increasing the Reynolds number, i.e., the flow becomes more convection dominated, brings the eigenvalues closer towards the imaginary axis $\jmath\mathbb{R}$. At a certain point between Re $= 200$ and Re $= 300$, a pair of eigenvalues crosses this axis and the pencil becomes unstable. As discussed in Subsection 4.2.3, in the event of an unstable pencil $(\boldsymbol{A}, \boldsymbol{M})$, an initial feedback $K^{(0)}$ has to be computed. The influence of this initial feedback modifies only the unstable eigenvalues as depicted by ($\circ$) in Figures 4.2c–4.2e. The computation of this initial feedback dependents only on the refinement level and the Reynolds number. Hence, it can be performed in the offline phase. Nevertheless, the computation of an initial feedback is denoted by an additional Newton step in the following results.

4.4.2. Feedback Stabilization for the NSE Scenario

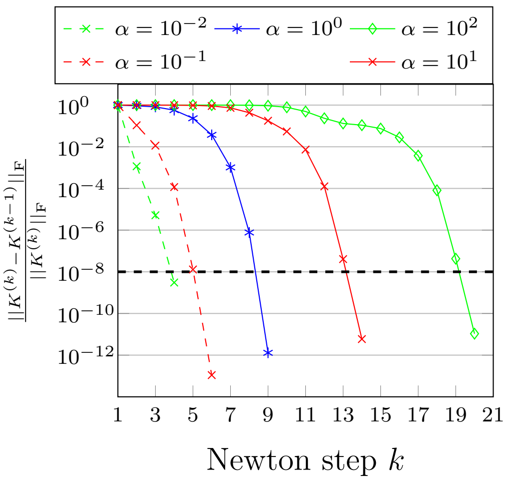
The first convergence results are depicted in Table 4.4. In detail, Table 4.4a shows the influence of the output weighting α for different Reynolds numbers. The computations are performed on the initial refinement Level 1. The number of Newton steps (#Newt) increases for increasing Reynolds number as well as for an increasing α. Both behaviors are natural consequences. As mentioned above, the Reynolds number influences the stability of the system drastically. Furthermore, an increasing α penalizes the output within the LQR setup further, such that the feedback needs to be of a higher quality to force the output faster towards zero. The total number of ADI steps (#ADI) follows this behavior. However, except for Level 6, a decreasing number of ADI steps can be observed between Re = 200 and Re = 300. This indicates that the used initial feedback for Re > 200 is a good starting point such that the costs of each Newton step decrease. In summary, one can say that the KN-ADI method can be used for all parameter configurations.

The results depicted in Table 4.4b investigate the influence of the different mesh refinement levels from Table 4.3a. Besides one exception, the number of Newton steps slightly increases with an increasing refinement level. By increasing the refinement level, more information needs to be processed in order to compute the feedback. Similar to an increasing output weight, this yields a growing number of Newton steps. The number of ADI steps does not show clear tendencies of growing or shrinking. However, the number of ADI steps appears to grow at first and to shrink towards Level 6. We believe that the physical behavior of the NSE scenario is not resolved sufficiently in the lower refinement levels such that the computation of its feedback is more complicated in each step. However, upon reaching Level 3 or Level 4, the physical behavior is resolved sufficiently and the costs for each Newton step stay on the same level. A significant difference occurs for Re = 500 in-between Level 5 and Level 6. The number of Newton steps increases drastically, hence, the number of ADI steps grows as well. This behavior is investigated next. Notice that the combination of a higher refinement level and a higher output weighting produces qualitatively similar results.

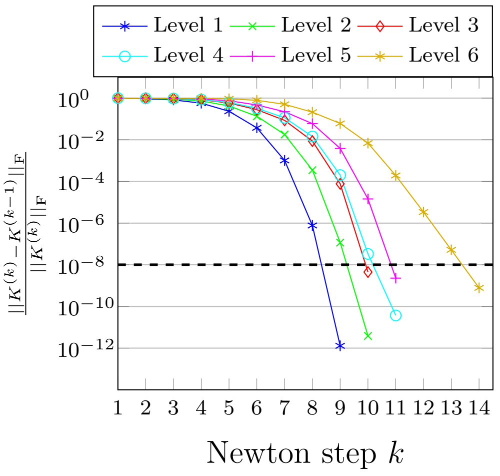
The result in Theorem 4.5 states a quadratic convergence of the Riccati solution. As explained in Subsection 4.3.2, the relative change on the feedback K is used as the stopping criterion, which shows a similar quadratic convergence behavior. In Figure 4.3, the evolution of the relative change for the results from Table 4.4 is depicted. The results are restricted to the most demanding case with a Reynolds number of 500. In detail, Table 4.4a illustrates the influence of the output weighting α. For an increasing value of α, the relative change stagnates around 1.0 before the typical quadratic convergence can be observed.

A similar behavior can be observed in Figure 4.3b for an increasing refinement level as in Table 4.4b. For the levels 1–5, the stagnation phase slightly prolongs with each level. As mentioned above for Level 6, the number of Newton steps increases drastically. However, it is not an even longer stagnation phase that yields this behavior, but a non quadratic convergence behavior. As carried out in [15, Sec. 4.2], in that case the ADI tolerance is not sufficient to ensure quadratic convergence of Newton's method. As shown in Figure 4.3c, increasing the ADI tolerance yields the expected quadratic convergence behavior. However, after a certain point, increasing the ADI tolerance does not influence the Newton convergence anymore. Hence, the enormously increasing computation costs might not be necessary as depicted in Figure 4.3d.

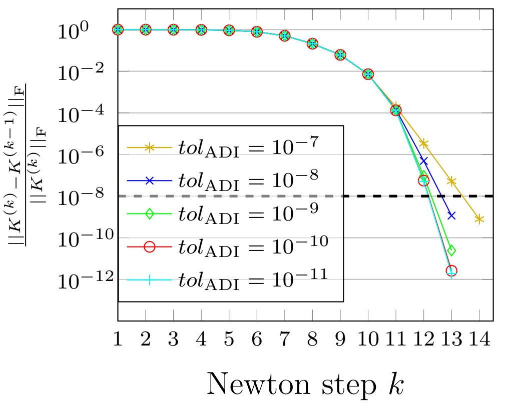
This behavior indicates that a fully automatic process to determine the tolerance of the inner ADI loop would be required to avoid unnecessary additional costs. The overall computation costs mainly consists of costs for the shift computation and the linear solve in each ADI step. The costs for the computation of the relative change, as well as the feedback accumulation, in each ADI step are negligible compared to the linear solve.



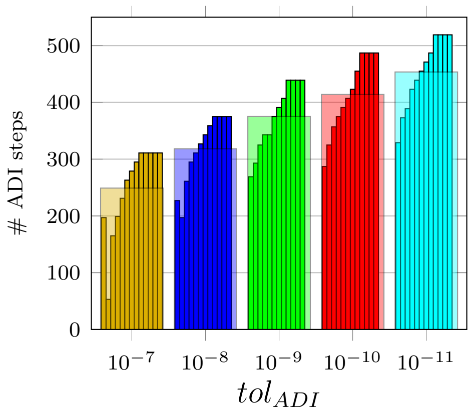
(a) Newton convergence for different output weights α (refinement: Level 1, $tol_{\text{ADI}} = 10^{-7}$).



(b) Newton convergence for different refinement levels ($\alpha = 10^{0}$, $tol_{\text{ADI}} = 10^{-7}$).



(c) Newton convergence for different ADI stopping tolerances (refinement: Level 6, $\alpha = 1$).



(d) Number of ADI steps in Figure 4.3c: per Newton step (small bars) and average over all Newton steps (wide bar) (refinement: Level 6, $\alpha = 1$).

Figure 4.3.: **NSE scenario:** Influence of output weight α, refinement levels from Table 4.3a, and ADI tolerance on Newton convergence behavior ($\text{Re} = 500$, $tol_{\text{Newton}} = 10^{-8}$).

Table 4.5.: **NSE scenario:** Detailed computation timings in seconds ($tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$, $\alpha = 10^0$).

		$\text{time}_{\text{lin\_solve}}$	$\text{time}_{\text{shift}}$	$\text{time}_{\text{rel\_ch}}$	$\mathbf{time_{total}}$
Re = 100					
Level	1	$2.24 \cdot 10^2$	$2.14 \cdot 10^1$	$1.19 \cdot 10^0$	$\mathbf{2.47 \cdot 10^2}$
	2	$4.22 \cdot 10^2$	$5.44 \cdot 10^1$	$2.34 \cdot 10^0$	$\mathbf{4.78 \cdot 10^2}$
	3	$1.19 \cdot 10^3$	$1.58 \cdot 10^2$	$5.95 \cdot 10^0$	$\mathbf{1.35 \cdot 10^3}$
	4	$2.44 \cdot 10^3$	$4.80 \cdot 10^2$	$1.11 \cdot 10^1$	$\mathbf{2.93 \cdot 10^3}$
	5	$9.47 \cdot 10^3$	$1.32 \cdot 10^3$	$5.25 \cdot 10^1$	$\mathbf{1.08 \cdot 10^4}$
	6	$3.60 \cdot 10^4$	$4.03 \cdot 10^3$	$2.39 \cdot 10^2$	$\mathbf{4.02 \cdot 10^4}$
Re = 200					
Level	1	$5.39 \cdot 10^2$	$4.37 \cdot 10^1$	$2.93 \cdot 10^0$	$\mathbf{5.85 \cdot 10^2}$
	2	$1.34 \cdot 10^3$	$1.14 \cdot 10^2$	$6.96 \cdot 10^0$	$\mathbf{1.46 \cdot 10^3}$
	3	$4.45 \cdot 10^3$	$3.68 \cdot 10^2$	$2.12 \cdot 10^1$	$\mathbf{4.84 \cdot 10^3}$
	4	$1.03 \cdot 10^4$	$9.55 \cdot 10^2$	$4.46 \cdot 10^1$	$\mathbf{1.13 \cdot 10^4}$
	5	$2.57 \cdot 10^4$	$3.25 \cdot 10^3$	$1.36 \cdot 10^2$	$\mathbf{2.91 \cdot 10^4}$
	6	$5.01 \cdot 10^4$	$6.83 \cdot 10^3$	$2.83 \cdot 10^2$	$\mathbf{5.72 \cdot 10^4}$
Re = 300					
Level	1	$4.13 \cdot 10^2$	$3.20 \cdot 10^1$	$1.99 \cdot 10^0$	$\mathbf{4.47 \cdot 10^2}$
	2	$1.60 \cdot 10^3$	$1.17 \cdot 10^2$	$8.99 \cdot 10^0$	$\mathbf{1.72 \cdot 10^3}$
	3	$7.65 \cdot 10^3$	$4.80 \cdot 10^2$	$4.34 \cdot 10^1$	$\mathbf{8.17 \cdot 10^3}$
	4	$1.75 \cdot 10^4$	$9.77 \cdot 10^2$	$8.19 \cdot 10^1$	$\mathbf{1.85 \cdot 10^4}$
	5	$4.62 \cdot 10^4$	$2.77 \cdot 10^3$	$2.51 \cdot 10^2$	$\mathbf{4.92 \cdot 10^4}$
	6	$1.06 \cdot 10^5$	$9.08 \cdot 10^3$	$5.73 \cdot 10^2$	$\mathbf{1.15 \cdot 10^5}$
Re = 400					
Level	1	$6.62 \cdot 10^2$	$3.63 \cdot 10^1$	$3.09 \cdot 10^0$	$\mathbf{7.01 \cdot 10^2}$
	2	$1.73 \cdot 10^3$	$1.50 \cdot 10^2$	$1.09 \cdot 10^1$	$\mathbf{1.89 \cdot 10^3}$
	3	$9.40 \cdot 10^3$	$4.20 \cdot 10^2$	$4.80 \cdot 10^1$	$\mathbf{9.87 \cdot 10^3}$
	4	$2.35 \cdot 10^4$	$1.09 \cdot 10^3$	$1.10 \cdot 10^2$	$\mathbf{2.47 \cdot 10^4}$
	5	$6.18 \cdot 10^4$	$3.17 \cdot 10^3$	$3.18 \cdot 10^2$	$\mathbf{6.53 \cdot 10^4}$
	6	$1.54 \cdot 10^5$	$1.00 \cdot 10^4$	$9.13 \cdot 10^2$	$\mathbf{1.65 \cdot 10^5}$
Re = 500					
Level	1	$9.10 \cdot 10^2$	$3.52 \cdot 10^1$	$4.44 \cdot 10^0$	$\mathbf{9.50 \cdot 10^2}$
	2	$2.25 \cdot 10^3$	$1.32 \cdot 10^2$	$1.30 \cdot 10^1$	$\mathbf{2.40 \cdot 10^3}$
	3	$1.17 \cdot 10^4$	$4.65 \cdot 10^2$	$6.01 \cdot 10^1$	$\mathbf{1.22 \cdot 10^4}$
	4	$3.17 \cdot 10^4$	$1.11 \cdot 10^3$	$1.35 \cdot 10^2$	$\mathbf{3.29 \cdot 10^4}$
	5	$7.41 \cdot 10^4$	$2.79 \cdot 10^3$	$4.13 \cdot 10^2$	$\mathbf{7.73 \cdot 10^4}$
	6	$1.91 \cdot 10^5$	$9.10 \cdot 10^3$	$1.18 \cdot 10^3$	$\mathbf{2.02 \cdot 10^5}$

Table 4.6.: Number of Newton steps (#Newt) and ADI steps (#ADI) during KN-ADI process for Stokes and CFM scenario ($tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$).

(a) **Stokes scenario:** Influence of output weighting α during the KN-ADI process (refinement: Level 1).

Re / α	100		200		300		400		500	
	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI
10^{-2}	3	61	3	65	3	72	3	75	3	69
10^{-1}	3	61	3	65	3	72	3	75	3	69
10^{0}	3	61	4	86	4	96	4	100	5	111
10^{1}	5	103	6	128	6	160	7	185	8	174
10^{2}	8	160	9	185	10	224	11	296	11	307

(b) **CFM scenario:** Influence of refinement levels on the KN-ADI process ($\alpha = 1$).

Set	I		II		III		IV		V	
	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI	#Newt	#ADI
Level 1	6	202	6	203	6	240	3	131	3	143
Level 2	7	236	8	375	8	287	7	376	7	460
Level 3	8	351	8	308	8	345	8	627	8	429
Level 4	9	362	10	376	10	486	8	529	8	410

A detailed overview of the different costs for $\alpha = 10^0$ is depicted in Table 4.5. The total time ($\text{time}_{\text{total}}$) is split into the time for the linear solves ($\text{time}_{\text{lin\_solve}}$), the time for the shift computation ($\text{time}_{\text{shift}}$), and the time to evaluate the norm of the relative change ($\text{time}_{\text{rel\_ch}}$). Thereby, the time for the linear solves dominates the entire process. Although the time to calculate the shifts is roughly one magnitude smaller, it is still a reasonable share of the entire costs. The computation of the relative changes is around two magnitudes smaller and, therefore, does not really contribute to the total time.

4.4.3. Feedback Stabilization for Stokes and CFM Scenarios

In this section, some results for the Stokes and CFM scenarios are depicted in Table 4.6. In detail, Table 4.6a shows the influence of the output weighting α to the Stokes scenario. The results are qualitatively similar to the results in Table 4.4a. However, the feedback computation of the Stokes scenario, which is the NSE scenario without the non-linearity $(\vec{v} \cdot \nabla)\vec{v}$, is much simpler. Less Newton steps are needed for the same parameters and each Newton step needs much less ADI steps.

Similar observations can be made for the CFM scenario as depicted in Table 4.6b. Although the CFM scenario has a more complicated block structure, its solution needs only slightly more Newton steps. Hence, the problem gets more demanding with increasing Reynolds and Schmidt numbers as well as a higher refinement level. In summary,

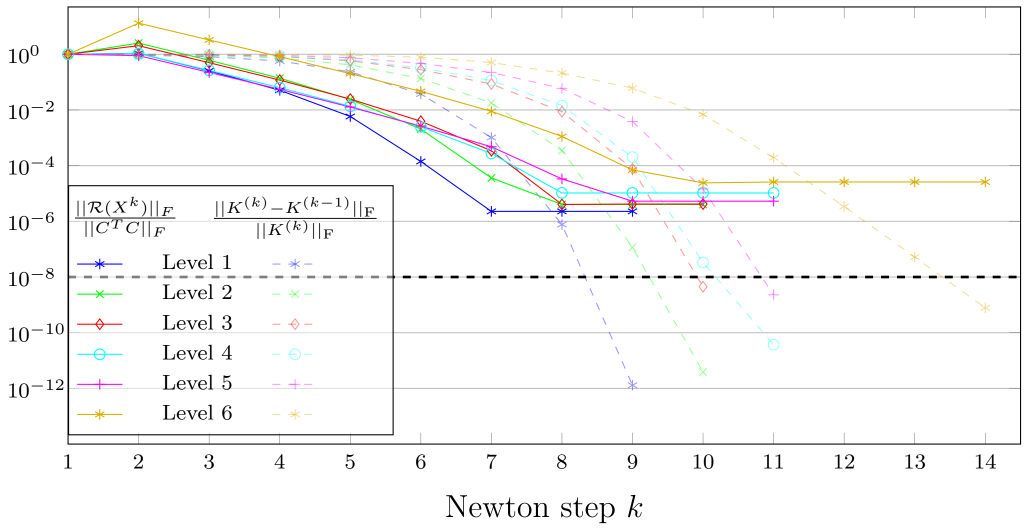


Figure 4.4.: **NSE scenario:** Evolution of relative change and projected residual dependent on the refinement level (Re $= 500$, $tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$, $\alpha = 10^{0}$).

the CFM scenario behaves in a similar way to the NSE scenario, where the increasing Reynolds number has a higher influence than an increasing Schmidt number. In [14], more numerical tests with a varying output weighting are performed. Notice that the non-quadratic convergence behavior mentioned above can be observed in the CFM scenario for Set V and $\alpha = 10$. However, increasing the ADI tolerance resolves this problem as well.

4.4.4. Residual Convergence of KN-ADI for NSE Scenario

The results in the previous sections are obtained by monitoring the relative change of the feedback K to determine the stopping criteria for the nested iteration. Thereby, the results in Figure 4.3b indicate that the expected convergence behavior cannot always be achieved. On the one hand, the relative change might stagnate for a couple of Newton steps. On the other hand, the final convergence is not quadratic. To investigate the reason for this unexpected behavior, some of the previous experiments are repeated including the explicit computation of the projected residual as introduced in Subsection 4.3.2. Hence, Figure 4.4 shows the evolution of the projected residual on top of the results from Figure 4.3b. As it turns out, the stagnation of the relative change results from an upwards jump of the residual after the first Newton step, i.e., Newton step 2 in the results, since the initial feedback is illustrated as a first Newton step. Furthermore, the non-quadratic convergence of the relative change results from a stagnation of the projected residual. Both behaviors are not reflected properly in the relative change of the feedback matrix. Notice that the stagnation of the residual is roughly around the same point if one ignores the upwards jump in the first Newton step. Hence, the non-quadratic behavior of the relative change for the Level 6 refinement results from a

Table 4.7.: **NSE scenario:** Detailed computation timings including the explicit residual computation in seconds (Re $= 500$, $tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$, $\alpha = 10^{0}$).

		$\text{time}_{\text{lin\_solve}}$	$\text{time}_{\text{shift}}$	$\text{time}_{\text{rel\_ch}}$	$\textbf{time}_{\textbf{res}}$	$\textbf{time}_{\textbf{total}}$
Re $= 500$						
Level	1	$9.70 \cdot 10^{2}$	$4.85 \cdot 10^{1}$	$6.90 \cdot 10^{0}$	$\mathbf{3.88 \cdot 10^{3}}$	$\mathbf{4.91 \cdot 10^{3}}$
	2	$2.29 \cdot 10^{3}$	$1.79 \cdot 10^{2}$	$1.44 \cdot 10^{1}$	$\mathbf{9.19 \cdot 10^{3}}$	$\mathbf{1.17 \cdot 10^{4}}$
	3	$1.16 \cdot 10^{4}$	$3.81 \cdot 10^{2}$	$5.77 \cdot 10^{1}$	$\mathbf{4.64 \cdot 10^{4}}$	$\mathbf{5.85 \cdot 10^{4}}$
	4	$3.27 \cdot 10^{4}$	$1.37 \cdot 10^{3}$	$1.53 \cdot 10^{2}$	$\mathbf{1.63 \cdot 10^{5}}$	$\mathbf{1.98 \cdot 10^{5}}$
	5	$7.45 \cdot 10^{4}$	$2.75 \cdot 10^{3}$	$4.21 \cdot 10^{2}$	$\mathbf{3.59 \cdot 10^{5}}$	$\mathbf{4.37 \cdot 10^{5}}$
	6	$1.93 \cdot 10^{5}$	$8.97 \cdot 10^{3}$	$1.14 \cdot 10^{3}$	$\mathbf{9.24 \cdot 10^{5}}$	$\mathbf{1.13 \cdot 10^{6}}$

combination of this upwards jump and the early stagnation of the residual.

However, computing the projected residual explicitly is enormously expensive as shown in Table 4.7. The time to evaluate the projected residuals (time_{res}) exceeds the times to solve the linear systems by nearly one magnitude such that it dominates the total computation time.

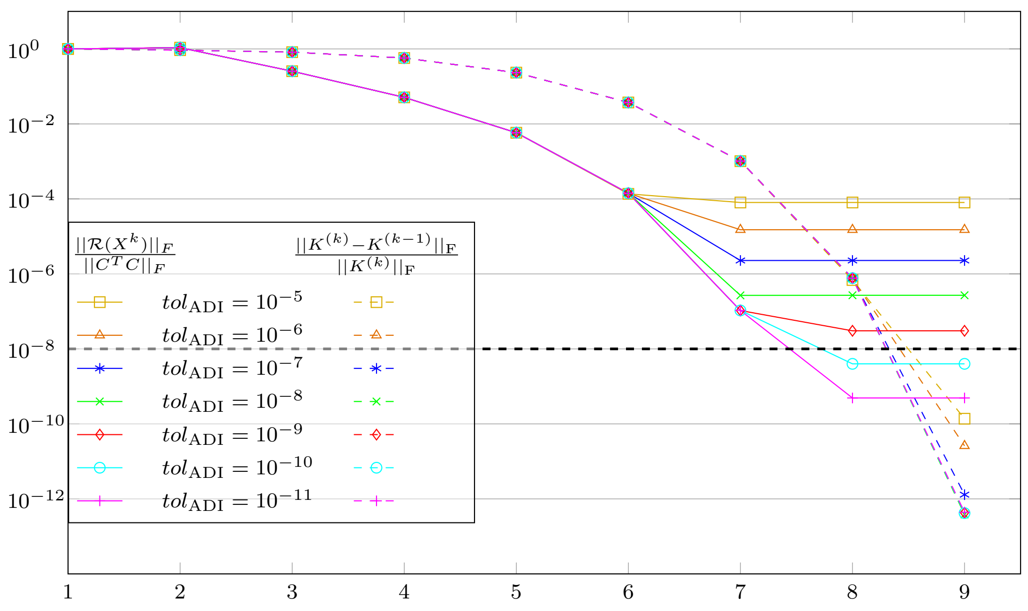
The residual stagnation for the coarsest refinement level is illustrated in further detail in Figure 4.5. Thereby, the ADI tolerance tol_{ADI} varies between 10^{-5} and 10^{-11}. The first example in Figure 4.5a considers an output weighting of $\alpha = 10^{0}$. As it turns out, the method seems to be converged after nine Newton steps independent of the ADI tolerance. However, the residual stagnates around a magnitude of $O(10 \cdot tol_{\text{ADI}})$.

Similar results are depicted in Figure 4.5b for the output weighting of $\alpha = 10^{2}$. Although the upwards jump of the first Newton step is drastic, the residual achieves similar results. Merely the lowest ADI tolerance of $tol_{\text{ADI}} = 10^{-5}$ shows the non-quadratic convergence behavior in the evolution of the relative change.

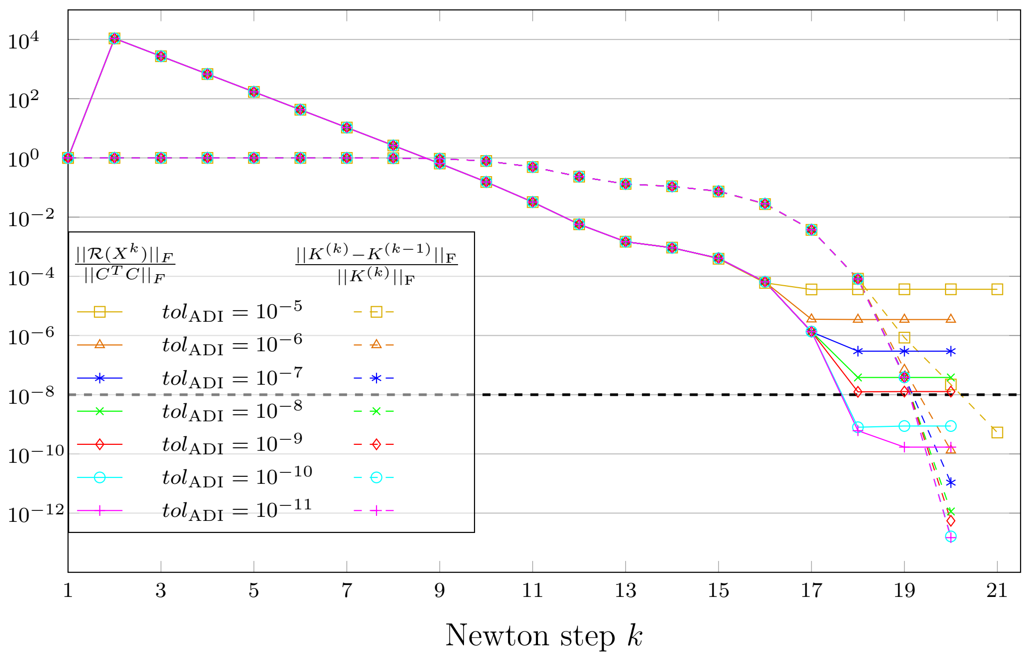
To complete these results, the detailed computation times for both experiments are depicted in Table 4.8. As in Table 4.7, the computation of the projected residual dominates the entire process.

4.5. Conclusion – Part I

All experiments in Section 4.4 show that the KN-ADI method in Algorithm 3 is able to compute the solution of the projected GCARE (4.16c) without the use of any explicit projection. Nevertheless, it turns out that convergence statements based on the relative change of the feedback K might not be suitable for all problem configurations. Certain important problems during the process, such as the upwards jump in the first Newton step or a residual stagnation depending on the ADI tolerance, cannot be observed. All achieved feedback matrices turned out to be stabilizing. We believe that the good initial Bernoulli feedback yields this behavior. If such an initial feedback is not available or the problem structure is more demanding, better methods to ensure convergence to the



(a) Output weighting $\alpha = 10^0$.



(b) Output weighting $\alpha = 10^2$.

Figure 4.5.: **NSE scenario:** Evolution of relative change and projected residual dependent on the ADI tolerance
(Re = 500, $tol_{\mathrm{Newton}} = 10^{-8}$, refinement: Level 1).

Table 4.8.: **NSE scenario:** Detailed computation timings for various ADI tolerances including the explicit residual computation in seconds ($\mathrm{Re} = 500$, $tol_{\text{Newton}} = 10^{-8}$, refinement: Level 1).

tol_{ADI}	#Newt	#ADI	$\text{time}_{\text{lin\_solve}}$	$\text{time}_{\text{shift}}$	$\text{time}_{\text{rel\_ch}}$	$\mathbf{time_{res}}$	$\mathbf{time_{total}}$
			$\alpha = 10^0$				
10^{-5}	8	1453	$6.64 \cdot 10^2$	$3.65 \cdot 10^1$	$3.23 \cdot 10^0$	$\mathbf{1.98 \cdot 10^3}$	$\mathbf{2.69 \cdot 10^3}$
10^{-6}	8	1785	$7.89 \cdot 10^2$	$3.59 \cdot 10^1$	$3.88 \cdot 10^0$	$\mathbf{2.48 \cdot 10^3}$	$\mathbf{3.30 \cdot 10^3}$
10^{-7}	8	2079	$9.70 \cdot 10^2$	$4.85 \cdot 10^1$	$6.90 \cdot 10^0$	$\mathbf{3.88 \cdot 10^3}$	$\mathbf{4.91 \cdot 10^3}$
10^{-8}	8	2405	$1.07 \cdot 10^3$	$3.56 \cdot 10^1$	$5.21 \cdot 10^0$	$\mathbf{3.55 \cdot 10^3}$	$\mathbf{4.66 \cdot 10^3}$
10^{-9}	8	2769	$1.26 \cdot 10^3$	$4.69 \cdot 10^1$	$9.23 \cdot 10^0$	$\mathbf{5.38 \cdot 10^3}$	$\mathbf{6.70 \cdot 10^3}$
10^{-10}	8	3067	$1.37 \cdot 10^3$	$3.63 \cdot 10^1$	$6.66 \cdot 10^0$	$\mathbf{5.41 \cdot 10^3}$	$\mathbf{6.83 \cdot 10^3}$
10^{-11}	8	3321	$1.47 \cdot 10^3$	$3.63 \cdot 10^1$	$7.11 \cdot 10^0$	$\mathbf{5.43 \cdot 10^3}$	$\mathbf{6.94 \cdot 10^3}$
			$\alpha = 10^2$				
10^{-5}	20	2914	$1.32 \cdot 10^3$	$9.40 \cdot 10^1$	$6.30 \cdot 10^0$	$\mathbf{3.85 \cdot 10^3}$	$\mathbf{5.28 \cdot 10^3}$
10^{-6}	19	3864	$1.73 \cdot 10^3$	$8.78 \cdot 10^1$	$8.35 \cdot 10^0$	$\mathbf{5.17 \cdot 10^3}$	$\mathbf{7.00 \cdot 10^3}$
10^{-7}	19	4923	$2.25 \cdot 10^3$	$1.15 \cdot 10^2$	$1.63 \cdot 10^1$	$\mathbf{8.65 \cdot 10^3}$	$\mathbf{1.10 \cdot 10^4}$
10^{-8}	19	6250	$2.79 \cdot 10^3$	$8.73 \cdot 10^1$	$1.35 \cdot 10^1$	$\mathbf{8.97 \cdot 10^3}$	$\mathbf{1.19 \cdot 10^4}$
10^{-9}	19	7447	$3.33 \cdot 10^3$	$8.78 \cdot 10^1$	$1.62 \cdot 10^1$	$\mathbf{1.11 \cdot 10^4}$	$\mathbf{1.45 \cdot 10^4}$
10^{-10}	19	8974	$4.05 \cdot 10^3$	$8.68 \cdot 10^1$	$1.96 \cdot 10^1$	$\mathbf{1.37 \cdot 10^4}$	$\mathbf{1.78 \cdot 10^4}$
10^{-11}	19	10150	$4.55 \cdot 10^3$	$8.77 \cdot 10^1$	$2.21 \cdot 10^1$	$\mathbf{1.61 \cdot 10^4}$	$\mathbf{2.08 \cdot 10^4}$

desired solution need to be developed. Monitoring the projected residual explicitly is one, but unfortunately highly expensive, method to ensure convergence. From our experience, we would recommend an ADI tolerance of $tol_{\text{ADI}} = 10^{-2} \cdot tol_{\text{Newton}}$ to overcome all difficulties. However, this might exceed the available computation time drastically.

A more efficient method that overcomes various problems is presented in Chapter 6. Beforehand, the solution process of the innermost saddle point system (4.32a) is examined in greater detail in the next chapter.

5 Comparison of Linear Solvers

Contents

After deriving the first major results from this thesis in Chapter 4, the most time consuming computational step is examined in more detail in this chapter. It is pointed out at the end of Subsection 4.2.2 that the crucial computation step within Algorithm 3 is the solution of a large-scale SPS of the form (4.32a), which is repeated here for better readability:

$$\underbrace{\begin{bmatrix} A^T + q_\ell M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix}}_{\boldsymbol{F}_\ell} \underbrace{\begin{bmatrix} \widehat{V}_\ell \\ * \end{bmatrix}}_{\widehat{\boldsymbol{V}}_\ell} = \underbrace{\begin{bmatrix} \widehat{Y}_{\ell-1}^{(k)} \\ 0 \end{bmatrix}}_{\widehat{\boldsymbol{Y}}_{\ell-1}^{(k)}}. \qquad \text{(cf. eq. (4.32a))}$$

The choice of a solver for linear systems depends highly on the structure and the properties of the system matrix $\boldsymbol{F}_\ell$. Two different solution strategies to solve (4.32a) are presented within this thesis. Therefore, the crucial properties of (4.32a) are reviewed shortly.

The system (4.32a) has the typical saddle point structure, compare, e.g., [42], and the right-hand side $\widehat{\boldsymbol{Y}}_{\ell-1}^{(k)}$ consists of $n_a + 2n_r$ columns. The system matrix $\boldsymbol{F}_\ell \in \mathbb{C}^{N\times N}$ depends on the varying ADI shift q_ℓ and is indefinite for all $q_\ell \in \mathbb{C}^-$. Since A is nonsymmetric in general, $\boldsymbol{F}_\ell$ is assumed to be nonsymmetric as well.

***Remark* 5.1** *$\boldsymbol{F}_\ell = \boldsymbol{A} + q_\ell \boldsymbol{M}$ is singular if and only if $q_\ell = -\lambda_{us}$, where $\lambda_{us} \in \mathbb{C}^+$ is an unstable eigenvalue of $(\boldsymbol{A}, \boldsymbol{M})$, as discussed in Subsection 4.2.3. Although it is unlikely that this ADI shift would be chosen in general, it would be sorted out by the used routine since the unstable eigenvalues λ_{us}, if they exist, are known from computing the initial feedback as explained in Subsection 4.2.3.*

The chapter is structured as follows. In the first section, a direct solver is investigated such as used for all numerical examples in Section 4.4. In Section 5.2, iterative solvers are considered and various preconditioning approaches are investigated. The usability of these approaches is demonstrated by repeating some of the numerical experiments from Section 4.4 using these preconditioned iterative solution methods. The results in Section 5.2 are partly published in [35] for the Stokes and in [36] for the CFM scenario.

5.1. Sparse Direct Solver

A direct solver computes the solution of a linear system explicitly, which should yield the highest achievable tolerance. Considering a general linear system of the form $\widetilde{F}\boldsymbol{x} = \boldsymbol{b}$, the simplest explicit definition of the sought solution is $\boldsymbol{x} = \widetilde{F}^{-1}\boldsymbol{b}$. However, the explicit computation of the inverse of $\widetilde{F}$ is highly expensive and usually numerically unstable. Even more important is the fact that for large and sparse matrices $\widetilde{F}$, the inverse $\widetilde{F}^{-1}$ is in general dense and cannot be stored. Thus, the computation of $\widetilde{F}^{-1}$ is numerically not feasible.

Generally, direct solvers do not form $\widetilde{F}^{-1}$ and can still explicitly solve the system $\widetilde{F}\boldsymbol{x} = \boldsymbol{b}$ as explained for one specific direct solver next.

5.1.1. *LU* Decomposition

One of the most reliable and stable techniques to directly solve a linear systems is the *LU* decomposition (also known as *LU* or triangular factorization). The main idea is based on the premise that "if a linear system $\widetilde{F}_\triangledown \boldsymbol{x} = \boldsymbol{b}$ has a non-singular triangular coefficient matrix $\widetilde{F}_\triangledown \in \mathbb{C}^{n\times n}$, computation of the unique solution $\boldsymbol{x}$ is remarkable easy." [79, Sec. 3.5] Thus, the *LU* decomposition computes a lower triangular factor $L \in \mathbb{C}^{n\times n}$ and an upper triangular factor $U \in \mathbb{C}^{n\times n}$ such that $\widetilde{F} = LU$, as described in detail in, e.g., [79, Sec. 3.5] or [63, Sec. 3.2]. "The solution to the original $\widetilde{F}\boldsymbol{x} = \boldsymbol{b}$ problem is then found by a two step triangular solve process

$$L\boldsymbol{y} = \boldsymbol{b}, \quad U\boldsymbol{x} = \boldsymbol{y} \quad \Rightarrow \quad \widetilde{F}\boldsymbol{x} = LU\boldsymbol{x} = L\boldsymbol{y} = \boldsymbol{b}. \tag{5.1}$$

[In this way] the *LU* factorization is a 'high level' algebraic description of Gaussian elimination." [63, p. 94]

In some cases such a factorization does not exist or its computation is numerically unstable. Hence, one considers additional permutation matrices as described in the following theorem.

Theorem 5.2 (cf. [79, Thm. 3.5.7]) *Let $\widetilde{F} \in \mathbb{C}^{n\times n}$. There exist permutation matrices $P, Q \in \mathbb{C}^{n\times n}$, a lower triangular matrix $L \in \mathbb{C}^{n\times n}$, and an upper triangular matrix $U \in \mathbb{C}^{n\times n}$ such that*

$$\widetilde{F} = PLUQ. \quad (\Rightarrow\ L\boldsymbol{y} = P^T\boldsymbol{b},\ U\boldsymbol{z} = \boldsymbol{y},\ \boldsymbol{x} = Q^T\boldsymbol{z})$$

If $\widetilde{F}$ is non-singular, one may take $Q = I_n$ and $\widetilde{F}$ may be written as

$$\widetilde{F} = PLU. \qquad (\Rightarrow\ L\boldsymbol{y} = P^T\boldsymbol{b},\ U\boldsymbol{x} = \boldsymbol{y})$$

The permutation matrices are used to pivot the rows (P) or columns (Q) in order to avoid numerical cancellation effects, which increases the stability of the method significantly. For non-singular matrices, the most stable variant is $\widetilde{F} = PLU$. Unfortunately, even for sparse matrices $\widetilde{F}$, one ends up with dense LU factors due to the so-called *fill-in*. The fill-in of a sparse matrix describes the entries of a matrix that are changed within the application of an algorithm from an original zero to a non-zero value. A highly efficient method to minimize fill-in is a special permutation technique as described in [51] and implemented in the UMFPACK package (version 4.3 an higher) that "is incorporated as a built-in operator in MATLAB (version 6.5 an higher) as `x=F\b` when $\widetilde{F}$ is sparse and nonsymmetric."[51, Abstract] The interested reader is referred to [51, 52, 55] and the references therein for further details. This sparse LU factorization uses a limited pivoting technique. Hence, its accuracy is limited compared to the complete pivoting in the dense case. In the following subsection, the error is investigated that occurs due to the sparse LU decomposition. As test examples, certain matrices $\widetilde{F} = \boldsymbol{F}_\ell$ from Section 4.4 are used.

5.1.2. Numerical Experiments for Sparse Direct Solver

To test the sparse direct solver from MATLAB, a couple of ADI shifts q_ℓ are selected which have been used during the computations of the different scenarios in Section 4.4. In detail, we choose the shifts with largest and smallest magnitude as well as the two shifts with the largest imaginary part. Additionally, we included one shift within each magnitude of the shift spectrum. If not already selected as a shift, the shift closest to the imaginary axis $\jmath\mathbb{R}$ is added as well. Although complex shifts occur as a pair, we selected only the shift with the negative imaginary part. The selected shifts for the NSE scenario for various Reynolds numbers are depicted in Figure 5.1. Each subfigure shows the used ADI shifts as ($*$) and the selected shifts as ($\circ$) for the different Reynolds levels. Additional, the above mentioned "bad shifts" $q_\ell = -\lambda_{us}$, which are detected during computation of the initial feedback in Subsection 4.2.3, are marked with ($\diamond$). As mentioned above, these shifts yield singular matrices. Within our numerical tests,

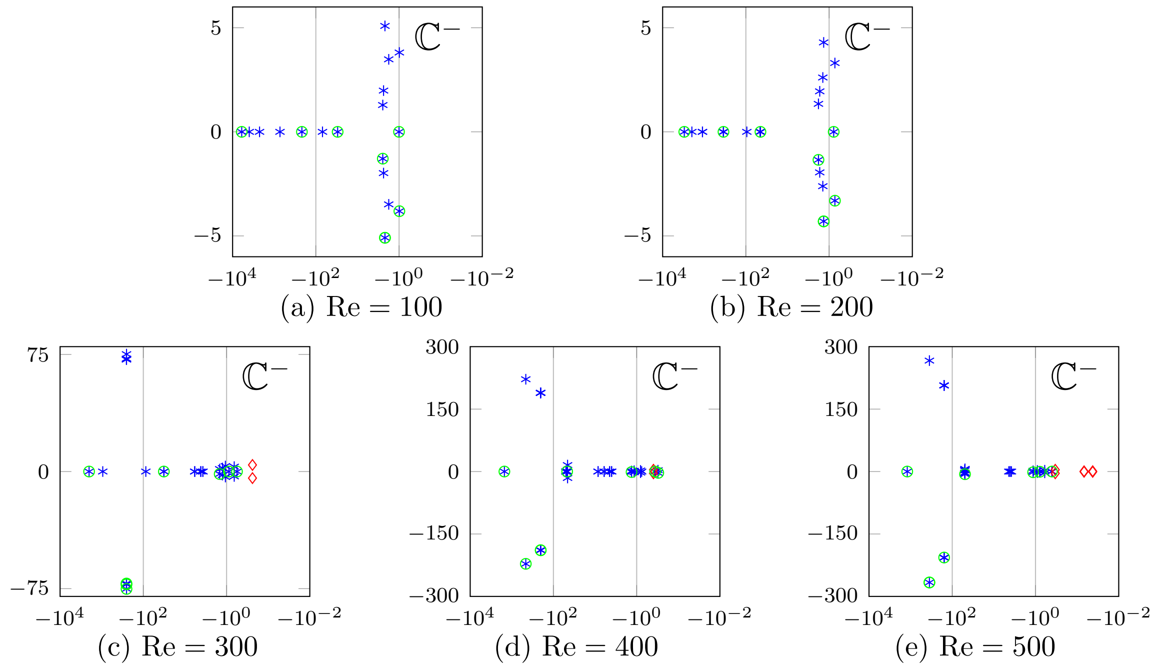


Figure 5.1.: **NSE scenario:** ADI shifts for various Reynolds numbers (used shifts: (∗), "bad shifts": (◊), selected shifts: (○)).

such a "bad shift" has never been computed during the shift selection process. For Re $= 400, 500$ the "bad shifts" seem to be close to the used shifts, but their distance is sufficient such that the used shifts do not yield badly conditioned matrices.

For each of these selected shifts, the shifted system matrix $\boldsymbol{F}_\ell$, as in (4.32a), is set up and the artificial right-hand side $\boldsymbol{b}_\ell = \boldsymbol{F}_\ell \cdot \mathbb{1}_n$ is built. Using $\boldsymbol{b}_\ell$, the exact error is defined as

$$\mathbf{err}_\ell := \mathbb{1}_n - \boldsymbol{x}_\ell \quad \text{with} \quad \boldsymbol{x}_\ell = \boldsymbol{F}_\ell \backslash \boldsymbol{b}_\ell.$$

Additionally, the 1-norm condition number of each $\boldsymbol{F}_\ell$ (`condest(`$\boldsymbol{F}_\ell$`)` in MATLAB) and the relative error $\dfrac{||\mathbf{err}_\ell||_1}{||\mathbb{1}_n||_1}$, using the 1-norm, are computed and the results are depicted in the Figures 5.2– 5.4. Thereby, the x-axis denotes $-|q_\ell|$. Notice that in this case the error is scaled by the length of the vector, since $||\mathbb{1}_n||_1 = n$. We choose the 1-norm for all considerations, since this norm is used for the computation of the condition number via `condest(`$\boldsymbol{F}_\ell$`)`. Using, e.g., the maximum norm results is qualitatively similar results. In detail, Figure 5.2 shows the results for the NSE scenario. The upper row depicts the result for varying Reynolds numbers for the coarsest refinement level. Increasing the magnitude of the shift q_ℓ increases the condition number drastically, as shown in Figure 5.2a. Thereby, the Reynolds number does not really affect the behavior. The relative error in Figure 5.2b increases in the same way as the condition number. This behavior can be observed for all refinement levels in Figure 5.2c. Moreover, the condition number for shifts $-10^3 < -|q_\ell| < -10^{-1}$ increases with each refinement level. For shifts with larger magnitude, the growing condition number seems to be similar for all refinement levels. However, the higher refinement levels choose shifts with larger magnitude such that the condition number grows further. The relative error behaves accordingly as depicted in Figure 5.2d.

Similar results are depicted for the Stokes scenario in Figure 5.3 (refinement: Level 1) and for the CFM scenario in Figure 5.4 (refinement: Level 4). In the Stokes case, it turns out that the condition number has its minimum for $|q_\ell| \approx 1$ and slightly grows for shifts with smaller magnitude. For the CFM scenario, one can notice a significant gap in the condition number and, therefore, in the relative error for the sets with $\mathrm{Re} = 1$ (Set I,II,IV) and sets with $\mathrm{Re} = 10$ (Set III,V). This gap vanishes for shifts $|q_\ell| > 10^3$.

All results show that the condition number has a huge influence on the accuracy. Although an ADI shift with a large magnitude increases the influence of the mass matrices and moves all eigenvalues towards the left, $\boldsymbol{F}_\ell$ is always indefinite due to its structure. Increasing the magnitude of the shift, therefore, moves the eigenvalues in $\mathbb{C}^+$ towards the imaginary axis $\jmath\mathbb{R}$ and decreases the magnitude of the smallest eigenvalue. Furthermore, the eigenvalues in $\mathbb{C}^-$ are shifted towards the left which increases the magnitude of the largest eigenvalue. Hence, the condition number grows drastically.

The solution produced by the sparse direct solver is often considered exact for the given finite arithmetic. Nevertheless, it is shown in the examples above that the relative error can be quite large.

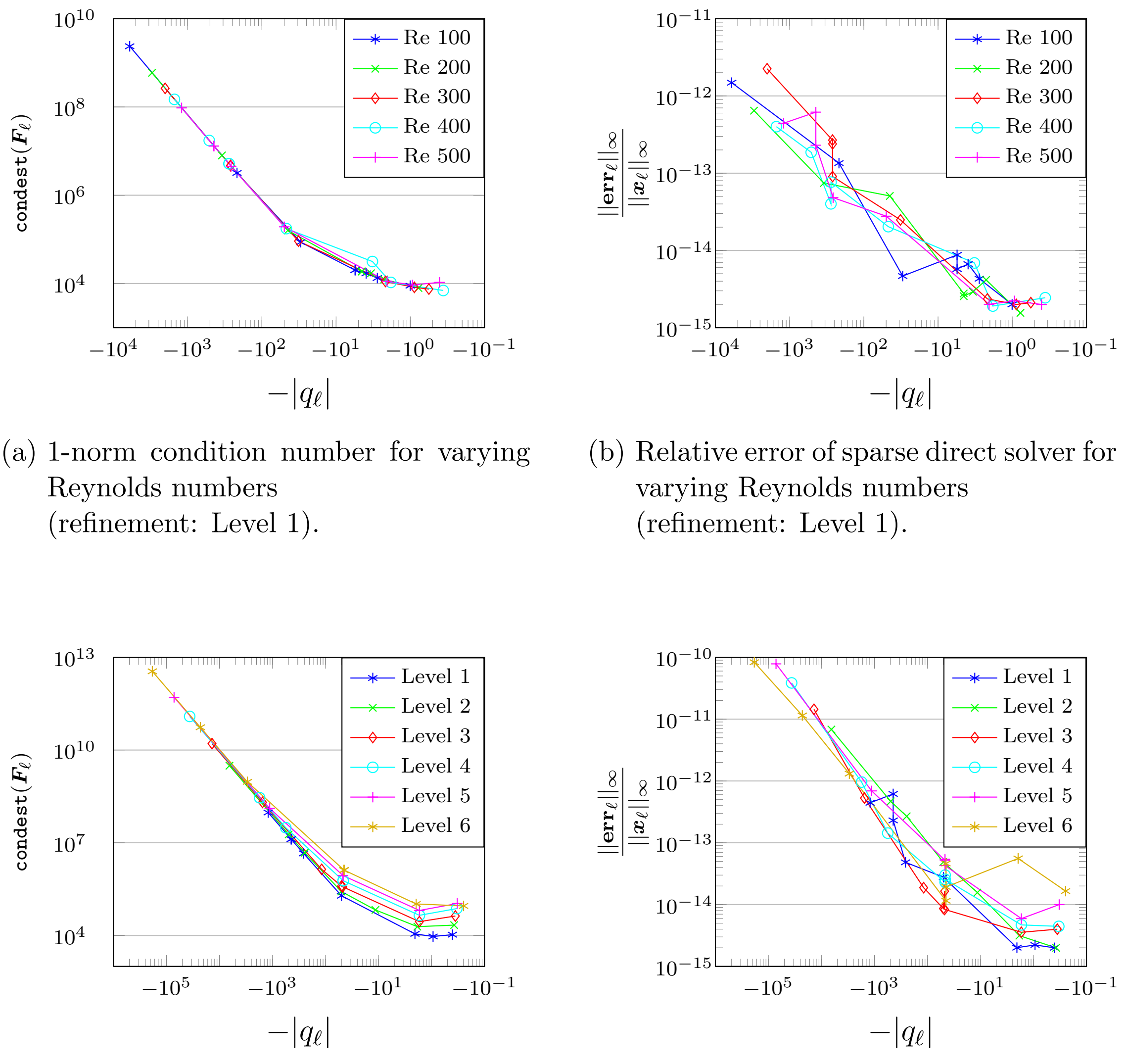
A big drawback with using sparse direct solvers is the still existing fill-in. Thus, the sparse direct solver cannot be applied anymore for increasing dimensions after a certain point since the memory requirements exceed the available memory. Specifying such a point is highly difficult and depends on the matrix structure and the available hardware architecture. Some more detailed studies, as well as special fill-in reducing ordering techniques, can be found in, e.g., [52, Chap. 6f]. “The practical limit for feasibility is often that direct sparse methods are competitive for two-dimensional PDE problems but iterative methods are required in three dimensions.”[56, p. 68]

Using an iterative method that only needs to perform matrix-vector products is examined in detail in the next section.

5.2. Iterative Solvers

In this section, the iterative solution of the SPS (4.32a) is investigated. The goal is to use the Krylov subspace method GMRES [114] that is introduced shortly in Subsection 2.4.3. The main advantage of such an iterative method is that it only requires matrix-vector multiplications with the sparse matrix $\boldsymbol{F}_\ell$. These matrix-vector products can be performed in a highly efficient manner.

Although most iterative methods “are well founded theoretically, they are all likely to suffer from slow convergence for problems that arise from typical applications such as fluid dynamics”[113, p. 261]. As it is pointed out in [35, 36] and the references therein, the performance of the iterative methods deteriorates with decreasing mesh-size. Furthermore, iterative methods are in general not very robust with regard to



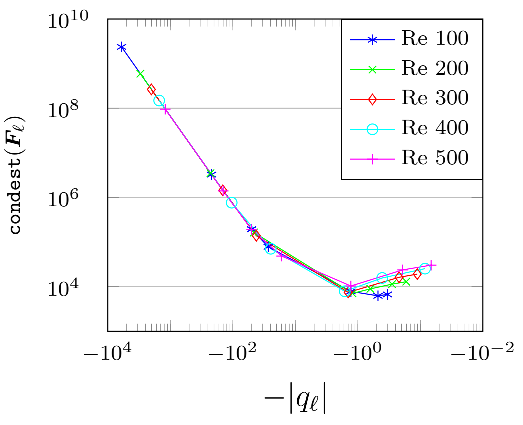
(a) 1-norm condition number for varying Reynolds numbers (refinement: Level 1).

(b) Relative error of sparse direct solver for varying Reynolds numbers (refinement: Level 1).

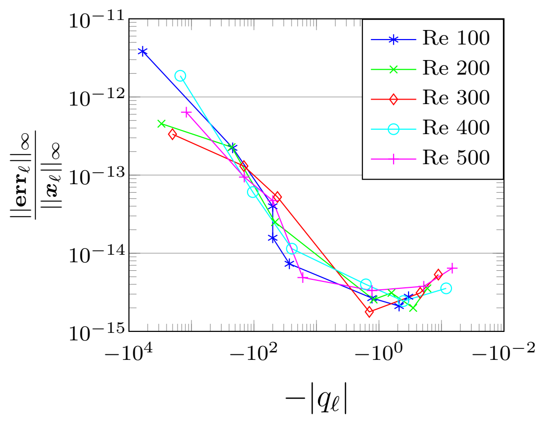
(c) 1-norm condition number for different refinement levels (Re = 500).

(d) Relative error of sparse direct solver for different refinement levels (Re = 500).

Figure 5.2.: **NSE scenario:** Influence of ADI shifts on the 1-norm condition number of $\boldsymbol{F}_\ell$ and the relative error in the 1-norm using the sparse direct solver in MATLAB.

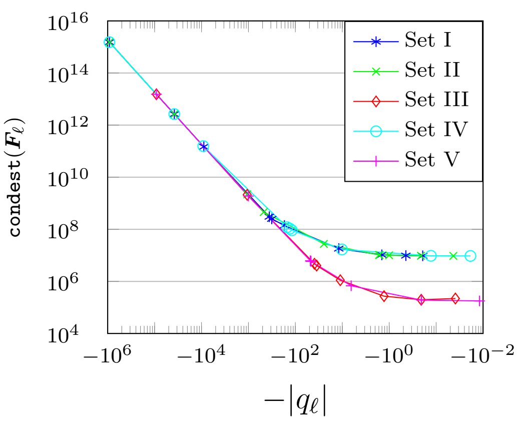


(a) 1-norm condition number for varying Reynolds numbers.

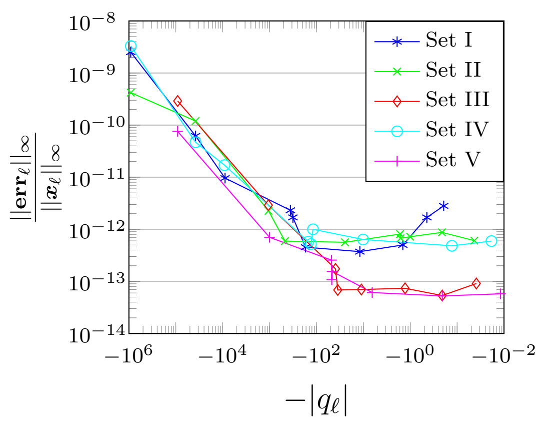


(b) Relative error of sparse direct solver for varying Reynolds numbers.

Figure 5.3.: **Stokes scenario:** Influence of ADI shifts on the 1-norm condition number of $\boldsymbol{F}_\ell$ and the relative error in the 1-norm using the sparse direct solver in MATLAB (refinement: Level 1).



(a) 1-norm condition number for varying Reynolds numbers.



(b) Relative error of sparse direct solver for varying Reynolds numbers.

Figure 5.4.: **CFM scenario:** Influence of ADI shifts on the 1-norm condition number of $\boldsymbol{F}_\ell$ and the relative error in the 1-norm using the sparse direct solver in MATLAB (refinement: Level 4).

parameter changes, as shown in, e.g., [113, Cha. 9].

To circumvent these problems, one usually considers a preconditioned SPS that can be defined for a suitable left preconditioner $\boldsymbol{P}_\ell \in \mathbb{C}^{N\times N}$ as

$$\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell\widehat{\boldsymbol{V}}_\ell = \boldsymbol{P}_\ell^{-1}\widehat{\boldsymbol{Y}}_{\ell-1}^{(k)}. \tag{5.2}$$

"Intuitively, if $\boldsymbol{P}_\ell$ can be chosen so that $\boldsymbol{P}_\ell^{-1}$ is an inexpensive approximate inverse of $\boldsymbol{F}_\ell$, then this might make a good preconditioner; however, it is not necessary for a good preconditioner to be such that $\boldsymbol{P}_\ell^{-1}$ is an approximate inverse of $\boldsymbol{F}_\ell$. A sufficient condition for a good preconditioner is that the preconditioned matrix $\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell$ has a low-degree minimum polynomial. This condition is more usually expressed in terms of $\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell$ having only a few distinct eigenvalues: in this form we must insist that $\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell$ is not degenerate (derogatory) or at least that its Jordan canonical form has Jordan blocks of only small dimension."[95, p. 1969] A more detailed convergence analysis of GMRES is not within the scope of this thesis but the interested reader is referred to, e.g., [90, Sec. 5.7.2ff] and [125] for such an analysis that also involves the right-hand side. More general details about preconditioned iterative methods can be found in [113, Chap. 9ff.] and [90] for general linear systems and in [56] for various transport problems.

The next subsection defines suitable block preconditioners $\boldsymbol{P}_\ell$, which are based on the generalized block structure (4.32a). Thereby, the approaches in [56, 95] are adapted straightforwardly. Some of these results are already published in [35] for the Stokes scenario and in [36] for the CFM scenario.

Notice that in this thesis only left preconditioners are considered. All derivations can be applied in a modified way as right or central preconditioner, as mentioned in, e.g., [95, Rem. 2]. Furthermore, the derivations in the following are restricted to the real-valued case, where $q_\ell \in \mathbb{R}^-$ and, thus, $\boldsymbol{F}_\ell \in \mathbb{R}^{N\times N}$ as used in [56, 95]. Later on in Subsection 5.2.3, an extension to the complex case is depicted.

5.2.1. Block Preconditioners

Efficient preconditioners for the SPS (4.32a) need to consider the block structure of the SPS in detail. Following the derivations in [95, Rem. 4] and [56, Sec. 8.1], one can define the block preconditioner $\boldsymbol{P}_\ell$ and its inverse via

$$\boldsymbol{P}_\ell = \begin{bmatrix} P_{F,\ell} & 0 \\ \widehat{G}^T & -P_{\mathrm{SC},\ell} \end{bmatrix} \in \mathbb{R}^{N\times N} \quad \Rightarrow \quad \boldsymbol{P}_\ell^{-1} = \begin{bmatrix} P_{F,\ell}^{-1} & 0 \\ P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1} & -P_{\mathrm{SC},\ell}^{-1} \end{bmatrix} \in \mathbb{R}^{N\times N}. \tag{5.3}$$

Thereby, $P_{F,\ell}$ is an approximation of $F_\ell := A^T + q_\ell M \in \mathbb{R}^{n\times n}$ and $P_{\mathrm{SC},\ell}$ an approximation of the *Schur complement* (SC) $\mathcal{T}_{\mathrm{SC},\ell} := \widehat{G}^T F_\ell^{-1}\widehat{G} \in \mathbb{R}^{m\times m}$; see, e.g., [56, eq. 8.6].

Applying $\boldsymbol{P}_\ell^{-1}$ from (5.3) to $\boldsymbol{F}_\ell$ yields

$$\begin{aligned} \boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell &= \begin{bmatrix} P_{F,\ell}^{-1} & 0 \\ P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1} & -P_{\mathrm{SC},\ell}^{-1} \end{bmatrix} \begin{bmatrix} F_\ell & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \\ &= \begin{bmatrix} P_{F,\ell}^{-1}F_\ell & P_{F,\ell}^{-1}\widehat{G} \\ P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1}F_\ell - P_{\mathrm{SC},\ell}^{-1}\widehat{G} & P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1}\widehat{G} \end{bmatrix}. \end{aligned} \tag{5.4a}$$

For an ideal preconditioner, i.e., $P_{F,\ell} = F_\ell$ and $P_{\mathrm{SC},\ell} = \mathcal{T}_{\mathrm{SC},\ell}$, one ends up with

$$\begin{bmatrix} I_n & F_\ell^{-1}\widehat{G} \\ \mathcal{T}_{\mathrm{SC},\ell}^{-1}\widehat{G}^T - \mathcal{T}_{\mathrm{SC},\ell}^{-1}\widehat{G} & \mathcal{T}_{\mathrm{SC},\ell}^{-1}\widehat{G}^T F_\ell^{-1}\widehat{G} \end{bmatrix} = \begin{bmatrix} I_n & * \\ 0 & I_m \end{bmatrix}, \tag{5.4b}$$

which is a preconditioned system with the sole eigenvalue 1, compare [95, Rem. 4], for which an iterative Krylov subspace method would converge within at most three iterations [95, Rem. 3].

***Remark* 5.3** *For the Stokes scenario, the SPS (4.32a) is symmetric and one could use a Krylov subspace method for symmetric systems together with a symmetric block diagonal preconditioner as explained in [56, Chap. 6] or [35, Sec. 3.2]. However, it is shown in [35, Sec. 4] that the block preconditioner (5.3) outperforms the symmetric version such that no further consideration is given to the symmetric version in this thesis.*

To apply the preconditioner (5.3) within the iterative method, the inverse $\boldsymbol{P}_\ell^{-1}$ is never formed explicitly, but the solution of the linear system

$$\begin{bmatrix} P_{F,\ell} & 0 \\ \widehat{G}^T & -P_{\mathrm{SC},\ell} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix} \tag{5.5a}$$

is considered. This system can be solved step-wise via

$$\text{Step I:} \qquad \boldsymbol{x}_1 = P_{F,\ell}^{-1}\boldsymbol{b}_1, \tag{5.5b}$$

$$\text{Step II:} \qquad \boldsymbol{x}_2 = P_{\mathrm{SC},\ell}^{-1}(\widehat{G}^T\boldsymbol{x}_1 - \boldsymbol{b}_2). \tag{5.5c}$$

The detailed block structures for the NSE and CFM scenarios are depicted in Table 5.1a for equation (5.3) and in Table 5.1b for the ideal preconditioned system (5.4) and for the preconditioning step (5.5). Thereby, all rectangular off-diagonal entries only yield a matrix-vector multiplication.

As mentioned in Remark 5.1, the shifted matrices $\widetilde{A} + q_\ell\widetilde{M}$ from Table 5.1a become singular if and only if the pencil $(\widetilde{A}, \widetilde{M})$ has unstable eigenvalues $\widetilde{\lambda}_{us} \in \mathbb{C}^+$ and $q_\ell = -\widetilde{\lambda}_{us}$. These unstable eigenvalues cannot be derived easily from the eigenvalues of the initially considered pencil $(\boldsymbol{A}, \boldsymbol{M})$, such that these situations cannot be recognized or prevented automatically. Since $\widetilde{M} \succ 0$, standard eigenvalue solvers can be used to efficiently compute the unstable eigenvalues $\widetilde{\lambda}_{us}$ during the offline process of computing the initial feedback if necessary. Later on in the shift computation algorithms, they can be sorted out as explained in Remark 5.1.

The crucial ingredient for an efficient preconditioner is a good approximation of the preconditioning blocks used in (5.5b)–(5.5c) and in Table 5.1. This means the steps (5.5b)–(5.5c) can be evaluated fast, without losing the "clustering of eigenvalues" property as explained in [56, Sec. 2.2]. Different approximations are explained in detail in the next subsection.

Table 5.1.: Overview of block structures of preconditioner.

(a) Block structure of preconditioner and its inverse.

$$\boldsymbol{P}_\ell = \begin{bmatrix} P_{F,\ell} & 0 \\ \widehat{G}^T & -P_{\mathrm{SC},\ell} \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{P}_\ell^{-1} = \begin{bmatrix} P_{F,\ell}^{-1} & 0 \\ P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1} & -P_{\mathrm{SC},\ell}^{-1} \end{bmatrix}, \qquad \begin{matrix} P_{\boldsymbol{z},\ell} \text{ approximates } F_{\boldsymbol{z},\ell} := A_{\boldsymbol{z}}^T + q_\ell M_{\boldsymbol{z}} \\ P_{\boldsymbol{c},\ell} \text{ approximates } F_{\boldsymbol{c},\ell} := A_{\boldsymbol{c}}^T + q_\ell M_{\boldsymbol{c}} \end{matrix}$$

	$P_{F,\ell}$	$\mathcal{T}_{\mathrm{SC},\ell}$	$P_{F,\ell}^{-1}$	$P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1}$
NSE	$P_{\boldsymbol{z},\ell}$	$G^T F_{\boldsymbol{z},\ell}^{-1} G$	$P_{\boldsymbol{z},\ell}^{-1}$	$P_{\mathrm{SC},\ell}^{-1} G^T P_{\boldsymbol{z}}^{-1}$
CFM	$\begin{bmatrix} P_{\boldsymbol{z},\ell} & -R_{\boldsymbol{z}}(c^{(\vec{w})})^T \\ 0 & P_{\boldsymbol{c},\ell} \end{bmatrix}$	$G^T F_{\boldsymbol{z},\ell}^{-1} G$	$\begin{bmatrix} P_{\boldsymbol{z},\ell}^{-1} & P_{\boldsymbol{z},\ell}^{-1} R_{\boldsymbol{z}}(c^{(\vec{w})})^T P_{\boldsymbol{c},\ell}^{-1} \\ 0 & P_{\boldsymbol{c},\ell}^{-1} \end{bmatrix}$	$\begin{bmatrix} P_{\mathrm{SC},\ell}^{-1} G^T P_{\boldsymbol{z},\ell}^{-1} & P_{\mathrm{SC},\ell}^{-1} G^T P_{\boldsymbol{z},\ell}^{-1} R_{\boldsymbol{z}}(c^{(\vec{w})})^T P_{\boldsymbol{c},\ell}^{-1} \end{bmatrix}$

(b) Block structure of applied preconditioner.

$$\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell = \begin{bmatrix} P_{F,\ell}^{-1} & 0 \\ P_{\mathrm{SC},\ell}^{-1}\widehat{G}^T P_{F,\ell}^{-1} & -P_{\mathrm{SC},\ell}^{-1} \end{bmatrix} \begin{bmatrix} F_\ell & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \approx \begin{bmatrix} I_n & * \\ 0 & I_m \end{bmatrix}$$

	ideal preconditioner	**preconditioning step**		
	$\boldsymbol{P}_\ell^{-1}\boldsymbol{F}_\ell^{-1}$	$\boldsymbol{x}$	$\boldsymbol{b}$	Solve: $\boldsymbol{x} = \boldsymbol{P}_\ell^{-1}\boldsymbol{b}$
NSE	$\begin{bmatrix} I_{n_{\boldsymbol{z}}} & F_{\boldsymbol{z},\ell}^{-1}G \\ 0 & I_{n_{\boldsymbol{p}}} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{x}_{\boldsymbol{z}} \\ \boldsymbol{x}_{\boldsymbol{p}} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{b}_{\boldsymbol{z}} \\ \boldsymbol{b}_{\boldsymbol{p}} \end{bmatrix}$	Step I: $\boldsymbol{x}_{\boldsymbol{z}} = P_{\boldsymbol{z},\ell}^{-1}\boldsymbol{b}_{\boldsymbol{z}}$ Step II: $\boldsymbol{x}_{\boldsymbol{p}} = P_{\mathrm{SC},\ell}^{-1}(G^T\boldsymbol{x}_{\boldsymbol{z}} - \boldsymbol{b})$
CFM	$\begin{bmatrix} I_{n_{\boldsymbol{z}}} & 0 & F_{\boldsymbol{z},\ell}^{-1}G \\ 0 & I_{n_{\boldsymbol{c}}} & 0 \\ 0 & 0 & I_{n_{\boldsymbol{p}}} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{x}_{\boldsymbol{z}} \\ \boldsymbol{x}_{\boldsymbol{c}} \\ \boldsymbol{x}_{\boldsymbol{p}} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{b}_{\boldsymbol{z}} \\ \boldsymbol{b}_{\boldsymbol{c}} \\ \boldsymbol{b}_{\boldsymbol{p}} \end{bmatrix}$	Step I.a: $\boldsymbol{x}_{\boldsymbol{c}} = P_{\boldsymbol{c},\ell}^{-1}\boldsymbol{b}_{\boldsymbol{c}}$ Step I.b: $\boldsymbol{x}_{\boldsymbol{z}} = P_{\boldsymbol{z},\ell}^{-1}(R_{\boldsymbol{z}}(c^{(\vec{w})})^T\boldsymbol{x}_{\boldsymbol{c}} + \boldsymbol{b}_{\boldsymbol{z}})$ Step II: $\boldsymbol{x}_{\boldsymbol{p}} = P_{\mathrm{SC},\ell}^{-1}(G^T\boldsymbol{x}_{\boldsymbol{z}} - \boldsymbol{b}_{\boldsymbol{p}})$

Table 5.2.: Substeps to apply the SC approximations in step II.

	$\boldsymbol{x} = \left(P_{\mathrm{SC},\ell}^{\mathrm{SPDC}}\right)^{-1} \boldsymbol{b}$		$\boldsymbol{x} = \left(P_{\mathrm{SC},\ell}^{\mathrm{LSC}}\right)^{-1} \boldsymbol{b}$	
Offline phase	Assemble:	$M_{\boldsymbol{p}},\ S_{\boldsymbol{p}},\ K_{\boldsymbol{p}}(\vec{w}) \in \mathbb{R}^{m\times m}$	Form:	$G_{\mathrm{LSC}} := \operatorname{diag}\left(M_{\boldsymbol{z}}\right)^{-1} G \in \mathbb{R}^{n\times m}$ $S_{\mathrm{LSC}} := G^T G_{\mathrm{LSC}} \in \mathbb{R}^{m\times m}$
Online phase	Substep II.a: Substep II.b: Substep II.c:	$\boldsymbol{x}_1 = S_{\boldsymbol{p}}^{-1}\boldsymbol{b}$ $\boldsymbol{x}_2 = F_{\boldsymbol{p},\ell}\,\boldsymbol{x}_1$ $\boldsymbol{x} = M_{\boldsymbol{p}}^{-1}\boldsymbol{x}_2$	Substep II.a: Substep II.b: Substep II.c:	$\boldsymbol{x}_1 = S_{\mathrm{LSC}}^{-1}\,\boldsymbol{b}$ $\boldsymbol{x}_2 = G_{\mathrm{LSC}}^T F_{\boldsymbol{z},\ell} G_{\mathrm{LSC}}\,\boldsymbol{x}_1$ $\boldsymbol{x} = S_{\mathrm{LSC}}^{-1}\,\boldsymbol{x}_2$

Table 5.3.: Configuration setups for block approximation methods.

	$P_{\boldsymbol{z}}^{-1}/P_{\boldsymbol{c}}^{-1}$	$S_{\boldsymbol{p}}^{-1}/S_{\mathrm{LSC}}^{-1}$	$M_{\boldsymbol{p}}^{-1}$
AAC	AGMG	AGMG	CSI
AAD	AGMG	AGMG	direct
ADD	AGMG	direct	direct
DDD	direct	direct	direct

5.2.2. Approximation Methods

The two main preconditioning blocks during the preconditioning step (5.5) are an approximation of the SC $\mathcal{T}_{\text{SC},\ell} \in \mathbb{R}^{m\times m}$ and an approximation of the shifted system matrix $F_\ell \in \mathbb{R}^{n\times n}$. Although $\mathcal{T}_{\text{SC},\ell}$ is of slightly smaller dimension $m \times m$, it involves the inverse F_ℓ^{-1} and is in general a dense matrix that should not be formed explicitly. In the following, two approximation techniques are explained.

Schur Complement (SC) Approximation The SC $\mathcal{T}_{\text{SC},\ell} = \widehat{G}^T F_\ell^{-1} \widehat{G}$ can be simplified in all scenarios to $\mathcal{T}_{\text{SC},\ell} = G^T F_{\boldsymbol{z},\ell}^{-1} G$ as depicted in Table 5.1a. Combining the statements from [56, Sec. 8.2] and [122, Sec. 3.2], two SC approximations are derived in the following. For both approximations, one considers that $F_{\boldsymbol{z},\ell}$ contains discrete components of the shifted Oseen operator

$$\mathscr{O}_{\vec{z},\ell} := -\frac{1}{\text{Re}}\Delta + \vec{w}\cdot\nabla + q_\ell \mathscr{I}.$$

Compared to the linearized NSE (3.14a), the reaction term $(\vec{z}\cdot\nabla)\vec{w}$ is omitted, which is common practice for the derivation of preconditioners; see, e.g., [56, Sec. 8.3.2]. Furthermore, one supposes that the analogous operator defined on the pressure space exists as

$$\mathscr{O}_{p,\ell} := (-\frac{1}{\text{Re}}\Delta + \vec{w}\cdot\nabla + q_\ell \mathscr{I})_p.$$

As stated in [56, Sec. 8.2], the second derivatives contradict the usual requirements on the pressure function $p(t,\vec{x})$ to be differentiable only once and one simply purports that these differential forms make sense. The key ingredient is to assume "that the commutator of the [shifted Oseen] operators with the gradient operator

$$\mathscr{C}_\ell := (\mathscr{O}_{\vec{z},\ell})\nabla - \nabla(\mathscr{O}_{p,\ell}) \approx 0 \tag{5.6}$$

is small in some sense"[56, p. 347]. The discrete version of (5.6) can be written as

$$(M_{\boldsymbol{z}}^{-1}F_{\boldsymbol{z},\ell})M_{\boldsymbol{z}}^{-1}G \approx M_{\boldsymbol{z}}^{-1}G(M_{\boldsymbol{p}}^{-1}F_{\boldsymbol{p},\ell}) \tag{5.7}$$

using the matrices from Subsection 3.5. Thereby, $F_{\boldsymbol{p},\ell} := -(\frac{1}{\text{Re}}S_{\boldsymbol{p}} + K_{\boldsymbol{p}}(\vec{w}))^T + q_\ell M_{\boldsymbol{p}}$ is the shifted system matrix defined on the pressure space.

The first SC approximation is based on [56, Sec. 8.2.1] and is called *shifted pressure diffusion-convection* (SPDC) approximation in the following. It can be derived if one multiplies (5.7) from the left with $G^T F_{\boldsymbol{z},\ell}^{-1} M_{\boldsymbol{z}}$ and from the right with $F_{\boldsymbol{p},\ell}^{-1} M_{\boldsymbol{p}}$ such that

$$G^T M_{\boldsymbol{z}}^{-1} G F_{\boldsymbol{p},\ell}^{-1} M_{\boldsymbol{p}} \approx G^T F_{\boldsymbol{z},\ell}^{-1} G = \mathcal{T}_{\text{SC},\ell}.$$

Similar to $\mathcal{T}_{\text{SC},\ell}$, the matrix $G^T M_{\boldsymbol{z}}^{-1} G \in \mathbb{R}^{m\times m}$ is in general a dense matrix that should not be formed explicitly. As shown in [56, Sec. 5.5.1], $G^T M_{\boldsymbol{z}}^{-1} G$ is spectrally equivalent to the stiffness matrix $S_{\boldsymbol{p}}$ if one uses an inf-sup stable discretization and considers an inflow-outflow problem. Hence,

$$P_{\mathrm{SC},\ell}^{\mathrm{SPDC}} := S_{\boldsymbol{p}} F_{\boldsymbol{p},\ell}^{-1} M_{\boldsymbol{p}} \quad \Rightarrow \quad \left(P_{\mathrm{SC},\ell}^{\mathrm{SPDC}}\right)^{-1} := M_{\boldsymbol{p}}^{-1} F_{\boldsymbol{p},\ell} S_{\boldsymbol{p}}^{-1}; \tag{5.8}$$

compare [36, Sec. 3.3]. In the end, applying $\left(P_{\mathrm{SC},\ell}^{\mathrm{SPDC}}\right)^{-1}$ in (5.5c) can be done within three substeps as depicted in the left row of Table 5.2. First, one needs to solve a pure Neumann problem on the pressure space that is formally denoted by $S_{\boldsymbol{p}}^{-1}$; compare [44]. The second substep is a matrix-vector multiplication with $F_{\boldsymbol{p},\ell}$ and substep three is the solution with the spd matrix $M_{\boldsymbol{p}}$. These substeps are evaluated as shown in the next paragraphs. For more details regarding this kind of SC approximation, the interested reader is referred to [49] for generalized Stokes problems, to [46, 92] for steady and unsteady Stokes problems, and to [93] for general PDEs.

The second approximation approach is based on [56, Sec. 8.2.2] and is only applicable for inf-sup stable discretizations. The idea is to "define an approximation to the matrix operator $F_{\boldsymbol{p},\ell}$ that makes the discrete commutator (5.7) small" [56, p. 353]. This technique is called *least-squares commutator* (LSC) approximation in the following and can be defined as

$$\begin{aligned} P_{\mathrm{SC},\ell}^{\mathrm{LSC}} &:= (G^T \widehat{M}_{\boldsymbol{z}}^{-1} G)(G^T \widehat{M}_{\boldsymbol{z}}^{-1} F_{\boldsymbol{z},\ell} \widehat{M}_{\boldsymbol{z}}^{-1} G)^{-1} (G^T \widehat{M}_{\boldsymbol{z}}^{-1} G) \\ \Rightarrow \quad \left(P_{\mathrm{SC},\ell}^{\mathrm{LSC}}\right)^{-1} &:= (G^T \widehat{M}_{\boldsymbol{z}}^{-1} G)^{-1} (G^T \widehat{M}_{\boldsymbol{z}}^{-1} F_{\boldsymbol{z},\ell} \widehat{M}_{\boldsymbol{z}}^{-1} G)(G^T \widehat{M}_{\boldsymbol{z}}^{-1} G)^{-1} \end{aligned} \tag{5.9}$$

with $\widehat{M}_{\boldsymbol{z}} = \operatorname{diag}(M_{\boldsymbol{z}})$; see [56, Sec. 8.2.2] for more details. Notice that $\widehat{M}_{\boldsymbol{z}}^{-1} G \in \mathbb{R}^{n \times m}$ and $G^T \widehat{M}_{\boldsymbol{z}}^{-1} G \in \mathbb{R}^{m \times m}$ are both sparse by construction and can be precomputed in the offline phase since they do not depend on any parameters. Applying $\left(P_{\mathrm{SC},\ell}^{\mathrm{LSC}}\right)^{-1}$ in (5.5c) can also be done in three substeps as illustrated in the right column of Table 5.2. Thereby, the first and third substeps are identical, namely the solution with $G^T \widehat{M}_{\boldsymbol{z}}^{-1} G$. As mentioned above, this is similar to solving with the stiffness matrix on the pressure space, which means solving a pure Neumann problem. The second substep consists of three matrix-vector multiplications using the precomputed matrix $\widehat{M}_{\boldsymbol{z}}^{-1} G$ and the shifted system matrix $F_{\boldsymbol{z},\ell}$.

More details about both SC approximations, as well as a performance comparison and an error analysis, can be found in [56, Sec. 8.2f]. Throughout the numerical tests all matrix-vector products are carried out directly, whereas all linear system solves are further approximated, which is sufficient for the application as preconditioner.

Algebraic Multigrid (AMG) Approximation In various steps and substeps of the above derived preconditioner the approximate solution of a sparse linear system

$$\widetilde{F} \boldsymbol{x} = \boldsymbol{b}$$

is needed, compare Table 5.1b and Table 5.2, with $\widetilde{F} \in \{P_{\boldsymbol{z},\ell}, P_{\boldsymbol{c},\ell}, M_{\boldsymbol{p}}, S_{\boldsymbol{p}}, S_{\mathrm{LSC}}\}$. All these matrices, except $M_{\boldsymbol{p}}$, are related to a stiffness matrix, which means they arise from discretizing second order elliptic PDEs. A highly efficient tool to approximate such systems is the *algebraic multigrid* (AMG) method [113, Sec. 13.6].

Standard multigrid techniques are based on a hierarchy of grids, such that the solution on the coarse grid is used as an initial guess for the fine grid. The system moves between these different grid levels through restrictions (this process is often called coarsening) and prolongations. Applying this idea recursively, the linear solves can be performed efficiently on a coarse grid of small dimension, where a direct solver can be applied. Afterwards, the solution is prolonged to the original space. Before coarsening the problem and after prolonging the solution, so-called smoothing steps are applied that quickly damp special error frequencies. Further details can be found in, e.g., [113, Sec. 13.4].

The main idea of an AMG method is to apply the multigrid idea in a purely algebraic way that only uses the information given in the matrix itself without the use of a discretization hierarchy. Thereby, strong and weak couplings within the connectivity graph are exploited to derive coarser versions of the original matrix. For more details, the interested reader is referred to, e.g., [67, 111].

In the numerical test in Subsection 5.2.4, the AGMG package developed by the group of Y. Notay [96, 99, 100] is used. The AGMG package needs to be interpreted as a nonlinear function. Hence, a flexible iterative method, e.g., FGMRES [113, Sec. 9.4.1], should be used. As stated in [36], no drawbacks can be observed while using a standard GMRES implementation, such that this fact is not further considered in the remainder.

Chebyshev Semi-Iteration As mentioned above, $M_{\boldsymbol{p}}$ is not related to a stiffness matrix and, therefore, can be approximated differently. Since $M_{\boldsymbol{p}}$ is a spd mass matrix, the conjugate gradient method [72] converges rapidly using a simple preconditioner like $\operatorname{diag}(M_{\boldsymbol{p}})$ as shown in [129]. An even faster approximation is given by the so-called *Chebyshev semi-iteration* (CSI) as described in detail in [130]. The CSI method exploits the fact that the eigenvalue bounds for (consistent) finite element mass matrices are known; compare [129]. During the numerical test our own implementation of [122, Alg. 2] is used.

To demonstrate the efficiency of the various approximation methods, four different configuration sets are considered as depicted in Table 5.3.

5.2.3. Complex Preconditioners

All considerations in subsection 5.2.2 are restricted to the real-valued case. The main reason for this restriction is the fact that all reviewed methods were originally developed for real-valued problems. Shifted SPS like (4.32a) arise in various applications, but the shift q_ℓ is usually related to a time step such that $q_\ell \in \mathbb{R}^+$. The ADI shifts q_ℓ are related to the spectrum of the pencil $(\boldsymbol{A} - \boldsymbol{B}\left(\boldsymbol{K}^{(k)}\right)^T, \boldsymbol{M})$, which becomes complex if $\boldsymbol{A}$ is not symmetric or $\boldsymbol{K}^{(k)}$ not empty. Hence, complex-valued shifts might arise even for symmetric problems at latest in the second Newton step.

In this subsection it is shown that the above introduced block preconditioner as well as the various approximation techniques can be applied straight forwardly to the complex case. Considering a complex linear system $A\boldsymbol{x} = \boldsymbol{b}$ with $A \in \mathbb{C}^{n\times n}$, $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{C}^n$, each component can be split into its real and imaginary parts such that $A\boldsymbol{x} = \boldsymbol{b}$ can be written as

$$\begin{aligned} &(A_r + \jmath A_i)(\boldsymbol{x}_r + \jmath \boldsymbol{x}_i) = (\boldsymbol{b}_r + \jmath \boldsymbol{b}_i) \\ \Leftrightarrow \quad &\begin{bmatrix} A_r & -A_i \\ A_i & A_r \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_r \\ \boldsymbol{x}_i \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_r \\ \boldsymbol{b}_i \end{bmatrix} \end{aligned} \tag{5.10}$$

with $A_r, A_i \in \mathbb{R}^{n\times n}$ and $\boldsymbol{x}_r, \boldsymbol{x}_i, \boldsymbol{b}_r, \boldsymbol{b}_i \in \mathbb{R}^n$.

Using a complex shift $q_\ell = q_{r,\ell} + \jmath q_{i,\ell} \in \mathbb{C}^-$ in (4.32a) and applying (5.10), yields

$$\begin{aligned} &\begin{bmatrix} A^T + (q_{r,\ell} + \jmath q_{i,\ell})M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} \widehat{V}_{r,\ell} + \jmath \widehat{V}_{i,\ell} \\ * \end{bmatrix} = \begin{bmatrix} \widehat{Y}^{(k)}_{r,\ell-1} + \jmath \widehat{Y}^{(k)}_{i,\ell-1} \\ 0 \end{bmatrix} \\ \Leftrightarrow \quad &\begin{bmatrix} A^T + q_{r,\ell}M & -q_{i,\ell}M & \widehat{G} & 0 \\ q_{i,\ell}M & A^T + q_{r,\ell}M & 0 & \widehat{G} \\ \widehat{G}^T & 0 & 0 & 0 \\ 0 & \widehat{G}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \widehat{V}_{r,\ell} \\ \widehat{V}_{i,\ell} \\ * \\ * \end{bmatrix} = \begin{bmatrix} \widehat{Y}^{(k)}_{r,\ell-1} \\ \widehat{Y}^{(k)}_{i,\ell-1} \\ 0 \\ 0 \end{bmatrix} \\ \Leftrightarrow \quad &\begin{bmatrix} \widetilde{\boldsymbol{F}}_\ell & \widetilde{\boldsymbol{G}} \\ \widetilde{\boldsymbol{G}}^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{V}}_\ell \\ * \end{bmatrix} = \begin{bmatrix} \widetilde{\boldsymbol{Y}}^{(k)}_{\ell-1} \\ 0 \end{bmatrix}, \end{aligned} \tag{5.11}$$

which is a linear system of the form (4.32a) with a system matrix in $\mathbb{R}^{2N\times 2N}$. Applying an ideal preconditioner like (5.3), which is adapted to the larger dimension, leads to a preconditioned system with a single eigenvalue of 1, as derived in (5.4).

It is shown next that the preconditioning steps (5.5) adapted to the real-valued system (5.11) are equivalent to the preconditioning steps applied to the original system with a complex shift q_ℓ. Considering the ideal preconditioner $P_{\widetilde{\boldsymbol{F}},\ell} = \widetilde{\boldsymbol{F}}_\ell \in \mathbb{R}^{2n\times 2n}$, (5.5b) yields

$$\begin{aligned} \begin{bmatrix} \boldsymbol{x}_{r,1} \\ \boldsymbol{x}_{i,1} \end{bmatrix} = \widetilde{\boldsymbol{F}}_\ell^{-1} \begin{bmatrix} \boldsymbol{b}_{r,1} \\ \boldsymbol{b}_{i,1} \end{bmatrix} \quad \Leftrightarrow \quad &\begin{bmatrix} A^T + q_{r,\ell}M & -q_{i,\ell}M \\ q_{i,\ell}M & A^T + q_{r,\ell}M \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{r,1} \\ \boldsymbol{x}_{i,1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_{r,1} \\ \boldsymbol{b}_{i,1} \end{bmatrix} \\ \Leftrightarrow \quad &\begin{matrix} (A^T + q_{r,\ell}M)\boldsymbol{x}_{r,1} - q_{i,\ell}M\boldsymbol{x}_{i,1} = \boldsymbol{b}_{r,1} \\ q_{i,\ell}M\boldsymbol{x}_{r,1} + (A^T + q_{r,\ell}M)\boldsymbol{x}_{i,1} = \boldsymbol{b}_{i,1} \end{matrix} \\ \overset{(5.10)}{\Leftrightarrow} \quad &(A^T + (q_{r,\ell} + \jmath q_{i,\ell})M)(\boldsymbol{x}_{r,1} + \jmath \boldsymbol{x}_{i,1}) = (\boldsymbol{b}_{r,1} + \jmath \boldsymbol{b}_{i,1}) \\ \Leftrightarrow \quad &(A^T + q_\ell M)\boldsymbol{x}_1 = \boldsymbol{b}_1 \\ \Leftrightarrow \quad &\boldsymbol{x}_1 = F_\ell^{-1}\boldsymbol{b}_1. \end{aligned} \tag{5.12}$$

Analogously, for $P_{\widetilde{\mathrm{SC}},\ell} = \mathcal{T}_{\widetilde{\mathrm{SC}},\ell} = \widetilde{\boldsymbol{G}}^T \widetilde{\boldsymbol{F}}_\ell^{-1} \widetilde{\boldsymbol{G}} \in \mathbb{R}^{2m\times 2m}$, the ideal preconditioning step

(5.5c) can be written as

$$\begin{aligned}
\begin{bmatrix}\boldsymbol{x}_{r,2}\\ \boldsymbol{x}_{i,2}\end{bmatrix} = \mathcal{T}^{-1}_{\widetilde{\text{SC}},\ell}\left(\widetilde{\boldsymbol{G}}^T\begin{bmatrix}\boldsymbol{x}_{r,1}\\ \boldsymbol{x}_{i,1}\end{bmatrix} - \begin{bmatrix}\boldsymbol{b}_{r,2}\\ \boldsymbol{b}_{i,2}\end{bmatrix}\right) \quad &\Leftrightarrow \quad \widetilde{\boldsymbol{G}}^T \underbrace{\widetilde{\boldsymbol{F}}_\ell^{-1}\begin{bmatrix}\widehat{G}\boldsymbol{x}_{r,2}\\ \widehat{G}\boldsymbol{x}_{i,2}\end{bmatrix}}_{=\begin{bmatrix}\boldsymbol{c}_r\\ \boldsymbol{c}_i\end{bmatrix}} = \left(\begin{bmatrix}\widehat{G}^T\boldsymbol{x}_{r,1}\\ \widehat{G}^T\boldsymbol{x}_{i,1}\end{bmatrix} - \begin{bmatrix}\boldsymbol{b}_{r,2}\\ \boldsymbol{b}_{i,2}\end{bmatrix}\right)\\
&\Leftrightarrow \quad \begin{bmatrix}\widehat{G}^T & 0\\ 0 & \widehat{G}^T\end{bmatrix}\begin{bmatrix}\boldsymbol{c}_r\\ \boldsymbol{c}_i\end{bmatrix} = \left(\begin{bmatrix}\widehat{G}^T\boldsymbol{x}_{r,1}\\ \widehat{G}^T\boldsymbol{x}_{i,1}\end{bmatrix} - \begin{bmatrix}\boldsymbol{b}_{r,2}\\ \boldsymbol{b}_{i,2}\end{bmatrix}\right)\\
&\Leftrightarrow \quad \widehat{G}^T\boldsymbol{c} = \widehat{G}^T\boldsymbol{x}_1 - \boldsymbol{b}_2\\
&\overset{(5.12)}{\Leftrightarrow} \quad \widehat{G}^T F_\ell^{-1}\widehat{G}\boldsymbol{x}_2 = \widehat{G}^T\boldsymbol{x}_1 - \boldsymbol{b}_2\\
&\Leftrightarrow \quad \boldsymbol{x}_2 = \mathcal{T}^{-1}_{\text{SC},\ell}\left(\widehat{G}^T\boldsymbol{x}_1 - \boldsymbol{b}_2\right).
\end{aligned}$$

In summary, the preconditioning steps for the $(2N \times 2N)$-dimensional real-valued system (5.11) are equivalent to steps (5.5) for the $(N \times N)$-dimensional system (5.4) using a complex shift. “Adapting GMRES to the complex case is fairly straightforward”[113, p. 184, Sec. 6.5.9], as implemented in the standard MATLAB implementation. Furthermore, AGMG is able to handle complex systems. However, it is required to provide a complex right-hand side for the AGMG method if the system matrix is complex. This cannot be guaranteed throughout the GMRES iteration. Furthermore, using the complex version turned out to be more cost intensive than solving the real-valued equivalent system. Therefore, all substeps involving AGMG are extended as described in (5.10). The basic GMRES iteration, however, operates on the original system independent of the property of the shift.

5.2.4. Numerical Experiments for the Iterative Solvers

To demonstrate the usability of the introduced preconditioning techniques, the same linear systems $\boldsymbol{F}_\ell\boldsymbol{x}_\ell = \boldsymbol{b}_\ell$ as in Subsection 5.1.2 are considered. For the various scenarios, each linear system is solved with the standard GMRES implementation within MATLAB using the block preconditioner (5.3). Depending on the used SC approximation, the results are labeled with SPDC or LSC; compare Table 5.2. Unless otherwise stated, the AAC setup from Table 5.3 is used in the following computations. Further GMRES starting parameters are the maximal iteration number of $n_{\max,\mathsf{GMRES}} = 500$, the GMRES tolerance $tol_{\mathsf{GMRES}} = 10^{-12}$, and a GMRES starting vector of all zeros. The results state the used shift q_ℓ, the number of iterations $\#_{\text{it}}$, the relative error $\frac{||\mathbf{err}_\ell||_1}{||\mathbb{1}_n||_1}$, and the relative residual $\frac{||\boldsymbol{F}_\ell\boldsymbol{x}_\ell - \boldsymbol{b}_\ell||_2}{||\boldsymbol{b}_\ell||_2}$. Notice that GMRES stops if the preconditioned residual fulfills

$$||\boldsymbol{P}_\ell^{-1}(\boldsymbol{F}_\ell\boldsymbol{x}_\ell - \boldsymbol{b}_\ell)||_2 \leq tol_{\mathsf{GMRES}}.$$

NSE: Tables 5.4–5.5 depict the results for the NSE scenario for the coarsest refinement level. The number of iterations, as well as the relative error and residual, stay relatively constant for an increasing Reynolds number and $|q_\ell| > 10$. For $|q_\ell| < 10$ and $\mathrm{Re} \leq 200$, the SPDC version uses less steps than the LSC method. In general, it seems that the relative error and residual are better if one uses the LSC method. In Tables 5.6–5.7 the results for $\mathrm{Re} = 500$ are depicted considering an increasing refinement level. In such a case the number of iterations increases with the refinement level if one uses the LSC approximation. Although the SPDC method uses more iterations than the LSC method for lower refinement levels, the number of iterations increases only slightly with an increasing refinement level. In fact, the LSC method does not converge within 500 iteration steps for the finest refinement level for small magnitude shifts. However, the relative error deteriorates for high refinement levels in the case of the SPDC method. For a more straightforward understanding of the methods convergence behavior, the numbers of iteration and the relative errors are also depicted in Figures 5.5–5.6.

Stokes: The results for the Stokes scenario are displayed in the Tables 5.8–5.9. Thereby, the LSC method clearly outperforms the SPDC method. The number of iterations are drastically smaller, especially for small magnitude shifts. Compared to the NSE scenario, the LSC methods need less iteration steps to converge. In general, the number of iterations required for the LSC method is independent of the Reynolds number.

CFM: For the CFM scenario, Tables 5.10–5.13 display the results for the refinements Level 1 and Level 4 for each parameter set in Table 4.2. It becomes apparent that the number of iterations stays constant for the various parameter sets for both approximation methods. As in the other scenarios, shifts with larger magnitude result in fewer iteration steps. Furthermore, the number of iterations depending on the used shift varies more for the LSC than for the SPDC approximation. Similar to the NSE results, the SPDC is independent of the used refinement level, whereas the number of iteration steps grows for an increasing refinement level for the LSC approximation.

All considered block preconditioners show good results for the different parameter variations. The SPDC approximation is, in general, slightly more expensive, but shows better robustness regarding the changing parameters. Especially the mesh refinement introduces a drastic growth of the iteration numbers for the LSC approximation in some cases. The fluctuating behavior regarding the changing ADI shift of the LSC method might be an indication that this approximation is not yet robust enough regarding the ADI shift. This behavior might be able to be incorporated in the shift selection process in the future. Furthermore, an adaptive switch between both methods depending on the used shift can be implemented easily. Finding the best switching point depends highly on the problem and is, therefore, not a simple task. The slightly deteriorated relative residual of the iterative methods might yield problems within the overall Kleinman–Newton process from Chapter 4. In [124], this influence on the convergence of the ADI iteration is investigated. Nevertheless, a method to determine the required tolerance adaptively during the solution process is not yet available.

Table 5.4.: **NSE scenario:** Results of GMRES iteration for varying Reynolds numbers – Part I (refinement: Level 1, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Re = 100						
$-1.0\cdot10^{0}$	101	$9.5\cdot10^{-13}$	$6.6\cdot10^{-12}$	74	$1.2\cdot10^{-11}$	$1.8\cdot10^{-10}$
$-2.5\cdot10^{0}-\jmath\,1.3\cdot10^{0}$	82	$9.8\cdot10^{-13}$	$4.3\cdot10^{-12}$	57	$5.2\cdot10^{-12}$	$3.1\cdot10^{-11}$
$-9.8\cdot10^{-1}-\jmath\,3.8\cdot10^{0}$	97	$1.1\cdot10^{-12}$	$4.1\cdot10^{-12}$	65	$7.0\cdot10^{-12}$	$3.9\cdot10^{-11}$
$-2.2\cdot10^{0}-\jmath\,5.1\cdot10^{0}$	82	$6.3\cdot10^{-13}$	$2.1\cdot10^{-12}$	52	$5.1\cdot10^{-12}$	$1.9\cdot10^{-11}$
$-2.2\cdot10^{0}-\jmath\,5.1\cdot10^{0}$	82	$6.3\cdot10^{-13}$	$2.1\cdot10^{-12}$	52	$5.1\cdot10^{-12}$	$1.9\cdot10^{-11}$
$-3.0\cdot10^{1}$	19	$2.7\cdot10^{-13}$	$1.9\cdot10^{-12}$	21	$4.8\cdot10^{-12}$	$8.0\cdot10^{-12}$
$-2.2\cdot10^{2}$	9	$2.5\cdot10^{-13}$	$6.3\cdot10^{-13}$	15	$2.3\cdot10^{-12}$	$4.1\cdot10^{-12}$
$-6.0\cdot10^{3}$	8	$1.0\cdot10^{-13}$	$8.8\cdot10^{-14}$	14	$1.0\cdot10^{-12}$	$2.7\cdot10^{-12}$
Re = 200						
$-7.8\cdot10^{-1}$	124	$1.9\cdot10^{-12}$	$7.4\cdot10^{-12}$	118	$7.6\cdot10^{-11}$	$7.1\cdot10^{-10}$
$-1.8\cdot10^{0}-\jmath\,1.3\cdot10^{0}$	103	$2.4\cdot10^{-12}$	$7.9\cdot10^{-12}$	91	$1.6\cdot10^{-11}$	$9.8\cdot10^{-11}$
$-7.2\cdot10^{-1}-\jmath\,3.3\cdot10^{0}$	125	$2.4\cdot10^{-12}$	$6.6\cdot10^{-12}$	109	$2.0\cdot10^{-11}$	$1.0\cdot10^{-10}$
$-1.4\cdot10^{0}-\jmath\,4.3\cdot10^{0}$	108	$2.2\cdot10^{-12}$	$4.2\cdot10^{-12}$	90	$1.6\cdot10^{-11}$	$5.8\cdot10^{-11}$
$-1.4\cdot10^{0}-\jmath\,4.3\cdot10^{0}$	108	$2.2\cdot10^{-12}$	$4.2\cdot10^{-12}$	90	$1.6\cdot10^{-11}$	$5.7\cdot10^{-11}$
$-4.5\cdot10^{1}$	14	$1.4\cdot10^{-13}$	$4.0\cdot10^{-13}$	20	$5.8\cdot10^{-12}$	$3.5\cdot10^{-12}$
$-3.5\cdot10^{2}$	8	$1.4\cdot10^{-13}$	$5.4\cdot10^{-13}$	14	$8.8\cdot10^{-12}$	$1.2\cdot10^{-11}$
$-3.0\cdot10^{3}$	8	$2.8\cdot10^{-13}$	$8.9\cdot10^{-14}$	14	$1.0\cdot10^{-12}$	$1.3\cdot10^{-12}$
Re = 300						
$-5.6\cdot10^{-1}$	155	$3.5\cdot10^{-12}$	$1.0\cdot10^{-11}$	166	$6.8\cdot10^{-11}$	$3.9\cdot10^{-10}$
$-8.8\cdot10^{-1}$	141	$4.4\cdot10^{-12}$	$1.1\cdot10^{-11}$	149	$1.2\cdot10^{-10}$	$7.7\cdot10^{-10}$
$-1.5\cdot10^{0}-\jmath\,1.6\cdot10^{0}$	123	$4.0\cdot10^{-12}$	$1.2\cdot10^{-11}$	126	$1.9\cdot10^{-11}$	$7.1\cdot10^{-11}$
$-3.2\cdot10^{1}$	14	$1.7\cdot10^{-13}$	$8.3\cdot10^{-13}$	26	$6.7\cdot10^{-12}$	$4.0\cdot10^{-12}$
$-2.5\cdot10^{2}-\jmath\,7.2\cdot10^{1}$	8	$1.1\cdot10^{-13}$	$7.1\cdot10^{-13}$	15	$1.6\cdot10^{-12}$	$3.1\cdot10^{-12}$
$-2.5\cdot10^{2}-\jmath\,7.2\cdot10^{1}$	8	$1.0\cdot10^{-13}$	$7.1\cdot10^{-13}$	15	$1.6\cdot10^{-12}$	$1.9\cdot10^{-12}$
$-2.5\cdot10^{2}-\jmath\,7.5\cdot10^{1}$	8	$8.3\cdot10^{-14}$	$7.1\cdot10^{-13}$	15	$1.7\cdot10^{-12}$	$3.5\cdot10^{-12}$
$-2.0\cdot10^{3}$	8	$7.9\cdot10^{-14}$	$1.3\cdot10^{-13}$	14	$1.7\cdot10^{-12}$	$1.4\cdot10^{-12}$

Table 5.5.: **NSE scenario:** Results of GMRES iteration for varying Reynolds numbers – Part II (refinement: Level 1, approximation methods: ACC).

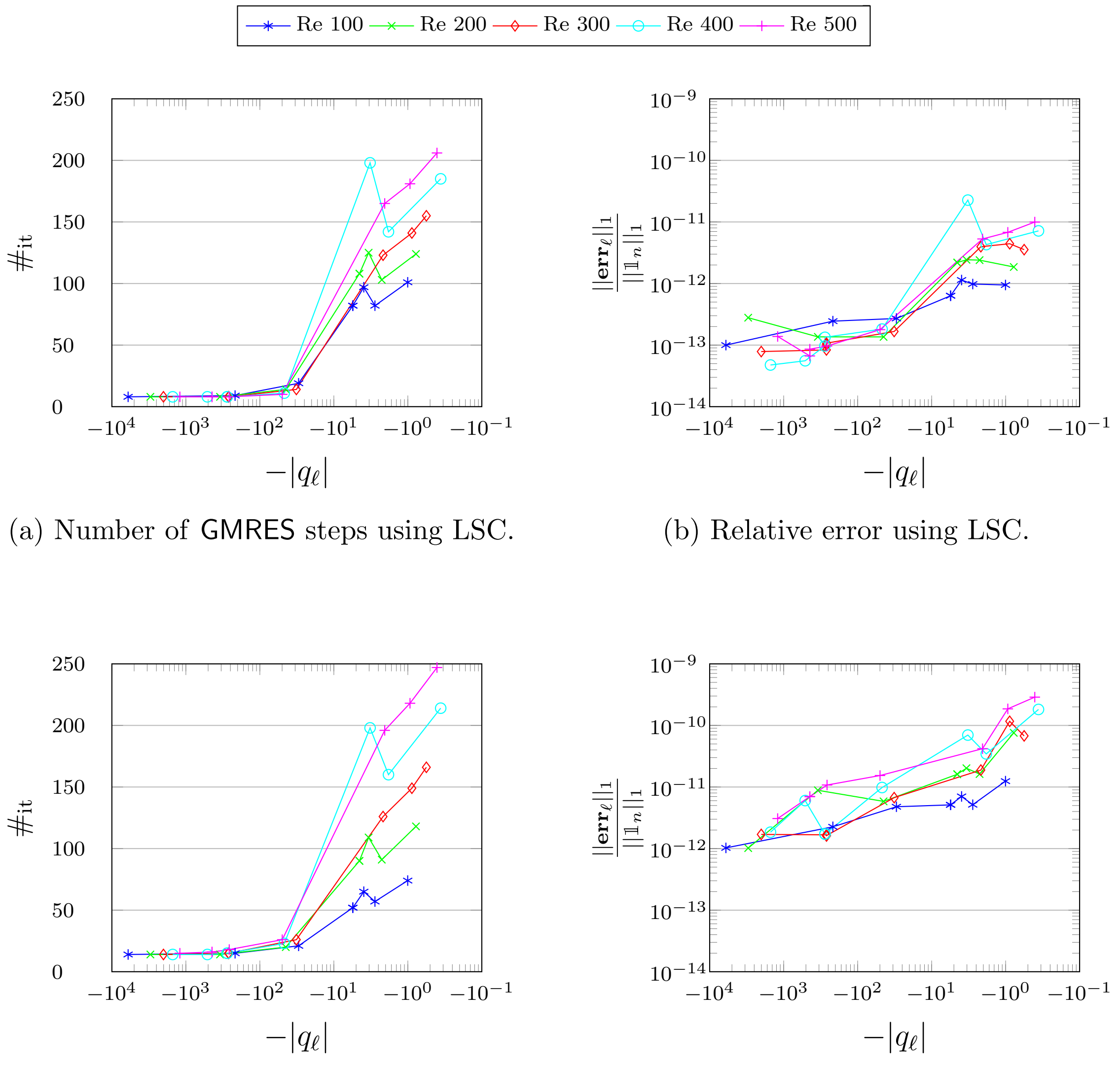
	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Re = 400						
$-3.6\cdot10^{-1}$	185	$7.2\cdot10^{-12}$	$1.8\cdot10^{-11}$	214	$1.8\cdot10^{-10}$	$1.0\cdot10^{-9}$
$-1.3\cdot10^{0}-\jmath\,1.3\cdot10^{0}$	142	$4.3\cdot10^{-12}$	$7.7\cdot10^{-12}$	160	$3.4\cdot10^{-11}$	$1.4\cdot10^{-10}$
$-3.0\cdot10^{-1}-\jmath\,3.2\cdot10^{0}$	198	$2.3\cdot10^{-11}$	$1.1\cdot10^{-11}$	198	$7.0\cdot10^{-11}$	$2.1\cdot10^{-10}$
$-4.7\cdot10^{1}$	11	$1.8\cdot10^{-13}$	$5.2\cdot10^{-13}$	23	$9.8\cdot10^{-12}$	$3.2\cdot10^{-12}$
$-2.0\cdot10^{2}-\jmath\,1.9\cdot10^{2}$	8	$1.3\cdot10^{-13}$	$6.7\cdot10^{-13}$	15	$1.7\cdot10^{-12}$	$2.2\cdot10^{-12}$
$-2.0\cdot10^{2}-\jmath\,1.9\cdot10^{2}$	8	$9.7\cdot10^{-14}$	$6.6\cdot10^{-13}$	15	$1.7\cdot10^{-12}$	$3.0\cdot10^{-12}$
$-4.6\cdot10^{2}-\jmath\,2.2\cdot10^{2}$	8	$5.6\cdot10^{-14}$	$3.6\cdot10^{-13}$	14	$6.0\cdot10^{-12}$	$6.1\cdot10^{-12}$
$-1.5\cdot10^{3}$	8	$4.8\cdot10^{-14}$	$1.4\cdot10^{-13}$	14	$1.8\cdot10^{-12}$	$2.9\cdot10^{-12}$
Re = 500						
$-4.0\cdot10^{-1}-\jmath\,2.7\cdot10^{-13}$	206	$1.0\cdot10^{-11}$	$1.5\cdot10^{-11}$	247	$2.9\cdot10^{-10}$	$1.2\cdot10^{-9}$
$-7.9\cdot10^{-1}-\jmath\,5.0\cdot10^{-1}$	181	$6.8\cdot10^{-12}$	$1.1\cdot10^{-11}$	218	$1.9\cdot10^{-10}$	$7.1\cdot10^{-10}$
$-1.1\cdot10^{0}-\jmath\,1.7\cdot10^{0}$	165	$5.3\cdot10^{-12}$	$8.3\cdot10^{-12}$	196	$4.2\cdot10^{-11}$	$1.3\cdot10^{-10}$
$-4.9\cdot10^{1}-\jmath\,6.4\cdot10^{0}$	10	$1.8\cdot10^{-13}$	$3.9\cdot10^{-13}$	26	$1.5\cdot10^{-11}$	$3.7\cdot10^{-12}$
$-1.6\cdot10^{2}-\jmath\,2.1\cdot10^{2}$	8	$9.7\cdot10^{-14}$	$7.2\cdot10^{-13}$	18	$1.1\cdot10^{-11}$	$4.4\cdot10^{-12}$
$-3.5\cdot10^{2}-\jmath\,2.7\cdot10^{2}$	8	$8.6\cdot10^{-14}$	$4.0\cdot10^{-13}$	16	$7.0\cdot10^{-12}$	$3.9\cdot10^{-12}$
$-3.5\cdot10^{2}-\jmath\,2.7\cdot10^{2}$	8	$6.7\cdot10^{-14}$	$4.0\cdot10^{-13}$	16	$7.0\cdot10^{-12}$	$3.1\cdot10^{-12}$
$-1.2\cdot10^{3}$	8	$1.4\cdot10^{-13}$	$1.9\cdot10^{-13}$	15	$3.1\cdot10^{-12}$	$3.6\cdot10^{-12}$

Table 5.6.: **NSE scenario:** Results of GMRES iteration for different refinement levels – Part I (Re = 500, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Level 1						
$-4.0\cdot10^{-1}-\jmath\,2.7\cdot10^{-13}$	206	$1.0\cdot10^{-11}$	$1.5\cdot10^{-11}$	247	$2.9\cdot10^{-10}$	$1.2\cdot10^{-9}$
$-7.9\cdot10^{-1}-\jmath\,5.0\cdot10^{-1}$	181	$6.8\cdot10^{-12}$	$1.1\cdot10^{-11}$	218	$1.9\cdot10^{-10}$	$7.1\cdot10^{-10}$
$-1.1\cdot10^{0}-\jmath\,1.7\cdot10^{0}$	165	$5.3\cdot10^{-12}$	$8.3\cdot10^{-12}$	196	$4.2\cdot10^{-11}$	$1.3\cdot10^{-10}$
$-4.9\cdot10^{1}-\jmath\,6.4\cdot10^{0}$	10	$1.8\cdot10^{-13}$	$3.9\cdot10^{-13}$	26	$1.5\cdot10^{-11}$	$3.7\cdot10^{-12}$
$-1.6\cdot10^{2}-\jmath\,2.1\cdot10^{2}$	8	$9.7\cdot10^{-14}$	$7.2\cdot10^{-13}$	18	$1.1\cdot10^{-11}$	$4.4\cdot10^{-12}$
$-3.5\cdot10^{2}-\jmath\,2.7\cdot10^{2}$	8	$8.6\cdot10^{-14}$	$4.0\cdot10^{-13}$	16	$7.0\cdot10^{-12}$	$3.9\cdot10^{-12}$
$-3.5\cdot10^{2}-\jmath\,2.7\cdot10^{2}$	8	$6.7\cdot10^{-14}$	$4.0\cdot10^{-13}$	16	$7.0\cdot10^{-12}$	$3.1\cdot10^{-12}$
$-1.2\cdot10^{3}$	8	$1.4\cdot10^{-13}$	$1.9\cdot10^{-13}$	15	$3.1\cdot10^{-12}$	$3.6\cdot10^{-12}$
Level 2						
$-3.7\cdot10^{-1}$	214	$3.9\cdot10^{-12}$	$1.8\cdot10^{-11}$	235	$1.0\cdot10^{-10}$	$1.5\cdot10^{-9}$
$-1.3\cdot10^{0}-\jmath\,1.3\cdot10^{0}$	171	$2.8\cdot10^{-12}$	$1.3\cdot10^{-11}$	183	$7.5\cdot10^{-11}$	$1.4\cdot10^{-9}$
$-1.1\cdot10^{1}$	44	$7.7\cdot10^{-13}$	$4.7\cdot10^{-12}$	58	$2.2\cdot10^{-11}$	$7.8\cdot10^{-11}$
$-5.0\cdot10^{1}-\jmath\,3.8\cdot10^{0}$	15	$3.0\cdot10^{-13}$	$1.0\cdot10^{-12}$	31	$9.1\cdot10^{-12}$	$1.4\cdot10^{-11}$
$-2.4\cdot10^{2}-\jmath\,2.8\cdot10^{1}$	9	$8.0\cdot10^{-14}$	$2.9\cdot10^{-13}$	22	$7.6\cdot10^{-12}$	$1.1\cdot10^{-11}$
$-4.9\cdot10^{2}$	9	$6.7\cdot10^{-14}$	$4.4\cdot10^{-13}$	20	$1.4\cdot10^{-11}$	$1.1\cdot10^{-11}$
$-6.5\cdot10^{3}$	9	$6.8\cdot10^{-13}$	$4.2\cdot10^{-13}$	17	$2.0\cdot10^{-11}$	$4.1\cdot10^{-12}$
Level 3						
$-3.6\cdot10^{-1}$	257	$6.7\cdot10^{-12}$	$7.2\cdot10^{-11}$	243	$2.1\cdot10^{-10}$	$2.3\cdot10^{-8}$
$-1.2\cdot10^{0}-\jmath\,1.2\cdot10^{0}$	224	$2.5\cdot10^{-12}$	$2.9\cdot10^{-11}$	190	$3.9\cdot10^{-11}$	$1.5\cdot10^{-9}$
$-4.4\cdot10^{1}-\jmath\,1.8\cdot10^{1}$	26	$2.9\cdot10^{-13}$	$1.3\cdot10^{-12}$	35	$8.9\cdot10^{-12}$	$3.6\cdot10^{-11}$
$-4.4\cdot10^{1}-\jmath\,1.8\cdot10^{1}$	27	$1.6\cdot10^{-13}$	$1.7\cdot10^{-12}$	34	$2.1\cdot10^{-11}$	$4.5\cdot10^{-11}$
$-4.9\cdot10^{1}-\jmath\,7.5\cdot10^{0}$	27	$1.0\cdot10^{-13}$	$1.8\cdot10^{-12}$	33	$1.1\cdot10^{-11}$	$2.9\cdot10^{-11}$
$-1.2\cdot10^{2}$	13	$4.3\cdot10^{-13}$	$2.5\cdot10^{-12}$	26	$7.9\cdot10^{-12}$	$2.1\cdot10^{-11}$
$-1.5\cdot10^{3}$	11	$4.1\cdot10^{-13}$	$1.1\cdot10^{-12}$	21	$6.7\cdot10^{-12}$	$3.7\cdot10^{-11}$
$-1.4\cdot10^{4}$	9	$1.2\cdot10^{-12}$	$3.8\cdot10^{-13}$	20	$4.0\cdot10^{-12}$	$4.2\cdot10^{-12}$

Table 5.7.: **NSE scenario:** Results of GMRES iteration for different refinement levels – Part II (Re = 500, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Level 4						
$-3.3\cdot10^{-1}$	379	$8.9\cdot10^{-12}$	$2.4\cdot10^{-10}$	275	$5.5\cdot10^{-10}$	$5.5\cdot10^{-8}$
$-1.2\cdot10^{0}-\jmath\,1.2\cdot10^{0}$	345	$1.9\cdot10^{-12}$	$5.0\cdot10^{-11}$	223	$5.2\cdot10^{-11}$	$4.3\cdot10^{-9}$
$-4.2\cdot10^{1}-\jmath\,2.1\cdot10^{1}$	52	$1.6\cdot10^{-13}$	$2.3\cdot10^{-12}$	40	$1.4\cdot10^{-11}$	$5.6\cdot10^{-11}$
$-4.2\cdot10^{1}-\jmath\,2.1\cdot10^{1}$	52	$6.3\cdot10^{-13}$	$1.8\cdot10^{-12}$	40	$1.4\cdot10^{-11}$	$5.4\cdot10^{-11}$
$-4.2\cdot10^{1}-\jmath\,2.1\cdot10^{1}$	52	$1.5\cdot10^{-13}$	$2.0\cdot10^{-12}$	40	$1.4\cdot10^{-11}$	$5.3\cdot10^{-11}$
$-5.6\cdot10^{2}$	15	$6.5\cdot10^{-13}$	$2.1\cdot10^{-12}$	28	$2.4\cdot10^{-12}$	$1.5\cdot10^{-10}$
$-1.7\cdot10^{3}$	8	$8.0\cdot10^{-13}$	$3.6\cdot10^{-12}$	22	$1.9\cdot10^{-10}$	$2.5\cdot10^{-10}$
$-3.6\cdot10^{4}$	8	$5.6\cdot10^{-12}$	$1.8\cdot10^{-12}$	20	$4.0\cdot10^{-10}$	$3.4\cdot10^{-10}$
Level 5						
$-3.3\cdot10^{-1}$	429	$1.2\cdot10^{-11}$	$5.3\cdot10^{-10}$	278	$3.1\cdot10^{-11}$	$5.5\cdot10^{-9}$
$-1.2\cdot10^{0}-\jmath\,1.2\cdot10^{0}$	418	$1.9\cdot10^{-12}$	$6.9\cdot10^{-11}$	229	$2.1\cdot10^{-11}$	$2.1\cdot10^{-9}$
$-4.2\cdot10^{1}-\jmath\,2.0\cdot10^{1}$	87	$3.5\cdot10^{-13}$	$3.6\cdot10^{-12}$	51	$1.5\cdot10^{-11}$	$2.0\cdot10^{-10}$
$-4.2\cdot10^{1}-\jmath\,2.0\cdot10^{1}$	85	$5.4\cdot10^{-13}$	$9.5\cdot10^{-12}$	51	$1.5\cdot10^{-11}$	$2.0\cdot10^{-10}$
$-4.2\cdot10^{1}-\jmath\,2.0\cdot10^{1}$	90	$4.2\cdot10^{-13}$	$3.6\cdot10^{-12}$	51	$1.5\cdot10^{-11}$	$1.8\cdot10^{-10}$
$-1.1\cdot10^{3}$	13	$2.2\cdot10^{-12}$	$3.6\cdot10^{-12}$	32	$5.4\cdot10^{-12}$	$4.5\cdot10^{-10}$
$-7.2\cdot10^{4}$	8	$1.8\cdot10^{-11}$	$2.5\cdot10^{-12}$	26	$2.2\cdot10^{-10}$	$1.2\cdot10^{-9}$
Level 6						
$-2.5\cdot10^{-1}$	500	$8.1\cdot10^{-9}$	$7.2\cdot10^{-7}$	302	$1.8\cdot10^{-9}$	$3.9\cdot10^{-7}$
$-9.9\cdot10^{-1}-\jmath\,1.7\cdot10^{0}$	500	$1.4\cdot10^{-9}$	$1.1\cdot10^{-7}$	257	$1.8\cdot10^{-11}$	$2.7\cdot10^{-9}$
$-3.8\cdot10^{1}-\jmath\,2.3\cdot10^{1}$	173	$1.4\cdot10^{-12}$	$9.8\cdot10^{-12}$	53	$2.4\cdot10^{-11}$	$1.1\cdot10^{-9}$
$-3.8\cdot10^{1}-\jmath\,2.3\cdot10^{1}$	173	$3.1\cdot10^{-12}$	$1.8\cdot10^{-11}$	53	$2.4\cdot10^{-11}$	$1.1\cdot10^{-9}$
$-3.8\cdot10^{1}-\jmath\,2.3\cdot10^{1}$	173	$5.7\cdot10^{-13}$	$2.5\cdot10^{-11}$	53	$2.4\cdot10^{-11}$	$1.3\cdot10^{-9}$
$-3.0\cdot10^{3}$	11	$4.0\cdot10^{-12}$	$3.7\cdot10^{-12}$	31	$1.9\cdot10^{-10}$	$3.6\cdot10^{-10}$
$-2.3\cdot10^{4}$	8	$4.4\cdot10^{-12}$	$1.4\cdot10^{-11}$	30	$1.2\cdot10^{-10}$	$4.2\cdot10^{-9}$
$-1.8\cdot10^{5}$	8	$3.2\cdot10^{-11}$	$1.0\cdot10^{-11}$	29	$1.9\cdot10^{-10}$	$2.0\cdot10^{-9}$



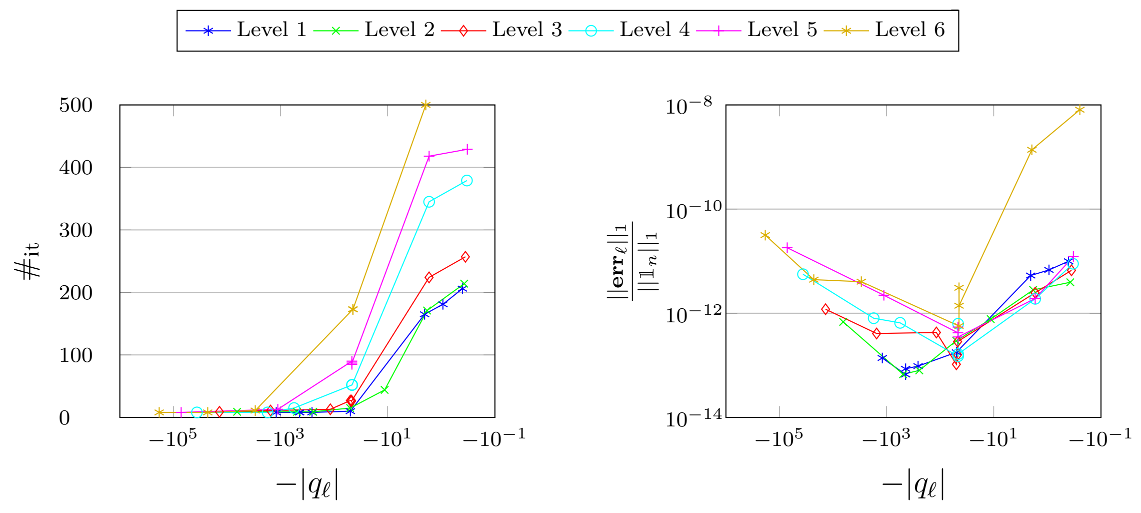
(a) Number of GMRES steps using LSC.

(b) Relative error using LSC.

(c) Number of GMRES steps using SPDC.

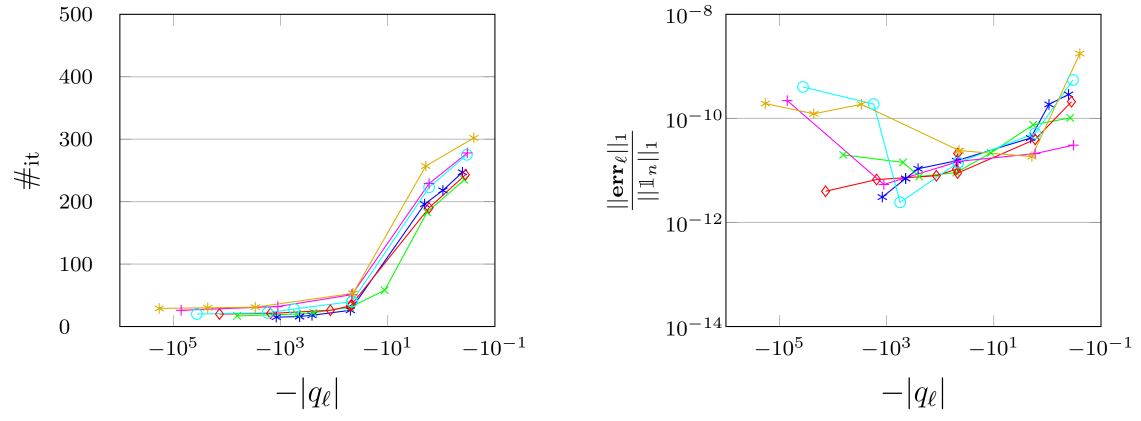
(d) Relative error using SPDC.

Figure 5.5.: **NSE scenario:** Influence of ADI shifts on the number of iterations $\#_{\mathrm{it}}$ and the relative error in the 1-norm using GMRES for different Reynolds numbers and different SC approximation methods
(refinement: Level 1, approximation methods: ACC).



(a) Number of GMRES steps using LSC.

(b) Relative error using LSC.



(c) Number of GMRES steps using SPDC.

(d) Relative error using SPDC.

Figure 5.6.: **NSE scenario:** Influence of ADI shifts on the number of iterations $\#_{\text{it}}$ and the relative error in the 1-norm using GMRES for refinement levels and different SC approximation methods
(Re = 500, approximation methods: ACC).

Table 5.8.: **Stokes scenario:** Results of GMRES iteration for varying Reynolds numbers – Part I (refinement: Level 1, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert \boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert \boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Re = 100						
$-3.4 \cdot 10^{-1}$	52	$2.8 \cdot 10^{-13}$	$6.3 \cdot 10^{-12}$	170	$3.1 \cdot 10^{-11}$	$9.5 \cdot 10^{-10}$
$-4.8 \cdot 10^{-1}$	50	$2.6 \cdot 10^{-13}$	$5.2 \cdot 10^{-12}$	162	$2.4 \cdot 10^{-11}$	$5.3 \cdot 10^{-10}$
$-1.3 \cdot 10^{0}$	42	$2.8 \cdot 10^{-13}$	$4.5 \cdot 10^{-12}$	133	$1.0 \cdot 10^{-11}$	$1.8 \cdot 10^{-10}$
$-2.7 \cdot 10^{1}$	17	$1.4 \cdot 10^{-13}$	$4.9 \cdot 10^{-13}$	32	$7.8 \cdot 10^{-12}$	$1.1 \cdot 10^{-11}$
$-5.0 \cdot 10^{1} - \jmath 8.2 \cdot 10^{-1}$	14	$1.5 \cdot 10^{-13}$	$1.3 \cdot 10^{-12}$	23	$5.1 \cdot 10^{-12}$	$6.2 \cdot 10^{-12}$
$-5.0 \cdot 10^{1} - \jmath 8.4 \cdot 10^{-1}$	14	$1.5 \cdot 10^{-13}$	$1.3 \cdot 10^{-12}$	23	$5.1 \cdot 10^{-12}$	$6.2 \cdot 10^{-12}$
$-2.2 \cdot 10^{2}$	9	$2.0 \cdot 10^{-13}$	$3.5 \cdot 10^{-13}$	15	$4.4 \cdot 10^{-12}$	$4.7 \cdot 10^{-12}$
$-6.0 \cdot 10^{3}$	8	$7.1 \cdot 10^{-14}$	$1.1 \cdot 10^{-13}$	14	$1.6 \cdot 10^{-12}$	$1.4 \cdot 10^{-12}$
Re = 200						
$-1.7 \cdot 10^{-1}$	53	$2.7 \cdot 10^{-13}$	$6.1 \cdot 10^{-12}$	244	$6.7 \cdot 10^{-10}$	$1.9 \cdot 10^{-8}$
$-2.8 \cdot 10^{-1} - \jmath 7.4 \cdot 10^{-3}$	50	$2.4 \cdot 10^{-13}$	$4.2 \cdot 10^{-12}$	235	$6.0 \cdot 10^{-11}$	$1.2 \cdot 10^{-9}$
$-6.3 \cdot 10^{-1}$	43	$2.9 \cdot 10^{-13}$	$5.2 \cdot 10^{-12}$	212	$1.6 \cdot 10^{-11}$	$1.2 \cdot 10^{-10}$
$-1.2 \cdot 10^{0}$	37	$2.5 \cdot 10^{-13}$	$4.4 \cdot 10^{-12}$	185	$7.7 \cdot 10^{-11}$	$2.1 \cdot 10^{-9}$
$-4.6 \cdot 10^{1}$	12	$2.1 \cdot 10^{-13}$	$1.2 \cdot 10^{-12}$	28	$6.7 \cdot 10^{-12}$	$6.1 \cdot 10^{-12}$
$-2.3 \cdot 10^{2}$	8	$1.1 \cdot 10^{-13}$	$5.5 \cdot 10^{-13}$	15	$4.0 \cdot 10^{-12}$	$4.2 \cdot 10^{-12}$
$-3.0 \cdot 10^{3}$	8	$8.9 \cdot 10^{-14}$	$8.2 \cdot 10^{-14}$	14	$1.1 \cdot 10^{-12}$	$1.7 \cdot 10^{-12}$
Re = 300						
$-1.1 \cdot 10^{-1}$	53	$4.0 \cdot 10^{-13}$	$9.0 \cdot 10^{-12}$	297	$4.8 \cdot 10^{-9}$	$1.2 \cdot 10^{-7}$
$-2.2 \cdot 10^{-1}$	49	$3.3 \cdot 10^{-13}$	$6.3 \cdot 10^{-12}$	288	$1.3 \cdot 10^{-9}$	$3.2 \cdot 10^{-8}$
$-1.4 \cdot 10^{0}$	32	$2.5 \cdot 10^{-13}$	$5.2 \cdot 10^{-12}$	205	$1.8 \cdot 10^{-11}$	$1.9 \cdot 10^{-10}$
$-4.2 \cdot 10^{1}$	11	$2.1 \cdot 10^{-13}$	$5.3 \cdot 10^{-13}$	32	$9.2 \cdot 10^{-12}$	$9.0 \cdot 10^{-12}$
$-1.4 \cdot 10^{2}$	8	$1.5 \cdot 10^{-13}$	$8.4 \cdot 10^{-13}$	16	$9.0 \cdot 10^{-12}$	$5.1 \cdot 10^{-12}$
$-2.0 \cdot 10^{3}$	8	$1.1 \cdot 10^{-13}$	$1.3 \cdot 10^{-13}$	14	$1.7 \cdot 10^{-12}$	$2.0 \cdot 10^{-12}$

Table 5.9.: **Stokes scenario:** Results of GMRES iteration for varying Reynolds numbers – Part II (refinement: Level 1, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$
Re = 400						
$-8.4 \cdot 10^{-2}$	53	$5.4 \cdot 10^{-13}$	$1.2 \cdot 10^{-11}$	330	$7.9 \cdot 10^{-9}$	$1.6 \cdot 10^{-7}$
$-4.1 \cdot 10^{-1}$	42	$2.0 \cdot 10^{-13}$	$3.0 \cdot 10^{-12}$	297	$8.1 \cdot 10^{-11}$	$1.4 \cdot 10^{-9}$
$-1.6 \cdot 10^{0}$	28	$2.5 \cdot 10^{-13}$	$4.6 \cdot 10^{-12}$	219	$2.7 \cdot 10^{-11}$	$4.1 \cdot 10^{-10}$
$-2.5 \cdot 10^{1}$	12	$2.2 \cdot 10^{-13}$	$9.3 \cdot 10^{-13}$	50	$1.1 \cdot 10^{-11}$	$2.0 \cdot 10^{-11}$
$-1.0 \cdot 10^{2}$	8	$2.0 \cdot 10^{-13}$	$1.2 \cdot 10^{-12}$	20	$5.4 \cdot 10^{-12}$	$4.6 \cdot 10^{-12}$
$-1.5 \cdot 10^{3}$	8	$1.2 \cdot 10^{-13}$	$1.6 \cdot 10^{-13}$	14	$1.9 \cdot 10^{-12}$	$3.2 \cdot 10^{-12}$
Re = 500						
$-6.7 \cdot 10^{-2}$	53	$6.5 \cdot 10^{-13}$	$1.5 \cdot 10^{-11}$	362	$1.3 \cdot 10^{-8}$	$2.1 \cdot 10^{-7}$
$-1.9 \cdot 10^{-1}$	46	$4.8 \cdot 10^{-13}$	$9.9 \cdot 10^{-12}$	347	$5.1 \cdot 10^{-10}$	$8.5 \cdot 10^{-9}$
$-1.3 \cdot 10^{0}$	29	$1.9 \cdot 10^{-13}$	$2.7 \cdot 10^{-12}$	256	$1.2 \cdot 10^{-10}$	$2.1 \cdot 10^{-9}$
$-1.7 \cdot 10^{1}$	13	$1.8 \cdot 10^{-13}$	$6.2 \cdot 10^{-13}$	75	$1.0 \cdot 10^{-11}$	$2.5 \cdot 10^{-11}$
$-5.0 \cdot 10^{1} - \jmath\, 2.1 \cdot 10^{0}$	9	$2.2 \cdot 10^{-13}$	$8.5 \cdot 10^{-13}$	39	$1.4 \cdot 10^{-11}$	$1.2 \cdot 10^{-11}$
$-1.4 \cdot 10^{2}$	8	$1.8 \cdot 10^{-13}$	$9.0 \cdot 10^{-13}$	24	$8.6 \cdot 10^{-12}$	$4.5 \cdot 10^{-12}$
$-1.2 \cdot 10^{3}$	8	$1.4 \cdot 10^{-13}$	$1.6 \cdot 10^{-13}$	15	$4.5 \cdot 10^{-12}$	$3.5 \cdot 10^{-12}$

Table 5.10.: **CFM scenario:** Results of GMRES iteration for different refinement levels – Part I (refinement: Level 1, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$
Set I						
$-1.9 \cdot 10^{-1}$	68	$4.6 \cdot 10^{-12}$	$3.1 \cdot 10^{-11}$	47	$1.4 \cdot 10^{-11}$	$9.3 \cdot 10^{-13}$
$-6.5 \cdot 10^{-1}$	67	$4.3 \cdot 10^{-12}$	$3.0 \cdot 10^{-11}$	40	$5.8 \cdot 10^{-12}$	$1.1 \cdot 10^{-12}$
$-1.5 \cdot 10^{0}$	66	$4.1 \cdot 10^{-12}$	$3.0 \cdot 10^{-11}$	36	$2.1 \cdot 10^{-12}$	$1.3 \cdot 10^{-12}$
$-4.1 \cdot 10^{0} - \jmath 4.2 \cdot 10^{-1}$	63	$2.9 \cdot 10^{-12}$	$2.2 \cdot 10^{-11}$	30	$4.4 \cdot 10^{-12}$	$2.2 \cdot 10^{-12}$
$-4.7 \cdot 10^{0} - \jmath 5.0 \cdot 10^{-1}$	63	$2.2 \cdot 10^{-12}$	$1.8 \cdot 10^{-11}$	30	$2.6 \cdot 10^{-12}$	$2.8 \cdot 10^{-12}$
$-1.4 \cdot 10^{1}$	57	$1.3 \cdot 10^{-12}$	$1.5 \cdot 10^{-11}$	26	$1.0 \cdot 10^{-11}$	$8.2 \cdot 10^{-12}$
$-2.2 \cdot 10^{2}$	28	$4.1 \cdot 10^{-13}$	$5.4 \cdot 10^{-13}$	23	$8.7 \cdot 10^{-12}$	$1.2 \cdot 10^{-11}$
$-1.6 \cdot 10^{3}$	12	$4.1 \cdot 10^{-13}$	$1.2 \cdot 10^{-13}$	22	$2.1 \cdot 10^{-10}$	$2.7 \cdot 10^{-11}$
$-4.3 \cdot 10^{4}$	9	$2.0 \cdot 10^{-11}$	$1.1 \cdot 10^{-13}$	23	$1.7 \cdot 10^{-10}$	$7.3 \cdot 10^{-11}$
Set II						
$-4.3 \cdot 10^{-2}$	68	$4.6 \cdot 10^{-12}$	$3.8 \cdot 10^{-11}$	50	$3.4 \cdot 10^{-11}$	$7.9 \cdot 10^{-13}$
$-2.1 \cdot 10^{-1} - \jmath 1.4 \cdot 10^{-2}$	68	$4.0 \cdot 10^{-12}$	$3.4 \cdot 10^{-11}$	46	$1.7 \cdot 10^{-11}$	$1.8 \cdot 10^{-12}$
$-1.1 \cdot 10^{0} - \jmath 1.1 \cdot 10^{0}$	67	$3.8 \cdot 10^{-12}$	$2.8 \cdot 10^{-11}$	38	$4.1 \cdot 10^{-12}$	$1.6 \cdot 10^{-12}$
$-1.1 \cdot 10^{0} - \jmath 1.1 \cdot 10^{0}$	67	$3.8 \cdot 10^{-12}$	$3.3 \cdot 10^{-11}$	38	$4.1 \cdot 10^{-12}$	$1.1 \cdot 10^{-12}$
$-1.1 \cdot 10^{0} - \jmath 1.1 \cdot 10^{0}$	67	$3.8 \cdot 10^{-12}$	$2.9 \cdot 10^{-11}$	38	$4.1 \cdot 10^{-12}$	$1.1 \cdot 10^{-12}$
$-1.7 \cdot 10^{1}$	55	$1.8 \cdot 10^{-12}$	$1.9 \cdot 10^{-11}$	26	$1.2 \cdot 10^{-12}$	$9.8 \cdot 10^{-12}$
$-5.8 \cdot 10^{2}$	18	$3.0 \cdot 10^{-13}$	$2.3 \cdot 10^{-13}$	22	$2.3 \cdot 10^{-10}$	$8.6 \cdot 10^{-12}$
$-1.5 \cdot 10^{3}$	13	$1.8 \cdot 10^{-13}$	$2.6 \cdot 10^{-13}$	23	$9.1 \cdot 10^{-11}$	$3.7 \cdot 10^{-11}$
$-4.3 \cdot 10^{4}$	9	$1.0 \cdot 10^{-11}$	$3.6 \cdot 10^{-13}$	23	$1.6 \cdot 10^{-10}$	$7.7 \cdot 10^{-11}$

Table 5.11.: **CFM scenario:** Results of GMRES iteration for different refinement levels – Part II (refinement: Level 4, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$	$\#_{\text{it}}$	$\frac{\lVert\mathbf{err}_\ell\rVert_1}{\lVert\mathbb{1}_n\rVert_1}$	$\frac{\lVert\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\rVert_2}{\lVert\boldsymbol{b}_\ell\rVert_2}$
Set I						
$-1.9\cdot10^{-1}$	134	$4.2\cdot10^{-11}$	$4.4\cdot10^{-9}$	51	$4.4\cdot10^{-12}$	$3.8\cdot10^{-12}$
$-4.4\cdot10^{-1}$	134	$6.2\cdot10^{-11}$	$3.8\cdot10^{-9}$	46	$2.1\cdot10^{-12}$	$2.9\cdot10^{-11}$
$-1.4\cdot10^{0}$	135	$1.3\cdot10^{-11}$	$1.0\cdot10^{-9}$	38	$5.8\cdot10^{-12}$	$3.9\cdot10^{-12}$
$-1.2\cdot10^{1}-\jmath\,2.8\cdot10^{-14}$	120	$1.5\cdot10^{-11}$	$3.2\cdot10^{-9}$	27	$8.2\cdot10^{-12}$	$1.6\cdot10^{-10}$
$-1.7\cdot10^{2}$	77	$2.2\cdot10^{-12}$	$4.9\cdot10^{-10}$	23	$2.4\cdot10^{-10}$	$4.8\cdot10^{-10}$
$-1.4\cdot10^{2}-\jmath\,2.9\cdot10^{2}$	75	$3.2\cdot10^{-12}$	$1.3\cdot10^{-10}$	24	$1.9\cdot10^{-10}$	$8.6\cdot10^{-10}$
$-2.0\cdot10^{2}-\jmath\,3.0\cdot10^{2}$	69	$4.1\cdot10^{-12}$	$1.1\cdot10^{-10}$	25	$1.4\cdot10^{-10}$	$9.3\cdot10^{-12}$
$-8.9\cdot10^{3}$	14	$5.4\cdot10^{-12}$	$1.5\cdot10^{-12}$	24	$4.1\cdot10^{-9}$	$9.2\cdot10^{-10}$
$-3.8\cdot10^{4}$	8	$6.7\cdot10^{-12}$	$3.8\cdot10^{-12}$	26	$7.2\cdot10^{-10}$	$8.5\cdot10^{-10}$
$-9.1\cdot10^{5}$	7	$1.4\cdot10^{-11}$	$3.6\cdot10^{-12}$	28	$1.9\cdot10^{-9}$	$1.5\cdot10^{-9}$
Set II						
$-4.3\cdot10^{-2}$	135	$1.6\cdot10^{-11}$	$4.5\cdot10^{-9}$	53	$1.0\cdot10^{-11}$	$3.6\cdot10^{-11}$
$-2.1\cdot10^{-1}-\jmath\,1.4\cdot10^{-2}$	134	$5.6\cdot10^{-11}$	$5.4\cdot10^{-9}$	50	$7.8\cdot10^{-12}$	$1.5\cdot10^{-11}$
$-8.6\cdot10^{-1}-\jmath\,5.2\cdot10^{-1}$	133	$6.9\cdot10^{-11}$	$4.1\cdot10^{-9}$	41	$7.3\cdot10^{-12}$	$7.4\cdot10^{-12}$
$-1.2\cdot10^{0}-\jmath\,1.1\cdot10^{0}$	133	$3.0\cdot10^{-11}$	$5.8\cdot10^{-9}$	43	$1.2\cdot10^{-12}$	$4.7\cdot10^{-11}$
$-1.2\cdot10^{0}-\jmath\,1.1\cdot10^{0}$	133	$3.6\cdot10^{-11}$	$2.9\cdot10^{-9}$	40	$1.8\cdot10^{-12}$	$4.3\cdot10^{-11}$
$-2.5\cdot10^{1}-\jmath\,1.4\cdot10^{-12}$	117	$4.9\cdot10^{-12}$	$2.9\cdot10^{-10}$	26	$1.3\cdot10^{-11}$	$3.8\cdot10^{-10}$
$-4.7\cdot10^{2}$	53	$4.1\cdot10^{-12}$	$5.5\cdot10^{-11}$	23	$3.8\cdot10^{-10}$	$5.1\cdot10^{-10}$
$-1.1\cdot10^{3}$	40	$1.7\cdot10^{-12}$	$2.6\cdot10^{-11}$	23	$5.4\cdot10^{-10}$	$4.2\cdot10^{-10}$
$-3.8\cdot10^{4}$	8	$1.4\cdot10^{-11}$	$2.3\cdot10^{-12}$	26	$7.7\cdot10^{-10}$	$9.4\cdot10^{-10}$
$-9.1\cdot10^{5}$	7	$3.4\cdot10^{-10}$	$1.2\cdot10^{-12}$	33	$2.1\cdot10^{-10}$	$5.7\cdot10^{-10}$

Table 5.12.: **CFM scenario:** Results of GMRES iteration for different refinement levels – Part I (refinement: Level 1, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell\boldsymbol{x}_\ell-\boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$
Set III						
$-3.9\cdot10^{-2}$	76	$4.3\cdot10^{-12}$	$3.1\cdot10^{-11}$	54	$3.6\cdot10^{-11}$	$3.5\cdot10^{-11}$
$-2.1\cdot10^{-1}-\jmath\,2.1\cdot10^{-2}$	74	$3.1\cdot10^{-12}$	$2.2\cdot10^{-11}$	45	$3.1\cdot10^{-11}$	$3.8\cdot10^{-11}$
$-1.1\cdot10^{0}-\jmath\,1.0\cdot10^{0}$	68	$1.9\cdot10^{-12}$	$1.3\cdot10^{-11}$	37	$1.3\cdot10^{-11}$	$1.2\cdot10^{-11}$
$-2.6\cdot10^{1}$	32	$3.4\cdot10^{-13}$	$8.7\cdot10^{-13}$	26	$4.1\cdot10^{-11}$	$9.4\cdot10^{-12}$
$-2.6\cdot10^{1}-\jmath\,2.6\cdot10^{1}$	29	$3.6\cdot10^{-13}$	$3.9\cdot10^{-13}$	26	$1.2\cdot10^{-10}$	$3.5\cdot10^{-12}$
$-2.6\cdot10^{1}-\jmath\,2.6\cdot10^{1}$	29	$2.9\cdot10^{-13}$	$4.8\cdot10^{-13}$	26	$1.2\cdot10^{-10}$	$1.6\cdot10^{-11}$
$-1.4\cdot10^{2}$	14	$1.8\cdot10^{-13}$	$2.5\cdot10^{-13}$	23	$5.5\cdot10^{-10}$	$3.8\cdot10^{-12}$
$-4.3\cdot10^{3}$	9	$4.1\cdot10^{-13}$	$1.4\cdot10^{-13}$	23	$1.0\cdot10^{-10}$	$6.0\cdot10^{-11}$
Set IV						
$-1.9\cdot10^{-2}$	68	$4.5\cdot10^{-12}$	$3.4\cdot10^{-11}$	51	$5.8\cdot10^{-11}$	$8.0\cdot10^{-13}$
$-1.4\cdot10^{-1}$	68	$4.2\cdot10^{-12}$	$3.2\cdot10^{-11}$	48	$1.7\cdot10^{-11}$	$1.0\cdot10^{-12}$
$-2.1\cdot10^{-1}-\jmath\,2.4\cdot10^{-1}$	68	$4.2\cdot10^{-12}$	$3.8\cdot10^{-11}$	45	$9.1\cdot10^{-12}$	$1.6\cdot10^{-12}$
$-3.1\cdot10^{-1}-\jmath\,4.0\cdot10^{-1}$	68	$4.2\cdot10^{-12}$	$3.2\cdot10^{-11}$	44	$2.5\cdot10^{-12}$	$9.5\cdot10^{-13}$
$-9.5\cdot10^{0}$	59	$1.7\cdot10^{-12}$	$1.8\cdot10^{-11}$	27	$9.3\cdot10^{-12}$	$9.9\cdot10^{-13}$
$-1.8\cdot10^{1}$	55	$1.4\cdot10^{-12}$	$1.4\cdot10^{-11}$	25	$2.2\cdot10^{-11}$	$7.6\cdot10^{-12}$
$-5.8\cdot10^{2}$	18	$3.4\cdot10^{-13}$	$4.8\cdot10^{-13}$	22	$2.3\cdot10^{-10}$	$3.9\cdot10^{-12}$
$-2.1\cdot10^{3}$	11	$6.2\cdot10^{-13}$	$2.3\cdot10^{-13}$	22	$6.6\cdot10^{-10}$	$4.1\cdot10^{-11}$
$-4.3\cdot10^{4}$	9	$3.2\cdot10^{-11}$	$2.4\cdot10^{-13}$	23	$2.2\cdot10^{-10}$	$9.5\cdot10^{-12}$
Set V						
$-1.2\cdot10^{-2}$	77	$4.3\cdot10^{-12}$	$3.2\cdot10^{-11}$	56	$1.2\cdot10^{-10}$	$1.3\cdot10^{-10}$
$-2.0\cdot10^{-1}-\jmath\,2.0\cdot10^{-1}$	75	$3.4\cdot10^{-12}$	$2.3\cdot10^{-11}$	46	$5.4\cdot10^{-11}$	$4.9\cdot10^{-11}$
$-2.7\cdot10^{-1}-\jmath\,3.5\cdot10^{-1}$	74	$3.7\cdot10^{-12}$	$2.5\cdot10^{-11}$	44	$3.9\cdot10^{-11}$	$2.9\cdot10^{-11}$
$-2.0\cdot10^{0}$	62	$1.9\cdot10^{-12}$	$1.2\cdot10^{-11}$	34	$2.4\cdot10^{-11}$	$1.4\cdot10^{-11}$
$-5.0\cdot10^{1}-\jmath\,1.0\cdot10^{1}$	23	$2.1\cdot10^{-13}$	$5.9\cdot10^{-13}$	25	$6.1\cdot10^{-11}$	$7.0\cdot10^{-12}$
$-1.4\cdot10^{2}$	14	$2.5\cdot10^{-13}$	$5.2\cdot10^{-13}$	23	$5.3\cdot10^{-10}$	$1.9\cdot10^{-11}$
$-4.3\cdot10^{3}$	9	$3.9\cdot10^{-13}$	$1.5\cdot10^{-13}$	23	$1.0\cdot10^{-10}$	$5.4\cdot10^{-12}$

Table 5.13.: **CFM scenario:** Results of GMRES iteration for different refinement levels – Part II (refinement: Level 4, approximation methods: ACC).

	LSC			SPDC		
q_ℓ	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$	$\#_{\text{it}}$	$\frac{\|\|\mathbf{err}_\ell\|\|_1}{\|\|\mathbb{1}_n\|\|_1}$	$\frac{\|\|\boldsymbol{F}_\ell \boldsymbol{x}_\ell - \boldsymbol{b}_\ell\|\|_2}{\|\|\boldsymbol{b}_\ell\|\|_2}$
Set III						
$-3.9 \cdot 10^{-2}$	147	$2.2 \cdot 10^{-11}$	$2.0 \cdot 10^{-9}$	55	$2.9 \cdot 10^{-10}$	$3.1 \cdot 10^{-10}$
$-2.1 \cdot 10^{-1} - \jmath\, 2.1 \cdot 10^{-2}$	148	$2.9 \cdot 10^{-11}$	$1.1 \cdot 10^{-9}$	46	$3.2 \cdot 10^{-10}$	$2.9 \cdot 10^{-10}$
$-9.1 \cdot 10^{-1} - \jmath\, 9.1 \cdot 10^{-1}$	140	$8.0 \cdot 10^{-12}$	$6.1 \cdot 10^{-10}$	38	$2.4 \cdot 10^{-11}$	$1.1 \cdot 10^{-10}$
$-1.1 \cdot 10^{1} - \jmath\, 2.3 \cdot 10^{-12}$	97	$3.6 \cdot 10^{-12}$	$5.1 \cdot 10^{-10}$	28	$9.9 \cdot 10^{-11}$	$6.4 \cdot 10^{-10}$
$-2.3 \cdot 10^{1} - \jmath\, 2.6 \cdot 10^{1}$	76	$4.0 \cdot 10^{-12}$	$1.1 \cdot 10^{-10}$	27	$1.6 \cdot 10^{-10}$	$7.3 \cdot 10^{-10}$
$-2.9 \cdot 10^{1} - \jmath\, 2.6 \cdot 10^{1}$	71	$4.4 \cdot 10^{-12}$	$1.2 \cdot 10^{-10}$	27	$7.7 \cdot 10^{-11}$	$6.2 \cdot 10^{-10}$
$-1.1 \cdot 10^{3}$	13	$1.1 \cdot 10^{-12}$	$3.6 \cdot 10^{-12}$	26	$7.7 \cdot 10^{-10}$	$5.4 \cdot 10^{-10}$
$-9.1 \cdot 10^{4}$	8	$3.6 \cdot 10^{-11}$	$9.2 \cdot 10^{-13}$	28	$1.1 \cdot 10^{-8}$	$1.7 \cdot 10^{-9}$
Set IV						
$-1.9 \cdot 10^{-2} - \jmath\, 7.2 \cdot 10^{-6}$	135	$4.5 \cdot 10^{-11}$	$3.6 \cdot 10^{-9}$	55	$9.5 \cdot 10^{-12}$	$3.3 \cdot 10^{-11}$
$-1.3 \cdot 10^{-1}$	135	$1.9 \cdot 10^{-11}$	$4.7 \cdot 10^{-9}$	52	$3.3 \cdot 10^{-12}$	$4.1 \cdot 10^{-11}$
$-1.0 \cdot 10^{1} - \jmath\, 3.0 \cdot 10^{-13}$	121	$1.5 \cdot 10^{-11}$	$3.7 \cdot 10^{-9}$	29	$1.5 \cdot 10^{-12}$	$4.6 \cdot 10^{-11}$
$-3.3 \cdot 10^{1} - \jmath\, 1.1 \cdot 10^{2}$	106	$1.1 \cdot 10^{-11}$	$2.5 \cdot 10^{-10}$	25	$1.3 \cdot 10^{-10}$	$6.5 \cdot 10^{-10}$
$-9.3 \cdot 10^{1} - \jmath\, 9.2 \cdot 10^{1}$	90	$1.1 \cdot 10^{-11}$	$3.5 \cdot 10^{-10}$	24	$7.7 \cdot 10^{-11}$	$8.6 \cdot 10^{-10}$
$-1.1 \cdot 10^{2} - \jmath\, 1.1 \cdot 10^{2}$	87	$3.7 \cdot 10^{-12}$	$1.4 \cdot 10^{-10}$	25	$8.3 \cdot 10^{-11}$	$3.2 \cdot 10^{-11}$
$-9.2 \cdot 10^{3}$	14	$5.8 \cdot 10^{-12}$	$2.5 \cdot 10^{-12}$	24	$5.0 \cdot 10^{-9}$	$9.0 \cdot 10^{-10}$
$-3.8 \cdot 10^{4}$	8	$1.5 \cdot 10^{-11}$	$4.1 \cdot 10^{-12}$	26	$7.5 \cdot 10^{-10}$	$6.0 \cdot 10^{-10}$
$-9.1 \cdot 10^{5}$	7	$2.6 \cdot 10^{-10}$	$2.5 \cdot 10^{-12}$	28	$2.2 \cdot 10^{-9}$	$2.3 \cdot 10^{-9}$
Set V						
$-1.2 \cdot 10^{-2}$	147	$3.9 \cdot 10^{-11}$	$2.3 \cdot 10^{-9}$	58	$6.5 \cdot 10^{-11}$	$9.1 \cdot 10^{-11}$
$-1.9 \cdot 10^{-1} - \jmath\, 7.2 \cdot 10^{-2}$	144	$1.9 \cdot 10^{-11}$	$1.8 \cdot 10^{-9}$	47	$2.4 \cdot 10^{-10}$	$2.4 \cdot 10^{-10}$
$-6.5 \cdot 10^{0}$	109	$8.4 \cdot 10^{-12}$	$4.2 \cdot 10^{-10}$	35	$2.5 \cdot 10^{-12}$	$2.3 \cdot 10^{-10}$
$-4.6 \cdot 10^{1} - \jmath\, 5.1 \cdot 10^{0}$	61	$3.4 \cdot 10^{-12}$	$9.0 \cdot 10^{-11}$	26	$2.0 \cdot 10^{-10}$	$2.0 \cdot 10^{-10}$
$-4.6 \cdot 10^{1} - \jmath\, 1.0 \cdot 10^{1}$	61	$9.0 \cdot 10^{-13}$	$3.1 \cdot 10^{-11}$	26	$1.8 \cdot 10^{-10}$	$6.2 \cdot 10^{-10}$
$-4.6 \cdot 10^{1} - \jmath\, 1.4 \cdot 10^{1}$	61	$3.1 \cdot 10^{-12}$	$7.5 \cdot 10^{-11}$	26	$1.6 \cdot 10^{-10}$	$3.8 \cdot 10^{-10}$
$-1.0 \cdot 10^{3} - \jmath\, 1.7 \cdot 10^{-11}$	15	$1.4 \cdot 10^{-12}$	$6.2 \cdot 10^{-13}$	26	$2.3 \cdot 10^{-10}$	$4.6 \cdot 10^{-10}$
$-9.1 \cdot 10^{4}$	7	$4.2 \cdot 10^{-11}$	$1.8 \cdot 10^{-12}$	33	$1.4 \cdot 10^{-10}$	$5.2 \cdot 10^{-10}$

The last part of this chapter investigates the efficiency of the approximation methods introduced in Subsection 5.2.2. Thereby, all systems of the NSE scenario with Re = 500 for the coarsest refinement level are considered. Using the different configuration setups in Table 5.3 for the LSC and the SPDC method, the GMRES convergence is identical. This indicates that the approximations are sufficient for the preconditioning process compared to solving the substeps numerically "exact" with a sparse direct solver. However, even for the coarsest mesh, the approximations (blue) outperform the direct solver (black) as depicted in Table 5.14.

The results in Table 5.14 are divided horizontally into two parts regarding the LSC and SPDC method. For each substep in Table 5.2, the times are accumulated during the entire GMRES process. Furthermore, the time to initialize the AGMG and the CSI method is measured. Notice that the substeps involving $S_{\text{LSC}}, S_{\boldsymbol{p}}, M_{\boldsymbol{p}}$ are independent of all parameters and need to be initialized only once during the entire process or could be initialized during offline phase of the procedure. The initialization times are displayed separately, indicated by a superscript "init". The table is vertically divided into three parts. The upper part shows the sum over all times for the eight different shifts q_ℓ. The middle and the lower part compare the influence of a complex shift and the subsequently extended linear systems as explained in Subsection 5.2.3. We chose shifts that needed the same amount of GMRES steps to be comparable.

Considering all shifts, the AAC method is three times faster than the DDD method. In all cases, the times to initialize the different approximations is at least one magnitude smaller than its actual execution. Furthermore, the LSC method is cheaper than the SPDC method. Regarding the complex shifts, the ACC method outperforms the DDD method to an even greater extent. If one considers finer mesh refinements, these speedups should increase further. Moreover, all approximation methods are based on a matrix-vector products such that an increasing dimension can be treated without the use of special hardware requirements.

5.3. Conclusion – Part II

The experiments in Subsection 5.1.2 and Subsection 5.2.4 show that both linear solvers are able to solve the SPS used in Chapter 4. Thereby, the ADI shift is the most influential component in the solution process.

On the one hand, the growing condition number for large magnitude ADI shifts effects the accuracy of the sparse direct solver drastically, as shown in Section 5.1. Nevertheless, all results are sufficient to compute the feedback during the KN-ADI method.

On the other hand, the block preconditioner introduced in Section 5.2 handles these ill-conditioned systems very well. However, ADI shifts with rather small magnitude result in longer and more expensive GMRES iterations. The robustness of the preconditioner should be investigated in more detail to come up with an overall robust method.

As mentioned above, an adaptive tolerance is not chosen and not yet investigated in detail. Nevertheless, the next chapter introduces such an adaptive accuracy one level higher in the nested KN-ADI method.

Table 5.14.: **NSE scenario:** Timings of the different configuration setups in Table 5.3 to solve the systems $\boldsymbol{F}_\ell \boldsymbol{x}_\ell = \boldsymbol{b}_\ell$ from Table 5.5 with Re = 500 (refinement: Level 1, blue highlights approximated solves).

		LSC				**SPDC**					
		$\text{AGMG}^{\text{init}}_{S_{\text{LSC}}}$	$\text{AGMG}_{F_\ell^{-1}}$	$\text{AGMG}_{S_{\text{LSC}}^{-1}}$	**total**	$\text{AGMG}^{\text{init}}_{S_{\boldsymbol{p}}}$	$\text{CSI}^{\text{init}}_{M_{\boldsymbol{p}}}$	$\text{AGMG}_{F_\ell^{-1}}$	$\text{AGMG}_{S_{\boldsymbol{p}}^{-1}}$	$\text{CSI}_{M_{\boldsymbol{p}}^{-1}}$	**total**
sum $\forall \mathbf{q}_\ell$	AAC	$1.8 \cdot 10^{-2}$	$4.3 \cdot 10^{0}$	$8.9 \cdot 10^{-1}$	$\mathbf{2.6 \cdot 10^{1}}$	$1.1 \cdot 10^{-2}$	$9.3 \cdot 10^{-4}$	$5.1 \cdot 10^{0}$	$7.0 \cdot 10^{-1}$	$3.9 \cdot 10^{-1}$	$\mathbf{3.3 \cdot 10^{1}}$
	AAD	$1.8 \cdot 10^{-2}$	$4.7 \cdot 10^{0}$	$9.5 \cdot 10^{-1}$	$\mathbf{2.6 \cdot 10^{1}}$	$1.1 \cdot 10^{-2}$	–	$5.4 \cdot 10^{0}$	$7.7 \cdot 10^{-1}$	$2.6 \cdot 10^{0}$	$\mathbf{4.0 \cdot 10^{1}}$
	ADD	–	$4.4 \cdot 10^{0}$	$7.1 \cdot 10^{0}$	$\mathbf{3.5 \cdot 10^{1}}$	–	–	$5.5 \cdot 10^{0}$	$9.1 \cdot 10^{-1}$	$2.5 \cdot 10^{0}$	$\mathbf{3.4 \cdot 10^{1}}$
	DDD	–	$4.6 \cdot 10^{1}$	$7.0 \cdot 10^{0}$	$\mathbf{7.5 \cdot 10^{1}}$	–	–	$6.1 \cdot 10^{1}$	$9.2 \cdot 10^{-1}$	$2.6 \cdot 10^{0}$	$\mathbf{9.4 \cdot 10^{1}}$
$q_\ell = -1205$	AAC	$1.8 \cdot 10^{-2}$	$1.3 \cdot 10^{-1}$	$3.0 \cdot 10^{-2}$	$\mathbf{1.5 \cdot 10^{-1}}$	$1.1 \cdot 10^{-2}$	$9.3 \cdot 10^{-4}$	$1.4 \cdot 10^{-1}$	$3.0 \cdot 10^{-2}$	$9.0 \cdot 10^{-3}$	$\mathbf{2.0 \cdot 10^{-1}}$
	AAD	$1.8 \cdot 10^{-2}$	$1.3 \cdot 10^{-1}$	$3.0 \cdot 10^{-2}$	$\mathbf{1.5 \cdot 10^{-1}}$	$1.1 \cdot 10^{-2}$	–	$1.5 \cdot 10^{-1}$	$3.1 \cdot 10^{-2}$	$5.2 \cdot 10^{-2}$	$\mathbf{2.5 \cdot 10^{-1}}$
	ADD	–	$1.2 \cdot 10^{-1}$	$9.1 \cdot 10^{-2}$	$\mathbf{2.1 \cdot 10^{-1}}$	–	–	$1.4 \cdot 10^{-1}$	$2.0 \cdot 10^{-2}$	$5.2 \cdot 10^{-2}$	$\mathbf{2.2 \cdot 10^{-1}}$
	DDD	–	$4.6 \cdot 10^{-1}$	$9.6 \cdot 10^{-2}$	$\mathbf{6.5 \cdot 10^{-1}}$	–	–	$9.4 \cdot 10^{-1}$	$2.0 \cdot 10^{-2}$	$5.2 \cdot 10^{-2}$	$\mathbf{1.1 \cdot 10^{0}}$
$q_\ell = -354 - \jmath 276$	AAC	$1.8 \cdot 10^{-2}$	$2.0 \cdot 10^{-1}$	$1.3 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{-1}}$	$1.1 \cdot 10^{-2}$	$9.3 \cdot 10^{-4}$	$2.4 \cdot 10^{-1}$	$1.5 \cdot 10^{-2}$	$9.1 \cdot 10^{-3}$	$\mathbf{2.8 \cdot 10^{-1}}$
	AAD	$1.8 \cdot 10^{-2}$	$2.0 \cdot 10^{-1}$	$1.4 \cdot 10^{-2}$	$\mathbf{1.7 \cdot 10^{-1}}$	$1.1 \cdot 10^{-2}$	–	$2.7 \cdot 10^{-1}$	$1.7 \cdot 10^{-2}$	$6.2 \cdot 10^{-2}$	$\mathbf{3.5 \cdot 10^{-1}}$
	ADD	–	$2.0 \cdot 10^{-1}$	$1.0 \cdot 10^{-1}$	$\mathbf{2.5 \cdot 10^{-1}}$	–	–	$2.4 \cdot 10^{-1}$	$2.2 \cdot 10^{-2}$	$6.0 \cdot 10^{-2}$	$\mathbf{3.2 \cdot 10^{-1}}$
	DDD	–	$7.1 \cdot 10^{-1}$	$1.0 \cdot 10^{-1}$	$\mathbf{9.3 \cdot 10^{-1}}$	–	–	$1.5 \cdot 10^{0}$	$2.2 \cdot 10^{-2}$	$6.0 \cdot 10^{-2}$	$\mathbf{1.7 \cdot 10^{0}}$

6 Inexact Low-Rank Kleinman–Newton-ADI Method

Contents

The sixth chapter introduces the second major contribution of this thesis, i.e., the inexact low-rank Kleinman–Newton-ADI method for index-2 DAE systems, which is a generalization of the presented method in [26] to the index-2 DAE case. In the first section, the derived method from Chapter 4 is reviewed and certain drawbacks are stated. In Section 6.2, the line search approach from [25] and the inexact Kleinman-Newton formulation from [59] are merged into a method, which is the major contribution established in this chapter. By adapting novel realizations of the involved methods, it is possible to incorporate and combine these well-known techniques, which could not be used together within this context before, to overcome the drawbacks indicated in Section 6.1. Section 6.3 introduces low-rank residual formulations, which are the key ingredients for an efficient implementation of this innovative method. In Section 6.4, numerical examples verify the usability of this novel approach.

6.1. Review of Kleinman–Newton-ADI Method

Although the applicability of the KN-ADI method for index-2 DAE systems has been verified in the articles [14, 15, 34–36], some drawbacks and numerical difficulties persists.

At first, consider the convergence behavior of the KN-ADI method. Although the KN-ADI for index-2 DAE systems converges globally for certain starting conditions, as shown in Theorem 4.5, the Riccati residual might grow drastically after the first Newton step. This behavior is not related to the special structure of the Riccati equation, but is well-known in the context of Newton's method, as described in [80, Chap. 8]. The numerical results in Section 4.4 show that this problem is not only an assumed possible consequence, but can be observed in, e.g., Figure 4.5b. In [25], Benner/Byers investigate the incorporation of a line search method into Newton's method to overcome this difficulty. Unfortunately, the method proposed in [25] is only applicable to small-scale problems since the computation involves the explicit handling of the dense Riccati residual (4.16c). Nevertheless, the line search approach from [25] is one of the major ingredients for the derivations of Chapter 6.

The second open problem is the determination of an appropriate accuracy for solving the linear Newton-step, i.e., the GCALE (4.21), iteratively. As stated in [80, Chap. 6] and [53], an inexact Newton method can be used to determine this accuracy. In [59], the approach is applied to large-scale Riccati equations. Unfortunately, the convergence proof in [59] is not applicable in the low-rank case, as shown in [26]. However, by combining the line search idea with the inexact Newton method, an innovative method can be derived. The fundamental key, which makes a combination of these different approaches possible and leads to a convergence proof, are low-rank residual formulations, as shown in [26]. The extension of the ideas in [26] to the index-2 DAE systems is straightforward and, additionally, avoids the explicit projection in each step of the computation of the projected residual from Subsection 4.3.2.

6.2. Inexact Kleinman–Newton Method with Line Search

To generalize the method in [26], the approach from [80, Sec. 8.2] is applied to the projected GCARE (4.16c) in the next subsection. Afterwards, two different line search approaches are investigated in Subsection 6.2.2 and the convergence behavior is analyzed in Subsection 6.2.3. The content is based on the statements in [26, Sec. 3] in a slightly adapted and extended version. All derivations in this section are formulated for the projected matrices in (4.15) that are never assembled explicitly, as explained in Remark 4.2. A way to efficiently perform all computations without the explicit use of the projected matrices is based on the statements in Subsection 4.2.2. To apply these statements for the inexact KNM with line search, some further modifications are necessary, as shown in Section 6.3.

6.2.1. Derivation of the Method

The basic inexact Newton result in [80, Thm. 6.1.2] states that one accepts the next Newton increment if the norm of the residual of the linear Newton step (4.20a) is smaller than the previous residual (4.16c) scaled by the so-called forcing parameter $\tau \in (0, 1)$. For the projected GCARE (4.16c), the increment $\mathcal{S} \in \mathbb{R}^{\widetilde{n} \times \widetilde{n}}$ in the k+1-st inexact Newton step is computed by solving (4.20a) iteratively until it holds that

$$||\mathcal{R}'(\mathcal{X}^{(k)})\mathcal{S}^{(k)} + \mathcal{R}(\mathcal{X}^{(k)})||_F \leq \tau_k ||\mathcal{R}(\mathcal{X}^{(k)})||_F. \tag{6.1}$$

To incorporate the line search idea from [80, Chap. 8], the increment in (4.20b) is scaled by $\xi_k > 0$ such that the new iterate is defined by

$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} + \xi_k \mathcal{S}^{(k)}. \tag{6.2}$$

Thereby, the step size ξ_k is chosen in order to fulfill the *sufficient decrease condition*

$$||\mathcal{R}\left(\mathcal{X}^{(k)} + \xi_k \mathcal{S}^{(k)}\right)||_F \leq (1 - \xi_k \beta) ||\mathcal{R}(\mathcal{X}^{(k)})||_F \tag{6.3}$$

with a certain safety parameter $\beta > 0$ and a step size ξ_k that is not unnecessarily small. In other words, the new iterate $\mathcal{X}^{(k+1)}$ has to ensure that the residual decreases monotonously. As shown in Subsection 6.2.3, the property that ξ_k is not unnecessarily small is important to ensure the convergence of the method and to prevent stagnation.

Instead of computing the new iterate $\mathcal{X}^{(k+1)}$ via (6.2), one sets $\xi_k = 1$ and defines the preliminary solution

$$\widetilde{\mathcal{X}}^{(k+1)} := \mathcal{X}^{(k)} + \mathcal{S}^{(k)}. \tag{6.4}$$

Following the statements in [26, Sec. 3.1], the residual of the Newton step (4.20) defines the preliminary projected Lyapunov residual

$$\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) := \mathcal{R}'(\mathcal{X}^{(k)})\mathcal{S}^{(k)} + \mathcal{R}(\mathcal{X}^{(k)}), \tag{6.5}$$

such that (6.1) simplifies to the *inexact Newton step condition*

$$||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F \leq \tau_k ||\mathcal{R}(\mathcal{X}^{(k)})||_F. \tag{6.6}$$

Using the definitions (4.18a), (4.21), (6.4), and (6.5), one can define the inexact Kleinman–Newton step

$$\left(\mathcal{A}^{(k)}\right)^T \widetilde{\mathcal{X}}^{(k+1)} \mathcal{M} + \mathcal{M} \widetilde{\mathcal{X}}^{(k+1)} \mathcal{A}^{(k)} = -\mathcal{W}^{(k)} \left(\mathcal{W}^{(k)}\right)^T + \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) \tag{6.7}$$

with a low-rank inhomogeneity $\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T$ that directly iterates on $\widetilde{\mathcal{X}}^{(k+1)}$ until the residual $\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})$ fulfills (6.6). Using (4.19) and (6.5), the residual of the projected CARE at (6.2) can be defined via

$$\begin{aligned}\mathcal{R}(\mathcal{X}^{(k)}+\xi_k\mathcal{S}^{(k)}) &= \mathcal{R}(\mathcal{X}^{(k)})+\xi_k\mathcal{R}'(\mathcal{X}^{(k)})\mathcal{S}^{(k)}+\frac{\xi_k^2}{2}\mathcal{R}''(\mathcal{X}^{(k)})(\mathcal{S}^{(k)},\mathcal{S}^{(k)})\\ &= (1-\xi_k)\mathcal{R}(\mathcal{X}^{(k)})+\xi_k\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})-\xi_k^2\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}.\end{aligned} \tag{6.8}$$

If the forcing parameters in (6.1) are limited by $\tau_k \le \bar{\tau} < 1$ and $\beta \in (0, 1-\bar{\tau})$, then (6.6) and (6.8) yield

$$\begin{aligned}&||\mathcal{R}\left(\mathcal{X}^{(k)}+\xi_k\mathcal{S}^{(k)}\right)||_F\\ &\le (1-\xi_k)||\mathcal{R}(\mathcal{X}^{(k)})||_F+\xi_k||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F+\xi_k^2\,||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||\\ &\le (1-\xi_k+\xi_k\bar{\tau})||\mathcal{R}(\mathcal{X}^{(k)})||_F+\xi_k^2\frac{||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||}{||\mathcal{R}(\mathcal{X}^{(k)})||_F}||\mathcal{R}(\mathcal{X}^{(k)})||_F\\ &= \left(1-\xi_k\left(1-\bar{\tau}-\xi_k\frac{||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||}{||\mathcal{R}(\mathcal{X}^{(k)})||_F}\right)\right)||\mathcal{R}(\mathcal{X}^{(k)})||_F\\ &\le (1-\xi_k\beta)||\mathcal{R}(\mathcal{X}^{(k)})||_F,\end{aligned}$$

where the step size ξ_k is limited by

$$0<\xi_k\le(1-\bar{\tau}-\beta)\frac{||\mathcal{R}(\mathcal{X}^{(k)})||_F}{||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||_F}. \tag{6.9}$$

Thus, every ξ_k for which (6.9) holds, satisfies the sufficient decrease condition (6.3) by construction.

As derived in Subsection 6.2.2, the norm of the residual (6.8) needs to be evaluated for various $\bar{\xi}_k \in (0,1)$ in the computation process of the sought step size ξ_k. The square of the residual norm can be defined via the quartic polynomial

$$\begin{aligned}f_{\mathcal{R}}^{(k)}(\bar{\xi}_k) &= ||\mathcal{R}\left(\mathcal{X}^{(k)}+\bar{\xi}_k\mathcal{S}^{(k)}\right)||_F^2\\ &= (1-\bar{\xi}_k)^2\upsilon_1^{(k)}+\bar{\xi}_k^2\upsilon_2^{(k)}+\bar{\xi}_k^4\upsilon_3^{(k)}+2\bar{\xi}_k(1-\bar{\xi}_k)\upsilon_4^{(k)}-2\bar{\xi}_k^2(1-\bar{\xi}_k)\upsilon_5^{(k)}-2\bar{\xi}_k^3\upsilon_6^{(k)}\end{aligned} \tag{6.10a}$$

with the scalar coefficients

$$\begin{aligned}\upsilon_1^{(k)} &= ||\mathcal{R}(\mathcal{X}^{(k)})||_F^2, & \upsilon_4^{(k)} &= \langle\mathcal{R}(\mathcal{X}^{(k)}),\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})\rangle,\\ \upsilon_2^{(k)} &= ||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F^2, & \upsilon_5^{(k)} &= \langle\mathcal{R}(\mathcal{X}^{(k)}),\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}\rangle,\\ \upsilon_3^{(k)} &= ||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||_F^2, & \upsilon_6^{(k)} &= \langle\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}),\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}\rangle.\end{aligned} \tag{6.10b}$$

Hence, after computing the coefficients (6.10b), one only needs to evaluate the scalar polynomial (6.10a) to compute the residual depending on $\bar{\xi}_k$.

To confirm that the Newton increment $\mathcal{S}^{(k)}$ describes a descent direction regarding the Riccati residual, one considers the first derivative of the polynomial (6.10a) with respect to a general ξ. Using the equivalences in (2.13), this derivative can be written as

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\xi} f_{\mathcal{R}}^{(k)}(\xi) &= \frac{\mathrm{d}}{\mathrm{d}\xi} ||\mathcal{R}\left(\mathcal{X}^{(k)} + \xi\mathcal{S}^{(k)}\right)||_F^2 \\
&= \frac{\mathrm{d}}{\mathrm{d}\xi} \operatorname{tr}\left(\mathcal{R}\left(\mathcal{X}^{(k)} + \xi\mathcal{S}^{(k)}\right)^2\right) \\
&= -2\operatorname{tr}\Big(\mathcal{R}\left(\mathcal{X}^{(k)} + \xi\mathcal{S}^{(k)}\right)\Big(\mathcal{R}(\mathcal{X}^{(k)}) - \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) + 2\xi\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}\Big)\Big) \\
&= -2\operatorname{tr}\Bigg(\Big((1-\xi)\mathcal{R}(\mathcal{X}^{(k)}) + \xi\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \xi^2\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}\Big) \\
&\qquad\qquad \times \Big(\mathcal{R}(\mathcal{X}^{(k)}) - \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) + 2\xi\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}\Big)\Bigg).
\end{aligned}$$

For $\xi = 0$ this yields

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\xi} f_{\mathcal{R}}^{(k)}(0) &= -2\operatorname{tr}\Big(\mathcal{R}(\mathcal{X}^{(k)})\Big(\mathcal{R}(\mathcal{X}^{(k)}) - \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})\Big)\Big) \\
&= -2\left(\operatorname{tr}\left(\mathcal{R}(\mathcal{X}^{(k)})^2\right) - \operatorname{tr}\left(\mathcal{R}(\mathcal{X}^{(k)})\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})\right)\right) \\
&= -2||\mathcal{R}(\mathcal{X}^{(k)})||_F^2 + 2\langle\mathcal{R}(\mathcal{X}^{(k)}), \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})\rangle \\
&\le -2||\mathcal{R}(\mathcal{X}^{(k)})||_F^2 + 2||\mathcal{R}(\mathcal{X}^{(k)})||_F||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F \\
&< -2||\mathcal{R}(\mathcal{X}^{(k)})||_F^2 + 2||\mathcal{R}(\mathcal{X}^{(k)})||_F||\mathcal{R}(\mathcal{X}^{(k)})||_F = 0,
\end{aligned}$$

using the Cauchy-Schwarz inequality $\langle\mathcal{R}(\mathcal{X}^{(k)}), \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})\rangle \le ||\mathcal{R}(\mathcal{X}^{(k)})||_F||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F$ and (6.6). This shows that $\mathcal{S}^{(k)}$ is a descent direction for $||\mathcal{R}(\mathcal{X}^{(k)} + \xi\mathcal{S}^{(k)})||_F$ in $||\mathcal{R}(\mathcal{X}^{(k)})||_F$ if (6.6) is fulfilled with $0 < \tau_k < 1$.

After choosing the step size ξ_k, the new iterate is defined as

$$\mathcal{X}^{(k+1)} = (1-\xi_k)\mathcal{X}^{(k)} + \xi_k\widetilde{\mathcal{X}}^{(k+1)}. \tag{6.11}$$

***Remark* 6.1 (cf. [26, Rem. 4])** *If the current iterate $\mathcal{X}^{(k)}$ is spsd, if the solution $\widetilde{\mathcal{X}}^{(k+1)}$ of (6.7) is spsd, and if $\xi_k \in (0,1]$, then $\mathcal{X}^{(k+1)}$ defined by (6.11) is also spsd.*

The entire algorithm of the inexact KNM with line search is depicted in Algorithm 4. The next subsection introduces two different approaches to determine the step size ξ_k.

6.2.2. Line Search Approaches

The line search idea is a well-known concept to improve globally convergent methods [80, Chap. 8]. In this subsection, two different approaches are reviewed that can be used to determine a step size ξ_k such that the sufficient decrease condition (6.3) is fulfilled. At first, the Armijo rule is examined based on the results in [80, Sec. 8.2] and [26, Sec. 3.2.1]. Secondly, the exact line search idea in [25] is modified for the inexact KNM.

Algorithm 4 Inexact Kleinman–Newton method with line search

Input: $\mathcal{A}$, $\mathcal{M}$, $\mathcal{B}$, $\mathcal{C}$, tol_{Newton}, initial stabilizing iterate $\mathcal{X}^{(0)}$, $\bar{\tau} \in (0,1)$, and $\beta \in (0, 1-\bar{\tau})$
Output: unique stabilizing solution $\mathcal{X}^{(*)}$ of GCARE (4.16c)

```
 1: Set $k = 0$.
 2: while $||\mathcal{R}(\mathcal{X}^{(k)})|| > tol_{\text{Newton}}$ do
 3:     $\mathcal{K}^{(k)} = \mathcal{M}\mathcal{X}^{(k)}\mathcal{B}$
 4:     Set $\mathcal{A}^{(k)} = \mathcal{A} - \mathcal{B}\left(\mathcal{K}^{(k)}\right)^T$, $\mathcal{W}^{(k)} = \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix}$.
 5:     Select $\tau_k \in (0, \bar{\tau}]$.
 6:     Compute a preliminary solution $\widetilde{\mathcal{X}}^{(k+1)}$ that solves:
            $\left(\mathcal{A}^{(k)}\right)^T \widetilde{\mathcal{X}}^{(k+1)}\mathcal{M} + \mathcal{M}\widetilde{\mathcal{X}}^{(k+1)}\mathcal{A}^{(k)} = -\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T + \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})$
        until $||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F \leq \tau_k ||\mathcal{R}(\mathcal{X}^{(k)})||_F$.
 7:     Set $\mathcal{S}^{(k)} = \widetilde{\mathcal{X}}^{(k+1)} - \mathcal{X}^{(k)}$.
 8:     Compute $\xi_k \in (0,1)$ such that $||\mathcal{R}\left(\mathcal{X}^{(k)} + \xi_k\mathcal{S}^{(k)}\right)||_F \leq (1-\xi_k\beta)||\mathcal{R}(\mathcal{X}^{(k)})||_F$.
 9:     Set $\mathcal{X}^{(k+1)} = (1-\xi_k)\mathcal{X}^{(k)} + \xi_k\widetilde{\mathcal{X}}^{(k+1)}$.
10:     $k = k + 1$
11: end while
12: $\mathcal{X}^{(*)} = \mathcal{X}^{(k)}$
```

Armijo Rule In [4], Armijo introduced a step size selection approach used for a steepest descent method. Given a real number $\bar{\xi}_k > 0$, the Armijo sequence computes $\xi_k^j = 2^{-j}\bar{\xi}_k$ and accepts $\xi_k = \xi_k^j$ as a new step size for j being the smallest integer such that ξ_k^j fulfills the sufficient decrease condition (6.3). Following the derivations in [80, Sec. 8.2], this approach can be formulated more generally by choosing the new ξ_k such that

$$\Upsilon_0 \xi_k^{\text{old}} \leq \xi_k^{\text{new}} \leq \Upsilon_1 \xi_k^{\text{old}},$$

where $0 < \Upsilon_0 \leq \Upsilon_1 < 1$. Thus, the parameter Υ_0 safeguards against ξ_k being too close to zero, which leads to a stagnation of the method.

By choosing $\Upsilon = \Upsilon_0 = \Upsilon_1 \in (0,1)$, the step size can be computed via $\xi_k = \Upsilon^j$, where j is the smallest integer such that (6.3) is fulfilled, as described in [26, Sec. 3.2.1]. Hence, the Armijo rule generates a step size that satisfies

$$\xi_k > \Upsilon(1-\bar{\tau}-\beta)\frac{||\mathcal{R}(\mathcal{X}^{(k)})||_F}{||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||_F}. \tag{6.12}$$

Setting $\Upsilon = \frac{1}{2}$ results in the method stated in [4].

The right hand side in (6.12) can be bounded by exploiting the structure of (4.16c), as shown in the next theorem, which is a generalization of [26, Thm. 5].

Theorem 6.2 *Let r_k denote the rank of the descent direction $\mathcal{S}^{(k)}$ and assume that $(\mathcal{A}^{(k)}, \mathcal{M})$ is stable. If all forcing parameters τ_k are bounded by $\tau_k \leq \bar{\tau} < 1$, then the step* *sizes obtained by the Armijo rule are bounded from below by*

$$\xi_k > \frac{\Upsilon(1-\bar{\tau}-\beta)}{r_k(1+\bar{\tau})^2}\frac{\lambda_{\mathcal{M},\min}^2}{\varkappa_F(\mathcal{M})^2\,||\mathcal{B}\mathcal{B}^T||_F||\mathcal{R}(\mathcal{X}^{(k)})||_F\left(\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,dt\right)^2} \tag{6.13}$$

with $\lambda_{\mathcal{M},\min} = \min\{|\lambda| : \lambda \in \Lambda(\mathcal{M})\}$.

Proof. The first step is to bound the solution $\mathcal{S}^{(k)}$ of (6.5), which can be rewritten as

$$\left(\mathcal{A}^{(k)}\mathcal{M}^{-1}\right)^T\mathcal{S}^{(k)} + \mathcal{S}^{(k)}\mathcal{A}^{(k)}\mathcal{M}^{-1} = \mathcal{M}^{-1}(\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)}))\mathcal{M}^{-1}. \tag{6.14}$$

Since the pencil $(\mathcal{A}^{(k)},\mathcal{M})$ is stable, the matrix $\mathcal{A}^{(k)}\mathcal{M}^{-1}$ that defines (6.14) is stable, such that $\mathcal{S}^{(k)}$ can be defined via

$$\mathcal{S}^{(k)} = \int_0^\infty e^{(\mathcal{A}^{(k)}\mathcal{M}^{-1})^Tt}(\mathcal{M}^{-1}(\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)}))\mathcal{M}^{-1})e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}\,\mathrm{d}t.$$

The spectral norm of $\mathcal{S}^{(k)}$ can be bounded by

$$||\mathcal{S}^{(k)}||_2 \le ||\mathcal{M}^{-1}||_2^2||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)})||_2\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t. \tag{6.15}$$

For any matrix $\mathcal{S}$ with $\operatorname{rank}(\mathcal{S}) = r$ it holds that $||\mathcal{S}||_2 \le ||\mathcal{S}||_F \le \sqrt{r}||\mathcal{S}||_2$, as it is shown in [63, eq. 2.3.7]. Hence, the Frobenius norm of $\mathcal{S}^{(k)}$ can be bounded by

$$\begin{aligned}||\mathcal{S}^{(k)}||_F &\le \sqrt{r_k}||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)})||_2||\mathcal{M}^{-1}||_2^2\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t\\ &\le \sqrt{r_k}||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)})||_F||\mathcal{M}^{-1}||_2^2\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t.\end{aligned}$$

Furthermore, the inexact Newton step condition (6.1) yields

$$||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) - \mathcal{R}(\mathcal{X}^{(k)})||_F \le ||\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})||_F + ||\mathcal{R}(\mathcal{X}^{(k)})||_F \le (1+\bar{\tau})||\mathcal{R}(\mathcal{X}^{(k)})||_F,$$

such that

$$||\mathcal{S}^{(k)}||_F \le \sqrt{r_k}(1+\bar{\tau})||\mathcal{R}(\mathcal{X}^{(k)})||_F||\mathcal{M}^{-1}||_2^2\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t.$$

The denominator in (6.12) can be bounded via

$$\begin{aligned}&||\mathcal{M}\mathcal{S}^{(k)}\mathcal{B}\mathcal{B}^T\mathcal{S}^{(k)}\mathcal{M}||_F \le ||\mathcal{M}||_F^2||\mathcal{B}\mathcal{B}^T||_F||\mathcal{S}^{(k)}||_F^2\\ &\le r_k(1+\bar{\tau})^2||\mathcal{M}||_F^2||\mathcal{B}\mathcal{B}^T||_F||\mathcal{R}(\mathcal{X}^{(k)})||_F^2||\mathcal{M}^{-1}||_2^4\left(\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t\right)^2\\ &\le r_k(1+\bar{\tau})^2\varkappa_F(\mathcal{M})^2\,||\mathcal{B}\mathcal{B}^T||_F||\mathcal{R}(\mathcal{X}^{(k)})||_F^2\frac{1}{\lambda_{\mathcal{M},\min}^2}\left(\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\,\mathrm{d}t\right)^2\end{aligned} \tag{6.16}$$

with $\varkappa_F(\mathcal{M}) := ||\mathcal{M}||_F||\mathcal{M}^{-1}||_F$ and

$$\lambda_{\mathcal{M},\min} := \min\{|\lambda| : \lambda \in \Lambda(\mathcal{M})\} = \frac{1}{\max\{|\lambda| : \lambda \in \Lambda(\mathcal{M}^{-1})\}} = \frac{1}{||\mathcal{M}^{-1}||_2}.$$

Inserting (6.16) into (6.12) concludes the proof.

□

In addition, the integral in the denominator of (6.13) can be bounded by the following remark; compare [26, Rem. 6].

***Remark* 6.3** *If the matrix pencil $(\mathcal{A}^{(k)}, \mathcal{M})$ is a normal matrix pair with spectral abscissa $\sigma(\mathcal{A}^{(k)}, \mathcal{M}) = \max\{\mathrm{Re}\,(\Lambda) : \lambda \in \Lambda(\mathcal{A}^{(k)}, \mathcal{M})\}$, as introduced in Definitions 2.1 (g) and 2.1 (j), then*

$$||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2 \leq e^{t\sigma(\mathcal{A}^{(k)},\mathcal{M})}.$$

If $\sigma(\mathcal{A}^{(k)}, \mathcal{M}) < 0$, then

$$\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\, dt \leq \frac{1}{2|\sigma(\mathcal{A}^{(k)}, \mathcal{M})|}.$$

If the matrix pencil $(\mathcal{A}^{(k)}, \mathcal{M})$ is not normal, one can use the ϵ-pseudospectral abscissa $\sigma_\epsilon(\mathcal{A}^{(k)}, \mathcal{M}) = \max\{\mathrm{Re}\,(\lambda) : \lambda \in \Lambda_\epsilon(\mathcal{A}^{(k)}, \mathcal{M})\}$, as established in Definition 2.1 (h). In [126, Thm. 15.2], it is stated that if the boundary arc length $L_{\epsilon,k}$ of $\sigma_\epsilon(\mathcal{A}^{(k)}, \mathcal{M})$ with $\epsilon > 0$ is bounded, then

$$||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2 \leq \frac{L_{\epsilon,k}e^{t\sigma_\epsilon(\mathcal{A}^{(k)},\mathcal{M})}}{2\pi\epsilon}, \quad \forall t \geq 0.$$

Hence, if $\sigma_\epsilon(\mathcal{A}^{(k)}, \mathcal{M}) < 0$, then

$$\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2\, dt \leq \frac{L_{\epsilon,k}^2}{8\pi^2\epsilon^2|\sigma_\epsilon(\mathcal{A}^{(k)}, \mathcal{M})|}.$$

Corollary 6.4 *In the setting of Theorem 6.2, using Remark 6.3, the step sizes ξ_k are bounded away from zero.*

These results ensure that the step size ξ_k is bounded away from zero such that Newton's method does not stagnate. Nevertheless, the Armijo rule does not necessarily compute the step size with the maximal possible descent in the residual. An approach that finds the best residual reduction is presented next.

Exact Line Search To determine the optimal step size $\xi_k > 0$ that yields the minimal possible residual norm, one needs to find the minimum of the scalar quartic polynomial (6.10) with respect to $\xi > 0$. Benner and Byers analyzed this line search approach for the KNM applied to a GCARE like (4.16c) in [25] and called it *exact line search* *method* (ELSM). Thereby, each step in the KNM is considered to be solved exactly, as it is considered in Section 4.2 of this thesis, such that $\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) = 0$. In [25], it is shown that a local minimum $\xi_k \in (0, 2]$ exists, which preserves stability of the closed loop pencil $(\mathcal{A}^{(k+1)}; \mathcal{M})$, assuming (6.11) and a stable pencil $(\mathcal{A}^{(k)}, \mathcal{M})$. Unfortunately, these results cannot be confirmed, in general, if the Kleinman–Newton step is solved inexactly. Moreover, for $\xi_k \in (1, 2]$ the spsd property of the new iterate in (6.11) cannot be ensured anymore, as stated in Remark 6.1.

Nevertheless, the derivations in (6.8) and (6.10) extend the approach in [25] to the case of inexact solves. To ensure a spsd iterate $X^{(k+1)}$, the ELSM chooses ξ_k as

$$\xi_k = \min_{\xi_k \in (0,1]} f_{\mathcal{R}}^{(k)}(\xi_k) \tag{6.17}$$

with $f_{\mathcal{R}}^{(k)}(\xi_k)$ as defined in (6.10). Since $\mathcal{S}^{(k)}$ is a descent direction if (6.6) is fulfilled for $0 < \tau_k < 1$, a local minimum $\xi_k \in (0, 1]$ that ensures (6.3) exists. In Table 6.3 in Subsection 6.4.1, some numerical tests show the performance of this line search approach compared to the Armijo method. In general, the Armijo rule is preferable, since the step size computed by the Armijo rule is bounded away from zero, as described in Theorem 6.2. To the author's knowledge, no such result exists for the ELSM at this time.

6.2.3. Convergence Results

The convergence of the inexact KNM with line search is examined in this subsection. The results are based on the statements in [26, Sec. 3.3] which are adapted to the notation in this thesis and partly extended to the projected GCARE (4.16c).

In [59], Feitzinger et al. establish a convergence result for the inexact KNM for a standard CARE that extends the classical convergence proof in [81]. Thereby, no line search method is considered. The key ingredients of their convergence proof are certain positive semi-definiteness assumptions for the Lyapunov residual $\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})$ in (6.5). All statements are derived under the following assumption.

***Assumption* 6.5** *The dynamical system $\widehat{\Phi}(\mathcal{A}, \mathcal{B}, \mathcal{C}; \mathcal{M})$ in (4.15) is given such that $(\mathcal{A}, \mathcal{B}; \mathcal{M})$ is stabilizable and $(\mathcal{C}, \mathcal{A}; \mathcal{M})$ is detectable.*

The well-posedness of the inexact KNM is given by the following theorem with $\mathcal{M} = I_{\tilde{n}}$ in our notation.

Theorem 6.6 (cf. [59, Thm. 4.3]) *Let $\mathcal{X}^{(k)}$ be symmetric and positive semi-definite such that $\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k)}$ is stable and*

$$\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) \preceq \mathcal{C}^T\mathcal{C} \tag{6.18}$$

holds. Then

(i) the iterate $\mathcal{X}^{(k+1)} = \widetilde{X}^{(k+1)}$ of the inexact KNM is well defined, symmetric and positive semi-definite,

(ii) and the matrix $\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(k+1)}$ is stable.

As shown in 4.2.2, a special version of the low-rank ADI method from Subsection 2.4.2 is used to approximately solve the Lyapunov equation (6.7). Hence, the preliminary iterates $\widetilde{\mathcal{X}}^{(k+1)}$ are of low rank. As shown in Section 6.3, the Lyapunov residual in the k+1-st Newton and ℓ-th ADI step can be written as low-rank product

$$\mathcal{L}(\widetilde{\mathcal{X}}_\ell^{(k+1)}) = \widetilde{\mathcal{W}}_\ell^{(k+1)} \left(\widetilde{\mathcal{W}}_\ell^{(k+1)}\right)^T = \mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)} \left(\mathcal{W}^{(k)}\right)^T \left(\mathcal{F}_\ell^{(k)}\right)^T, \tag{6.19}$$

where $\mathcal{W}^{(k)} = \begin{bmatrix}\mathcal{C}^T & \mathcal{K}^{(k)}\end{bmatrix}, \widetilde{\mathcal{W}}_\ell^{(k+1)} \in \mathbb{R}^{\tilde{n}\times(n_a+n_r)}$. Furthermore, if the closed-loop matrix pencil $(\mathcal{A}^{(k)}, \mathcal{M})$ is stable and the ADI shifts $\{q_i\}_{i=1}^{\ell}$ are in the open left half-plane $\mathbb{C}^-$, then $\mathcal{F}_\ell^{(k)} = \mathcal{F}(\mathcal{A}^{(k)}, q_1, \ldots, q_\ell; \mathcal{M};)$ is an analytic matrix function.

Lemma 6.7 (cf. [26, Lem. 8]) *If $\mathcal{M}, \mathcal{N} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$ are spsd matrices with $\mathcal{M} \succeq \mathcal{N}$, i.e., $\mathcal{M} - \mathcal{N} \succeq 0$, then* $\operatorname{null}(\mathcal{M}) \subset \operatorname{null}(\mathcal{N})$ *and* $\operatorname{range}(\mathcal{N}) \subset \operatorname{range}(\mathcal{M})$.

Proof. Assume there exists $\boldsymbol{x} \in \operatorname{null}(\mathcal{M})$ with $\boldsymbol{x} \notin \operatorname{null}(\mathcal{N})$, then $\boldsymbol{x}^T\mathcal{M}\boldsymbol{x} - \boldsymbol{x}^T\mathcal{N}\boldsymbol{x} = -\boldsymbol{x}^T\mathcal{N}\boldsymbol{x} < 0$, which contradicts $\mathcal{M} \succeq \mathcal{N}$. Hence, $\operatorname{range}(\mathcal{M})^\perp = \operatorname{null}(\mathcal{M}) \subset \operatorname{null}(\mathcal{N}) = \operatorname{range}(\mathcal{N})^\perp$ and, consequently, $\operatorname{range}(\mathcal{N}) \subset \operatorname{range}(\mathcal{M})$.

□

As it is stated subsequently to [26, Lem. 8], "[t]he definition of $\mathcal{W}^{(k)}$ and application of the previous lemma give that $\mathcal{C}^T\mathcal{C} \succeq \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) = \mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)} \left(\mathcal{W}^{(k)}\right)^T \left(\mathcal{F}_\ell^{(k)}\right)^T$ implies $\operatorname{range}\left(\mathcal{F}_\ell^{(k)}\mathcal{C}^T\right) \subset \operatorname{range}\left(\mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)}\right) \subset \operatorname{range}\left(\mathcal{C}^T\right) \subset \operatorname{range}\left(\mathcal{W}^{(k)}\right)$. However, the invariance property $\operatorname{range}\left(\mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)}\right) \subset \operatorname{range}\left(\mathcal{C}^T\right)$, or even $\operatorname{range}\left(\mathcal{F}_\ell^{(k)}\mathcal{C}^T\right) \subset \operatorname{range}\left(\mathcal{C}^T\right)$, is typically not satisfied. Recall that $\mathcal{C}^T \in \mathbb{R}^{\tilde{n}\times n_a}$ while $\mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)} \in \mathbb{R}^{n\times(n_a+n_r)}$ for $k > 1$. Therefore, in general $\mathcal{C}^T\mathcal{C} \not\succeq \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})$."

In the following theorem, Feitzinger et al. prove the quadratic convergence of the inexact KNM using another semi-definiteness condition on $\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)})$.

Theorem 6.8 (cf. [59, Thm. 4.4]) *Let Assumption 6.5 be satisfied and let $\mathcal{X}^{(0)}$, spsd, be such that $\mathcal{A}^{(0)}$ is stable. Assume that (6.18) and*

$$0 \preceq \mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) \preceq (\mathcal{X}^{(k+1)} - \mathcal{X}^{(k)})\mathcal{B}\mathcal{B}^T(\mathcal{X}^{(k+1)} - \mathcal{X}^{(k)}) \tag{6.20}$$

hold for all $k \in \mathbb{N}$. Then the iterates of inexact KNM (6.7) and (6.11) with step size $\xi_k = 1$ satisfy

(i) $\lim_{k\to\infty} \mathcal{X}^{(k)} = \mathcal{X}^{(*)}$ *and* $0 \preceq \mathcal{X}^{(*)} \preceq \cdots \preceq \mathcal{X}^{(k+1)} \preceq \mathcal{X}^{(k)} \preceq \cdots \preceq \mathcal{X}^{(1)}$,

(ii) $(\mathcal{A} - \mathcal{B}\mathcal{B}^T\mathcal{X}^{(*)})$ is stable and $\mathcal{X}^{(*)}$ is the maximal solution of $\mathcal{R}(\mathcal{X}) = 0$,

(iii) $||\mathcal{X}^{(k+1)} - \mathcal{X}^{(*)}||_F \leq \kappa||\mathcal{X}^{(k)} - \mathcal{X}^{(*)}||_F^2$, $k \in \mathbb{N}$.

In the proof of [59, Thm. 4.4], the condition (6.20) is used to prove the monotonicity $0 \preceq \cdots \preceq \mathcal{X}^{(k+1)} \preceq \mathcal{X}^{(k)} \preceq \cdots \preceq \mathcal{X}^{(1)}$. As pointed out in [26, Sec. 3.3], it is interesting that the inexact KNM converges q-quadratically autonomous from the forcing parameter τ_k in (6.6). Using the results in Lemma 6.7, the relation of the main assumption (6.20) yields range $\left(\mathcal{F}_\ell^{(k)}\mathcal{W}^{(k)}\right) \subset$ range $\left((\mathcal{X}^{(k+1)} - \mathcal{X}^{(k)})\mathcal{B}\right)$, which is not satisfied, in general. "Therefore, the convergence analysis in [59] is not applicable if the low-rank ADI method, or any other low-rank solver, is used to approximately solve the Lyapunov equation." [26, p. 8]

The convergence proof for the inexact KNM in this thesis follows the statements in [26, Thm. 10], which is based on a general proof for the inexact Newton method, see, e.g., [80, Sec. 8.2]. In detail, the convergence $||\mathcal{R}(\mathcal{X}^{(k)})||_F \to 0$ is proven and the structure of the GCARE (4.16c) is used to show the convergence of the iterates $\mathcal{X}^{(k)}$. In [25, Lem. 6], it is proven that if $(\mathcal{A}, \mathcal{B}; \mathcal{M})$ is controllable and $\{\mathcal{R}(\mathcal{X}^{(k)})\}$ is bounded, then $\{\mathcal{X}^{(k)}\}$ is bounded as well. Controllability of a system implies stabilizability, see, e.g., [91, Thm. A.1], and is, therefore, stronger than Assumption 6.5. Gou and Laub established a proof in [66] that shows $\{X^{(k)}\}$ is bounded under the assumption of a stabilizable system, bounded residuals $\{\mathcal{R}(\mathcal{X}^{(k)})\}$, and stable closed-loop pencils $(\mathcal{A}^{(k)}, \mathcal{M})$.

As it is stated in [26] prior to [26, Lem. 10], "[t]he papers [81] on exact Kleinman–Newton, [25] on Kleinman–Newton with line search, and [59] on inexact Kleinman–Newton contain proofs that the [pencils $(\mathcal{A}^{(k)}, \mathcal{M})$] corresponding to the iterates $\mathcal{X}^{(k)}$ are stable, provided that [$(\mathcal{A}^{(0)}, \mathcal{M})$] is stable. This implies the unique solution of the Lyapunov equation (4.21) and, therefore, the well-posedness of the respective method. Since the definiteness assumption in [59, Thm. 4.3] typically does not hold in the low-rank case, there is no result yet on the well-posedness of the inexact Kleinman–Newton method and we have to assume existence of $\widetilde{\mathcal{X}}^{(k+1)}$ such that (6.7) and (6.6) are satisfied."

Theorem 6.9 (cf. [26, Thm. 10]) *Let Assumption 6.5 be satisfied and assume that for all k, there exists a symmetric positive semi-definite $\widetilde{\mathcal{X}}^{(k+1)}$ such that (6.7) and (6.6) hold.*

(i) If the step sizes are bounded away from zero, i.e., $\xi_k \geq \xi_{\min} > 0$ for all k, then $||\mathcal{R}(\mathcal{X}^{(k)})||_F \to 0$.

(ii) If, in addition to 6.9 (i), the pencils $(\mathcal{A}^{(k)}, \mathcal{M})$ are stable for $k \geq k_0$, and $\mathcal{X}^{(k)} \succeq 0$ for all $k \geq k_0$, then $\mathcal{X}^{(k)} \to \mathcal{X}^{(*)}$, where $\mathcal{X}^{(*)} \succeq 0$ is the unique stabilizing solution of the GCARE (4.16c).

Proof. (i) The first part follows from a standard line search argument. Using a telescoping series, the sufficient decrease condition (6.3) implies that for any integer $\widetilde{k}$,

$$\begin{aligned}||\mathcal{R}(\mathcal{X}^{(0)})||_F &\geq ||\mathcal{R}(\mathcal{X}^{(0)})||_F - ||\mathcal{R}(\mathcal{X}^{(\widetilde{k}+1)})||_F \\ &= \sum_{k=0}^{\widetilde{k}} \left(||\mathcal{R}(\mathcal{X}^{(k)})||_F - ||\mathcal{R}(\mathcal{X}^{(k+1)})||_F\right) \geq \sum_{k=0}^{\widetilde{k}} \xi_k \beta ||\mathcal{R}(\mathcal{X}^{(k)})||_F \geq 0.\end{aligned}$$

Taking the limit $\widetilde{k} \to \infty$ and using $\xi_k \geq \xi_{\min} > 0$ implies $||\mathcal{R}(\mathcal{X}^{(k)})||_F \to 0$.

(ii) If the pencils $(\mathcal{A}^{(k)}, \mathcal{M})$ are stable for $k \geq k_0$ and $\{\mathcal{R}(\mathcal{X}^{(k)})\}$ is bounded, [66, Lem. 2.3] guarantees that $\{\mathcal{X}^{(k)}\}$ is bounded. Hence, $\{\mathcal{X}^{(k)}\}$ has a converging subsequence. For any converging subsequence it holds that

$$\lim_{j\to\infty} \mathcal{X}^{(k_j)} \succeq 0 \quad \text{and} \quad \lim_{j\to\infty} ||\mathcal{R}(\mathcal{X}^{(k_j)})||_F = ||\mathcal{R}(\lim_{j\to\infty} \mathcal{X}^{(k_j)})||_F = 0.$$

The spsd solution of the GCARE (4.16c) is unique and stabilizing by Theorem 2.26. Hence, every converging subsequence of $\{\mathcal{X}^{(k)}\}$ has the same limit $\mathcal{X}^{(*)}$. Therefore, the entire sequence converges.

□

***Remark* 6.10 (cf. [26, Rem. 10])**

1. *If the step size $\xi_k \in (0, 1]$, then $\mathcal{X}^{(k)} \succeq 0$ for all k, see Remark 6.1.*

2. *Lower bounds for the step size computed by the Armijo rule are established in Theorem 6.2. In particular if*

$$\left\{ ||\mathcal{R}(\mathcal{X}^{(k)})||_F \int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2 \, dt \right\}_{k\in\mathbb{N}} \tag{6.21}$$

is bounded, the step size computed by the Armijo rule is bounded away from zero. Since for all k it holds that $||\mathcal{R}(\mathcal{X}^{(k)})||_F < ||\mathcal{R}(\mathcal{X}^{(0)})||_F$, the sequence (6.21) is bounded if $\int_0^\infty ||e^{\mathcal{A}^{(k)}\mathcal{M}^{-1}t}||_2^2 \, dt$ is bounded, which is a condition on the uniform stability of the pencils $(\mathcal{A}^{(k)}, \mathcal{M})$, $k \in \mathbb{N}$.

As established in, e.g, [80, Sec. 8.2], the convergence rate of the inexact KNM is determined by the forcing parameter τ_k in (6.6). In detail, for $\tau_k \to 0$ the convergence rate is superlinear and for $\tau_k = \mathcal{O}(||\mathcal{R}(\mathcal{X}^{(k)})||_F)$ the convergence is quadratic, both provided that the assumptions in Theorem 6.9 hold.

The inexact Newton scheme and the line search methods explicitly use the Lyapunov and Riccati residuals, as well as their norms. As it is shown above, the generalization of the methods in [26] to deal with a mass matrix $\mathcal{M} \neq I_{\widetilde{n}}$ is straightforward. To integrate this entire procedure into the scheme of KN-ADI for index-2 DAE systems from Chapter 4, the projected Lyapunov and Riccati residuals need to be considered. As pointed out in Subsection 4.3.2, the computation of these projected, dense, and large-scale matrices needs to be avoided. Hence, the performance of the entire method depends on an efficient computation of the residual norm, as well as an efficient method to store these dense matrices. The latter problem led Benner and Byers to restrict themselves in [25] to small dense test examples. The next section introduces techniques that circumvent this drawback and enable the inexact KNM with line search to be used for large-scale index-2 DAE systems. Most of these techniques are extensions of the methods published in [26, Sec. 4f] for a standard non-projected CARE to the index-2 DAE case.

6.3. Low-Rank Residual KN-ADI Method

In this section, various algorithmic improvements from the last years are combined that have not been available at the beginning of the author's PhD. The key ingredient are the improvements on the low-rank ADI iteration, which have been worked out in [27–29]. In the next subsection, these improvements are reviewed and adapted to the KN-ADI for index-2 DAE systems from Sections 4.2 & 4.3.

6.3.1. Improved Low-Rank ADI Method

The first significant improvement is published in [28, Sec. 4.4.1] and introduces a low-rank residual formulation for the ADI. To adapt this method to the index-2 DAE structure, one considers the GCALE (4.21) for a fixed Newton step $k+1$. Multiplying (4.21) from the left and the right by $\mathcal{M}^{-1}$ yields the CALE

$$F\mathcal{X} + \mathcal{X}F^T = -\widehat{W}\widehat{W}^T \tag{6.22}$$

with $F := \mathcal{M}^{-1}\left(\mathcal{A}^{(k)}\right)^T$, $\widehat{W} = \mathcal{M}^{-1}\mathcal{W}^{(k)}$, and $\mathcal{X} = \mathcal{X}^{(k+1)}$, which is solved by the ADI method from Subsection 2.4.2 with the mass matrix $M = I_{\tilde{n}}$. Hence, the recursion (2.37a) can be written as

$$\begin{aligned}\widehat{\mathcal{V}}_\ell &= (I_{\tilde{n}} - (q_\ell + \overline{q}_{\ell-1})(F + q_\ell I_{\tilde{n}})^{-1}(\widehat{\mathcal{V}}_{\ell-1})) = (F - \overline{q_{\ell-1}}\, I_{\tilde{n}})(F + q_\ell I_{\tilde{n}})^{-1}\widehat{\mathcal{V}}_{\ell-1}\\ &= \left(\prod_{j=2}^{\ell}(F - \overline{q_{j-1}}\, I_{\tilde{n}})(F + q_j I_{\tilde{n}})^{-1}\right)(F + q_1 I_{\tilde{n}})^{-1}\widehat{W} \in \mathbb{C}^{\tilde{n}\times(n_a+n_r)},\end{aligned}$$

such that the low-rank solution factor $\widehat{\mathcal{Z}}_\ell \in \mathbb{C}^{\tilde{n}\times\ell(n_r+n_a)}$ can be defined via (2.37b). Hence, $\mathcal{X} = \widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H$ solves (6.22). Notice that the Newton index $k+1$ is omitted at the solution factor $\widehat{\mathcal{Z}}_\ell$. As introduced in [28, Sec. 4.2], the involved shifted system matrices $(F - q_1 I_{\tilde{n}})$ and $(F - q_2 I_{\tilde{n}})^{-1}$ commute for all q_1, $q_2 \in \mathbb{C}\backslash\Lambda(F)$, such that $\widehat{\mathcal{V}}_\ell$ can be regrouped as

$$\widehat{\mathcal{V}}_\ell = (F + q_\ell I_{\tilde{n}})^{-1}\underbrace{\left(\prod_{j=1}^{\ell-1}(F - \overline{q_{j-1}}\, I_{\tilde{n}})(F + q_j I_{\tilde{n}})^{-1}\right)\widehat{W}}_{=:\widehat{W}_{\ell-1}} = (F + q_\ell I_{\tilde{n}})^{-1}\widehat{W}_{\ell-1}, \tag{6.23a}$$

where $\widehat{W}_0 = \widehat{W}$ and

$$\begin{aligned}\widehat{W}_\ell &= (F - \overline{q_\ell}\, I_{\tilde{n}})\widehat{\mathcal{V}}_\ell = (F - \overline{q_\ell}\, I_{\tilde{n}})(F + q_\ell I_{\tilde{n}})^{-1}\widehat{W}_{\ell-1}\\ &= \left(I_{\tilde{n}} - 2\,\mathrm{Re}\,(q_\ell)\,(F + q_\ell I_{\tilde{n}})^{-1}\right)\widehat{W}_{\ell-1}\\ &= \widehat{W}_{\ell-1} - 2\,\mathrm{Re}\,(q_\ell)\,\widehat{\mathcal{V}}_\ell \in \mathbb{C}^{\tilde{n}\times(n_a+n_r)}.\end{aligned} \tag{6.23b}$$

Hence, the definition of $\widehat{W}_\ell$ yields

$$\widehat{W}_\ell = \underbrace{\left(\prod_{j=1}^{\ell-1}(F - \overline{q_{j-1}}\, I_{\widetilde{n}})(F + q_j I_{\widetilde{n}})^{-1}\right)}_{=:\widehat{\mathcal{F}}_\ell} \widehat{W} = \widehat{\mathcal{F}}_\ell \widehat{W} \tag{6.23c}$$

with the matrix function $\widehat{\mathcal{F}}_\ell = \widehat{\mathcal{F}}(F, q_1, \ldots, q_\ell)$. In [28, Sec. 4.4.1], Benner et al. show that the iteration defined by (6.23) is mathematically equivalent to the original ADI method in [31, 88], as introduced in Subsection 2.4.2. Furthermore, the residual of (6.22) after the ℓ-th ADI step can be written as low-rank factorization

$$\widehat{\mathcal{L}}(\widehat{\mathcal{Z}}_\ell \widehat{\mathcal{Z}}_\ell^H) = F\widehat{\mathcal{Z}}_\ell \widehat{\mathcal{Z}}_\ell^H + \widehat{\mathcal{Z}}_\ell \widehat{\mathcal{Z}}_\ell^H F^T + \widehat{W}\widehat{W}^T = \widehat{\mathcal{F}}_\ell \widehat{W}\widehat{W}^T \widehat{\mathcal{F}}_\ell^H = \widehat{W}_\ell \widehat{W}_\ell^H \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}}. \tag{6.24}$$

As mentioned in Subsection 2.3.4, the transformation that leads to (6.22) is never performed explicitly. Hence, the equations (6.23) are reformulated using the original matrices. In detail, (6.23a) yields

$$\widehat{\mathcal{V}}_\ell = \left(\mathcal{M}^{-1}\left(\mathcal{A}^{(k)}\right)^T + q_\ell I_{\widetilde{n}}\right)^{-1} \widehat{W}_{\ell-1} = \left(\left(\mathcal{A}^{(k)}\right)^T + q_\ell \mathcal{M}\right)^{-1} \underbrace{\mathcal{M}\widehat{W}_{\ell-1}}_{=:\widehat{\mathcal{W}}_{\ell-1}} \tag{6.25a}$$

with $\widehat{\mathcal{W}}_0 = \mathcal{M}\mathcal{M}^{-1}\mathcal{W}^{(k)} = \mathcal{W}^{(k)}$, (6.23b) yields

$$\begin{aligned} & \widehat{W}_\ell = \widehat{W}_{\ell-1} - 2\,\mathrm{Re}\,(q_\ell)\,\widehat{\mathcal{V}}_\ell = \mathcal{M}^{-1}(\mathcal{M}\widehat{W}_{\ell-1} - 2\,\mathrm{Re}\,(q_\ell)\,\mathcal{M}\widehat{\mathcal{V}}_\ell) \\ \Leftrightarrow\quad & \widehat{\mathcal{W}}_\ell = \widehat{\mathcal{W}}_{\ell-1} - 2\,\mathrm{Re}\,(q_\ell)\,\mathcal{M}\widehat{\mathcal{V}}_\ell, \end{aligned} \tag{6.25b}$$

and (6.23c) leads to

$$\begin{aligned} \widehat{W}_\ell &= \left(\prod_{j=1}^{\ell-1}\left(\mathcal{M}^{-1}\left(\mathcal{A}^{(k)}\right)^T - \overline{q_{j-1}}\, I_{\widetilde{n}}\right)\left(\mathcal{M}^{-1}\left(\mathcal{A}^{(k)}\right)^T + q_j I_{\widetilde{n}}\right)^{-1}\right)\mathcal{M}^{-1}\mathcal{W}^{(k)} \\ &= \mathcal{M}^{-1}\left(\prod_{j=1}^{\ell-1}\left(\left(\mathcal{A}^{(k)}\right)^T - \overline{q_{j-1}}\,\mathcal{M}\right)\left(\left(\mathcal{A}^{(k)}\right)^T + q_j\mathcal{M}\right)^{-1}\right)\mathcal{M}\mathcal{M}^{-1}\mathcal{W}^{(k)} \\ \Leftrightarrow\quad \widehat{\mathcal{W}}_\ell &= \underbrace{\left(\prod_{j=1}^{\ell-1}\left(\left(\mathcal{A}^{(k)}\right)^T - \overline{q_{j-1}}\,\mathcal{M}\right)\left(\left(\mathcal{A}^{(k)}\right)^T + q_j\mathcal{M}\right)^{-1}\right)}_{=:\widetilde{\mathcal{F}}_\ell^{(k)}}\mathcal{W}^{(k)}. \end{aligned} \tag{6.25c}$$

Multiplying (6.24) form the left and the right by $\mathcal{M}$ yields

$$\begin{aligned} \mathcal{M}\widehat{\mathcal{L}}(\widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H)\mathcal{M} &= \mathcal{M}\mathcal{M}^{-1}\left(\mathcal{A}^{(k)}\right)^T\widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H\mathcal{M} + \mathcal{M}\widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H\mathcal{A}^{(k)}\mathcal{M}^{-1}\mathcal{M} + \mathcal{M}\widehat{W}\widehat{W}^T\mathcal{M} \\ &= \left(\mathcal{A}^{(k)}\right)^T\widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H\mathcal{M} + \mathcal{M}\widehat{\mathcal{Z}}_\ell\widehat{\mathcal{Z}}_\ell^H\mathcal{A}^{(k)} + \mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T \\ &= \widehat{\mathcal{W}}_\ell\widehat{\mathcal{W}}_\ell^H = \widetilde{\mathcal{F}}_\ell^{(k)}\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T\left(\widetilde{\mathcal{F}}_\ell^{(k)}\right)^H = \mathcal{L}(\mathcal{X}_\ell^{(k+1)}) \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}} \end{aligned} \tag{6.25d}$$

with $\widetilde{\mathcal{F}}_\ell^{(k)} = \widetilde{\mathcal{F}}(\mathcal{A}^{(k)}, q_1, \ldots, q_\ell;\, \mathcal{M}) \in \mathbb{C}^{\widetilde{n}\times\widetilde{n}}$. The ADI method defined by (6.25) shows that the Lyapunov residual can be written as low-rank decomposition, where the actual residual factor defines the right-hand side in the next ADI step. A remaining issue refers to the complex components $\widehat{\mathcal{V}}_\ell$, $\widehat{\mathcal{W}}_\ell \in \mathbb{C}^{\widetilde{n}\times(n_a+n_r)}$, $\widehat{\mathcal{Z}}_\ell \in \mathbb{C}^{\widetilde{n}\times\ell(n_a+n_r)}$, and $\widetilde{\mathcal{F}}_\ell^{(k)} \in \mathbb{C}^{\widetilde{n}\times\widetilde{n}}$, which yield considerable higher storage usage. This happens whenever complex ADI shifts are used. For nonsymmetric pencils $(\mathcal{A}^{(k)}, \mathcal{M})$, complex ADI shifts usually improve the ADI convergence significantly. This naturally leads to complex factors, although the solution $\mathcal{X}_\ell^{(k+1)} = \widehat{\mathcal{Z}}_\ell \widehat{\mathcal{Z}}_\ell^H \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}}$ is supposed to be real.

In [27, 29], Benner et al. introduced the second significant improvement that deals with this issue. It is well established that to ensure a real-valued solution $\mathcal{X}_\ell^{(k+1)}$, the ADI shifts are obliged to occur either as real numbers $q_\ell \in \mathbb{R}^-$ or as complex conjugate pairs $q_\ell \in \mathbb{C}^-$, $q_{\ell+1} = \overline{q_\ell} \in \mathbb{C}^-$. Exploiting this fact leads to an ADI scheme with real-valued factors $\mathcal{W}_\ell \in \mathbb{R}^{\widetilde{n}\times(n_a+n_r)}$ and $\mathcal{Z}_\ell \in \mathbb{R}^{\widetilde{n}\times\ell(n_a+n_r)}$, such that $\mathcal{X}_\ell^{(k+1)} = \mathcal{Z}_\ell \mathcal{Z}_\ell^T$ is the approximate solution of (4.21) in the ℓ-th ADI step. Furthermore, the Lyapunov residual is defined by the real-valued matrix function $\mathcal{F}_\ell^{(k)} = \mathcal{F}(\mathcal{A}^{(k)}, q_1, \ldots, q_\ell; \mathcal{M}) \in \mathbb{R}^{\widetilde{n}\times\widetilde{n}}$ as

$$\mathcal{L}(\mathcal{X}_\ell^{(k+1)}) = \mathcal{F}_\ell^{(k)} \mathcal{W}^{(k)} \left(\mathcal{W}^{(k)}\right)^T \left(\mathcal{F}_\ell^{(k)}\right)^T = \mathcal{W}_\ell^{(k+1)} \left(\mathcal{W}_\ell^{(k+1)}\right)^T. \tag{6.26}$$

As stated in [26, Sec. 4], the analytic matrix function $\mathcal{F}_\ell^{(k)} \equiv \widetilde{\mathcal{F}}_\ell^{(k)}$ if and only if the ADI shifts are closed under complex conjugation, i.e., $\{q_i\}_{i=1}^{\ell} = \overline{\{q_i\}_{i=1}^{\ell}}$. Using (6.26), eigenvalue-based norms of the Lyapunov residual $\mathcal{L}(\mathcal{X}_\ell^{(k+1)})$ can be computed highly efficiently by (2.7), i.e.,

$$\left\|\mathcal{L}(\mathcal{X}_\ell^{(k+1)})\right\|_{F/2} = \left\|\left(\mathcal{W}_\ell^{(k+1)}\right)^T \mathcal{W}_\ell^{(k+1)}\right\|_{F/2}. \tag{6.27}$$

Summarizing, the improved low-rank ADI method with low-rank and real-valued residual and solution factors to solve the GCALE (4.21) adapts [27, Alg. 1]. The entire procedure is depicted in Algorithm 5. For more details regarding this real-valued ADI iteration, the interested reader is referred to [27, 29, 84].

To use this novel approach for index-2 DAE systems, one needs to consider certain modifications to combine it with the methods introduced in Subsection 4.2.2 that avoid explicit projections.

6.3.2. Low-Rank Residual ADI for Index-2 DAE systems

The improved ADI method in Algorithm 5 is derived for the projected matrices in (4.15). As stated in detail in Remark 4.2, this explicit projection needs to be avoided. By adapting the definition of the right-hand side in the k+1-st Newton step in (4.26) to

$$\mathcal{Y}_{\ell-1} = \mathcal{W}_{\ell-1} = \widehat{\Theta}_r^T W_{\ell-1}, \quad \ell \geq 1, \quad W_0 = W^{(k)}, \tag{6.28}$$

the entire procedure in Algorithm 5 can be adapted to the ideas in Subsection 4.2.2 straightforwardly without the use of any explicitly performed projection. However, to

Algorithm 5 Generalized real-valued low-rank residual ADI method

```
Input: $\mathcal{A}^{(k)}, \mathcal{W}^{(k)}, tol_{\text{ADI}}$, shifts $\{q_i\}_{i=1}^{\ell} = \overline{\{q_i\}_{i=1}^{\ell}} \in \mathbb{C}^-$.
Output: $\mathcal{Z}_\ell$ such that $\mathcal{Z}\mathcal{Z}^T \approx \mathcal{X}^{(k+1)}$ solves equation (4.21).
 1: Set $\ell = 1$, $\mathcal{Z} = [\,]$, $\mathcal{W}_0 = \mathcal{W}^{(k)}$.
 2: while $||\mathcal{W}_{\ell-1}^T \mathcal{W}_{\ell-1}||_F > tol_{\text{ADI}} ||\mathcal{W}_0^T \mathcal{W}_0||_F$ do
 3:     $\mathcal{V}_\ell = \left( \left(\mathcal{A}^{(k)}\right)^T + q_\ell \mathcal{M} \right)^{-1} \mathcal{W}_{\ell-1}$
 4:     if $\operatorname{Im}(q_\ell) = 0$ then
 5:         $\mathcal{W}_\ell = \mathcal{W}_{\ell-1} - 2 q_\ell \mathcal{M} \mathcal{V}_\ell$
 6:         $\widetilde{\mathcal{V}}_\ell = \sqrt{-2q_\ell}\, \mathcal{V}_\ell$
 7:     else
 8:         $\gamma_\ell = 2\sqrt{-\operatorname{Re}(q_\ell)}, \quad \delta_\ell = \operatorname{Re}(q_\ell) / \operatorname{Im}(q_\ell)$
 9:         $\mathcal{W}_{\ell+1} = \mathcal{W}_{\ell-1} + \gamma_\ell^2 \mathcal{M} \left(\operatorname{Re}(\mathcal{V}_\ell) + \delta_\ell \operatorname{Im}(\mathcal{V}_\ell)\right)$
10:         $\widetilde{\mathcal{V}}_{\ell+1} = \left[\gamma_\ell \left(\operatorname{Re}(\mathcal{V}_\ell) + \delta_\ell \operatorname{Im}(\mathcal{V}_\ell)\right) \quad \gamma_\ell \sqrt{(\delta_\ell^2 + 1)} \operatorname{Im}(\mathcal{V}_\ell)\right]$
11:         $\ell = \ell + 1$
12:     end if
13:     $\mathcal{Z} = \left[\mathcal{Z} \quad \widetilde{\mathcal{V}}_\ell\right]$
14:     $\ell = \ell + 1$
15: end while
```

compute the projected Lyapunov residual in (4.36a), (6.26) needs to be multiplied by $\widehat{\Theta}_l$ from the left and $\widehat{\Theta}_l^T$ from the right, such that the computation of

$$\begin{aligned} &\widehat{\Theta}_l \mathcal{L}(\mathcal{X}_\ell^{(k+1)}) \widehat{\Theta}_l^T = \widehat{\Theta}_l \mathcal{W}_\ell \mathcal{W}_\ell^T \widehat{\Theta}_l^T = \widehat{\Theta}_l \widehat{\Theta}_r^T W_\ell W_\ell^T \widehat{\Theta}_r \widehat{\Theta}_l^T \\ \Rightarrow \quad &\mathcal{L}(X_\ell^{(k+1)}) = \widehat{\Pi} W_\ell W_\ell^T \widehat{\Pi}^T =: \overline{W}_\ell \overline{W}_\ell^T \end{aligned} \tag{6.29}$$

would involve the explicitly projected low-rank residual factor $\overline{W}_\ell := \widehat{\Pi} W_\ell$. Multiplying line 5 in Algorithm 5 with $\widehat{\Theta}_l$ from the left yields

$$\begin{aligned} &\widehat{\Theta}_l \widehat{\Theta}_r^T W_\ell = \widehat{\Theta}_l \widehat{\Theta}_r^T W_{\ell-1} - 2 q_\ell \widehat{\Theta}_l \widehat{\Theta}_r^T M \widehat{\Theta}_r \mathcal{V}_\ell \\ \Leftrightarrow \quad &\widehat{\Pi} W_\ell = \widehat{\Pi} W_{\ell-1} - 2 q_\ell \widehat{\Pi} M V_\ell = \widehat{\Pi} W_{\ell-1} - 2 q_\ell M \widehat{\Pi}^T V_\ell. \end{aligned}$$

Using (4.28), i.e., $\widehat{\Pi}^T V_\ell = V_\ell$, the projected low-rank residual factor can be accumulated via

$$\overline{W}_\ell = \overline{W}_{\ell-1} - 2 q_\ell M V_\ell \tag{6.30}$$

without using any explicit projections. Hence, only the initial right hand side $W^{(k)}$ needs to be projected to define

$$\overline{W}_0 := \widehat{\Pi} W^{(k)}. \tag{6.31}$$

This one projection at the beginning of the ADI method is computed by (4.9) using (4.11) with the original sparse matrices. Notice that this projection is less expensive than a single ADI step and does not considerably increase the overall computation costs. Moreover, the right-hand side $W_{\ell-1}$ in (6.28) can be directly replaced by $\overline{W}_{\ell-1}$, since the right-hand side enters the projected equation (4.29) as $\widehat{\varPi} Y_{\ell-1}$. Due to the construction of $\overline{W}_\ell$, the right-hand side remains unchanged as

$$\widehat{\varPi} Y_{\ell-1} = \widehat{\varPi}\overline{W}_{\ell-1} = \widehat{\varPi}\widehat{\varPi} W_{\ell-1} = \widehat{\varPi} W_{\ell-1}.$$

To incorporate this improved ADI method into Algorithm 4, some remaining issues, such as the storage of the Newton step $\mathcal{S}^{(k)}$ and the projected Riccati residual, needs to be addressed. This is done in the next subsection, expanding the statements in [26, Sec. 5.2].

6.3.3. Low-Rank Riccati Residual for Index-2 DAE systems

The Newton step $\mathcal{S}^{(k)}$ is only used in the computation of the step size ξ_k, since the inexact Kleinman–Newton step (6.7) directly iterates over the preliminary solution $\widetilde{\mathcal{X}}^{(k+1)}$. Furthermore, $\mathcal{S}^{(k)}$ always occurs in products $\mathcal{M}\mathcal{S}^{(k)}\mathcal{B} \in \mathbb{R}^{\widetilde{n}\times n_r}$. Using (6.2), (6.4), and the definition of the feedback matrix in (4.16b), this product can be written as

$$\mathcal{M}\mathcal{S}^{(k)}\mathcal{B} = \begin{cases} \mathcal{M}\widetilde{\mathcal{X}}^{(k+1)}\mathcal{B} - \mathcal{M}\mathcal{X}^{(k)}\mathcal{B} =: \widetilde{\mathcal{K}}^{(k+1)} - \mathcal{K}^{(k)} =: \Delta\widetilde{\mathcal{K}}^{(k+1)}, & \xi_k \neq 1, \\ \mathcal{M}\mathcal{X}^{(k+1)}\mathcal{B} - \mathcal{M}\mathcal{X}^{(k)}\mathcal{B} =: \mathcal{K}^{(k+1)} - \mathcal{K}^{(k)} =: \Delta\mathcal{K}^{(k+1)}, & \xi_k = 1, \end{cases} \tag{6.32}$$

which characterizes the feedback change corresponding to the preliminary or definite new iterate $\widetilde{\mathcal{X}}^{(k+1)}$ or $\mathcal{X}^{(k+1)}$. Hence, the dense Newton step $\mathcal{S}^{(k)}$ is never formed explicitly.

To define a low-rank Riccati residual for the new iterate $\mathcal{X}^{(k+1)}$, one needs to distinguish between the case $\xi_k = 1$ and $\xi_k \in (0,1)$. For $\xi_k = 1$, the feedback change $\Delta\mathcal{K}^{(k+1)}$ and the Lyapunov residual (6.26) employed in (6.8) yield

$$\mathcal{R}(\mathcal{X}^{(k+1)}) = \mathcal{W}^{(k+1)}\left(\mathcal{W}^{(k+1)}\right)^T - \Delta\mathcal{K}^{(k+1)}\left(\Delta\mathcal{K}^{(k+1)}\right)^T =: \mathcal{U}^{(k+1)}\mathcal{D}\left(\mathcal{U}^{(k+1)}\right)^T, \tag{6.33a}$$

which is an indefinite low-rank product as stated in Definition 2.4, whose spectral or Frobenius norm can be computed efficiently using (2.8). The factors are defined independently of ξ_k by

$$\mathcal{U}^{(k+1)} = \begin{bmatrix} \mathcal{W}^{(k+1)} & \Delta\mathcal{K}^{(k+1)} \end{bmatrix}, \quad \mathcal{D} = \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}, \tag{6.33b}$$

where I is of appropriate size.

To compute the Riccati residual dependent on $\xi_k \in (0,1)$, one needs to consider the preliminary feedback change $\Delta\widetilde{\mathcal{K}}^{(k+1)}$ in (6.8), as well as the Lyapunov residual for the preliminary solution defined via (6.26) as $\mathcal{L}(\widetilde{\mathcal{X}}^{(k+1)}) = \widetilde{\mathcal{W}}^{(k+1)}\left(\widetilde{\mathcal{W}}^{(k+1)}\right)^T$. Furthermore, the previous residual $\mathcal{R}(\mathcal{X}^{(k)})$ is said to be of the form (6.33), such that

$$\begin{aligned}
\mathcal{R}(\mathcal{X}^{(k+1)}) &= \mathcal{R}(\mathcal{X}^{(k)} + \xi_k \mathcal{S}^{(k)}) \\
&= (1-\xi_k)\mathcal{U}^{(k)}\mathcal{D}\left(\mathcal{U}^{(k)}\right)^T + \xi_k \widetilde{\mathcal{W}}^{(k+1)}\left(\widetilde{\mathcal{W}}^{(k+1)}\right)^T - \xi_k^2 \Delta\widetilde{\mathcal{K}}^{(k+1)}\left(\Delta\widetilde{\mathcal{K}}^{(k+1)}\right)^T \\
&= (1-\xi_k)\left(\mathcal{W}^{(k)}\left(\mathcal{W}^{(k)}\right)^T - \Delta\mathcal{K}^{(k)}\left(\Delta\mathcal{K}^{(k)}\right)^T\right) + \xi_k \widetilde{\mathcal{W}}^{(k+1)}\left(\widetilde{\mathcal{W}}^{(k+1)}\right)^T \\
&\quad - \xi_k^2 \Delta\widetilde{\mathcal{K}}^{(k+1)}\left(\Delta\widetilde{\mathcal{K}}^{(k+1)}\right)^T \\
&= \left[\left[\sqrt{(1-\xi_k)}\,\mathcal{W}^{(k)}\ \ \sqrt{\xi_k}\,\widetilde{\mathcal{W}}^{(k+1)}\right]\ \left[\sqrt{(1-\xi_k)}\,\Delta\mathcal{K}^{(k)}\ \ \xi_k\Delta\widetilde{\mathcal{K}}^{(k+1)}\right]\right] \\
&\quad \times \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} \\
&\quad \times \left[\left[\sqrt{(1-\xi_k)}\,\mathcal{W}^{(k)}\ \ \sqrt{\xi_k}\,\widetilde{\mathcal{W}}^{(k+1)}\right]\ \left[\sqrt{(1-\xi_k)}\,\Delta\mathcal{K}^{(k)}\ \ \xi_k\Delta\widetilde{\mathcal{K}}^{(k+1)}\right]\right]^T,
\end{aligned} \tag{6.34}$$

which simplifies to (6.33) for $\xi_k = 1$. By defining the new block matrices as

$$\begin{aligned}
\mathcal{W}^{(k+1)} &:= \left[\sqrt{(1-\xi_k)}\,\mathcal{W}^{(k)} \quad \sqrt{\xi_k}\,\widetilde{\mathcal{W}}^{(k+1)}\right] \in \mathbb{R}^{\tilde{n}\times(s+1)(n_a+n_r)}, \\
\Delta\mathcal{K}^{(k+1)} &:= \left[\sqrt{(1-\xi_k)}\,\Delta\mathcal{K}^{(k)} \quad \xi_k\Delta\widetilde{\mathcal{K}}^{(k+1)}\right] \in \mathbb{R}^{\tilde{n}\times(s+1)n_r},
\end{aligned} \tag{6.35}$$

the residual (6.34) is again of the form (6.33). Thereby, the size of the residual blocks depends on $s \geq 0$, which is the largest number, such that all step sizes between ξ_{k-1} and ξ_{k-s} are smaller than one, but $\xi_{k-s-1} = 1$. In other words, the number of iterations on a stretch immediately before the current step, where line search is performed, i.e., $\xi_k \in (0,1)$.

Similar to the projected Lyapunov residual in (6.29), the Riccati residual (6.33) needs to be multiplied by $\widehat{\Theta}_l$ from the left and $\widehat{\Theta}_l^T$ from the right to result in the projected Riccati residual (4.36b), i.e,

$$\begin{aligned}
\widehat{\Theta}_l\mathcal{R}(\mathcal{X}^{(k+1)})\widehat{\Theta}_l^T &= \widehat{\Theta}_l\mathcal{W}^{(k+1)}\left(\mathcal{W}^{(k+1)}\right)^T\widehat{\Theta}_l^T - \widehat{\Theta}_l\Delta\mathcal{K}^{(k+1)}\left(\Delta\mathcal{K}^{(k+1)}\right)^T\widehat{\Theta}_l^T \\
\Rightarrow \qquad \mathcal{R}(X^{(k+1)}) &= \widehat{\Pi}W^{(k+1)}\left(W^{(k+1)}\right)^T\widehat{\Pi}^T - \widehat{\Pi}\Delta K^{(k+1)}\left(\Delta K^{(k+1)}\right)^T\widehat{\Pi}^T \\
&= \overline{W}^{(k+1)}\left(\overline{W}^{(k+1)}\right)^T - \Delta K^{(k+1)}\left(\Delta K^{(k+1)}\right)^T \\
&=: U^{(k+1)}\mathcal{D}\left(U^{(k+1)}\right)^T
\end{aligned} \tag{6.36}$$

with $U^{(k+1)} = \left[\overline{W}^{(k+1)} \quad \Delta K^{(k+1)}\right]$ and $\Delta K^{(k)}$ being invariant regarding multiplication with $\widehat{\Pi}$ from the left. This invariance follows from (4.23) and

$$\widehat{\Pi}\Delta K^{(k+1)} = \widehat{\Pi}M(X^{(k+1)} - X^{(k)})B = M\widehat{\Pi}^T(X^{(k+1)} - X^{(k)})B = \Delta K^{(k+1)}. \tag{6.37}$$

Hence, the Riccati residual can be computed without any additional explicit projection since $\Delta K^{(k+1)}$ is $\widehat{\Pi}$ invariant. Furthermore, $\mathcal{L}(X^{(k+1)}) = \overline{W}^{(k+1)} \left(\overline{W}^{(k+1)}\right)^T$ is the projected Lyapunov residual for the solution $X^{(k+1)}$. The extension of this formulation for the case of $\xi_k \in (0, 1)$ is straightforward, using the projected Lyapunov residual for the preliminary solution defined by

$$\mathcal{L}(\widetilde{X}^{(k+1)}) = \widetilde{W}^{(k+1)} \left(\widetilde{W}^{(k+1)}\right)^T. \tag{6.38}$$

Before stating the final algorithm, some additional details regarding its implementation are discussed in the next subsection.

6.3.4. Implementation Details

The last two subsections discussed the incorporation of the improved low-rank ADI method (Subsection 6.3.1) into the inexact KNM with line search for index-2 DAE systems (Section 6.2). The following aspects are crucial for an efficient implementation.

Low-Rank Residual Accumulation: The computation of the Riccati residual depends on the actual Lyapunov residual and the feedback change. To monitor the Riccati residual during the ADI iteration, both quantities need to be accumulated efficiently. This can be done for the Lyapunov residual factor via (6.30). To accumulate the feedback change, the feedback accumulation in (4.35) is adapted to

$$\begin{aligned} \Delta\widetilde{K}_\ell^{(k+1)} &= \widetilde{K}_\ell^{(k+1)} - K^{(k)} = \widetilde{K}_{\ell-1}^{(k+1)} + M\widetilde{V}_\ell(\widetilde{V}_\ell^T B) - K^{(k)} \\ &= \Delta\widetilde{K}_{\ell-1}^{(k+1)} + M\widetilde{V}_\ell(\widetilde{V}_\ell^T B), \end{aligned} \qquad \forall \ell \geq 1 \tag{6.39}$$

with $\Delta\widetilde{K}_0^{(k+1)} = -K^{(k)}$; compare [26, Sec. 5.2]. Using (6.38) and (6.39), the actual projected Riccati residual in the k+1-st Newton and ℓ-th ADI step for the preliminary solution is defined by $\widetilde{U}_\ell^{(k+1)} := \begin{bmatrix} \widetilde{W}_\ell^{(k+1)} & \Delta\widetilde{K}_\ell^{(k+1)} \end{bmatrix}$ via

$$\mathcal{R}(\widetilde{X}_\ell^{(k+1)}) = \widetilde{W}_\ell^{(k+1)} \left(\widetilde{W}_\ell^{(k+1)}\right)^T - \Delta\widetilde{K}_\ell^{(k+1)} \left(\Delta\widetilde{K}_\ell^{(k+1)}\right)^T =: \widetilde{U}_\ell^{(k+1)} \mathcal{D} \left(\widetilde{U}_\ell^{(k+1)}\right)^T. \tag{6.40}$$

Low-Rank Line Search: To evaluate the quartic polynomial (6.10) efficiently, the computation of the coefficients (6.10b) has to exploit the introduced low-rank factorizations. Furthermore, the coefficients are formulated for the projected low-rank factors (6.36)–(6.38). Hence, the squared norm of the projected Riccati residual for the new iterate $X^{(k+1)}$, which depends on $X^{(k)}$, ξ, and the preliminary solution $\widetilde{X}^{(k+1)}$ as shown in (6.11), is defined via

$$\begin{aligned} \widetilde{f}_{\mathcal{R}}^{(k)}(\xi) &= ||\mathcal{R}(X^{(k+1)})||_F^2 \\ &= (1-\xi)^2\widetilde{\upsilon}_1^{(k)} + \xi^2\widetilde{\upsilon}_2^{(k)} + \xi^4\widetilde{\upsilon}_3^{(k)} + 2\xi(1-\xi)\widetilde{\upsilon}_4^{(k)} - 2\xi^2(1-\xi)\widetilde{\upsilon}_5^{(k)} - 2\xi^3\widetilde{\upsilon}_6^{(k)} \end{aligned} \tag{6.41a}$$

with the scalar coefficients

$$\begin{aligned}
\widetilde{\upsilon}_1^{(k)} &= ||\mathcal{R}(X^{(k)})||_F^2 = \operatorname{tr}\left(\left(U^{(k)}\mathcal{D}\left(U^{(k)}\right)^T\right)^2\right),\\
\widetilde{\upsilon}_2^{(k)} &= ||\mathcal{L}(\widetilde{X}^{(k+1)})||_F^2 = \operatorname{tr}\left(\left(\widetilde{W}^{(k+1)}\left(\widetilde{W}^{(k+1)}\right)^T\right)^2\right),\\
\widetilde{\upsilon}_3^{(k)} &= ||MS^{(k)}BB^TS^{(k)}M||_F^2 = \operatorname{tr}\left(\left(\Delta\widetilde{K}^{(k+1)}\left(\Delta\widetilde{K}^{(k+1)}\right)^T\right)^2\right),\\
\widetilde{\upsilon}_4^{(k)} &= \langle\mathcal{R}(X^{(k)}),\mathcal{L}(\widetilde{X}^{(k+1)})\rangle = \operatorname{tr}\left(U^{(k)}\mathcal{D}\left(U^{(k)}\right)^T\widetilde{W}^{(k+1)}\left(\widetilde{W}^{(k+1)}\right)^T\right),\\
\widetilde{\upsilon}_5^{(k)} &= \langle\mathcal{R}(X^{(k)}),MS^{(k)}BB^TS^{(k)}M\rangle = \operatorname{tr}\left(U^{(k)}\mathcal{D}\left(U^{(k)}\right)^T\Delta\widetilde{K}^{(k+1)}\left(\Delta\widetilde{K}^{(k+1)}\right)^T\right),\\
\widetilde{\upsilon}_6^{(k)} &= \langle\mathcal{L}(\widetilde{X}^{(k+1)}),MS^{(k)}BB^TS^{(k)}M\rangle = \operatorname{tr}\left(\widetilde{W}^{(k+1)}\left(\widetilde{W}^{(k+1)}\right)^T\Delta\widetilde{K}^{(k+1)}\left(\Delta\widetilde{K}^{(k+1)}\right)^T\right).
\end{aligned} \tag{6.41b}$$

Using these low-rank representations, the scalar coefficients (6.41b) can be evaluated highly efficiently as described in Lemma 2.7. Hence, the computation of a step size ξ_k can be performed efficiently using the Armijo rule or the ELSM from Subsection 6.2.2.

Low-Rank Newton Update After computing the step size $\xi_k \in (0,1]$, various components need to be updated before continuing the outer Newton iteration. Notice that ξ_k is chosen such that (6.3) is fulfilled. At first, the final projected low-rank residual factors are defined for $k \geq 0$ via

$$\begin{aligned}
\overline{W}^{(k+1)} &:= \begin{cases}\left[\sqrt{1-\xi_k}\,\overline{W}^{(k)} \quad \sqrt{\xi_k}\,\widetilde{W}_\ell^{(k+1)}\right], & \xi_k \in (0,1),\\ \widetilde{W}_\ell^{(k+1)}, & \xi_k = 1,\end{cases}\\
\text{with}\quad \overline{W}^{(0)} &:= \begin{cases}\widehat{\Pi}C^T, & \Lambda(\mathcal{A},\mathcal{M}) \subset \mathbb{C}^-,\\ \widehat{\Pi}\left[C^T \quad K^{(0)}\right], & \Lambda(\mathcal{A},\mathcal{M}) \not\subset \mathbb{C}^-,\end{cases}\\
\Delta K^{(k+1)} &:= \begin{cases}\left[\sqrt{1-\xi_k}\Delta K^{(k)} \quad \xi_k\Delta\widetilde{K}_\ell^{(k+1)}\right], & \xi_k \in (0,1),\\ \Delta\widetilde{K}_\ell^{(k+1)}, & \xi_k = 1.\end{cases}
\end{aligned} \tag{6.42}$$

Secondly, the final feedback at the end of the k+1-st Newton step is defined via

$$K^{(k+1)} = (1-\xi_k)K^{(k)} + \xi_k\Delta\widetilde{K}_\ell^{(k+1)}. \tag{6.43}$$

Assuming the previous Riccati iterate is defined via $X^{(k)} = Z^{(k)}\left(Z^{(k)}\right)^T$ and the preliminary solution is defined via $\widetilde{X}^{(k+1)} = \widetilde{Z}^{(k+1)}\left(\widetilde{Z}^{(k+1)}\right)^T$, the new Riccati iterate can be

Algorithm 6 Inexact low-rank Kleinman–Newton-ADI for index-2 DAE systems

```
Input: $M, A, \widehat{G}, B, C$, initial feedback $K^{(0)}$, $tol_{\text{Newton}}$, $\bar{\tau} \in (0,1)$, and $\alpha$, $\beta \in \mathbb{R}^+$
Output: feedback matrix $K$
 1: Set $\overline{W}^{(0)} = \widehat{\Pi}\left[\alpha C^T \quad K^{(0)}\right]$, $\Delta K^{(0)} = 0$, $U^{(0)} = \left[\overline{W}^{(0)} \quad \Delta K^{(0)}\right]$.
 2: Set $k = 0$.
 3: while $\left(\|U^{(k)}\mathcal{D}\left(U^{(k)}\right)^T\|_F > tol_{\text{Newton}}\|U^{(0)}\mathcal{D}\left(U^{(0)}\right)^T\|_F\right)$ do
 4:   Compute ADI shifts $\{q_i\}_{i=1}^{n_{\text{ADI}}} = \overline{\{q_i\}_{i=1}^{n_{\text{ADI}}}} \subset \mathbb{C}^-$ and choose $\tau_k \in (0, \bar{\tau}]$.
 5:   Set $\widetilde{W}_0 = \widehat{\Pi}\left[\alpha C^T \quad K^{(k)}\right]$, $\Delta\widetilde{K}_0 = -K^{(k)}$.
 6:   Set $\ell = 1$.
 7:   while $\left(\|\widetilde{W}_{\ell-1}^T\widetilde{W}_{\ell-1}\|_F > \tau_k\|U^{(k)}\mathcal{D}\left(U^{(k)}\right)\|_F\right)$ do
 8:     Get $V_\ell$ by solving
        $$\begin{bmatrix} A^T - K^{(k)}B^T + q_\ell M & \widehat{G} \\ \widehat{G}^T & 0 \end{bmatrix} \begin{bmatrix} V_\ell \\ * \end{bmatrix} = \begin{bmatrix} \widetilde{W}_{\ell-1} \\ 0 \end{bmatrix}.$$
 9:     if $\operatorname{Im}(q_\ell) = 0$ then
10:       $\widetilde{W}_\ell = \widetilde{W}_{\ell-1} - 2q_\ell M V_\ell$
11:       $\widetilde{V}_\ell = \sqrt{-2q_\ell}V_\ell$
12:       $\Delta\widetilde{K}_{\ell+1} = \Delta\widetilde{K}_{\ell-1} + M\widetilde{V}_\ell(\widetilde{V}_\ell^T B)$
13:     else
14:       $\gamma_\ell = 2\sqrt{-\operatorname{Re}(q_\ell)}, \quad \delta_\ell = \operatorname{Re}(q_\ell)/\operatorname{Im}(q_\ell)$
15:       $\widetilde{W}_{\ell+1} = \widetilde{W}_{\ell-1} + \gamma_\ell^2 M\left(\operatorname{Re}(V_\ell) + \delta_\ell \operatorname{Im}(V_\ell)\right)$
16:       $\widetilde{V}_{\ell+1} = \left[\gamma_\ell\left(\operatorname{Re}(V_\ell) + \delta_\ell\operatorname{Im}(V_\ell)\right) \quad \gamma_\ell\sqrt{(\delta_\ell^2+1)}\operatorname{Im}(V_\ell)\right]$
17:       $\ell = \ell + 1$
18:       $\Delta\widetilde{K}_{\ell+1} = \Delta\widetilde{K}_{\ell-2} + M\widetilde{V}_\ell(\widetilde{V}_\ell^T B)$
19:     end if
20:     $\widetilde{U}_{\ell+1} = \left[\widetilde{W}_{\ell+1} \quad \Delta\widetilde{K}_{\ell+1}\right]$
21:     $\ell = \ell + 1$
22:   end while
23:   if $\|\widetilde{U}_\ell\mathcal{D}\widetilde{U}_\ell^T\|_F > (1-\beta)\|U^{(k)}\mathcal{D}\left(U^{(k)}\right)^T\|_F$ then
24:     Compute $\xi_k \in (0,1)$ using Armijo rule or ELSM.
25:   else
26:     $\xi_k = 1$.
27:   end if
28:   $\overline{W}^{(k+1)} = \left[\sqrt{1-\xi_k}\,\overline{W}^{(k)} \quad \sqrt{\xi_k}\,\widetilde{W}_\ell\right]$
29:   $\Delta K^{(k+1)} = \left[\sqrt{1-\xi_k}\Delta K^{(k)} \quad \xi_k\Delta\widetilde{K}_\ell\right]$
30:   $U^{(k+1)} = \left[\overline{W}^{(k+1)} \quad \Delta K^{(k+1)}\right]$
31:   $K^{(k+1)} = (1-\xi_k)K^{(k)} + \xi_k\Delta\widetilde{K}_\ell$
32:   $k = k + 1$
33: end while
34: $K = K^{(k)}$
```

written as

$$\begin{aligned} X^{(k+1)} &= (1-\xi_k)X^{(k)} + \xi_k\widetilde{X}^{(k+1)} \\ &= (1-\xi_k)Z^{(k)}\left(Z^{(k)}\right)^T + \xi_k\widetilde{Z}^{(k+1)}\left(\widetilde{Z}^{(k+1)}\right)^T \\ &= \left[\sqrt{1-\xi_k}\,Z^{(k)} \quad \sqrt{\xi_k}\,\widetilde{Z}^{(k+1)}\right]\left[\sqrt{1-\xi_k}\,Z^{(k)} \quad \sqrt{\xi_k}\,\widetilde{Z}^{(k+1)}\right]^T, \end{aligned} \tag{6.44}$$

whose size depend on the number of ADI steps in the k-th and k+1-st Newton iteration. Notice that only the inexpensively accumulated feedback (6.43) is necessary in the right-hand side to proceed the Newton iteration. Furthermore, using line search in the convergence phase of the Newton iteration is unlikely, such that (6.44) is, in general, never used and the new iterate can be formed by $X^{(k+1)} = \widetilde{Z}^{(k+1)}\left(\widetilde{Z}^{(k+1)}\right)^T$.

The entire process of the inexact low-rank KN-ADI method is depicted in Algorithm 6. Thereby, only the feedback matrix K is computed.

6.4. Numerical Experiments for the Improved KN-ADI Method

In this section, the theoretical results from the current chapter are verified by various numerical tests. At first, the CTP scenario, which is introduced in Subsection 3.1.4, is considered to demonstrate the usability of the inexact low-rank KN-ADI including line search for a generalized state-space system without algebraic constraints. The example is used to show further results from [26, Sec. 6]. Afterwards, the experiments from Section 4.4 are repeated to show the obtained speed-ups of the improved KN-ADI method in Subsection 6.4.2.

6.4.1. Comparison of Different Low-Rank KN-ADI methods

To demonstrate the efficiency of the improved KN-ADI method, four different versions of the methods are considered. All methods are based on Algorithm 6 using a certain parameter setup. This means, all variants use the low-rank residual formulations, as well as the real-valued version of the ADI method. Furthermore, the novel shift approach from [30] is used to compute ADI shifts adaptively during the ADI process. In detail, the solution factor V_ℓ is used as projection basis of the closed-loop pencil $(A - BK^T; M)$. Afterwards, the eigenvalues of the projected pencil are used to determine a single ADI shift using the `lp_mnmx` routine from [104]. In the first ADI step, the right hand side $\widetilde{W}_0$ is used as projection basis. In all examples, we set $tol_{\text{Newton}} = 10^{-8}$ and $\beta = 10^{-4}$.

All different variants of the improved KN-ADI method are summarized in Table 6.1. The first method is abbreviated with “KN” and refers to the “exact” KN-ADI method that is the low-rank residual version of the method used in Section 4.4. Thereby, no line search is performed and we set $tol_{\text{ADI}} = tol_{\text{Newton}} \cdot 10^{-1}$. The abbreviation “KNLS” uses the same setup as “KN”, but line search is enabled. This version is comparable to [25]

Table 6.1.: Variants of improved KN-ADI method.

method	$\mathbf{tol_{ADI}}$	$\tau_{\mathbf{k}}$	**line search**	**details**
1: KN	10^{-9}	–	no	“exact”
2: KNLS	10^{-9}	–	yes	“exact”
3: iKNsLS	adaptive	$\frac{1}{k^3+1}$	yes	inexact superlinear
4: iKNqLS	adaptive	$\min\{0.1, 0.9\|\|\mathcal{R}(X^{(k)})\|\|_F\}$	yes	inexact quadratic

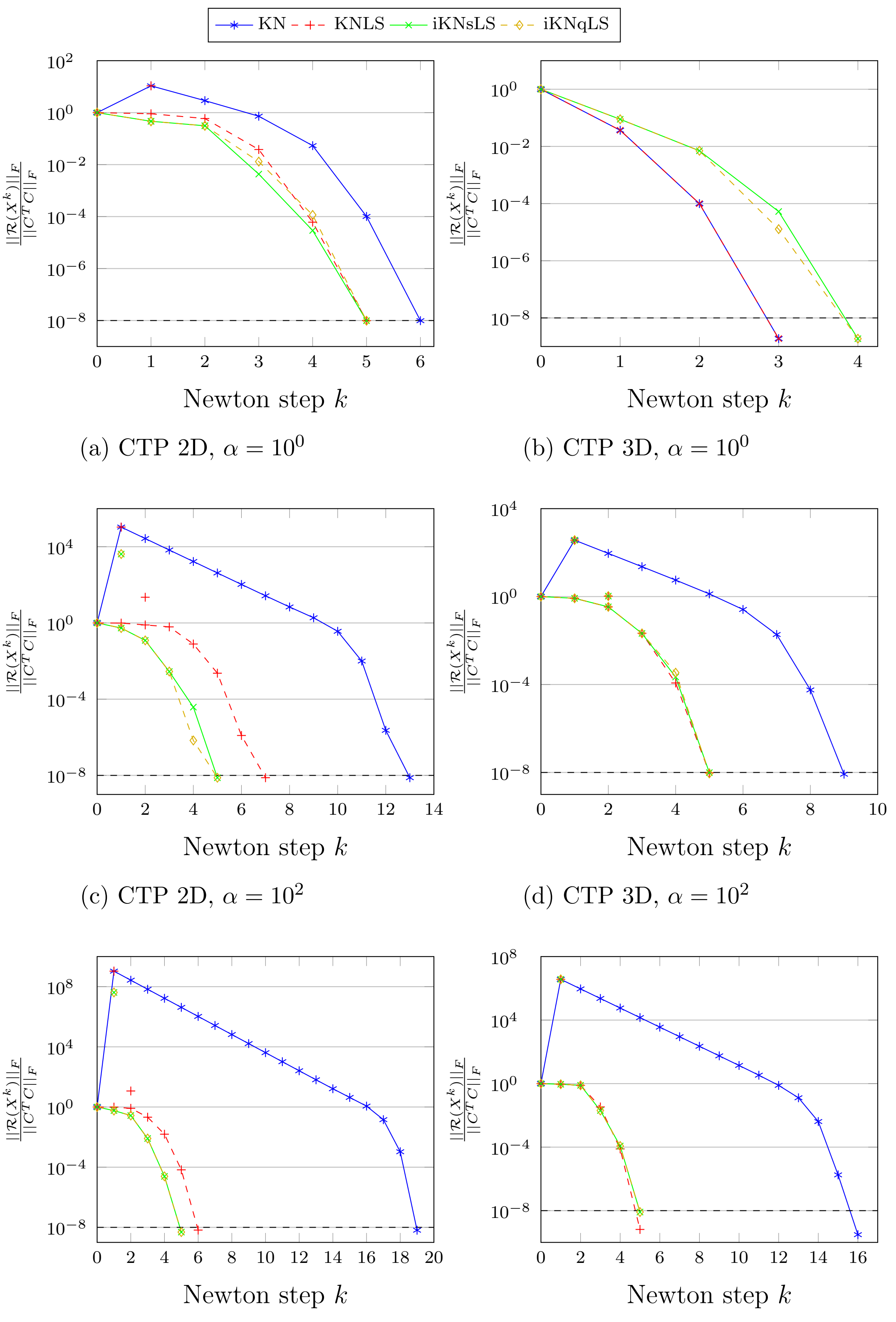
but improved by the various low-rank techniques. The third and fourth method use all introduced improvements and are abbreviated with “iKNsLS” and “iKNqLS”. Thereby, the small letters “s,q” stand for superlinear (s) or quadratic (q) convergence behavior.

The CTP setup is identical to the numerical example in [26, Sec. 6]. In detail, we use the combined transport problem (3.7) defined over the two or three dimensional UCD Ω_{UCD}, introduced in Subsection 3.1.5. The parameters are defined as $k_D = 1$, $\vec{w} = [0, -20]^T$ (in 2D), $\vec{w} = [0, -20, 0]^T$ (in 3D), $k_R = 100$, and

$$f(\vec{x}) = \begin{cases} 100, & \vec{x} \in \Omega_c. \\ 0, & \text{else.} \end{cases} \tag{6.45}$$

Furthermore, the control domain is defined as $\Omega_d = [0.1, 0.3] \times [0.4, 0.6]$ (in 2D) and $\Omega_d = [0.1, 0.3] \times [0.4, 0.6] \times [0.1, 0.3]$ (in 3D). Notice that this setup is also used in [59, 94]. The output operator is set to $C = B^T$. The mesh size is set to $h_{2D} = 1/75$ in 2D and $h_{3D} = 1/30$ in 3D. Hence, the finite element spaces are of dimension $n_{2D} = 5\,476$ and $n_{3D} = 24\,389$. The considered system is stable for all configurations such that no initial feedback is necessary.

In Figure 6.1, the evolution of the norm of the relative Riccati residuals is depicted. Thereby, the left row shows the results of the 2D and the right row of the 3D discretization. Except from the 3D, $\alpha = 10^0$ example, all examples show a significant upward jump in the Riccati residual after the first Newton step for the “KN” method. If necessary, the line search in the “KNLS” method ensures a monotonic decay of the residual. Thereby, a single mark denotes the residual before the line search is applied. If no line search is necessary, the convergence behavior is identical to the “KN” method. For an increasing output weight, the residual jump increases drastically. Hence, the amount of Newton steps that can be saved due to the line search application increases as well. The inexact method “iKNsLS” and “iKNqLS” behave in a similar fashion to the “KNLS”. The upward jump in the residual is avoided in all cases. The 3D, $\alpha = 10^0$ example shows a slower convergence than the “exact” methods. This is a natural affect of solving the Newton step inexactly. However, this inexact solve also results in a smaller upward jump after the first Newton step, since the Riccati residual can grow during the first Newton step with each ADI step. Applying the line search method in these cases results in an even faster Newton convergence as seen in all 2D examples.



(a) CTP 2D, $\alpha = 10^0$
(b) CTP 3D, $\alpha = 10^0$
(c) CTP 2D, $\alpha = 10^2$
(d) CTP 3D, $\alpha = 10^2$
(e) CTP 2D, $\alpha = 10^4$
(f) CTP 3D, $\alpha = 10^4$

Figure 6.1.: **CTP scenario:** Newton convergence for different output weights α in 2D and 3D for the different methods in Table 6.1 ($tol_{\text{Newton}} = 10^{-8}$, Armijo rule).

Table 6.2.: **CTP scenario:** Detailed iteration numbers and computation timings in seconds for the different methods in Table 6.1 ($tol_{\text{Newton}} = 10^{-8}$, Armijo rule).

			#Newt	#ADI	#LS	$\text{time}_{\text{KN-ADI}}$	$\text{time}_{\text{shift}}$	time_{LS}	$\textbf{time}_{\textbf{total}}$
$\alpha = 10^0$	CTP 2D								
	method	1	6	156	0	$5.7 \cdot 10^0$	$2.3 \cdot 10^{-1}$	–	$\mathbf{5.9 \cdot 10^0}$
		2	5	135	1	$4.8 \cdot 10^0$	$1.9 \cdot 10^{-1}$	$5.3 \cdot 10^{-3}$	$\mathbf{5.0 \cdot 10^0}$
		3	5	54	2	$1.9 \cdot 10^0$	$7.7 \cdot 10^{-2}$	$5.4 \cdot 10^{-3}$	$\mathbf{2.0 \cdot 10^0}$
		4	5	51	2	$1.8 \cdot 10^0$	$7.2 \cdot 10^{-2}$	$1.1 \cdot 10^{-4}$	$\mathbf{1.9 \cdot 10^0}$
	CTP 3D								
	method	1	3	68	0	$1.7 \cdot 10^2$	$8.5 \cdot 10^{-1}$	–	$\mathbf{1.7 \cdot 10^2}$
		2	3	68	0	$1.8 \cdot 10^2$	$8.5 \cdot 10^{-1}$	–	$\mathbf{1.8 \cdot 10^2}$
		3	4	44	0	$1.3 \cdot 10^2$	$6.3 \cdot 10^{-1}$	–	$\mathbf{1.3 \cdot 10^2}$
		4	4	45	0	$1.3 \cdot 10^2$	$6.2 \cdot 10^{-1}$	–	$\mathbf{1.3 \cdot 10^2}$
$\alpha = 10^2$	CTP 2D								
	method	1	13	359	0	$1.3 \cdot 10^1$	$5.3 \cdot 10^{-1}$	–	$\mathbf{1.4 \cdot 10^1}$
		2	7	180	2	$6.3 \cdot 10^0$	$2.5 \cdot 10^{-1}$	$6.4 \cdot 10^{-3}$	$\mathbf{6.6 \cdot 10^0}$
		3	5	28	1	$9.8 \cdot 10^{-1}$	$3.9 \cdot 10^{-2}$	$2.4 \cdot 10^{-3}$	$\mathbf{1.0 \cdot 10^0}$
		4	5	40	1	$1.4 \cdot 10^0$	$5.7 \cdot 10^{-2}$	$2.4 \cdot 10^{-3}$	$\mathbf{1.5 \cdot 10^0}$
	CTP 3D								
	method	1	9	201	0	$4.2 \cdot 10^2$	$3.0 \cdot 10^0$	–	$\mathbf{4.3 \cdot 10^2}$
		2	5	106	2	$2.1 \cdot 10^2$	$1.4 \cdot 10^0$	$7.4 \cdot 10^{-3}$	$\mathbf{2.1 \cdot 10^2}$
		3	5	33	2	$6.9 \cdot 10^1$	$4.6 \cdot 10^{-1}$	$4.3 \cdot 10^{-3}$	$\mathbf{6.9 \cdot 10^1}$
		4	5	33	2	$6.5 \cdot 10^1$	$4.8 \cdot 10^{-1}$	$2.9 \cdot 10^{-3}$	$\mathbf{6.6 \cdot 10^1}$
$\alpha = 10^4$	CTP 2D								
	method	1	19	524	0	$2.4 \cdot 10^1$	$1.0 \cdot 10^0$	–	$\mathbf{2.5 \cdot 10^1}$
		2	6	144	2	$6.2 \cdot 10^0$	$2.7 \cdot 10^{-1}$	$6.3 \cdot 10^{-3}$	$\mathbf{6.5 \cdot 10^0}$
		3	5	15	1	$6.5 \cdot 10^{-1}$	$2.9 \cdot 10^{-2}$	$5.5 \cdot 10^{-3}$	$\mathbf{6.8 \cdot 10^{-1}}$
		4	5	15	1	$6.5 \cdot 10^{-1}$	$2.9 \cdot 10^{-2}$	$5.8 \cdot 10^{-3}$	$\mathbf{6.9 \cdot 10^{-1}}$
	CTP 3D								
	method	1	16	371	0	$9.5 \cdot 10^2$	$4.7 \cdot 10^0$	–	$\mathbf{9.5 \cdot 10^2}$
		2	5	101	1	$2.4 \cdot 10^2$	$1.2 \cdot 10^0$	$2.0 \cdot 10^{-2}$	$\mathbf{2.4 \cdot 10^2}$
		3	5	25	1	$6.0 \cdot 10^1$	$3.1 \cdot 10^{-1}$	$1.3 \cdot 10^{-2}$	$\mathbf{6.1 \cdot 10^1}$
		4	5	25	1	$6.3 \cdot 10^1$	$3.0 \cdot 10^{-1}$	$1.2 \cdot 10^{-2}$	$\mathbf{6.3 \cdot 10^1}$

Table 6.3.: **CTP scenario:** Comparison of Armijo rule and ELSM ($tol_{\text{Newton}} = 10^{-8}$).

			Armijo				ELSM			
			#Newt	#ADI	#LS	time$_{\text{LS}}$	#Newt	#ADI	#LS	time$_{\text{LS}}$
$\alpha = 10^2$	CTP 2D									
	method	3	5	28	1	$2.4 \cdot 10^{-3}$	5	**27**	1	$3.4 \cdot 10^{-3}$
	method	4	5	40	1	$2.4 \cdot 10^{-3}$	5	**32**	1	$3.3 \cdot 10^{-3}$
	CTP 3D									
	method	3	5	**33**	2	$4.3 \cdot 10^{-3}$	6	51	1	$5.6 \cdot 10^{-3}$
	method	4	5	**33**	2	$2.9 \cdot 10^{-3}$	6	48	1	$3.7 \cdot 10^{-3}$

The difference between the superlinear and the quadratic method is not crucial for the Newton convergence of this example. So far, the "KNLS" seems to perform approximately as good as the inexact versions. However, examining the detailed iteration numbers and computation timings in Table 6.2 show that the actual runtimes (time$_{\text{total}}$) differ quite drastically. The major amount of time is spent by solving the SPS and computing the residuals; denoted by time$_{\text{KN-ADI}}$. This time is roughly proportional to the amount of ADI steps (#ADI). The time to compute the ADI shifts (time$_{\text{shift}}$) and the time to evaluate the line search method (time$_{\text{LS}}$) are of no consequence. Notice that the line search method is applied at most twice in each example. Hence, the number of ADI steps is the most important property in this table. As it turns out, the inexact solution of the Newton step improves the performance of the algorithm drastically; compare, e.g., [59]. The combination of the line search approach with the inexact solution of each Newton step enhances the performance tremendously without introducing significant additional costs. Similar results have been published in [26, Sec. 6]. Further examples in [26] show that the convergence behavior is strongly influenced by the choice of the output matrix C.

All results in Figure 6.1 and Table 6.2 use the Armijo rule from Subsection 6.2.2. For this method, a lower bound for the step size τ_k is established in Theorem 6.2. To compare this method with the ELSM from Subsection 6.2.2, all tests have been carried out again using the ELSM. As it turns out, both methods behave very similarly. One example, where significant differences occur, is depicted in Table 6.3. For $\alpha = 10^2$ the ELSM performs slightly better in the 2D case. However, in the 3D case, the Armijo rule performs better. In general, the Armijo rule can be evaluated faster. Due to this advantage and the non existing lower bound for the step size computed with the ELSM, all further computations use the Armijo rule.

6.4.2. Improved KN-ADI for Incompressible Flows

In this subsection, the improved KN-ADI method is applied to the Stokes, NSE, and CFM scenarios. These results have not yet been published elsewhere. As in Section 4.4, we mainly use the NSE scenario.

The first part of this subsection shows the influence of the various improvements to the overall performance of the algorithm. Afterwards, all numerical experiments from Section 4.4 are repeated with the most efficient parameter setup. Subsection 4.4.4 shows that the convergence of the relative change in the feedback K does not necessarily yield a convergence of the actual Riccati residual. Thus, one needs to consider a configuration whose relative residual dropped below the Newton tolerance $tol_{\text{Newton}} = 10^{-8}$. This is an important fact that the interested reader needs to keep in mind for further comparisons. In Figure 4.5a, it is shown that for $\mathrm{Re} = 500$, $\alpha = 10^0$, refinement: Level 1, and an ADI tolerance of $tol_{\text{ADI}} = 10^{-10}$, the relative Riccati residual drops below tol_{Newton} after $k = 8$ Newton steps. This configuration, including the computation of the explicitly projected residual, is the basis for the following comparison and is denoted by "Setup i". This and four additional setups are summarized in Table 6.4a. For each setup, the detailed iteration numbers (#Newt, #ADI, #LS) and the various timings are depicted in Table 6.4b. Furthermore, the convergence behavior of the relative Riccati residuals are depicted in Figure 6.2.

The first improvement, denoted by "Setup ii" uses the improved KN-ADI method in the "KN" setup from Table 6.1 with $tol_{\text{ADI}} = 10^{-8}$ and an explicit computation of the projected Riccati residual. This explicit residual computation is not necessary, but demonstrates the accuracy of the low-rank residual. In the third setup, all these unnecessary projections are avoided.

Setup iv incorporates a modified version of the adaptive shifts in [30]. The single shifts, as used for the CTP scenario, do not yield a converging low-rank ADI method. Hence, at most 15 ADI shifts are adaptively computed in each call. Using the right hand side $\widetilde{W}_0$ during the first call, the projected pencil has $n_a + n_r$ eigenvalues that are entered into the `lp_mnmx` routine from [104] to determine $r = \min\{15, n_a + n_r\}$ shifts. For further shift determinations, all blocks V_ℓ are stored during the ADI iteration until all previously determined shifts are used. The entire block $Z_{\text{tmp}} = [V_1, \ldots, V_r]$ is entered into the adaptive shift computation method and a thin QR-decomposition (using `qr(`Z_{tmp}`,0)` in MATLAB) is performed to determine a new projection basis. Afterwards, 15 ADI shifts are determined via `lp_mnmx`. A similar approach is described in [37]. The final setup, Setup v, uses the "iKNqLS" method from Table 6.1 using the Armijo rule and adaptively determined ADI shifts.

The incorporation of the real-valued ADI formulation in Setup ii reduces the number of linear solves (#lin_solve) and, therefore, the time to solve these systems ($\text{time}_{\text{lin\_solve}}$) drastically. Furthermore, the costs to compute the projected residuals are reduced by at least two magnitudes. Avoiding the computation of the projected residuals completely decreases the costs further, since the costs to evaluate the low-rank residuals are another magnitude smaller. An even more drastic performance improvement can be obtained by the adaptively determined ADI shifts. These shifts reduce the number of ADI steps and linear solves by a factor of five. Additionally, the computation of these adaptive ADI shifts is at least one magnitude less expensive.

Table 6.4.: Comparison of various KN-ADI methods.

(a) Detailed setups for the different KN-ADI methods.

i	KN-ADI, **explicit Lyap. and Ric. residual computation**, heuristic shifts (Alg. 3, $tol_{\text{ADI}} = 10^{-10}$)
ii	real-valued, low-rank residual KN-ADI, **expl. Ric. residual computation**, heuristic shifts (cf. Alg. 5, $tol_{\text{ADI}} = 10^{-8}$)
iii	real-valued, low-rank residual KN-ADI, **heuristic shifts** (cf. Alg. 5, $tol_{\text{ADI}} = 10^{-8}$)
iv	real-valued, low-rank residual KN-ADI, **adaptive shifts** (cf. Alg. 5, $tol_{\text{ADI}} = 10^{-8}$)
v	**inexact low-rank** KN-ADI, adaptive shifts (Alg. 6, $\eta_k = \min\{0.1, 0.9\|\|\mathcal{R}(X^{(k)})\|\|_F\}$, Armijo method)

(b) **NSE scenario:** Iteration numbers and timings in seconds for the different methods in Table 6.4a (Re = 500, refinement: Level 1, $tol_{\text{Newton}} = 10^{-8}$, $\alpha = 10^0$).

	#Newt	#ADI	#lin_solve	#LS	$\text{time}_{\text{lin\_solve}}$	$\text{time}_{\text{shift}}$	$\text{time}_{\text{rel\_ch}}$	$\text{time}_{\text{proj-res}}$	$\text{time}_{\text{lr-res}}$	time_{LS}	$\mathbf{time_{total}}$
i	8	3067	3067	–	$1.4 \cdot 10^3$	$3.6 \cdot 10^1$	$6.7 \cdot 10^0$	$5.4 \cdot 10^3$	–	–	$\mathbf{6.8 \cdot 10^3}$
ii	8	3031	1721	–	$7.0 \cdot 10^2$	$3.6 \cdot 10^1$	$8.5 \cdot 10^{-3}$	$1.0 \cdot 10^1$	$1.8 \cdot 10^0$	–	$\mathbf{7.5 \cdot 10^2}$
iii	8	3031	1721	–	$7.0 \cdot 10^2$	$3.7 \cdot 10^1$	$7.6 \cdot 10^{-3}$	–	$1.8 \cdot 10^0$	–	$\mathbf{7.4 \cdot 10^2}$
iv	8	600	346	–	$1.4 \cdot 10^2$	$2.8 \cdot 10^0$	$3.2 \cdot 10^{-3}$	–	$6.0 \cdot 10^{-1}$	–	$\mathbf{1.5 \cdot 10^2}$
v	7	305	176	1	$7.3 \cdot 10^1$	$1.9 \cdot 10^0$	$5.1 \cdot 10^{-3}$	–	$2.4 \cdot 10^{-1}$	$1.5 \cdot 10^{-2}$	$\mathbf{7.5 \cdot 10^1}$

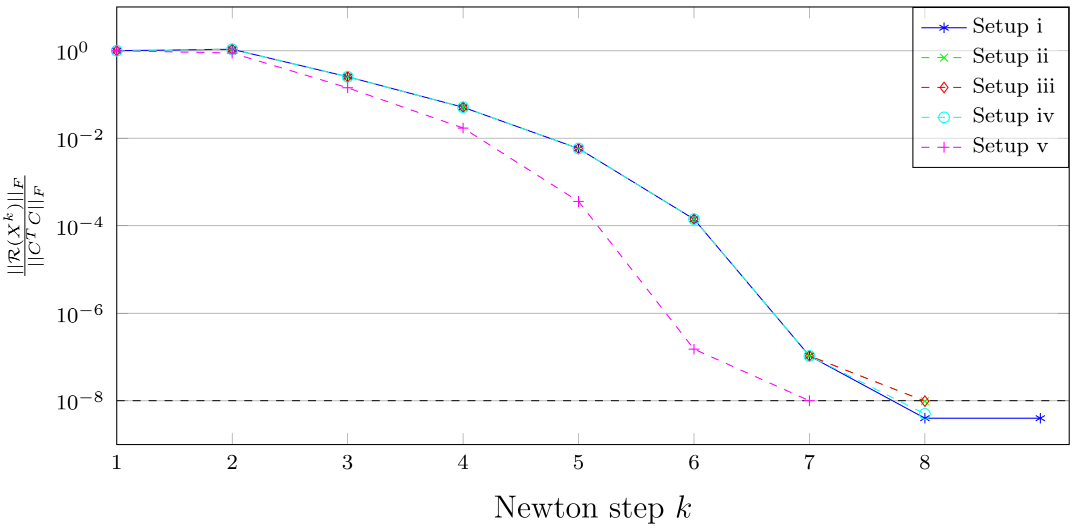


Figure 6.2.: **NSE scenario:** Newton convergence for the different KN-ADI methods in Table 6.4a (Re = 500, refinement: Level 1, $tol_{\text{Newton}} = 10^{-8}$, $\alpha = 10^0$).

Using Setup v improves the method further. The number of ADI steps is reduced by a factor of two, which also yields less time for the shift computation and the evaluation of the low-rank residual. Thereby, the costs for the single line search call can be neglected. Comparing the total computation times between Setup i and Setup v, a speedup of 90 can be achieved. One could argue now that the ninth Newton step for Setup i is not necessary. Removing the costs for this step, one ends up with 2696 linear solves and a speedup of approximately 80. Although each improvement reduces the overall computation costs, it is shown in Figure 6.2 that the Riccati residuals are identical, except from the last step, for the setups i–iv. This shows that the low-rank residual can be used to determine the actual residual accurately. As for the CTP example in the previous subsection, the "iKNqLS" method improves the convergence behavior of the Riccati residual. These improvements become more important if the output weighting α increases. By repeating all numerical experiments from Section 4.4, the performance improvements of the relevant scenarios for this thesis are established in the following.

In Table 6.5, the influence of the output weighting α (Table 6.5a), as well as the influence of the refinement levels (Table 6.5b), are depicted. Similar to the results in Table 4.4, the amount of Newton steps increases with an increasing α. Nevertheless, significant savings can be achieved by the "iKNqLS" method. Furthermore, line search is only necessary for higher Reynolds numbers and higher output weights. Considering the increasing refinement level, the saving regarding the Newton steps are not that significant. Especially for Re $\geq$ 300 and a refinement level larger than two, the number of Newton steps is nearly as high as for the KN-ADI in Table 4.4b. Additionally, an unusual amount of line search runs can be observed. We believe that this effect is a result of the instability of the considered pencil. Solving the first Newton step inexactly might yield an intermediate solution that is nearly not stabilizing. Therefore, the following ADI iteration tends to diverge.

Table 6.5.: **NSE scenario:** Number of Newton steps (#Newt), ADI steps (#ADI), and line search runs (#LS) during the "iKNqLS" process ($tol_{\text{Newton}} = 10^{-8}$, $\eta_k = \min\{0.1, 0.9||\mathcal{R}(X^{(k)})||_F\}$, Armijo method).

(a) Influence of output weighting α during the "iKNqLS" process (refinement: Level 1).

Re	100			200			300			400			500		
α	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS
10^{-2}	3	38	–	4	74	–	4	73	–	4	87	–	5	79	–
10^{-1}	4	53	–	5	109	–	5	84	–	4	74	–	5	109	–
10^{0}	5	80	–	6	118	–	7	119	–	6	115	1	7	176	1
10^{1}	7	98	–	7	134	–	8	153	1	10	212	2	9	201	2
10^{2}	7	109	–	9	199	1	12	296	3	12	331	3	12	340	4

(b) Influence of refinement levels during the "iKNqLS" process ($\alpha = 1$).

Re	100			200			300			400			500		
	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS
Level 1	5	80	–	6	118	–	7	119	–	6	115	1	7	176	1
Level 2	4	73	–	6	118	1	7	144	1	7	148	1	7	168	1
Level 3	5	99	–	5	124	–	10	221	3	8	200	2	7	183	–
Level 4	4	72	–	6	176	1	11	198	6	10	199	5	10	243	3
Level 5	5	126	–	6	160	1	11	244	4	11	273	4	10	267	3
Level 6	6	189	–	6	184	1	11	280	4	11	279	4	13	344	6

Table 6.6.: **NSE scenario:** Comparison of “exact” and inexact start ($tol_{\text{Newton}} = 10^{-8}$, $\eta_k = \min\{0.1, 0.9||\mathcal{R}(X^{(k)})||_F\}$, Armijo method).

		start inexact					start “exact” with $tol_{\text{ADI}} = 10^{-2}$				
		#Newt	#ADI	#LS	time_{LS}	$\mathbf{time_{total}}$	#Newt	#ADI	#LS	time_{LS}	$\mathbf{time_{total}}$
Re = 300											
Level	3	10	221	3	$1.0 \cdot 10^{-1}$	$\mathbf{7.2 \cdot 10^{2}}$	8	186	1	$2.5 \cdot 10^{-2}$	$\mathbf{5.9 \cdot 10^{2}}$
Level	4	11	198	6	$8.2 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{3}}$	8	177	0	–	$\mathbf{1.4 \cdot 10^{3}}$
Level	5	11	244	4	$9.5 \cdot 10^{-2}$	$\mathbf{4.8 \cdot 10^{3}}$	8	215	0	–	$\mathbf{4.1 \cdot 10^{3}}$
Level	6	11	280	4	$1.0 \cdot 10^{-4}$	$\mathbf{1.2 \cdot 10^{4}}$	9	259	0	–	$\mathbf{1.2 \cdot 10^{4}}$
Re = 400											
Level	3	8	200	2	$3.8 \cdot 10^{-2}$	$\mathbf{6.1 \cdot 10^{2}}$	6	158	1	$5.4 \cdot 10^{-2}$	$\mathbf{5.2 \cdot 10^{2}}$
Level	4	10	199	5	$9.1 \cdot 10^{-2}$	$\mathbf{1.5 \cdot 10^{3}}$	7	197	1	$6.7 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{3}}$
Level	5	11	273	4	$5.6 \cdot 10^{-1}$	$\mathbf{5.4 \cdot 10^{3}}$	8	244	1	$1.4 \cdot 10^{-1}$	$\mathbf{4.6 \cdot 10^{3}}$
Level	6	11	279	4	$1.2 \cdot 10^{0}$	$\mathbf{1.3 \cdot 10^{4}}$	8	272	1	$3.0 \cdot 10^{-1}$	$\mathbf{1.3 \cdot 10^{4}}$
Re = 500											
Level	3	7	183	0	–	$\mathbf{6.2 \cdot 10^{2}}$	7	179	1	$4.1 \cdot 10^{-2}$	$\mathbf{6.0 \cdot 10^{2}}$
Level	4	10	243	3	$1.3 \cdot 10^{-1}$	$\mathbf{2.0 \cdot 10^{3}}$	8	192	1	$4.0 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{3}}$
Level	5	10	267	3	$5.7 \cdot 10^{-1}$	$\mathbf{5.5 \cdot 10^{3}}$	9	261	1	$8.7 \cdot 10^{-2}$	$\mathbf{5.5 \cdot 10^{3}}$
Level	6	13	344	6	$1.6 \cdot 10^{0}$	$\mathbf{1.6 \cdot 10^{4}}$	7	248	1	$6.6 \cdot 10^{-1}$	$\mathbf{1.2 \cdot 10^{4}}$

Our algorithm detects this behavior by monitoring the Riccati and Lyapunov residual continuously. Notice that this behavior is not covered by the convergence proof in Theorem 6.9, where a stabilizing solution for $k \geq k_0$ is required. Nevertheless, our algorithm handles this situation by deleting the last ADI step and performing line search. This yields convergence in all considered examples, although the relative Riccati residual seems to stagnate for a couple of steps. Hence, an increasing amount of line search runs is required.

Another approach to circumvent this problem is starting the process in the “KN” setup with a fixed ADI tolerance for the first Newton step. Based on the experiences of the “KN” setup, we set $tol_{\mathrm{ADI}} = 10^{-2}$ for the first two Newton steps. If the relative Riccati residual decreases and drops below $5 \cdot 10^{-1}$, the method switches to the “iKNqLS” scheme. A comparison of both starting procedures is depicted in Table 6.6 for Re ≥ 300 and the refinements Level 3–6. As it turns out, the latter setup prevents the stagnation of the relative Riccati residual and reduces the amount of Newton steps. However, the “exact” solved first Newton steps increase the number of ADI steps and, therefore, the savings due to less Newton steps do not really pay off. In total, the amount of ADI steps reduces only slightly in most of the examples.

In the end, the “iKNqLS” setup is able to solve all setups. A summary of the detailed timings for the “iKNqLS” setup is depicted in Table 6.7. Comparing these results with the results in Table 4.5, significant speedups are achieved. The interested reader is reminded that the results in Table 4.5 only ensured convergence of the norm of the relative change of the feedback matrix. Hence, the speedups compared to the results in Section 4.4 are even more significant if one would consider the KN-ADI method with an explicit calculation of the projected residuals.

The results from the use of the “iKNqLS” setup for the Stokes and CFM scenario are depicted in Table 6.8. Both scenarios benefit from the improved method. Due to the already relatively small amount of ADI steps using the KN-ADI method, the improvements are not as significant as for the NSE scenario. Nevertheless, the “iKNqLS” setup reduces the amount of ADI steps and, therefore, the overall computation costs.

6.5. Conclusion – Part III

Linking to the conclusions in Section 4.5, the inexact low-rank KN-ADI method is a powerful tool to solve GARE for index-2 DAE systems. Besides the various algorithmic improvements, which make the method highly efficient, a convergence proof could be established that combines the inexact Newton idea with a line search approach. On the one hand, the inexact solution of the Newton steps decreases the amount of ADI steps significantly. On the other hand, the line search approach prevents the upward jump of the Riccati residual such that the previously observed stagnation phase in the relative change of the feedback can be avoided. Both tools are only applicable for large-scale systems due to the various low-rank formulations. These low-rank formulations are the key ingredient to establish the method in Algorithm 6 as a competitive tool to solve large-scale GARE.

Table 6.7.: **NSE scenario:** Detailed computation timings in seconds for “iKNqLS” process ($tol_{\text{Newton}} = 10^{-8}$, $tol_{\text{ADI}} = 10^{-7}$, $\alpha = 10^{0}$, $\eta_k = \min\{0.1, 0.9||\mathcal{R}(X^{(k)})||_F\}$, Armijo method).

		$\text{time}_{\text{lin\_solve}}$	$\text{time}_{\text{shift}}$	$\text{time}_{\text{lr\_res}}$	time_{LS}	$\mathbf{time_{total}}$
			Re = 100			
Level	1	$2.7 \cdot 10^{1}$	$4.5 \cdot 10^{-1}$	$1.5 \cdot 10^{-1}$	–	$\mathbf{2.8 \cdot 10^{1}}$
	2	$7.4 \cdot 10^{1}$	$9.8 \cdot 10^{-1}$	$2.3 \cdot 10^{-1}$	–	$\mathbf{7.5 \cdot 10^{1}}$
	3	$2.9 \cdot 10^{2}$	$2.8 \cdot 10^{0}$	$5.6 \cdot 10^{-1}$	–	$\mathbf{2.9 \cdot 10^{2}}$
	4	$5.0 \cdot 10^{2}$	$4.3 \cdot 10^{0}$	$8.2 \cdot 10^{-1}$	–	$\mathbf{5.0 \cdot 10^{2}}$
	5	$2.2 \cdot 10^{3}$	$1.8 \cdot 10^{1}$	$3.2 \cdot 10^{0}$	–	$\mathbf{2.2 \cdot 10^{3}}$
	6	$8.6 \cdot 10^{3}$	$7.4 \cdot 10^{1}$	$9.0 \cdot 10^{0}$	–	$\mathbf{8.7 \cdot 10^{3}}$
			Re = 200			
Level	1	$4.5 \cdot 10^{1}$	$8.5 \cdot 10^{-1}$	$1.9 \cdot 10^{-1}$	–	$\mathbf{4.6 \cdot 10^{1}}$
	2	$1.3 \cdot 10^{2}$	$1.6 \cdot 10^{0}$	$3.1 \cdot 10^{-1}$	$1.8 \cdot 10^{-2}$	$\mathbf{1.3 \cdot 10^{2}}$
	3	$3.9 \cdot 10^{2}$	$3.7 \cdot 10^{0}$	$6.8 \cdot 10^{-1}$	–	$\mathbf{3.9 \cdot 10^{2}}$
	4	$1.3 \cdot 10^{3}$	$1.2 \cdot 10^{1}$	$2.1 \cdot 10^{0}$	$1.2 \cdot 10^{-1}$	$\mathbf{1.3 \cdot 10^{3}}$
	5	$3.0 \cdot 10^{3}$	$2.8 \cdot 10^{1}$	$4.0 \cdot 10^{0}$	$2.9 \cdot 10^{-1}$	$\mathbf{3.0 \cdot 10^{3}}$
	6	$8.7 \cdot 10^{3}$	$8.4 \cdot 10^{1}$	$1.1 \cdot 10^{1}$	$9.6 \cdot 10^{-1}$	$\mathbf{8.8 \cdot 10^{3}}$
			Re = 300			
Level	1	$4.5 \cdot 10^{1}$	$8.9 \cdot 10^{-1}$	$1.8 \cdot 10^{-1}$	–	$\mathbf{4.6 \cdot 10^{1}}$
	2	$1.7 \cdot 10^{2}$	$2.0 \cdot 10^{0}$	$4.1 \cdot 10^{-1}$	$3.1 \cdot 10^{-2}$	$\mathbf{1.8 \cdot 10^{2}}$
	3	$5.8 \cdot 10^{2}$	$5.3 \cdot 10^{0}$	$9.4 \cdot 10^{-1}$	$2.5 \cdot 10^{-2}$	$\mathbf{5.9 \cdot 10^{2}}$
	4	$1.4 \cdot 10^{3}$	$1.1 \cdot 10^{1}$	$1.8 \cdot 10^{0}$	–	$\mathbf{1.4 \cdot 10^{3}}$
	5	$4.0 \cdot 10^{3}$	$3.6 \cdot 10^{1}$	$5.1 \cdot 10^{0}$	–	$\mathbf{4.1 \cdot 10^{3}}$
	6	$1.2 \cdot 10^{4}$	$1.2 \cdot 10^{2}$	$1.5 \cdot 10^{1}$	$1.0 \cdot 10^{-4}$	$\mathbf{1.2 \cdot 10^{4}}$
			Re = 400			
Level	1	$4.8 \cdot 10^{1}$	$1.0 \cdot 10^{0}$	$1.8 \cdot 10^{-1}$	$1.8 \cdot 10^{-2}$	$\mathbf{4.9 \cdot 10^{1}}$
	2	$1.8 \cdot 10^{2}$	$2.3 \cdot 10^{0}$	$3.8 \cdot 10^{-1}$	$1.9 \cdot 10^{-2}$	$\mathbf{1.8 \cdot 10^{2}}$
	3	$5.1 \cdot 10^{2}$	$5.4 \cdot 10^{0}$	$8.7 \cdot 10^{-1}$	$5.4 \cdot 10^{-2}$	$\mathbf{5.2 \cdot 10^{2}}$
	4	$1.6 \cdot 10^{3}$	$1.6 \cdot 10^{1}$	$2.2 \cdot 10^{0}$	$6.7 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{3}}$
	5	$4.6 \cdot 10^{3}$	$3.9 \cdot 10^{1}$	$6.1 \cdot 10^{0}$	$1.4 \cdot 10^{-1}$	$\mathbf{4.6 \cdot 10^{3}}$
	6	$1.3 \cdot 10^{4}$	$1.3 \cdot 10^{2}$	$1.5 \cdot 10^{1}$	$1.2 \cdot 10^{0}$	$\mathbf{1.3 \cdot 10^{4}}$
			Re = 500			
Level	1	$7.3 \cdot 10^{1}$	$1.9 \cdot 10^{0}$	$2.4 \cdot 10^{-1}$	$1.5 \cdot 10^{-2}$	$\mathbf{7.5 \cdot 10^{1}}$
	2	$2.0 \cdot 10^{2}$	$2.8 \cdot 10^{0}$	$4.1 \cdot 10^{-1}$	$2.2 \cdot 10^{-2}$	$\mathbf{2.1 \cdot 10^{2}}$
	3	$6.0 \cdot 10^{2}$	$6.6 \cdot 10^{0}$	$9.5 \cdot 10^{-1}$	$4.1 \cdot 10^{-2}$	$\mathbf{6.0 \cdot 10^{2}}$
	4	$1.6 \cdot 10^{3}$	$1.5 \cdot 10^{1}$	$2.3 \cdot 10^{0}$	$4.0 \cdot 10^{-2}$	$\mathbf{1.6 \cdot 10^{3}}$
	5	$5.4 \cdot 10^{3}$	$5.0 \cdot 10^{1}$	$6.1 \cdot 10^{0}$	$8.7 \cdot 10^{-2}$	$\mathbf{5.5 \cdot 10^{3}}$
	6	$1.6 \cdot 10^{4}$	$1.7 \cdot 10^{2}$	$1.9 \cdot 10^{1}$	$1.6 \cdot 10^{0}$	$\mathbf{1.6 \cdot 10^{4}}$

Table 6.8.: Number of Newton steps (#Newt), ADI steps (#ADI), and line search runs (#LS) during the "iKNqLS" process for Stokes and CFM scenario ($tol_{\text{Newton}} = 10^{-8}$, $\eta_k = \min\{0.1, 0.9||\mathcal{R}(X^{(k)})||_F\}$, Armijo method).

(a) **Stokes scenario:** Influence of output weighting α during the "iKNqLS" process (refinement: Level 1).

Re	100			200			300			400			500		
α	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS
10^{-2}	4	37	–	4	40	–	4	38	–	4	38	–	4	42	–
10^{-1}	4	38	–	4	40	–	4	38	–	4	39	–	4	39	–
10^{0}	4	38	–	4	40	–	4	38	–	4	40	–	4	41	–
10^{1}	4	36	–	4	37	–	4	40	–	5	55	–	5	48	–
10^{2}	5	61	–	6	60	–	6	61	1	6	55	1	6	64	1

(b) **CFM scenario:** Influence of refinement levels on the "iKNqLS" process ($\alpha = 1$).

Set	I			II			III			IV			V		
	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS	#Newt	#ADI	#LS
Level 1	5	116	–	5	134	–	6	153	–	4	74	–	3	60	–
Level 2	6	172	–	5	144	–	6	177	–	6	168	1	6	177	–
Level 3	4	95	–	6	190	–	6	170	–	6	175	1	5	152	–
Level 4	7	237	1	6	161	1	8	269	1	7	215	–	8	250	–

7 Closed-Loop Simulation

Contents

Chapter 7 examines the usability of the feedback matrix K in a closed-loop forward simulation to stabilize the NSE scenario from Subsection 3.2.1. The results extend the statements in [15, Sec. 4.5]. Notice that we have chosen the NSE scenario since the CFM scenario would have required additional implementation steps within NAVIER to perform a closed-loop simulation. These implementations are part of future research.

In the context of this thesis, the process of stabilizing the NSE scenario refers to preventing or extinguishing vortexes behind the obstacle $\Omega_{\mathrm{K}}^{\mathrm{O}}$ in Figure 3.1 that occur for $\mathrm{Re} > 200$.

7.1. Discretized Model

As it is explained in Section 3.4, the nonlinear NSE (3.8) are linearized around a stationary but possibly unstable solution $(\vec{w}(\vec{x}), \chi_s(\vec{x}))$. The stationary flow field $\vec{w}(\vec{x})$ is chosen to be a laminar flow without any vortexes behind the obstacle. Such a flow field can be obtained by solving the stationary NSE (3.13). In cases where the stationary solution $\vec{w}(\vec{x})$ does not exist or does not fulfill the desired properties, an open-loop controller might be applied to the original system (3.8) to obtain the desired laminar flow $\vec{w}(\vec{x})$, compare, e.g., [76].

Raymond showed in [106] that the optimal control $\boldsymbol{u}(t)$, which is computed to asymptotically stabilize the linearized system (3.13), also stabilizes the original nonlinear system (3.8), assuming that $\vec{v}(t, \vec{x}) \approx \vec{w}(\vec{x})$. In other words, for small perturbations between

$\vec{w}(\vec{x})$ and $\vec{v}(t,\vec{x})$, the optimal control $\boldsymbol{u_z}(t)$ forces the velocity field $\vec{v}(t,\vec{x})$ towards $\vec{w}(\vec{x})$. Since $\vec{w}(\vec{x})$ is assumed to be laminar, the resulting velocity field $\vec{v}(t,\vec{x})$ is laminar as well.

The closed-loop forward simulation is executed within the finite element flow solver NAVIER [12]. As mentioned in Section 3.5, the FEM package NAVIER is used to assemble the matrices for the discrete representations of the Stokes/NSE and CFM scenarios. Thus, the spatially discretized nonlinear NSE system is defined by

$$M_{\boldsymbol{z}}\frac{d}{dt}\boldsymbol{v}(t) = A_{\boldsymbol{v}}(\boldsymbol{v}(t))\boldsymbol{v}(t) + G\boldsymbol{\chi}(t), \tag{7.1a}$$

$$0 = G^T\boldsymbol{v}(t), \tag{7.1b}$$

$$\boldsymbol{y}(t) = C\boldsymbol{v}(t). \tag{7.1c}$$

The mass matrix $M_{\boldsymbol{z}}$ and the discretized gradient G are defined as in (3.20). The discretized velocity $\boldsymbol{v}(t) \in \mathbb{R}^{n_{\boldsymbol{z}}}$ is restricted to the interior of Ω to match the dimension of $\boldsymbol{z}(t)$. The system matrix $A_{\boldsymbol{v}}(\boldsymbol{v}(t)) \in \mathbb{R}^{n_{\boldsymbol{z}} \times n_{\boldsymbol{z}}}$ represents the nonlinear Navier–Stokes operator in (3.8a), which is determined by the current velocity field $\boldsymbol{v}(t)$.

The system (7.1) does not represent the discretized boundary Γ, such that discretized versions of the boundary conditions (3.8d)–(3.8f) are considered straightforward. Additionally, the initial condition

$$\boldsymbol{v}(0) = \boldsymbol{w}$$

is considered to fulfill the requirement of $\vec{v}(t,\vec{x}) \approx \vec{w}(\vec{x})$ for $t = 0$. The interested reader is referred to [12] for more details about the actual solution strategy for the nonlinear system (7.1).

7.2. Dirichlet Boundary Control Input

As it is introduced in Subsection 4.1.3, the feedback stabilization is supposed to influence the system (7.1) via a boundary input. To apply the feedback within the forward simulation, the optimal control $\boldsymbol{u_z}(t) \in \mathbb{R}^{n_r}$ needs to be reflected appropriately in the Dirichlet BC. The optimal control is defined via

$$\boldsymbol{u_z}(t) = -K\boldsymbol{z}(t) = -(K\boldsymbol{v}(t) - K\boldsymbol{w}) \tag{7.2}$$

with K being the feedback matrix, which is computed via Algorithm 6 to stabilize the linearized and discretized NSE (3.20). Using the matrix market format [45], the feedback K which is computed via MATLAB can be imported into NAVIER. Afterwards, the feedback K is applied to the stationary velocity field $\boldsymbol{w}(\vec{x})$ and the recent velocity field $\boldsymbol{v}(t)$.

The physical interpretation of $\boldsymbol{u_z}(t)$ describes the magnitude of the parabolic in-/outflow conditions over each of the n_r control boundaries $\Gamma_{feed,i}$, scaled by the corresponding entry $u_i(t)$, as depicted in Figure 4.1. This behavior needs to be interpreted as

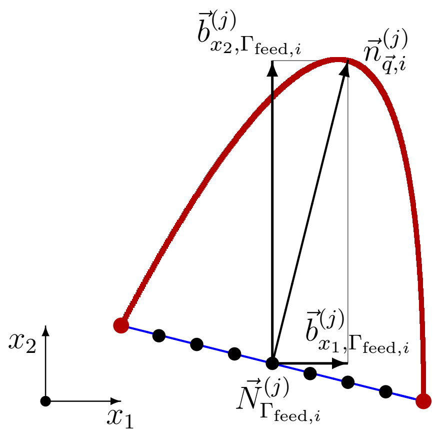

$\vec{b}^{(j)}_{x_2,\Gamma_{\text{feed},i}}$
$\vec{n}^{(j)}_{\vec{q},i}$
$\vec{b}^{(j)}_{x_1,\Gamma_{\text{feed},i}}$
x_2
x_1
$\vec{N}^{(j)}_{\Gamma_{\text{feed},i}}$

Figure 7.1.: Decomposition of control input over $\Gamma_{\text{feed},i}$ to assemble $\widehat{B}$ ($|\vec{n}^{(j)}_{\vec{q},i}| = 1$).

Dirichlet BC. Therefore, the parabolic inflow profile over $\Gamma_{\text{feed},i}$ with a maximum height of $|\vec{n}^{(j)}_{\vec{q},i}| = 1$ is decomposed into the x_1 and x_2 velocity components corresponding to each boundary node $\vec{N}^{(j)}_{\Gamma_{\text{feed},i}}$, as depicted for one node in Figure 4.1. For the particular node number j, the velocity perpendicular to the boundary also describes the maximum height of 1. The decomposition is defined by

$$\vec{n}^{(j)}_{\vec{q},i} = \vec{b}^{(j)}_{x_1,\Gamma_{\text{feed},i}} + \vec{b}^{(j)}_{x_2,\Gamma_{\text{feed},i}} = \begin{bmatrix} b^{(j)}_{x_1,\Gamma_{\text{feed},i}} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ b^{(j)}_{x_2,\Gamma_{\text{feed},i}} \end{bmatrix}.$$

All components of the decomposition depend only on the used discretization and can be assembled into a special boundary input operator $\widehat{B}$ in the offline grid generating phase. During the online phase, the optimal control is computed via (7.2) in each time step and the Dirichlet BC on Γ_{feed} is assigned by

$$\boldsymbol{v}(t) = \boldsymbol{g}_{\text{feed},i}(t) := \widehat{B}u_i(t) \text{ on } \Gamma_{\text{feed},i},\ \forall i = 1, \ldots, n_r.$$

These BC complete the necessary information to solve the closed-loop forward simulation of (7.1).

As explained in Subsection 4.4.1, the output operator $C_{\boldsymbol{z}}$ measures the vertical velocity components at the observation nodes $\boldsymbol{P}_{\text{obs}}$, as marked in Figure 3.3. Only these components are reflected throughout the feedback computation. Hence, these components should remain as small as possible to minimize the cost functional (4.16a), which also yields a laminar flow field.

7.3. Numerical Experiments for Closed-Loop Simulation

The quality of the closed-loop simulation is affected by the output weight α. However, choosing α appropriately is an optimization process by itself and not within the scope of this thesis. For Re $= 500$ and refinement Level 6, we determined $\alpha = 2.0$ throughout various numerical test as a suitable choice.

The major issue of the closed-loop simulation is that the control $\boldsymbol{u_z}$ is not restricted explicitly. Hence, the computed $\boldsymbol{u_z}(t)$ might exceed the limits of the underlying FE discretization. Therefore, we restrict each component of the input $\boldsymbol{u_z}(t)$ by 3.0.

The result of the closed-loop forward simulation is illustrated by a snapshot for $t = 16$ in Figure 7.2. In detail, the vertical velocity component, which is measured by the output matrix C, is depicted for three different configurations. Thereby, positive velocity components are depicted by red and negative components are depicted by blue. The entire domain is divided into a 5×5 grid. All observation points are located in the third and fourth column of the third row. Hence, the vertical velocity in these two rectangles should remain zero which is depicted by the color green.

The middle picture in Figure 7.2 shows the forward simulation without the influence of any feedback. The occurring vortexes are distinctly visible by alternating red and blue areas that fade with a growing distance to the obstacle.

The lower picture shows the influence of the initial feedback which has been computed by solving the corresponding GABE as discussed in Subsection 4.2.3. The vortexes are smoothed slightly, but the flow remains unstable. This shows that the Bernoulli feedback is not sufficient in this setup, although the pencil arising from the linearized NSE is stable using the initial feedback.

The upper picture shows the influence of the Riccati feedback. All vertical components are significantly smoothed. Due to the restriction of $\boldsymbol{u_z}(t)$ and an empirical determined output weight, one might be able to improve these results further. A video of the closed-loop simulation is available in the supplementary material submitted with this thesis and on the website of the research project: `http://www2.mpi-magdeburg.mpg.de/mpcsc/projekte/optconfeestabmultiflow/nse_re_500_level_6_lambda_2.php`.

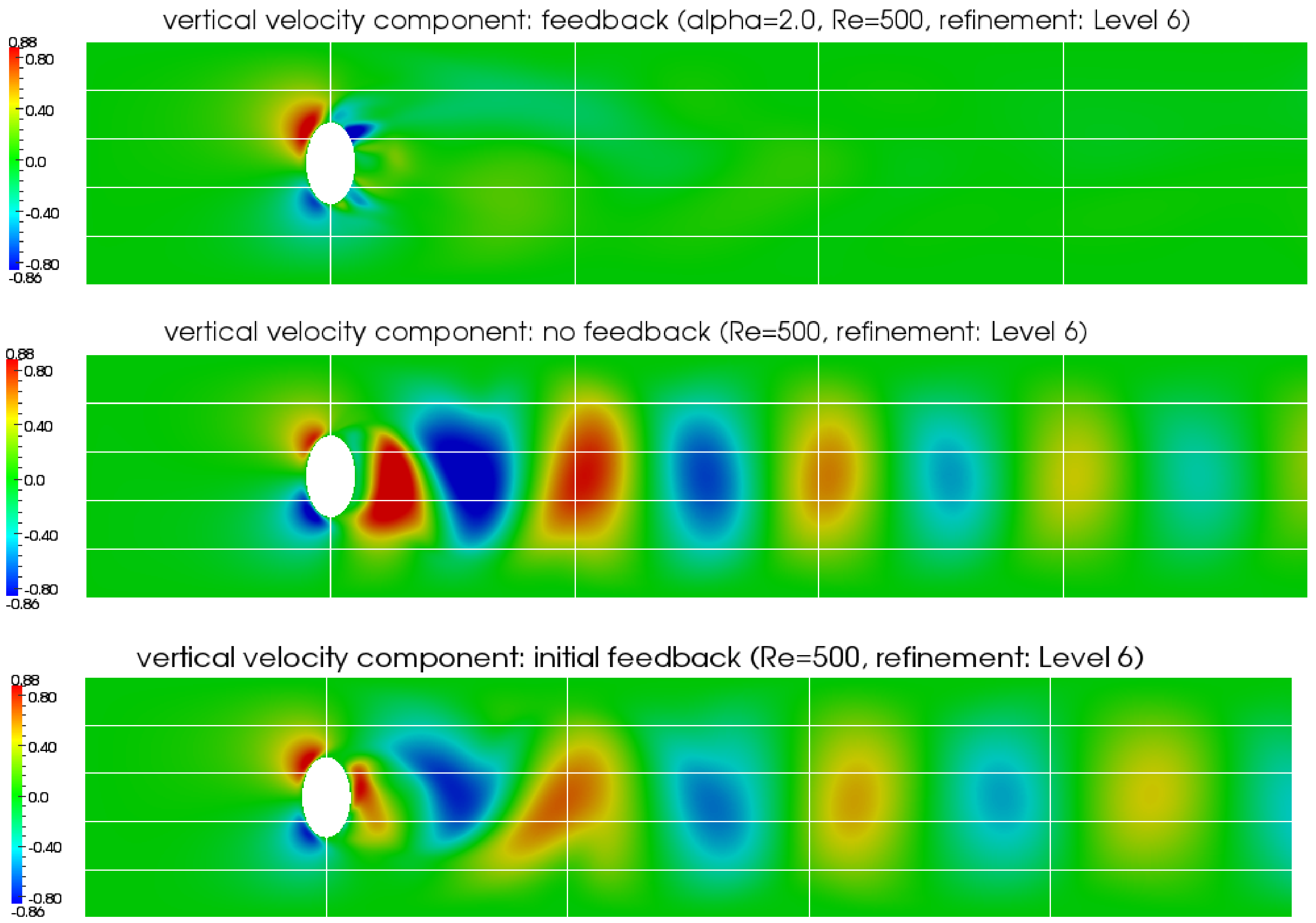


Figure 7.2.: **NSE scenario:** Snapshots of closed-loop simulation at $t = 16$ (Re $= 500$, $\alpha = 2$, refinement: Level 6).

8 Conclusion and Outlook

Contents

8.1. Conclusion

In this thesis, we have investigated various numerical aspects of the feedback stabilization of scalar and vector-valued transport problems; one scalar and three vector-valued test scenarios have been defined in Chapter 3. The first scenario is the scalar one and describes a combined transport process defined on a unit cube in two or three dimensions. The focus of this thesis lies on incompressible multi-field flow problems that can be interpreted as vector-valued transport problems. Hence, scenario two and three are defined by the nonlinear Navier–Stokes and the linear Stokes equations. Both scenarios use the two-dimensional *Kármán vortex street* as the test domain and are constrained by the so-called divergence-free condition. It is this condition that assures the mass conservation of the entire system. The last scenario combines the nonlinear Navier–Stokes flow with a scalar diffusion-convection process that is defined on a two-dimensional reactor model.

All scenarios have been linearized and discretized using the inf-sup stable *Taylor–Hood* finite elements. Including certain input and output operators, one ends up with a generalized state-space system whose structure is identical for all scenarios. However, all vector-valued transport problems yield index-2 differential-algebraic equations (DAE) due to the divergence-free condition. The major task of this thesis is to handle these systems, avoiding expensive general-purpose DAE solvers.

In Chapter 4, a feedback stabilization approach for these index-2 DAE systems has been derived. This approach mimics the ideas in [106] in a numerical way. The central problem of Chapter 4 is the solution of an implicitly projected large-scale generalized algebraic Riccati equation. In order to solve it, a nested iteration has been derived which incorporated the projection method from [71] into the combination of a specially tailored Kleinman–Newton method and an *alternating directions implicit* (ADI) method. Moreover, a convergence proof for this version of Newton's method has been derived. Thereby, the similar approach in [41] differs from our approach in the definition of the used projector. While both projectors project on the same subspace using different topologies, the projection never needs to be performed explicitly in our approach. Notice that the entire process is also directly applicable to the scalar transport problem.

The algorithmic key ingredients for our approach are the efficient feedback accumulation as well as an efficiently evaluable stopping criterion. The usability of the proposed method has been shown throughout intensive numerical experiments in Section 4.4. Although our proposed method was able to solve all considered test scenarios, certain drawbacks occurred. It turned out that the relative change is not sufficient as stopping criterion for certain configurations. Using the explicitly projected residual would overcome this drawback, but the computation of this residual is highly expensive. Furthermore, the choice of the accuracy of the inner ADI method, which depends on the problem, is not known beforehand. Hence, each Newton step might be solved more accurately than necessary, which wastes a lot of computation time, or is not solved accurately enough such that the entire algorithm does not converge. Another problem is the upward jump of the Riccati residual after the first Newton step. This behavior is well-known throughout literature and results in significantly higher computation times. Nevertheless, by choosing an appropriate combination of the stopping criteria for each nested level, convergence of the Kleinman–Newton-ADI method can be guaranteed.

The main computational step within our algorithm is the solution of a large-scale indefinite saddle point system. Hence, Chapter 5 investigates two different methods: the sparse direct solver from MATLAB as well as the Krylov subspace method GMRES. To use the latter method efficiently, a specially tailored block preconditioner has been derived in Section 5.2. The main issue is the efficient approximation of the dense and large-scale Schur complement. Based on [56], two approximation methods have been introduced. Therefore, two block approximation methods have been adapted. In detail, an *algebraic multigrid* method as well as a *Chebyshev semi-iteration* have been used to derive an efficient preconditioner. The issue of complex-valued systems for these kinds of approximation techniques is, in general, not considered but has been addressed in this thesis. Further intensive numerical tests have shown the usability of all our methods. It has been discovered that each method has advantages and disadvantages regarding the different scenarios and parameters. However, all approximation methods improve the efficiency of the introduced preconditioner drastically.

In Chapter 6, we have revisited the Kleinman–Newton-ADI method from Chapter 4 and have investigated techniques to overcome the examined drawbacks. On the one hand, considering the line search approach in [25], the upward jumps of the Riccati residual can be prevented. However, this method involves the computation of dense matrices such that it is not applicable for large-scale systems. On the other hand, the inexact Kleinman–Newton method in [59] determines the ADI accuracy adaptively such that the wasted computational time can be reduced. However, we could show that the proposed convergence proof in [59] is not applicable in a low-rank case. Nevertheless, we have been able to incorporate the line search method in [25] as well as the inexact Kleinman–Newton method in [59]. The resulting method is the inexact low-rank Kleinman–Newton-ADI method for index-2 DAE systems which is depicted in Algorithm 6. To combine all of these approaches, the low-rank residual ADI formulations in [27, 28] have been extended to a low-rank Riccati residual formulation. Additionally, we have stated a convergence proof for this novel method. Further improvements have been achieved by adapting the real-valued ADI in [29] and the adaptively chosen ADI shifts from [30] to the index-2 DAE case. The usability of the inexact Kleinman–Newton-ADI method for index-2 DAE systems has been validated throughout intensive numerical experiments. As a result, significant speedups up to a factor of 90 could be achieved.

In Chapter 7, a closed-loop simulation of the Navier–Stokes equations on the *Kármán vortex street* validates the usability of our proposed Riccati feedback. This means that our flow stabilization technique is applicable in this case using standard inf-sup stable finite elements.

The majority of the achievements in this thesis are connected to the iterative solution of large scale index-2 DAE-based Riccati equations. From our point of view, there are only very few parts within this algorithm that can be improved further. Using the various low-rank structures and techniques, one ends up with a highly efficient algorithm.

8.2. Outlook

Whilst working on this thesis a few open problems appeared.

The first open problem is that a more accurate realization of the theory in [106] to build the boundary input operator is desirable. Therefore, a more detailed study of the functional analytic connections is necessary. Strongly connected to this is the problem of a proper approximation theory for the computed feedback matrix K. Using a finite element approximation to solve indefinite dimensional partial differential equations, one is always interested in the quality of the resulting approximation. This concept needs to be extended to the feedback operator to solve the infinite dimensional control problem, similar to the results for parabolic systems in [10, 115].

The iterative solution of the large-scale saddle point systems within the nested iteration yields two open problems. On the one hand, it has been shown in Section 5.2 that the introduced preconditioners are suitable to use GMRES as iterative solver. However, the number of GMRES steps varies drastically for certain configurations. Hence, a more robust preconditioner, especially regarding the used ADI shift, would increase the efficiency. Further possible improvements are block Krylov and Krylov subspace recycling techniques.

On the other hand, as investigated in detail in Chapter 6, the adaptive selection of stopping tolerances within the nested iteration is crucial to ensure convergence without wasting computational time. Extending the concept of inexact solves for the innermost linear solver, depending on the needs of each single ADI step, is crucial to use iterative methods efficiently. Some basic work for this problem has been done in [116, 124]. Extending these ideas to the index-2 DAE case using preconditioned Krylov subspace methods and incorporating this in the nested inexact Kleinman–Newton-ADI method is a highly non-trivial problem. However, first numerical tests show that the results in [116, 124] also hold for the index-2 DAE case.

The last open problem is the extension of the closed-loop simulations to other scenarios. Therefore, one needs to extend the modifications within the used finite element flow solver NAVIER, as well as some conceptual considerations regarding the design of the control problem.

Appendices

A

Appendix

Contents

In this chapter, some additional material is provided. First, eleven theses are formulated that summarize the core statements of this thesis. Secondly, the relations between the content of this thesis and other publications by the author are established. Afterwards, the *declaration of honor* and the computer specifications of the used compute server are stated.

Theses

1. This thesis investigates a Riccati-based feedback stabilization approach for scalar and vector-valued transport problems that are discretized by a standard finite element method.

2. All considered scenarios yield generalized state-space systems after linearization and discretization. The vector-valued scenarios yield additional algebraic constraints.

3. A linear-quadratic regulator approach is introduced, whose optimal solution is obtained by a Riccati-based feedback.

4. A specially tailored Kleinman–Newton-ADI method can be used to solve the arising large-scale and implicitly projected Riccati equations as it is established in a convergence proof.

5. Inside the innermost step of the nested iteration, large-scale indefinite saddle point systems have to be solved efficiently.

6. Sparse direct solvers, as well as the Krylov subspace method GMRES, are investigated in detail to perform the demanding linear solves.

7. A block preconditioner improves the convergence of GMRES significantly. Thereby, various approximation techniques, such as a Schur complement approximation, an algebraic multigrid method, and a Chebyshev semi-iteration, are investigated to increase the efficiency of the preconditioner.

8. The combination of an inexact Newton scheme, a line search approach, a real-valued ADI formulation, as well as the extension of the low-rank residual ADI method, yields a highly efficient method to solve large-scale Riccati equations that are based on index-2 DAE systems.

9. Two different line search techniques are investigated and a convergence proof for the inexact low-rank Kleinman–Newton-ADI method for index-2 DAE systems has been established.

10. All introduced techniques have been validated through intensive numerical tests. The results indicate large performance gains and significant runtime savings.

11. The overall usability of the proposed approach is demonstrated by a closed-loop feedback simulation of the Navier–Stokes equations on the *Kármán vortex street.*

Publications

Many parts of this thesis are extensions of already published publications of the author. In this section, the content of this thesis is associated with these publications. Furthermore, a few statements of this thesis, which have been derived by other authors and that have not yet been published, are assigned.

The definition of the CFM scenario, as well as the extension of the projector $\widehat{\Pi}^T$ for the CFM case is based on

[14] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. "Optimal Control-Based Feedback Stabilization of Multi-Field Flow Problems". In: *Trends in PDE Constrained Optimization*. Ed. by G. Leugering, P. Benner, S. Engell, A. Griewank, H. Harbrecht, M. Hinze, R. Rannacher, and S. Ulbrich. Vol. 165. Internat. Ser. Numer. Math. Basel: Birkhäuser, 2014, pp. 173–188 (cit. on pp. 4, 43, 44, 46, 54, 55, 58, 67, 70, 76, 77, 85, 122).

The definition of the NSE scenario as well as most derivations from Chapter 4 have been published in a condensed version in

[15] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. "Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flows". In: *SIAM J. Sci. Comput.* 37.2 (2015), A832–A858 (cit. on pp. 4, 39, 42, 44, 46, 47, 50, 51, 54, 55, 58, 61, 62, 67, 68, 70, 76, 80, 81, 122, 155).

The introduction of the Stokes scenario as well as various segments of the block preconditioner, defined in Section 5.2, have been investigated in

[35] 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". In: *SIAM J. Sci. Comput.* 35.5 (2013), S150–S170 (cit. on pp. 4, 41, 42, 46, 47, 50, 51, 54, 55, 58, 59, 67, 70, 76, 90, 93, 96, 97, 122).

The block preconditioner used for the CFM scenario in Section 5.2 has been introduced in

[36] P. Benner, J. Saak, M. Stoll, and H. K. Weichelt. "Efficient Solvers for Large-Scale Saddle Point Systems Arising in Feedback Stabilization of Multi-Field Flow Problems". In: *System Modeling and Optimization*. Ed. by C. Pötzsche, C. Heuberger, B. Kaltenbacher, and F. Rendl. Vol. 443. IFIP Adv. Inf. Commun. Technol. New York: Springer, 2014, pp. 11–20 (cit. on pp. 4, 43, 46, 67, 90, 93, 96, 101, 102, 122).

The statements in Chapter 6 are extensions to the index-2 DAE case of the results in

[26] P. Benner, M. Heinkenschloss, J. Saak, and H. K. Weichelt. "An inexact low-rank Newton-ADI method for large-scale algebraic Riccati equations". In: *Appl. Numer. Math.* 108 (Oct. 2016), pp. 125–142. ISSN: 0168-927 (cit. on pp. 5, 24, 38, 47, 61, 121–123, 125, 126, 128–132, 135, 137, 139, 142, 143, 146, IV).

Notice that the proof of Proposition 2.17 had been derived by M. Heinkenschloss throughout working on [26], but had not been published there. Furthermore, the proof of Theorem 2.26 is based on ideas of P. Benner.

Declaration of Honor

I, hereby, declare that I produced this thesis without prohibited assistance and that all sources of information that were used in producing this thesis, including my own publications, have been clearly marked and referenced.
In particular I have not wilfully:

- Fabricated data or ignored or removed undesired results.
- Misused statistical methods with the aim of drawing other conclusions than those warranted by the available data.
- Plagiarised data or publications or presented them in a disorted way.

I know that violations of copyright may lead to injunction and damage claims from the author or prosecution by the law enforcement authorities. This work has not previously been submitted as a doctoral thesis in the same or a similar form in Germany or in any other country. It hast not previously been published as a whole.

Schriftliche Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht.
Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert oder verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadenersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann.
Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, den 28. Januar 2016

Heiko Weichelt

Computer Specifications

The specific computer specifications of the used compute server nodes are listed below.

```
RELEASE INFO
Description:    CentOS release 5.5 (Final)
Release:        5.5
Codename:       Final

SYSTEM INFO
operating system: GNU/Linux
kernel name: Linux
kernel release: 2.6.18-194.el5
kernel version: #1 SMP Fri Apr 2 14:58:14 EDT 2010
plattfrom type: x86_64 (64Bit)

PROCESSOR INFO
CPU type: Intel(R) Xeon(R) CPU X5650 @ 2.67GHz
number of physical CPUs: 2
number of cores (virtual): 12
Processor 1
    number of physical cores:  6
    cache size : 12288 KB
Processor 2
    number of physical cores:  6
    cache size : 12288 KB
total number of physical cores: 12

MEMORY INFO
RAM installed: 49.449.320 kB
SWAP installed: 0 kB

default MATLAB version
8.0.0.783 (R2012b)

Currently loaded modules:
  comp/gcc/4.5.1
  lib/il32p64/suitesparse/3.7
  lib/il32p64/blas/openblas-0.2.4
  lib/il32p64/lapack/3.4.1
  apps/matlab/2012b
```

B Bibliography

[1] L. Amodei and J. M. Buchot. “A stabilization algorithm of the Navier–Stokes equations based on algebraic Bernoulli equation”. In: *Numer. Linear Algebra Appl.* 19.4 (2012), pp. 700–727 (cit. on pp. 3, 26, 69, 70).

[2] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems.* SIAM Publications, 2005 (cit. on pp. 18, 19).

[3] A. C. Antoulas, D. C. Sorensen, and Y. Zhou. “On the decay rate of Hankel singular values and related issues”. In: *Systems Control Lett.* 46.5 (2002), pp. 323–342. ISSN: 0167-6911 (cit. on p. 22).

[4] L. Armijo. “Minimization of functions having Lipschitz continuous first partial derivatives”. In: *Pacific J. Math.* 16.1 (1966), pp. 1–3 (cit. on p. 126).

[5] W. F. Arnold and A. J. Laub. “Generalized eigenproblem algorithms and software for algebraic Riccati equations”. In: *Proc. IEEE* 72 (1984), pp. 1746–1754 (cit. on p. 64).

[6] I. Babuška and T. Strouboulis. *The Finite Element Method and its Reliability.* Oxford: Clarendon Press, 2001, pp. XI+802 (cit. on pp. 8, 9, 46).

[7] M. Badra. “Feedback stabilization of the 2-D and 3-D Navier-Stokes equations based on an extended system”. In: *ESAIM: COCV* 15 (04 Oct. 2009), pp. 934–968. ISSN: 1262-3377 (cit. on p. 3).

[8] M. Badra. “Lyapunov function and local feedback boundary stabilization of the Navier–Stokes equations”. In: *SIAM J. Control Optim.* 48.3 (2009), pp. 1797–1830 (cit. on p. 3).

[9] H.-D. Baehr and K. Stephan. *Heat and Mass Transfer.* 2., rev. ed. Berlin: Springer, 2006, pp. XXII+688 (cit. on p. 42).

[10] H. T. Banks and K. Kunisch. “The linear regulator problem for parabolic systems”. In: *SIAM J. Control Optim.* 22 (1984), pp. 684–698 (cit. on p. 163).

[11] E. Bänsch. “Local mesh refinement in 2 and 3 dimensions”. In: *IMPACT Comput. Sci. Eng.* 3.3 (1991), pp. 181–191 (cit. on pp. 9, 78).

[12] E. Bänsch. “Simulation of Instationary, Incompressible Flows”. In: *Acta Math. Univ. Comenianae* 67.1 (1998), pp. 101–114 (cit. on pp. 49, 156).

[13] E. Bänsch and P. Benner. “Stabilization of incompressible flow problems by Riccati-based feedback”. In: *Constrained Optimization and Optimal Control for Partial Differential Equations*. Ed. by G. Leugering, S. Engell, A. Griewank, M. Hinze, R. Rannacher, V. Schulz, M. Ulbrich, and S. Ulbrich. Vol. 160. Internat. Ser. Numer. Math. Basel: Birkhäuser, 2012, pp. 5–20 (cit. on pp. 2, 3, 55).

[14] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. “Optimal Control-Based Feedback Stabilization of Multi-Field Flow Problems”. In: *Trends in PDE Constrained Optimization*. Ed. by G. Leugering, P. Benner, S. Engell, A. Griewank, H. Harbrecht, M. Hinze, R. Rannacher, and S. Ulbrich. Vol. 165. Internat. Ser. Numer. Math. Basel: Birkhäuser, 2014, pp. 173–188 (cit. on pp. 4, 43, 44, 46, 54, 55, 58, 67, 70, 76, 77, 85, 122).

[15] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. “Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flows”. In: *SIAM J. Sci. Comput.* 37.2 (2015), A832–A858 (cit. on pp. 4, 39, 42, 44, 46, 47, 50, 51, 54, 55, 58, 61, 62, 67, 68, 70, 76, 80, 81, 122, 155).

[16] V. Barbu. “Feedback stabilization of the Navier-Stokes equations”. In: *ESAIM: Control Optim. Calc. Var.* 9 (2003), pp. 197–206 (cit. on p. 3).

[17] V. Barbu, I. Lasiecka, and R. Triggiani. *Tangential boundary stabilization of Navier-Stokes equations*. Vol. 181. American Mathematical Society, 2006 (cit. on p. 3).

[18] V. Barbu and R. Triggiani. “Internal stabilization of Navier-Stokes equations with finite-dimensional controllers”. In: *Indiana Univ. Math. J.* 53.5 (2004), pp. 1443–1494 (cit. on p. 3).

[19] S. Barrachina, P. Benner, and E. S. Quintana-Ortí. “Parallel Solution of Large-Scale Algebraic Bernoulli Equations via the Matrix Sign Function Method”. In: *Proc. 2005 Intl. Conf. Parallel Processing (ICPP-05)*. 2005, pp. 189–193 (cit. on p. 26).

[20] S. Barrachina, P. Benner, and E. S. Quintana-Ortí. “Efficient algorithms for generalized algebraic Bernoulli equations based on the matrix sign function”. In: *Numer. Algorithms* 46.4 (2007), pp. 351–368 (cit. on pp. 25, 26, 29, 69).

[21] U. Baur and P. Benner. “Efficient Solution of Algebraic Bernoulli Equations Using $\mathscr{H}$-Matrix Arithmetic”. In: *Proceedings of Enumath 2007 the 7th European Conference on Numerical Mathematics and Advanced Applications*. Ed. by K. Kunisch, G. Of, and O. Steinbach. Springer-Verlag, Berlin/Heidelberg, Germany, 2008, pp. 127–134. ISBN: 978-3-540-69776-3 (cit. on p. 26).

[22] P. Benner. “Computational Methods for Linear-Quadratic Optimization”. In: *Supplemento ai Rendiconti del Circolo Matematico di Palermo, Serie II* No. 58 (1999), pp. 21–56 (cit. on p. 27).

[23] P. Benner. “Solving Large-Scale Control Problems”. In: *IEEE Control Syst. Mag.* 14.1 (2004), pp. 44–59 (cit. on pp. xxv, 31, 32).

[24] P. Benner, M. Bollhöfer, D. Kressner, C. Mehl, and T. Stykel, eds. *Numerical Algebra, Matrix Theory, Differential-Algebraic Equations and Control Theory.* Festschrift in honor of Volker Mehrmann. Springer, 2015, pp. XLIV+608. ISBN: 978-3-319-15259-2 (cit. on p. 25).

[25] P. Benner and R. Byers. “An exact line search method for solving generalized continuous-time algebraic Riccati equations”. In: *IEEE Trans. Automat. Control* 43.1 (Jan. 1998), pp. 101–107 (cit. on pp. 64, 121, 122, 125, 128, 129, 131, 132, 142, 162, 163).

[26] P. Benner, M. Heinkenschloss, J. Saak, and H. K. Weichelt. “An inexact low-rank Newton-ADI method for large-scale algebraic Riccati equations”. In: *Appl. Numer. Math.* 108 (Oct. 2016), pp. 125–142. ISSN: 0168-927 (cit. on pp. 5, 24, 38, 47, 61, 121–123, 125, 126, 128–132, 135, 137, 139, 142, 143, 146, IV).

[27] P. Benner, P. Kürschner, and J. Saak. “A Reformulated Low-Rank ADI Iteration with Explicit Residual Factors”. In: *Proc. Appl. Math. Mech.* 13.1 (2013), pp. 585–586. ISSN: 1617-7061 (cit. on pp. 133, 135, 163).

[28] P. Benner, P. Kürschner, and J. Saak. “An improved numerical method for balanced truncation for symmetric second order systems”. In: *Math. Comput. Model. Dyn. Syst.* 19.6 (2013), pp. 593–615 (cit. on pp. 133, 134, 163).

[29] P. Benner, P. Kürschner, and J. Saak. “Efficient handling of complex shift parameters in the low-rank Cholesky factor ADI method”. In: *Numer. Algorithms* 62.2 (2013), pp. 225–251 (cit. on pp. 133, 135, 163).

[30] P. Benner, P. Kürschner, and J. Saak. “Self-Generating and Efficient Shift Parameters in ADI Methods for Large Lyapunov and Sylvester Equations”. In: *Electron. Trans. Numer. Anal.* 43 (2014), pp. 142–162 (cit. on pp. 32, 142, 147, 163).

[31] P. Benner, J.-R. Li, and T. Penzl. “Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems”. In: *Numer. Linear Algebra Appl.* 15.9 (2008), pp. 755–777 (cit. on pp. xxv, 22, 25, 31, 32, 65, 70–72, 74, 75, 134).

[32] P. Benner, R.-C. Li, and N. Truhar. “On the ADI method for Sylvester equations”. In: *J. Comput. Appl. Math.* 233.4 (2009), pp. 1035–1045 (cit. on pp. 12, 22).

[33] P. Benner and J. Saak. “Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey”. In: *GAMM Mitteilungen* 36.1 (2013), pp. 32–52 (cit. on p. 28).

[34] P. Benner, J. Saak, F. Schieweck, P. Skrzypacz, and H. K. Weichelt. “A Non-Conforming Composite Quadrilateral Finite Element Pair for Feedback Stabilization of the Stokes Equations”. In: *J. Numer. Math.* 22.3 (Oct. 2014), pp. 191–220 (cit. on pp. 62, 122).

[35] 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”. In: *SIAM J. Sci. Comput.* 35.5 (2013), S150–S170 (cit. on pp. 4, 41, 42, 46, 47, 50, 51, 54, 55, 58, 59, 67, 70, 76, 90, 93, 96, 97, 122).

[36] P. Benner, J. Saak, M. Stoll, and H. K. Weichelt. “Efficient Solvers for Large-Scale Saddle Point Systems Arising in Feedback Stabilization of Multi-Field Flow Problems”. In: *System Modeling and Optimization.* Ed. by C. Pötzsche, C. Heuberger, B. Kaltenbacher, and F. Rendl. Vol. 443. IFIP Adv. Inf. Commun. Technol. New York: Springer, 2014, pp. 11–20 (cit. on pp. 4, 43, 46, 67, 90, 93, 96, 101, 102, 122).

[37] P. Benner, J. Saak, and M. M. Uddin. “Structure preserving model order reduction of large sparse second-order index-1 systems and application to a mechatronics model”. In: *Math. Comput. Model. Dyn. Syst.* 22.6 (2016), pp. 509–523 (cit. on p. 147).

[38] P. Benner and A. Schneider. “Balanced Truncation Model Order Reduction for LTI Systems with many Inputs or Outputs”. In: *Proc. of the 19th International Symposium on Mathematical Theory of Networks and Systems.* Ed. by András Edelmayer. Budapest, Hungary, 2010, pp. 1971–1974. ISBN: 978-963-311-370-7 (cit. on p. 50).

[39] P. Benner and A. Schneider. “Model Reduction for Linear Descriptor Systems with Many Ports”. In: *Progress in Industrial Mathematics at ECMI 2010.* Ed. by M. Günther, A. Bartel, M. Brunk, S. Schöps, and M. Striebel. Vol. 17. Mathematics in Industry. Berlin: Springer, 2012, pp. 137–143 (cit. on p. 50).

[40] P. Benner and A. Schneider. *Balanced Truncation for Descriptor Systems with Many Terminals.* Preprint MPIMD/13-17. Available from `http://www.mpi-magdeburg.mpg.de/preprints/`. Max Planck Institute Magdeburg, Oct. 2013 (cit. on pp. 50, 54).

[41] P. Benner and T. Stykel. “Numerical solution of projected algebraic Riccati equations”. In: *SIAM J. Numer. Anal* 52.2 (2014), pp. 581–600 (cit. on pp. 75, 162).

[42] M. Benzi, G. H. Golub, and J. Liesen. “Numerical solution of saddle point problems”. In: *Acta Numerica* 14 (2005), pp. 1–137 (cit. on pp. 50, 90).

[43] D. Bini, B. Iannazzo, and B. Meini. *Numerical Solution of Algebraic Riccati Equations.* Society for Industrial and Applied Mathematics, 2011 (cit. on pp. 3, 25).

[44] P. Bochev and R. B. Lehoucq. “On the finite element solution of the pure Neumann problem”. In: *SIAM Review* 47.1 (2005), pp. 50–66 (cit. on pp. 50, 101).

[45] R. F. Boisvert, R. Pozo, and K. A. Remington. *The Matrix Market Exchange Formats: Initial Design*. NIST Interim Report 5935. National Institute of Standards and Technology, Dec. 1996 (cit. on p. 156).

[46] J. H. Bramble and J. E. Pasciak. "Iterative techniques for time dependent Stokes problems". In: *Computers & Mathematics with Applications* 33.1-2 (Jan. 1997), pp. 13–30 (cit. on p. 101).

[47] 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)*. Amsterdam: North-Holland, 1987, pp. 73–187 (cit. on p. 40).

[48] J. Bührle. "Properties of Time-Dependent Stokes Flow and the Regularization of Velocity Fluctuations in Praticle Suspensions". PhD thesis. Philipps-Universität Marburg, 2007 (cit. on p. 41).

[49] J. Cahouet and J.-P. Chabard. "Some fast 3D finite element solvers for the generalized Stokes problem". In: *International Journal for Numerical Methods in Fluids* 8.8 (1988), pp. 869–895 (cit. on p. 101).

[50] K. A. Cliffe, T. J. Garratt, and A. Spence. "Eigenvalues of block matrices arising from problems in fluid mechanics". In: *SIAM J. Matrix Anal. Appl.* 15.4 (1994), pp. 1310–1318 (cit. on pp. 16, 17, 51, 59).

[51] T. A. Davis. "Algorithm 832: UMFPACK V4.3, an unsymmetric-pattern multifrontal method". In: *ACM Trans. Math. Softw.* 30.2 (June 2004), pp. 196–199. ISSN: 0098-3500 (cit. on p. 91).

[52] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Fundamentals of Algorithms 2. Philadelphia, PA, USA: SIAM, 2006. ISBN: 978-0-898716-13-9 (cit. on pp. 91, 93).

[53] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. "Inexact Newton methods". In: *SIAM J. Numer. Anal* 19.2 (1982), pp. 400–408 (cit. on p. 122).

[54] E. Deriaz and V. Perrier. "Orthogonal Helmholtz decomposition in arbitrary dimension using divergence-free and curl-free wavelets". In: *Appl. Comput. Harmon. Anal.* 26.2 (2009), pp. 249–269 (cit. on p. 56).

[55] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices*. Oxford, UK: Clarendon Press, 1989 (cit. on p. 91).

[56] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. Oxford, UK: Oxford University Press, 2005 (cit. on pp. 8, 9, 30, 33, 37, 46, 48, 93, 96, 97, 100, 101, 162).

[57] J. A. Fay. *Introduction to Fluid Mechanics*. Cambridge, Mass.: MIT Press, 1994, pp. XVIII+605 (cit. on pp. 38–41).

[58] C. L. Fefferman. *Existence and smoothness of the Navier-Stokes equation.* The Millennium Prize Problems. Available at http://www.claymath.org/millennium-problems/navier%E2%80%93stokes-equation. Peterborough, USA: Clay Mathematical Institute, 2000 (cit. on p. 1).

[59] F. Feitzinger, T. Hylla, and E. W. Sachs. "Inexact Kleinman–Newton method for Riccati equations". In: *SIAM J. Matrix Anal. Appl.* 31.2 (Mar. 2009), pp. 272–288 (cit. on pp. 38, 121, 122, 129–131, 143, 146, 162, 163).

[60] C. Foias, O. Manley, R. Rosa, and R. Temam. *Navier-Stokes Equations and Turbulence.* Cambridge: Cambridge Univ. Press, 2001, pp. XIV+347 (cit. on pp. 55, 56).

[61] A. V. Fursikov. "Stabilization for the 3D Navier-Stokes system by feedback boundary control". In: *Discrete Contin. Dyn. Syst.* 10.1/2 (2004), pp. 289–314 (cit. on p. 3).

[62] V. Girault and P. A. Raviart. *Finite Element Methods for Navier–Stokes Equations. Theory and Algorithms.* Berlin, Germany: Springer-Verlag, 1986 (cit. on p. 56).

[63] G. H. Golub and C. F. van Loan. *Matrix Computations.* 3rd. Baltimore: Johns Hopkins University Press, 1996 (cit. on pp. 67, 71, 90, 127).

[64] I. Granet. *Fluid Mechanics.* 4. ed. Englewood Cliffs, NJ: Prentice Hall, 1996, pp. XI+460 (cit. on p. 39).

[65] L. Grasedyck. "Existence of a low rank or $\mathscr{H}$-matrix approximant to the solution of a Sylvester equation". In: *Numer. Linear Algebra Appl.* 11.4 (2004), pp. 371–389. ISSN: 1099-1506 (cit. on p. 22).

[66] C.-H. Guo and A. J. Laub. "On a Newton-like method for solving algebraic Riccati equations". In: *SIAM J. Matrix Anal. Appl.* 21.2 (1999), pp. 694–698 (cit. on pp. 131, 132).

[67] W. Hackbusch. *Multi-Grid Methods and Applications.* Vol. 4. Springer Series in Computational Mathematics. Springer Verlag, 1985 (cit. on p. 102).

[68] J. Heiland. "Decoupling and Optimization of Differential-Algebraic Equations with Application in Flow Control". Dissertation. TU Berlin, 2014 (cit. on p. 3).

[69] J. Heiland and V. Mehrmann. "Distributed control of linearized Navier–Stokes equations via discretized input/output maps". In: *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 92.4 (2012), pp. 257–274 (cit. on p. 3).

[70] S. Hein. "MPC-LQG-Based Optimal Control of Parabolic PDEs". Available from http://archiv.tu-chemnitz.de/pub/2010/0013. Dissertation. Technische Universität Chemnitz, Feb. 2009 (cit. on p. 68).

[71] M. Heinkenschloss, D. C. Sorensen, and K. Sun. "Balanced truncation model reduction for a class of descriptor systems with application to the Oseen equations". In: *SIAM J. Sci. Comput.* 30.2 (2008), pp. 1038–1063 (cit. on pp. 54–60, 66, 67, 162).

[72] M. R. Hestenes and E. Stiefel. "Methods of conjugate gradients for solving linear systems". In: *J. Res. Nat. Bur. Standards* 49.6 (1952), pp. 409–436 (cit. on pp. 33, 102).

[73] J. G. Heywood, R. Rannacher, and S. Turek. "Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations". In: *Internat. J. Numer. Methods Fluids* 22.5 (1996), pp. 325–352. ISSN: 1097-0363 (cit. on p. 40).

[74] N. J. Higham. *Functions of Matrices: Theory and Computation.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008, pp. xx+425. ISBN: 978-0-898716-46-7 (cit. on pp. 12, 14–16).

[75] D. Hinrichsen and J. O'Halloran. "Limits of generalized state space systems under proportional and derivative feedback". In: *Math. Control Signals Systems* 10.2 (1997), pp. 97–124. ISSN: 0932-4194 (cit. on p. 28).

[76] M. Hinze and K. Kunisch. "Second order methods for boundary control of the instationary Navier–Stokes system". In: *Z. Angew. Math. Mech.* 84.3 (2004), pp. 171–187 (cit. on p. 155).

[77] L. Hogben, ed. *Handbook of Linear Algebra.* 2nd edition. Boca Raton, London, New York: Chapman & Hall/CRC, 2014 (cit. on pp. 19, 20).

[78] P. Hood and C. Taylor. "Navier–Stokes equations using mixed interpolation". In: *Finite Element Methods in Flow Problems.* Ed. by J. T. Oden, R. H. Gallagher, C. Taylor, and O. C. Zienkiewicz. University of Alabama in Huntsville Press, 1974, pp. 121–132 (cit. on pp. 9, 47).

[79] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge: Cambridge University Press, 1985 (cit. on pp. 10, 11, 90, 91).

[80] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* Philadelphia: SIAM, 1995 (cit. on pp. 30, 31, 122, 123, 125, 126, 131, 132).

[81] D. L. Kleinman. "On an iterative technique for Riccati equation computations". In: *IEEE Trans. Automat. Control* AC-13 (1968), pp. 114–115 (cit. on pp. xxv, 3, 64, 65, 129, 131).

[82] A. N. Krylov. "On the numerical solution of the equation by which the frequency of small oscillations is determined in technical problems". In: *Izv. Akad. Nauk SSSR.* Ser. Fiz.-Mat. 4 (1931), pp. 491–539 (cit. on p. 33).

[83] P. Kunkel and V. Mehrmann. *Differential-Algebraic Equations: Analysis and Numerical Solution.* Textbooks in Mathematics. EMS Publishing House, 2006 (cit. on pp. 3, 10, 27, 29).

[84] Patrick Kürschner. “Efficient Low-Rank Solution of Large-Scale Matrix Equations”. Dissertation. Apr. 2016. ISBN: 978-3-8440-4385-3. URL: http://hdl.handle.net/11858/00-001M-0000-0029-CE18-2 (cit. on pp. 21, 22, 28, 32, 79, 135).

[85] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford, UK: Oxford University Press, 1995 (cit. on pp. 3, 19, 20, 22–25, 64, 65).

[86] N. Lang, H. Mena, and J. Saak. “On the benefits of the LDL^T factorization for large-scale differential matrix equation solvers”. In: *Linear Algebra Appl.* 480 (2015), pp. 44–71 (cit. on p. 12).

[87] A. Lee. “Normal matrix pencils”. In: *Period. Math. Hungar.* 1.4 (1971), pp. 287–301 (cit. on p. 11).

[88] J.-R. Li and J. White. “Low rank solution of Lyapunov equations”. In: *SIAM J. Matrix Anal. Appl.* 24.1 (2002), pp. 260–280 (cit. on pp. 22, 134).

[89] T. Li, P. C.-Y. Weng, E. K.-w. Chu, and W.-W. Lin. “Large-scale Stein and Lyapunov equations, Smith method, and applications”. In: *Numer. Algorithms* 63.4 (2013), pp. 727–752. ISSN: 1017-1398; 1572-9265/e (cit. on p. 22).

[90] J. Liesen and Z. Strakoš. *Krylov Subspace Methods: Principles and Analysis*. Oxford: Oxford Univ. Press, 2013, pp. XV+391 (cit. on pp. 33, 96).

[91] A. Locatelli. *Optimal Control: An Introduction*. Basel: Birkhäuser, 2001 (cit. on pp. 3, 18, 26, 27, 131).

[92] 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”. In: *SIAM J. Sci. Comput.* 14.2 (1993), pp. 310–337 (cit. on p. 101).

[93] K.-A. Mardal and R. Winther. “Preconditioning discretizations of systems of partial differential equations”. In: *Numer. Linear Algebra Appl.* 18.1 (Jan. 2011), pp. 1–40 (cit. on p. 101).

[94] K. Morris and C. Navasca. “Solution of algebraic Riccati equatons arising in control of partial differential equations.” In: *Control and Boundary Analysis*. Ed. by J. P. Zolesio and J. Cagnol. Vol. 240. Lecture Notes in Pure Appl. Math. Chapman & Hall/CRC, Boca Raton, FL, 2005, pp. 257–280 (cit. on pp. 38, 143).

[95] M. F. Murphy, G. H. Golub, and A. J. Wathen. “A note on preconditioning for indefinite linear systems”. In: *SIAM J. Sci. Comput.* 21.6 (2000), pp. 1969–1972 (cit. on pp. 96, 97).

[96] A. Napov and Y. Notay. “An algebraic multigrid method with guaranteed convergence rate”. In: *SIAM J. Sci. Comput.* 34 (2012), A1079–A1109 (cit. on p. 102).

[97] I. Newton. *The Mathematical Papers of Isaac Newton*. Ed. by D. T. Whiteside. Vol. 2. 1667–1670. Cambridge University Press, 2008 (cit. on p. 30).

[98] P. A. Nguyen and J.-P. Raymond. "Boundary stabilization of the Navier–Stokes equations in the case of mixed boundary conditions". In: *SIAM J. Control Optim.* 53.5 (2015), pp. 3006–3039 (cit. on p. 3).

[99] Y. Notay. "An aggregation-based algebraic multigrid method". In: *ETNA* 37 (2010), pp. 123–146 (cit. on p. 102).

[100] Y. Notay. "Aggregation-based algebraic multigrid for convection-diffusion equations". In: *SIAM J. Sci. Comput.* 34 (2012), A2288–A2316 (cit. on p. 102).

[101] D. Peaceman and H. Rachford. "The numerical solution of elliptic and parabolic differential equations". In: *J. Soc. Indust. Appl. Math.* 3 (1955), pp. 28–41 (cit. on p. 31).

[102] T. Penzl. "Numerical solution of generalized Lyapunov equations". In: *Adv. Comput. Math.* 8.1-2 (1998), pp. 33–48. ISSN: 1019-7168 (cit. on p. 28).

[103] T. Penzl. "Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case". In: *Systems Control Lett.* 40.2 (2000), pp. 139–144. ISSN: 0167-6911 (cit. on p. 22).

[104] T. Penzl. LYAPACK *Users Guide.* Preprint SFB393/00-33. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`. Chemnitz, Germany: Technische Universität Chemnitz, 2000 (cit. on pp. 32, 78, 79, 142, 147).

[105] J.-P. Raymond. "Local Boundary Feedback Stabilization of the Navier–Stokes Equations". In: *Proceedings of Control Systems: Theory, Numerics and Applications.* Available from `http://pos.sissa.it`. Rome, 2005 (cit. on pp. 3, 54).

[106] J.-P. Raymond. "Feedback boundary stabilization of the two-dimensional Navier–Stokes equations". In: *SIAM J. Control Optim.* 45.3 (2006), pp. 790–828 (cit. on pp. 2, 3, 54–56, 61, 155, 161, 163).

[107] J.-P. Raymond. "Feedback boundary stabilization of the three-dimensional incombressible Navier–Stokes equations". In: *J.Math. Pures Appl. (9)* 87.6 (2007), pp. 627–669 (cit. on pp. 3, 54).

[108] J.-P. Raymond. "Stokes and Navier–Stokes equations with nonhomogeneous boundary conditions". In: *Ann. Inst. H. Poincaré – Anal. Non Linéare* 24.6 (2007), pp. 921–951. URL: `http://eudml.org/doc/78770` (cit. on pp. 3, 54, 61, 62).

[109] J.-P. Raymond and L. Thevenet. "Boundary feedback stabilization of the two dimensional Navier–Stokes equations with finite dimensional controllers". In: *Discrete Contin. Dyn. Syst.* 27.3 (2010), pp. 1159–1187. ISSN: 1078-0947; 1553-5231/e (cit. on p. 54).

[110] S. S. Rodrigues. *Feedback boundary stabilization to trajectories for 3D Navier-Stokes equations.* arXiv e-prints 1508.00829. Cornell University, 2015. URL: `http://arxiv.org/abs/1508.00829` (cit. on p. 3).

[111] J. W. Ruge and K. Stüben. "Algebraic multigrid (AMG)". In: *Multigrid Methods.* Ed. by Editor S. McCormich. Vol. 5. Frontiers in Applied Mathematics. SIAM, 1987, pp. 73–130 (cit. on p. 102).

[112] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. Manchester, UK: Manchester University Press, 1992. ISBN: 0719033861 (cit. on p. 16).

[113] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, 2003. ISBN: 0898715342 (cit. on pp. 30–33, 58, 59, 78, 93, 96, 102, 104).

[114] Y. Saad and M. H. Schultz. “GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems”. In: *SIAM J. Sci. and Stat. Comput.* 7.3 (1986), pp. 856–869 (cit. on pp. 33, 93).

[115] J. Saak. “Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction”. Available from `http://nbn-resolving.de/urn:nbn:de:bsz:ch1-200901642`. Dissertation. Technische Universität Chemnitz, July 2009 (cit. on pp. 67, 79, 163).

[116] J. Sabino. “Solution of large-scale Lyapunov equations via the block modified Smith method”. Available from `https://scholarship.rice.edu/handle/1911/20641`. PhD thesis. RICE University, 2007 (cit. on p. 164).

[117] A. A. Samarskij and P. N. Vabiščevič. *Computational Heat Transfer, Volume 1, Mathematical Modelling*. Chichester: John Wiley & Sons, 1995, pp. XII+406 (cit. on pp. 36, 37).

[118] W. E. Schiesser. *Numerical Methods of Lines*. New York: Academic Press, 1991. ISBN: 978-0-12-624130-3 (cit. on p. 46).

[119] B. K. Shivamoggi. *Theoretical Fluid Dynamics*. Dordrecht: Nijhoff, 1985, pp. XVIII +429 (cit. on pp. 38, 39).

[120] Valeria Simoncini. “Computational methods for linear matrix equations”. Available at `http://www.dm.unibo.it/~simoncin/`. Mar. 2013 (cit. on p. 21).

[121] D. C. Sorensen and Y. Zhou. *Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations*. Tech Reports TR02-07. CAAM, Rice University, Houston, 2002 (cit. on p. 22).

[122] M. Stoll and A. J. Wathen. “All-at-once solution of time-dependent Stokes control”. In: *J. Comp. Phys.* 232 (2013), pp. 498–515 (cit. on pp. 100, 102).

[123] T. Stykel. “Analysis and Numerical Solution of Generalized Lyapunov Equations”. Dissertation. TU Berlin, June 2002 (cit. on p. 72).

[124] K. Sun. “Model Order Reduction and Domain Decomposition for Large-Scale Dynamical Systems”. PhD thesis. RICE University, 2008 (cit. on pp. 105, 164).

[125] D. Titley-Peloquin, J. Pestana, and A. J. Wathen. “GMRES convergence bounds that depend on the right-hand-side vector”. In: *IMA J. Numer. Anal.* 34.2 (2014), pp. 462–479 (cit. on p. 96).

[126] L. N. Trefethen and M. Embree. *Spectra and Pseudospectra. The Behavior of Nonnormal Matrices and Operators*. Princeton, NJ: Princeton University Press, 2005 (cit. on pp. 11, 128).

[127] M. Voigt. “On Linear-Quadratic Optimal Control and Robustness of Differential-Algebraic-Systems”. Dissertation. OVGU Magdeburg, Germany, July 2015 (cit. on pp. 3, 29, 72).

[128] E. L. Wachspress. *The ADI Model Problem*. New York: Springer, 2013. ISBN: 978-1-4614-5121-1 (cit. on pp. 31, 32).

[129] A. J. Wathen. “Realistic eigenvalue bounds for the Galerkin mass matrix”. In: *IMA J. Numer. Anal.* 7 (1987), pp. 449–457 (cit. on p. 102).

[130] A. J. Wathen and T. Rees. “Chebyshev semi-iteration in preconditioning for problems including the mass matrix”. In: *ETNA, Electron. Trans. Numer. Anal.* 34 (2009), pp. 125–135 (cit. on p. 102).

[131] H. K. Weichelt. “Feedback-Stabilisierung von instationären, inkompressiblen Strömungen mit Riccati-Ansatz”. Available from http://www.mpi-magdeburg.mpg.de/mpcsc/projekte/optconfeestabmultiflow/data/101221_weichelt_diplomarbeit.pdf. Diplomarbeit. D-09107 Chemnitz: Technische Universität Chemnitz, Dec. 2010 (cit. on p. 3).

[132] H. K. Weichelt. “Navier–Stokes-Gleichung gekoppelt mit dem Transport von (reaktiven) Substanzen”. Available from http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa-63214. Abschlussbericht Modellierungsseminar. D-09107 Chemnitz: Technische Universität Chemnitz, Apr. 2010 (cit. on pp. 3, 43).

[133] J. Weickert. *Navier–Stokes equations as a differential-algebraic system*. Preprint SFB393/96-08. Chemnitz, Germany: Technische Universität Chemnitz, Aug. 1996 (cit. on pp. 51, 59).

[134] John F. Wendt, ed. *Computational Fluid Dynamics an Introduction*. 2. ed. Berlin: Springer, 1996, pp. XII+301 (cit. on p. 39).

[135] F. M. White. *Viscous Fluid Flow*. 2. ed. New York: McGraw-Hill, 1991, pp. XXI+614 (cit. on pp. 37, 39, 40).