THE UNIVERSITY *of* EDINBURGH

Efficient and Parallelizable Numerics for Optimal Control and Nonlinear Partial Differential Equations

Bernhard Heinzelreiter

Doctor of Philosophy

The University of Edinburgh

2025

To my fiancée,

Jennifer

Abstract

The analysis and optimal control of differential equations are of central interest in various fields of research as well as industrial applications. Given the limitations of analytical methods for solving related problems, one must often devise potent numerical approximations. In practice, attaining accurate solutions will necessitate large-scale approximations that require tailored methods to facilitate efficient computation. This thesis focuses on three topics related to the overarching theme of numerical methods for large-scale optimal control problems and the numerical analysis of complex nonlinear dynamical systems.

In the first part, we derive a new parallel-in-time approach for solving large-scale optimization problems constrained by time-dependent partial differential equations arising from fluid dynamics. The solver consists of a preconditioner used within a flexible GMRES iteration. The preconditioner involves the use of a block circulant approximation of the original matrices, enabling parallelization-in-time via the use of fast Fourier transforms. We devise bespoke matrix approximations which may be applied within this framework. These make use of block row and column operations, saddle-point approximations, commutator arguments for divergence and gradient terms, as well as inner solvers such as the Uzawa method, Chebyshev semi-iteration, and multigrid. Theoretical results underpin our strategy of applying a block circulant strategy, and numerical experiments demonstrate the effectiveness and robustness of our approach on Stokes and Oseen problems. Notably, satisfying results for the strong and weak scaling of our methods are provided within a fully parallel architecture. To our knowledge, this is the first parallelizable preconditioner for fluid flow problems with such a strong theoretical foundation.

While the first part of this thesis addresses optimal control for linear systems, which are well understood, the analysis and computation of nonlinear problems pose significantly greater challenges. Linearization methods for nonlinear systems can provide a partial solution by converting the problem into a linear system. This allows for the application of techniques designed for linear systems to be utilized for nonlinear systems as well. However, linearization methods currently lack a solid mathematical framework when applied to partial differential equations and in the context of their optimal control. In the second part of this thesis, we explore how the Carleman linearization—one particular class of linearization methods—can be extended to dynamical systems on infinite-dimensional Hilbert spaces with quadratic nonlinearities. We demonstrate the well-posedness and convergence of the truncated Carleman linearization under suitable assumptions on the dynamical system, which encompass many common parabolic semilinear partial differential equations. Upon discretization, we show that the total approximation error of the linearization decomposes into two independent components: the discretization error

and the linearization error. This decomposition yields convergence bounds of the linearization independent of the discretization. Furthermore, it motivates the use of non-standard structure-exploiting numerical methods. Finally, we verify the theoretical convergence results with numerical experiments.

Lastly, we consider the solution of saddle-point systems with a tree-based block structure. Such problems stem from optimal control problems that consist of small, distributed control problems linked by coupling on a limited number of degrees of freedom with a specific graph structure. As our key contribution, we propose several structure-exploiting preconditioners to be used during applications of the GMRES algorithm and analyze their properties. We adapt several concepts originating in the field of multigrid methods, obtaining a variety of adapted multi-level methods. We analyze the complexity of all algorithms, and derive a number of results on eigenvalues of the preconditioned system and convergence of iterative methods. We validate our theoretical findings through a range of numerical experiments, demonstrating the convergence and efficacy of the developed preconditioners.

Lay Summary

This thesis investigates innovative mathematical and computational techniques for addressing complex problems involving differential equations, which are fundamental tools for modeling a wide array of phenomena in real-world applications, such as oceanic flows, weather systems, and industrial processes. Obtaining solutions to these problems can be challenging and resource-intensive. Consequently, there is a need for bespoke methods that enable scientists and engineers to achieve fast and accurate solutions.

The first part of the thesis presents a new approach to make calculations faster and more robust, especially for problems that change over time, like those found in fluid dynamics. Specifically, we aim to find optimal actions to steer a system of fluids into a certain desired state. Instead of trying to solve a huge problem all at once, this method splits it into many smaller, simpler parts that can be worked on at the same time. As a result, supercomputers can be employed to significantly accelerate the solution process. Tests show that this approach works well even for very large and complicated problems, making it a promising idea for future research.

The second part tackles even more challenging situations where the equations are nonlinear, meaning they can behave unpredictably or chaotically. The thesis investigates a method called Carleman linearization, which transforms these complex problems into forms compatible with established computational methods originally intended for simpler cases. It provides new mathematical guarantees for when and how this method works, and demonstrates its effectiveness through practical examples.

Finally, the thesis addresses a special type of problem that arises when many smaller problems are linked together, such as in networked systems. Each subproblem is framed as an optimization problem, aiming to find the most effective and cost-efficient solutions. We introduce new techniques to solve these interconnected problems more efficiently, drawing on ideas from other areas of mathematics. The results show that these methods are both efficient and robust in theory and practice.

Overall, this work contributes new tools that enhance fast solution methods for central scientific and engineering problems, paving the way for advances in fields that rely on complex mathematical models.

Acknowledgements

Firstly, I would like to thank my supervisor, John W. Pearson, for his invaluable guidance throughout my PhD journey. He not only taught me how to conduct rigorous research but also provided constant encouragement and support, both academically and beyond, and always made time for inspiring discussions about our projects and mathematics. I am especially grateful for his advice, particularly for enabling me to attend conferences, helping me build my professional network, and for his efforts in promoting the outreach of my research. Furthermore, I would like to thank my collaborators, Christoph Hansknecht and Andreas Potschka, for their insightful discussions and their continuous support throughout my work.

I am profoundly grateful to my fiancée, Jennifer, for her unwavering support throughout my PhD. From encouraging me during the initial application and my move to Edinburgh, to joining me here and turning this into our shared adventure, she has been a constant source of strength. Her encouragement during moments of doubt, her patience with the long hours spent in the office, and her understanding of the time apart during conferences and workshops have all been invaluable.

I deeply appreciate my close family, Johann, Gudrun, Marco, and Christian, for being a steadfast support network throughout this journey. Especially, I would like to thank my father, who initially sparked my interest and joy for mathematics and has been a constant inspiration ever since. Furthermore, I would like to thank my international friends, with special gratitude to my best man, Moritz, who frequently checked in on my well-being from overseas and often made the long journey from Austria to visit me in Edinburgh.

Lastly, I am grateful to the friends I have met during my time in Edinburgh. In particular, I wish to thank Deven, Simon, and Tom for helping me switch off from mathematics and always being up for meetups at our local. I also thank the entire group of office JCMB 6207 for the lunches, the other walks around the campus, and the occasional English grammar debate, making the office such a fun and welcoming space. A special mention goes to my cat, Knödel, my ever-reliable mathematical proof-reader, who reminded me that treats are sometimes more important than proofs.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Bernhard Heinzelreiter

Contents

Figures, Tables, and Algorithms

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This thesis examines differential equations and their optimal control, with an emphasis on numerical analysis and numerical linear algebra. Differential equations form the backbone of models in numerous research fields and industrial applications. For example, they play pivotal roles in physics (e.g., fluid flow problems [1, 2, 3]), biology (see, e.g., [4, 5, 6]; with specific applications in physiology [7, 8], epidemiology [9], and cancer cell evolution [10]), as well as in machine learning (e.g., neural networks inspired by differential equations [11]). Beyond being invaluable tools for understanding and simulating real-world phenomena, these models can also be used to optimize the systems they replicate through optimal control, where the objective is to determine control actions that maximize or minimize a given performance measure. Examples can be found in physics (e.g., fluid flow control [12]) and engineering (e.g., topology optimization [12]), among many others. In practice, efficiently solving differential equations and their associated optimal control problems is crucial for fully realizing their potential as a modeling tool. For instance, differential equations must be solved efficiently to enable reliable predictions. Likewise, the optimization problems arising in optimal control have to be addressed in order to design and realize control strategies. This necessitates the development of computational methods for differential equations and related problems.

Given the limitations of analytical methods, one must often devise potent numerical approximations to address problems associated with differential equations and optimal control. In practice, attaining accurate solutions will necessitate large-scale approximations, such as those encountered when finite element methods are applied or when tackling large network problems. The approximations can often be described as the solution to systems of equations with large numbers of degrees of freedom. The difficulties in solving such systems stem not only from their sheer size but also from challenges such as ill-posedness, complex nonlinearities, and the curse of dimensionality, together with the practical requirement for computational efficiency. Recent developments, such as the rise of highly parallel infrastructures on high-performance computing (HPC) clusters and GPUs, along with advances in cutting-edge technologies like quantum computing, have transformed the landscape of computational science and significantly expanded the range of tractable problems, providing the means to address aforementioned large-scale problems. Nevertheless, these innovations require novel approaches to fully harness their capabilities.

In this work, we introduce advances in numerical methods for differential equations and their optimal control, leveraging the increased availability of computational resources through tailored efficient numerical methods. At the core of these methods lies the identification and exploitation of the underlying structural features of the problems. In this thesis, we specifically utilize block, recursive, tree-coupled, and factorizable forms, as well as certain structured nonlinearities. These approaches not only make it possible to solve previously intractable problems but are also highly efficient and inherently parallelizable, making them well-suited for modern HPC architectures.

1.1 Organization of the Thesis

The research conducted during the author's studies can be organized into three projects with the overarching theme of numerical methods for large-scale optimal control problems and the numerical analysis of complex nonlinear dynamical systems. This resulted in the following papers, of which two have been accepted for publication in relevant journals:

[13] Bernhard Heinzelreiter and John W. Pearson. "Diagonalization-based parallel-in-time preconditioners for instationary fluid flow control problems". In: *IMA Journal of Numerical Analysis* (2025). DOI: 10.1093/imanum/draf088,

[14] Bernhard Heinzelreiter and John W. Pearson. "Carleman linearization of parabolic PDEs: Well-posedness, convergence, and efficient numerical methods". arXiv:2510.00722 [math.NA]. 2025, and

[15] Christoph Hansknecht, Bernhard Heinzelreiter, John W. Pearson, and Andreas Potschka. "A framework for the solution of tree-coupled saddle-point systems". In: *Numerical Linear Algebra with Applications* (2025). DOI: 10.1002/nla.70038.

The thesis presents these branches of research and is structured as follows:

In Chapter 2, we address key concepts of differential equations, optimal control, and numerical linear algebra that form the foundation of the subsequent chapters.

Chapter 3 presents the work in [13], wherein we derive a novel parallel-in-time approach for solving large-scale optimization problems constrained by time-dependent partial differential equations (PDEs) arising from fluid dynamics. The solver consists of a preconditioner that enables parallelization-in-time through matrix approximation techniques. We establish theoretical results that support the proposed method and demonstrate its effectiveness and robustness through numerical experiments on Stokes and Oseen problems. The runtime scaling properties of our approach are demonstrated within a fully parallel architecture, verifying the method's efficacy. To our knowledge, this is the first parallelizable preconditioner for fluid flow control problems with such a strong theoretical foundation.

While the first part of this thesis addresses optimal control for linear systems, which are well understood, the analysis and computation of nonlinear problems pose significantly greater challenges. Linearization methods for nonlinear systems can provide a partial solution by converting the problem into a linear system. This allows for the application of techniques designed for

linear systems to be utilized for nonlinear systems as well. However, linearization methods currently lack a solid mathematical framework when applied to PDEs and in the context of their optimal control. In Chapter 4 we present the results from [14], and explore how the Carleman linearization—one particular class of linearization methods—can be extended to dynamical systems on infinite-dimensional Hilbert spaces with quadratic nonlinearities. We demonstrate the well-posedness and convergence of the truncated Carleman linearization for dynamical systems, which encompass many common parabolic semi-linear PDEs. Furthermore, we discuss how the theory facilitates non-standard structure-exploiting numerical methods, extending the applicability of the linearization technique to large-scale problems. This framework is the first to rigorously address the convergence of the linearization for a general class of nonlinear problems, including fluid flow, and represents, to our knowledge, the first application of theoretically justified low-rank methods to Carleman linearization.

Lastly, Chapter 5 considers the solution of saddle-point systems with a tree-based block structure and presents the findings of [15]. Such problems stem from optimal control problems that consist of small, distributed control problems linked by coupling on a limited number of degrees of freedom with a specific graph structure. We propose several structure-exploiting preconditioners to be used in iterative linear solvers and analyze their properties. We adapt concepts originating from the field of multigrid methods, obtaining a variety of adapted multi-level methods. This framework offers a unified perspective on existing direct solvers and newly developed iterative solvers for tree-coupled systems, facilitating direct comparison between methods and promoting advances in both algorithm design and theoretical understanding.

1.2 Contributions by the Author of the Thesis

Since the research presented in this thesis originated from collaborations with other researchers, this section highlights the contributions of the author of this thesis.

Chapters 3 and 4 are the result of collaborations between the author and John Pearson, with the author of this thesis serving as the lead scientist.

Chapter 5 presents joint work with Christoph Hansknecht, John Pearson, and Andreas Potschka. The primary contributions of this thesis' author lie in the algorithmic design, development, and testing of solver ideas, as well as designing and implementing numerical experiments. This includes contributing to the implementation of the numerical methods, as well as conducting comprehensive numerical experiments to demonstrate the efficiency of these methods and validate the theoretical findings. The author acknowledges that many of the theoretical results were primarily driven by Christoph Hansknecht.

Chapter 2

Background Material

In this chapter, we introduce a series of problems and methods that are crucial for the understanding of the topics covered in this thesis. Section 2.1 focuses on optimal control of both ordinary and partial differential equations, covering their theory and computational aspects. Optimal control problems form the central objectives of Chapters 3 and 5. The discussion extends to the role of nonlinearity and associated difficulties, which lays the basis and motivation for Chapter 4 in the scope of optimal control. Section 2.2 introduces fundamental concepts of numerical linear algebra, including linear solvers such as MINRES, GMRES, and FGMRES, as well as basic preconditioning techniques within the context of finite element methods. These tools are utilized in Chapters 3, 4, and 5.

2.1 Differential Equations and Optimal Control

In this thesis, we predominantly consider time-dependent (instationary) differential equations that take the form

$$\begin{aligned} y'(t) + \mathcal{L}(t, y(t)) &= f(t) \quad \text{for } t \in [0, T), \\ y(0) &= y_0, \end{aligned} \tag{2.1}$$

where $y(t) \in H$ denotes the state of the differential equation at time t and H is a Hilbert space, which may be either finite- or infinite-dimensional. It is noted that we occasionally use the term *state* to denote the entire trajectory y over the time interval. The operator

$$\mathcal{L} : [0, T) \times H \to H$$

is, in general, nonlinear and may depend explicitly on time. The final time T may be finite or infinite, i.e., $T \in (0, \infty]$. Such equations naturally arise in physical, biological, and chemical applications, among others. While the form (2.1) is suitable for a broad range of problems—particularly finite-dimensional ones—a significant class of problems requires unbounded operators $\mathcal{L}$, such as those encountered in time-dependent PDEs. In these cases, the concept of a weak formulation provides a rigorous framework for defining the differential equation. Specifically, this involves formulating the equation on a rigged Hilbert space (Gelfand triple) $V \hookrightarrow H \hookrightarrow V'$, with $\mathcal{L} : [0, T) \times V \to V'$. The vector space V' denotes the topological dual of V. The objective in the weak formulation is then to seek a function $y \in L^2(0, T; V)$ that satisfies an abstract equation

$e(y, f) = 0$, where $e : L^2(0, T; V) \times L^2(0, T; V') \to L^2(0, T; V')$. Thus, the problem is reformulated as an equation on Banach spaces. A more detailed discussion of the weak formulation is provided in later sections of this thesis where its rigorous analysis becomes relevant. It should be noted that, in the finite-dimensional case and under suitable assumptions on f, the weak formulation is equivalent to the form (2.1); thus, the weak form serves as a generalization.

Another class of differential equations considered here are stationary, meaning they do not involve time-dependence. We specifically consider elliptic problems of the form

$$\mathcal{L}(y)(x) = f(x),$$

where $\mathcal{L}$ is an elliptic operator defined on a bounded domain $\Omega \subset \mathbb{R}^d$ of spatial dimension $d \in \mathbb{N}$. The functions y and f are defined on Ω, and the problem is equipped with appropriate boundary conditions. As with the time-dependent equation, the stationary problem typically requires a weak formulation, which leads to an abstract problem: find $y \in V$ such that $e(y, f) = 0$ for $e : V \times V' \to V'$. Therefore, both the stationary and instationary cases can be formulated as equations on Banach spaces, a perspective that allows the upcoming discussion to remain general with respect to the types of equations considered.

The study of both stationary and instationary equations in this thesis is motivated by their central role in optimal control, where the goal is to influence the evolution of the state $y(t)$ (or the configuration $y(x)$ in the stationary case) through controls in order to optimize a given cost functional, subject to the constraint of the differential equation. For a detailed discussion, we refer to [16, 17] for the topic of PDEs, and to [18, 19, 20, 21] for optimal control theory.

2.1.1 Optimal Control Problems

We will broadly follow the introduction to optimal control provided in [21], where such problems are approached as optimization problems in Banach spaces. A wide range of optimal control problems can be formulated as the optimization problem

$$\begin{aligned} \min_{y \in Y,\, u \in U} \quad & J(y, u) \\ \text{s.t.} \quad & e(y, u) = 0. \end{aligned} \tag{2.2}$$

In this setting, the sets Y and U are function spaces and—in accordance with the introduction to differential equations—are assumed to be Banach and Hilbert spaces, respectively. An element $y \in Y$ is called the *state* of the system and $u \in U$ is the *control*. The objective $J : Y \times U \to \mathbb{R}$ is a functional which assigns a penalization to each state and control. Lastly, the equation $e(y, u) = 0$ describes a differential equation with the operator $e : Y \times U \to Z$, where Z is a Banach space as well. Specifically, if this equation refers to a PDE, the optimal control problem is referred to as a PDE-constrained optimization problem. The precise choices of the spaces Y and U depends on the underlying differential equation, since they are naturally chosen in such a way that existence and uniqueness of solutions are guaranteed. We additionally require that the assumptions given

in [21, p. 55] are fulfilled. Those assumptions include that $e(y,u) = 0$ has a bounded solution operator, i.e., there is a mapping $y : U \to Y$ such that $e(y(u),u) = 0$ for all $u \in U$. This assumption refers to the well-posedness of the differential equation constraint. Furthermore, it is assumed that e is continuous under weak convergence, and that J is sequentially weakly lower semicontinuous. These assumption are fulfilled for the optimal control problems considered in this thesis and guarantee existence of a minimizer to problem (2.2), cf. Theorem 1.45 in [21, p. 55].

A concrete example for the problem (2.2) is the distributed heat control problem with homogeneous Dirichlet boundary conditions. Let Ω be a bounded Lipschitz domain and $T \in (0,\infty]$. The evolution of heat can be modelled by the PDE

$$\begin{aligned} \frac{\partial y}{\partial t}(t,x) - \Delta y(t,x) &= u(t,x) && \text{in } (0,T)\times\Omega, \\ y(0,x) &= y_0(x) && \text{on } \Omega, \\ y(t,x) &= 0 && \text{on } (0,T)\times\partial\Omega, \end{aligned} \tag{2.3}$$

where y denotes the temperature field and u a heat source. In order to obtain existence and uniqueness of the solution to the PDE, we consider a weak formulation of this problem (cf. [21, p. 41]) with the function spaces $Y = L^2(0,T;H_0^1(\Omega))$, $U = L^2(0,T;L^2(\Omega))$, where $H^1(\Omega)$ denotes the Sobolev space with bounded first derivatives and homogeneous boundary conditions. The PDE can then be written as an equation

$$e(y,u) = \mathcal{E}y + \mathcal{B}u - g = 0, \tag{2.4}$$

where $Z = L^2(0,T;(H_0^1(\Omega))')$ and $\mathcal{E} : Y \to Z$ and $\mathcal{B} : U \to Z$ are bounded operators. The right-hand side $g \in Z$ incorporates the initial condition and boundary conditions. A typical objective is to obtain a temperature field y close to a given target temperature y_d. This can be described by the objective function

$$J(y,u) = \frac{1}{2}\int_0^T \int_\Omega \|y(t,x) - y_d(t,x)\|^2 \,\mathrm{d}x\mathrm{d}t + \frac{\beta}{2}\int_0^T \int_\Omega \|u(t,x)\|^2 \,\mathrm{d}x\mathrm{d}t,$$

which can be written in the (more general) form

$$J(y,u) = \frac{1}{2}\left\|\mathcal{Q}\,(y - y_d)\right\|_H^2 + \frac{\beta}{2}\|u\|_U^2 \tag{2.5}$$

with $\mathcal{Q} : Y \to H$. The space H denotes a reflexive Banach space, which in our example of heat control is $H = L^2(0,T;L^2(\Omega))$. This concludes the definition of all necessary parts of (2.2). The parameter $\beta > 0$ is a modelling parameter and is referred to as the regularization parameter. It determines how strongly a control is penalized. Large values of β correspond to a high cost

of applying control, such as incurring significant expense for heat sources in the heat control example. Although this makes the control strategy more economical, it also means the state y deviates further from the desired state y_d. On the other hand, small values of β allow for stronger control actions, which may increase costs but result in a state y that more closely matches y_d.

Having outlined the general form (2.2) of the optimal control problems of interest, we now address the question of how minimizers y and u can be determined.

2.1.2 First-Order Optimality Conditions

The first-order optimality conditions are necessary conditions and therefore conditions for identifying candidates for minimizers. Moreover, they form the basis of various optimization algorithms, so-called first-order methods, underscoring their importance. We introduce the additional assumptions that J and e are continuously Fréchet differentiable and that $e_y(y(u), u) : Y \to Z$ has a bounded inverse for all $u \in U$. In order to derive the first-order optimality conditions of system (2.2), we introduce the Lagrangian of the minimization problem, which we define as

$$\mathcal{L}(y, u, p) = J(y, u) + \langle p, e(y, u) \rangle_{Z'}, \tag{2.6}$$

where $\mathcal{L} : Y \times U \times Z' \to \mathbb{R}$ and $\langle \cdot, \cdot \rangle_{Z'}$ is the duality pairing of Z'. Now, assume that y and u are minimizers. It holds that all derivatives of $\mathcal{L}$ are zero, i.e.,

$$e(y, u) = 0 \quad \text{(state equation)}, \tag{2.7a}$$
$$J_y(y, u) + e_y(y, u)^* p = 0 \quad \text{(adjoint equation)}, \tag{2.7b}$$
$$J_u(y, u) + e_u(y, u)^* p = 0 \quad \text{(gradient equation)}, \tag{2.7c}$$

where $\mathcal{D}^* : Y' \to X'$ denotes the adjoint operator of an operator $\mathcal{D} : X \to Y$ between Banach spaces X and Y. As denoted in the equations, (2.7a) is called the *state equation*, (2.7b) the *adjoint equation*, and lastly (2.7c) the *gradient equation*. Solving these coupled equations gives candidates for minimizers of the optimization problem. In the case where e is linear (i.e., of the form (2.4)) and J is quadratic (i.e., of the form (2.5)), or after linearizing the system (e.g., via Newton's method), the system of equations (2.7) can be expressed as the block-operator system

$$\left(\begin{array}{cc:c} \mathcal{Q}^*\mathcal{Q} & 0 & \mathcal{E}^* \\ 0 & \beta\mathcal{I} & \mathcal{B}^* \\ \hdashline \mathcal{E} & \mathcal{B} & 0 \end{array}\right) \begin{pmatrix} y \\ u \\ p \end{pmatrix} = \begin{pmatrix} \mathcal{Q}^*\mathcal{Q}y_d \\ 0 \\ g \end{pmatrix}, \tag{2.8}$$

where $\mathcal{I}$ denotes the canonical embedding $\mathcal{I} : B \to B^*$. The partitioning indicated by the dashed lines reveals the saddle-point problem structure of the optimality system. Saddle-point problems form a fundamental component of optimal control problems and their solution poses pivotal challenges.

Since equations (2.7) are rather abstract, we show what these conditions can mean in the scope of the example of distributed heat control. For these equations the state equation remains unchanged and is the weak formulation of (2.3). The adjoint equation results in the weak form of the PDE

$$\begin{aligned} \frac{\partial p}{\partial t}(t,x) + \Delta p(t,x) &= y(t,x) - y_d(t,x) && \text{in } (0,T)\times\Omega, \\ p(T,x) &= 0 && \text{on } \Omega, \\ p(t,x) &= 0 && \text{on } (0,T)\times\partial\Omega. \end{aligned} \tag{2.9}$$

It is noted that this equation is the backward-in-time heat equation, which means that instead of an initial condition a terminal condition at $t = T$ has to be fulfilled. Finally, the gradient equation translates to

$$\beta u(t,x) = p(t,x) \quad \text{in } (0,T)\times\Omega. \tag{2.10}$$

Hence, candidates for minimizers are determined by solving the coupled PDEs (2.3), (2.9), and (2.10). As this system involves both initial and terminal conditions, standard time integration methods used for evolution equations are not applicable. Instead, the system must be solved simultaneously in space and time (or *all-at-once*), which can greatly increase the problem size of the systems that have to be solved at each step of the algorithmic process. This inherent complexity poses a central challenge in time-dependent PDE-constrained optimization problems, as it leads to large-scale saddle-point systems upon discretization.

Instead of considering an all-at-once system, one can also use a gradient descent method for solving a reduced optimization problem. Since the constraint $e(y,u) = 0$ has a solution operator, the constraint can be substituted into the cost functional, which rephrases the problem as an unconstrained optimization problem of the form

$$\min_{u\in U} \quad J(y(u),u).$$

This allows one to compute the gradient of $J(y(\hat{u}),\hat{u})$ for a guess $\hat{u}$, and then use it to improve the current guess by the gradient descent method. The resulting equations are closely linked to the first-order optimality conditions and require solving (3.3) and (3.4) separately, rather than all-at-once systems. Specifically, the gradient direction can be achieved by first solving the state equation using the current control estimate $\hat{u}$, which yields a state $\hat{y}$. This state is then used to solve the adjoint equation, resulting in the adjoint variable $\hat{p}$. With $\hat{p}$, a search direction can be computed. For example, in the heat control scenario, the approximation of the control can be updated using $\hat{u} \leftarrow \hat{u} + s(\beta\hat{u} - \hat{p})$, where s is an appropriately chosen step size. The optimization problem is, therefore, reduced to solving evolution equations. This approach is often referred to as the reduced gradient method. While this approach circumvents the challenges of dealing with an all-at-once system, it requires repeatedly solving potentially large systems of differential equations and may introduce numerical issues such as slow or even lack of convergence.

Previously, we derived optimality conditions and solution techniques in Banach spaces. The resulting equations are therefore continuous in time and—when dealing with PDEs—undiscretized in space. To obtain an optimal solution, it is often necessary to use numerical methods and discretize the equations, as analytical solutions are typically unattainable. This approach is known as *optimize-then-discretize* (OTD), reflecting the sequence of deriving optimality conditions before discretizing. Conversely, the *discretize-then-optimize* (DTO) approach first discretizes the problem and then applies optimization techniques to the discrete system. These two strategies can produce different results unless the discretization is chosen carefully, and each has its advantages and disadvantages. In the OTD approach, the solution corresponds to the original optimal control problem, and the discretization of the state and adjoint equations can be selected independently of each other. Furthermore, OTD allows one to apply standard convergence and stability results for time integration and finite element methods to achieve guaranteed error bounds, which might not be given in the DTO case. For example, [22] showcases that the OTD approach can offer improved convergence for certain PDE-constrained optimization problems. However, OTD can introduce disadvantages; for instance, the optimality system may become asymmetric, even if it is symmetric in the infinite-dimensional setting, which affects the choice of numerical solvers. In contrast, the DTO approach tends to preserve symmetry in the resulting systems and can be easier to implement in optimization algorithms such as interior point methods. One way to circumvent the need to choose between OTD and DTO is to use commutative discretizations, which yield the same discretized system regardless of the approach. Both methods have their strengths and weaknesses, and the choice between them depends on the specific application. Accordingly, both are utilized in this thesis. For a brief discussion and comparison of these methods, see [21, pp. 160–164].

Let us showcase a discretization of an optimality system by the example of heat control. If the problem is discretized by a finite element method in space and by the backward Euler method in time, the OTD approach applied to the abstract optimality conditions (2.8) results in a linear system of the form

$$\begin{pmatrix} I_{n_t} \otimes M & 0 & \frac{1}{\tau}E^T \otimes M + I_{n_t} \otimes L_a \\ 0 & \beta I_{n_t} \otimes M & I_{n_t} \otimes M \\ \frac{1}{\tau}E \otimes M + I_{n_t} \otimes L & I_{n_t} \otimes M & 0 \end{pmatrix} \begin{pmatrix} \underline{y} \\ \underline{u} \\ \underline{p} \end{pmatrix} = \begin{pmatrix} \underline{y}_d \\ 0 \\ 0 \end{pmatrix} \tag{2.11}$$

In this system, n_t is the number of time steps and τ is the time step size. The matrix I_{n_t} denotes the identity matrix of size n_t, while E has ones on the diagonal and negative ones on the lower off-diagonal. The matrix M refers to the mass matrix, L to the discretization of the spatial differential operator (specifically, the negative diffusion term in our case), and L_a denotes the discretization of the adjoint operator. For our particular problem, $L_a = L^T$, but this equality may not hold for other cases, as previously discussed. The underlined variables represent the discretized functions and right-hand side compoments.

2.1.3 Nonlinear Equations

The preceding theory in the abstract setting encompasses both linear and nonlinear equations. However, the analysis and solution process for optimal control problems become significantly more challenging when transitioning from linear to general nonlinear equations. In the following, we highlight key aspects that must be considered when addressing nonlinearities, emphasizing the increased complexity inherent in such problems.

Linear constraints e and quadratic cost functionals J guarantee the uniqueness of a minimizer, and the first-order optimality conditions become both necessary and sufficient, forming a system of linear equations. Furthermore, the reduced gradient method converges if the step size is chosen sufficiently small. However, when dealing with nonlinear operators, these properties may no longer hold. The first-order optimality system then becomes a system of coupled nonlinear equations, making their solution more difficult, as iterative methods such as Newton's method or SQP (see, e.g., [21, pp. 140 ff.]) must be employed. Additionally, candidates for minimizers may not actually be minimizers. When solving the nonlinear system of first-order conditions or applying a reduced gradient method, convergence cannot be guaranteed without sufficiently good initial guesses. Computing such guesses can itself be a challenging task.

Another aspect, which signifies the complexity associated with the shift from linear to nonlinear problems, is optimal feedback control design for time-dependent control problems, see, e.g., [20, p. xviii], [19, pp. 132–174], and [23]. For feedback control problems, the primary objective remains to find minimizers of (2.2). However, instead of solving the problem for a specific set of data, the goal is to identify a mapping π such that the optimal control can be expressed as a function of the state, i.e., $u(t) = \pi(y(t))$, or even as a function of just an observation of the state, $u(t) = \pi(\mathcal{C}(y(t)))$. Feedback controls have practical advantages, including increased robustness to observation errors and model inaccuracies. This problem is closely related to solving the Hamilton–Jacobi–Bellman (HJB) equation, see, e.g., [20, pp. 137 ff.] and [19, p. 150 f.]. For linear time-dependent differential equations, such problems are well-studied in the literature, often reducing to solving the operator equation known as the Riccati equation. However, when nonlinearities are introduced, the problem becomes significantly more challenging, with a substantial increase in dimensionality.

Furthermore, nonlinearities can have adversarial consequences for the discretization. When iterative nonlinear solvers are employed to address the nonlinearity, the matrices involved in the discretized optimality conditions—such as those in (2.11)—may vary with each iteration, resulting in additional computational overhead. Moreover, linear problems may exhibit exploitable structures that facilitate the design of efficient numerical methods. However, these structures can be foiled when nonlinearities are present in the equations. For example, system (2.11) possesses a Kronecker product structure. This allows the matrices for spatial discretizations to be constructed once, rather than at every time step, and enables the development of fast parallelizable numerical methods, as will be discussed in Chapter 3.

In summary, transitioning from linear to nonlinear optimization entails more analytical and computational difficulties. The lack of global guarantees and the need for sophisticated solution strategies underscore the importance of careful analysis, good initialization, tailored linearization techniques, and numerical methods for nonlinear problems.

2.2 Iterative Methods for Linear Equations

The numerical solution of differential equations and optimal control problems—especially those that involve PDEs—often narrows down to the solution of sparse large-scale linear systems. These linear systems arise from the discretization of all-at-once systems of evolutionary equations or first-order optimality conditions. The sparsity usually allows for the explicit construction of the required matrices and vectors. However, black-box solvers such as direct solvers, available in libraries like SuperLU [24], UMFPACK [25], and MUMPS [26], frequently fail on these systems due to their excessive problem size and the resulting substantial memory requirements. Designing an effective solver requires careful consideration of the characteristics of the underlying structure of the problem and the differential equations. Iterative solvers, especially Krylov subspace methods, provide a framework for integrating information about the system into the solver through preconditioning. When coupled with efficient preconditioning techniques, they have overcome the shortcomings of direct solvers and have been applied successfully to an array of large-scale problems over the past decades. In the following sections, we present iterative solvers and preconditioning techniques that are crucial for the upcoming chapters and form the basis for a large set of methods used for partial differential equations and optimal control. This is broadly based on [27] and citations therein.

Let the linear system we consider be given by

$$\mathcal{A}x = f,$$

where $\mathcal{A} \in \mathbb{R}^{n\times n}$ is a sparse invertible square matrix, and x, $f \in \mathbb{R}^n$ are vectors. Note that the assumption of $\mathcal{A}$ being invertible is not essential, as some iterative methods can solve singular linear systems. Moreover, these linear problems can be further generalized to complex systems in $\mathbb{C}$. The following ideas often extend naturally to such cases by replacing the transpose with the conjugate transpose and using the appropriate scalar products. The general form of an iterative solver considered in this work can be formulated as

$$x^{(k+1)} = \Phi(k, f, x^{(k)}), \quad x^{(0)} = x_0. \tag{2.12}$$

The vector x_0 denotes the initial guess. The function $\Phi : \mathbb{N} \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ maps each iterate to the next; Φ may be linear, but it is not required to be. Furthermore, we define the residual at iterate k as

$$r^{(k)} = f - \mathcal{A}x^{(k)}.$$

An iterative method is said to converge if $\|r^{(k)}\| \to 0$ as $k \to \infty$. In practice, an iterative solver is terminated once a specified convergence criterion is met, typically when the residual satisfies $\|r^{(k)}\| < \varepsilon$ for some suitably small $\varepsilon > 0$. The appropriate choice of norm for measuring the residual can depend on both the specific problem and the linear solver used.

If $\mathcal{A}$ is ill-conditioned, iterative solvers may converge slowly and encounter numerical difficulties. In this situation, it is beneficial to precondition $\mathcal{A}$ and solve an equivalent linear system. An invertible matrix $\mathcal{P} \in \mathbb{R}^{n\times n}$ that approximates $\mathcal{A}$ and whose inverse can be applied efficiently is called a preconditioner. The goal is to solve the modified linear system

$$\mathcal{P}^{-1}\mathcal{A}x = \mathcal{P}^{-1}f,$$

which is referred to as left-preconditioning, or

$$\mathcal{A}\mathcal{P}^{-1}\tilde{x} = f, \quad x = \mathcal{P}^{-1}\tilde{x},$$

referred to as right-preconditioning. Either case can be handled as an alternative linear system with matrices $\mathcal{P}^{-1}\mathcal{A}$ and $\mathcal{A}\mathcal{P}^{-1}$; however, for most methods, it is beneficial to explicitly take the preconditioner into account. Most methods are discussed for the preconditioned case, which covers the unpreconditioned case by setting $\mathcal{P} = I$, where $I \in \mathbb{R}^{n\times n}$ is the identity matrix.

2.2.1 MINRES

The first presented solver is called *minimal residual method* (MINRES), which was originally introduced in [28]. This discussion is based on insights from [27]. MINRES is designed to solve systems with symmetric $\mathcal{A}$ and symmetric positive-definite (SPD) $\mathcal{P}$, and belongs to the class of Krylov subspace methods. A Krylov subspace method is an iterative technique that incrementally constructs a basis for the Krylov subspace and seeks an approximate solution within this space, shifted by the initial guess. The kth-order Krylov supspace of a matrix–vector tuple $\mathcal{M}$ and g is defined as the vector space

$$\mathcal{K}_k(\mathcal{M}, g) \coloneqq \operatorname{span}\left\{g, \mathcal{M}g, \mathcal{M}^2g, \ldots, \mathcal{M}^{k-1}g\right\}.$$

It is noted that solely the matrix–vector product of $\mathcal{M}$ is necessary for the construction of $\mathcal{K}_k(\mathcal{M}, g)$. This enables one to design a non-intrusive solver, i.e., a method that does not require an explicit construction of $\mathcal{A}$ or $\mathcal{P}^{-1}$ but only their actions. This becomes useful if $\mathcal{A}$, for example, has a block-matrix structure with repetitive subblocks, and in the case of preconditioning, prevents an explicit construction of the inverse of $\mathcal{P}$.

The goal of (left-)preconditioned MINRES is now to find a solution that minimizes a norm of the residual over the shifted space

$$V_k = x_0 + \mathcal{K}_k(\mathcal{P}^{-1}\mathcal{A}, \mathcal{P}^{-1}f),$$

Algorithm 2.1: The left-preconditioned MINRES method [27, p. 192]

Function MINRES($\mathcal{A}$, f, $\mathcal{P}$, $x^{(0)}$)**:**

Input : Matrix $\mathcal{A} \in \mathbb{R}^{n\times n}$
Right-hand side $f \in \mathbb{R}^n$
Preconditioner $\mathcal{P} \in \mathbb{R}^{n\times n}$
Initial guess $x^{(0)} \in \mathbb{R}^n$

Output: Approximation x solving $\mathcal{A}x = f$ to a certain tolerance

1 Initialize $v^{(0)} \leftarrow 0$, $w^{(0)} \leftarrow 0$, $w^{(1)} \leftarrow 0$, $\gamma_0 \leftarrow 0$
2 Compute $v^{(1)} \leftarrow f - \mathcal{A}x^{(0)}$
3 Solve $\mathcal{P}z^{(1)} = v^{(1)}$, set $\gamma_1 \leftarrow \sqrt{\langle z^{(1)}, v^{(1)}\rangle}$
4 Set $\eta \leftarrow \gamma_1$, $s_0 \leftarrow s_1 \leftarrow 0$, $c_0 \leftarrow c_1 \leftarrow 1$
5 **for** $k = 1$ **to** *convergence* **do**
6 $z^{(k)} \leftarrow z^{(k)}/\gamma_k$
7 $\delta_k \leftarrow \langle \mathcal{A}z^{(k)}, z^{(k)}\rangle$
8 $v^{(k+1)} \leftarrow \mathcal{A}z^{(k)} - (\delta_k/\gamma_k)v^{(k)} - (\gamma_k/\gamma_{k-1})v^{(k-1)}$
9 Solve $\mathcal{P}z^{(k+1)} = v^{(k+1)}$
10 $\gamma_{k+1} \leftarrow \sqrt{\langle z^{(k+1)}, v^{(k+1)}\rangle}$
11 $\alpha_0 \leftarrow c_k\delta_k - c_{k-1}s_k\gamma_k$
12 $\alpha_1 \leftarrow \sqrt{\alpha_0^2 + \gamma_{k+1}^2}$
13 $\alpha_2 \leftarrow s_k\delta_k + c_{k-1}c_k\gamma_k$
14 $\alpha_3 \leftarrow s_{k-1}\gamma_k$
15 $c_{k+1} \leftarrow \alpha_0/\alpha_1$
16 $s_{k+1} \leftarrow \gamma_{k+1}/\alpha_1$
17 $w^{(k+1)} \leftarrow (z^{(k)} - \alpha_3 w^{(k-1)} - \alpha_2 w^{(k)})/\alpha_1$
18 $x^{(k)} \leftarrow x^{(k-1)} + c_{k+1}\eta w^{(k+1)}$
19 $\eta \leftarrow -s_{k+1}\eta$
20 **return** $x^{(k)}$

i.e., the kth iterate translates to

$$x^{(k)} = \operatorname{argmin}_{x\in V_k} \|f - \mathcal{A}x\|_{\mathcal{P}^{-1}}, \tag{2.13}$$

where $\|\cdot\|_{\mathcal{P}^{-1}} \coloneqq \|\mathcal{P}^{-1/2}\cdot\|$. Due to the symmetry of $\mathcal{A}$ and the SPD property of $\mathcal{P}$, an orthonormal basis of V_k can be constructed through the so-called Lanczos iteration. This way, the optimization problem (2.13) can be solved cheaply without keeping track of the basis vectors. Algorithm 2.1 shows the complete procedure of MINRES, where line 8 refers to the Lanczos iterate, line 11 and the following to the sequential QR factorization, and line 15 and 16 to the necessary Givens rotation.

The convergence of MINRES can be estimated by the eigenvalue distribution of the preconditioned matrix, i.e., the set of all eigenvalues $\lambda(\mathcal{P}^{-1}\mathcal{A})$. In particular, the following inequality holds

$$\frac{\|r^{(k)}\|_{\mathcal{P}^{-1}}}{\|r^{(0)}\|_{\mathcal{P}^{-1}}} \leq \min_{p_k\in\Pi_k,\, p_k(0)=1} \max_{\mu\in\lambda(\mathcal{P}^{-1}\mathcal{A})} |p_k(\mu)|\,, \tag{2.14}$$

where Π_k denotes the set of all polynomials of degree at most k (cf. [27, 210 f.]). Thus, the convergence rate of MINRES is determined by how well a polynomial can be fitted to the set of eigenvalues. The fewer degrees required for a well-fitted polynomial, the better the convergence. In particular, clustering of eigenvalues leads to improved convergence. Therefore, an effective preconditioner should often be designed to cluster the eigenvalues of $\mathcal{A}$. Furthermore, let the eigenvalues of $\mathcal{P}^{-1}\mathcal{A}$ be contained in the intervals $[-a,-b]\cup[c,d]$, where a, b, c, $d>0$ with $a-b=d-c$. Then, the convergence bound can be refined to

$$\frac{\|r^{(2k)}\|_{\mathcal{P}^{-1}}}{\|r^{(0)}\|_{\mathcal{P}^{-1}}} \leq 2\left(\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}\right)^k.$$

This estimate further emphasizes that tighter intervals $[-a,-b]$ and $[c,d]$, corresponding to improved eigenvalue clustering, lead to faster convergence.

2.2.2 GMRES

A generalization of MINRES for nonsymmetric $\mathcal{A}$ and general invertible $\mathcal{P}$ was developed in [29], known as the *generalized minimal residual* (GMRES) method. GMRES is also a Krylov subspace method and applies the same principle to minimize the objective in problem (2.13). However, since $\mathcal{A}$ is not necessarily symmetric, the Lanczos method can no longer be used. Instead, GMRES employs Arnoldi's method, which constructs an orthonormal basis using the Gram–Schmidt process. This approach is more computationally expensive than the orthonormalization in MINRES and requires storing and solving for a Hessenberg matrix H_m whose size grows with the number of iterations. Moreover, it is necessary to store a vector of size n for each iteration throughout the process, which can be prohibitive for excessive problem sizes and large numbers of iterations. Consequently, a maximum number of iterations m is typically specified. If convergence is not achieved within m iterations, GMRES can be restarted using the current solution as the initial guess. The procedure is summarized in Algorithm 2.2. To facilitate the transition to the subsequent solver, we only outline the right-preconditioned case of GMRES. The left-preconditioned version of the method is structured similarly but omitted in this introduction.

The convergence behavior of GMRES is similar to that of MINRES. If the preconditioned matrix is diagonalizable, i.e., there is an invertible V and a diagonal matrix Λ such that $\mathcal{A}\mathcal{P}^{-1}=V\Lambda V^{-1}$, then for the kth iterate, it holds that

$$\frac{\|r^{(k)}\|_2}{\|r^{(0)}\|_2} \leq \kappa(V) \min_{p_k\in\Pi_k,\, p_k(0)=1} \max_{\mu\in\lambda(\mathcal{A}\mathcal{P}^{-1})} |p_k(\mu)|\,.$$

Even though this convergence bound is only applicable to diagonalizable linear systems and is hard to analyze further for general matrices, convergence and, therefore, termination of GMRES is always guaranteed at the nth iterate if $m=n$ under the assumption of exact arithmetic. In practice, however, the choice $m=n$ would defy the purpose of using GMRES as a solver for large-scale sparse systems, as this would require storing and solving for a dense Hessenberg

Algorithm 2.2: The right-preconditioned GMRES method [30]

Function GMRES($\mathcal{A}$, f, $\mathcal{P}$, $x^{(0)}$, m)**:**

Input : Matrix $\mathcal{A} \in \mathbb{R}^{n\times n}$
Right-hand side $f \in \mathbb{R}^n$
Preconditioner $\mathcal{P} \in \mathbb{R}^{n\times n}$
Initial guess $x^{(0)} \in \mathbb{R}^n$
Number of iterations per restarted iteration $m \in \mathbb{N}$

Output: Approximation x solving $\mathcal{A}x = f$ to a certain tolerance

1 Initialize $H_m \in \mathbb{R}^{(m+1)\times m}$ with zeros
2 Compute $r^{(0)} \leftarrow f - \mathcal{A}x^{(0)}$, $\beta \leftarrow \|r^{(0)}\|_2$, $v_1 \leftarrow r^{(0)}/\beta$
3 **for** $k = 1$ **to** m **do**
4 Solve $\mathcal{P}z^{(k)} = v^{(k)}$
5 Compute $w \leftarrow \mathcal{A}z^{(k)}$
6 **for** $i \leftarrow 1$ **to** k **do**
7 $h_{i,k} \leftarrow \langle w, v^{(i)}\rangle$
8 $w \leftarrow w - h_{i,k}v^{(i)}$
9 **end**
10 $h_{k+1,k} \leftarrow \|w\|_2$
11 $v^{(k+1)} \leftarrow w/h_{k+1,k}$
12 **end**
13 Let $V_m \leftarrow (v^{(1)}, \ldots, v^{(m)})$
14 Compute $y^{(m)} \leftarrow \arg\min_y \|\beta e_1 - H_m y\|_2$ with $e_1 = (1, 0, \ldots, 0)^T$
15 $x^{(m)} \leftarrow x^{(0)} + \mathcal{P}^{-1}V_m y^{(m)}$
16 **if** *not converged* **then**
17 Set $x^{(0)} \leftarrow x^{(m)}$ and go to line 2
18 **end**
19 **return** $x^{(m)}$

system of size n. This necessitates restarting with relatively small m to make GMRES a practical solver. In many applications, convergence is still achieved under restarting, provided the system is sufficiently well preconditioned, making GMRES a potent method for unsymmetric linear problems.

2.2.3 FGMRES

While GMRES is applicable to all invertible linear systems, the preconditioner has to be a constant linear operator throughout the whole iterative procedure. In certain cases, this poses a strong restriction in the design of preconditioners. For example, some linear systems can be reduced to more tractable subproblems that can be solved efficiently by other iterative methods. This naturally leads to the idea of nesting an iterative solver within $\mathcal{P}$, referred to as an inner solver. Even if such inner solvers take the form of (2.12) with a mappings Φ linear in $x^{(k)}$ and f, the overall iteration becomes a constant linear operator only if the number of iterations is fixed. However, if the inner solver is halted according to a convergence criterion based on a residual

threshold, the preconditioner becomes non-constant and, thus, incompatible within GMRES. Similarly, if the inner solver is itself a Krylov supspace method, the overall preconditioner $\mathcal{P}$ becomes nonlinear due to the inherent nonlinearity of Krylov subspace methods. In this thesis, we refer to varying preconditioners as nonlinear or non-constant.

GMRES was generalized to flexible GMRES (FGMRES) in [30] to allow for preconditioners $\mathcal{P}_k$ that can change in each iteration k. FGMRES relies on the same principle as MINRES and GMRES by finding an optimal solution over a shifted Krylov subspace that is adapted to account for varying preconditioners. As in GMRES, this is done by employing Arnoldi's iteration to the vectors generated by the adapted Krylov subspace, which then allows to solve the optimization problem based on the resulting orthonormal basis. The procedure is outlined in Algorithm 2.3, and mostly aligns with the GMRES iteration. Since this method also relies on the storage and computation of a dense Hessenberg matrix, the number of iterations is restricted and the algorithm is equipped with a restarting strategy. It is noted that the gained flexibility in the choice of preconditioners comes with a caveat. Since the preconditioners can now change in each iteration, it becomes necessary to keep track of all m vectors $z^{(k)}$ and $v^{(k)}$, doubling the memory requirements. Moreover, the convergence theory for FGMRES is less well understood than that of MINRES and GMRES, and it is generally not possible to establish error bounds on the residuals as the ones previously presented. Nevertheless, FGMRES often demonstrates strong convergence, even when nonlinear preconditioners are applied—a context in which the preceding solvers typically fail.

2.2.4 Chebyshev Semi-Iteration

We now turn to common iterative solvers that are particularly effective when embedded within Krylov subspace methods, especially for structured problems such as those encountered in finite element methods. This section discusses the *Chebyshev semi-iteration* and is based on [31]. The Chebyshev semi-iteration is a polynomial iterative technique that utilizes Chebyshev polynomials to minimize the residual $x - x^{(k)}$ using information from previous iterates. Even though Chebyshev semi-iteration is applicable to more general systems, we only apply it to problems where $\mathcal{A}$ is symmetric and $\mathcal{P}$ is SPD, and therefore, assume these properties subsequently.

The starting point for this method is an iterative splitting method for (2.12), which can be written in the form

$$x^{(k+1)} = \underbrace{\left(I - \mathcal{P}^{-1}\mathcal{A}\right)}_{=\mathcal{T}} x^{(k)} + \mathcal{P}^{-1}f = \mathcal{T}x^{(k)} + \mathcal{P}^{-1}f. \tag{2.15}$$

For example, if $\mathcal{P}$ is the diagonal of $\mathcal{A}$, (2.15) describes the Jacobi method. The iteration yields a sequence of vectors $\{x^{(j)}\}_{j=0}^{k}$, and can serve as a solver if $\rho(\mathcal{T}) < 1$, where $\rho(\cdot)$ denotes the spectral radius of a matrix. Chebyshev semi-iteration aims to improve this method at the kth iterative by combining the sequence of vectors $\{x^{(j)}\}_{j=0}^{k}$ linearly into an improved approximation

Algorithm 2.3: The right-preconditioned FGMRES method [30]

Function FGMRES($\mathcal{A}$, f, $\{\mathcal{P}_k\}$, $x^{(0)}$, m)**:**

Input : Matrix $\mathcal{A} \in \mathbb{R}^{n\times n}$
Right-hand side $f \in \mathbb{R}^n$
Preconditioners $\mathcal{P}_k \in \mathbb{R}^{n\times n}$
Initial guess $x^{(0)} \in \mathbb{R}^n$
Number of iterations per restarted iteration $m \in \mathbb{N}$

Output: Approximation x solving $\mathcal{A}x = f$ to a certain tolerance

1 Initialize $H_m \in \mathbb{R}^{(m+1)\times m}$ with zeros
2 Compute $r^{(0)} \leftarrow f - \mathcal{A}x^{(0)}$, $\beta \leftarrow \|r^{(0)}\|_2$, $v_1 \leftarrow r^{(0)}/\beta$
3 **for** $k \leftarrow 1$ **to** m **do**
4 Solve $\mathcal{P}_k z^{(k)} = v^{(k)}$
5 Compute $w \leftarrow \mathcal{A}z^{(k)}$
6 **for** $i \leftarrow 1$ **to** k **do**
7 $h_{i,k} \leftarrow \langle w, v^{(i)}\rangle$
8 $w \leftarrow w - h_{i,k}v^{(i)}$
9 **end**
10 $h_{k+1,k} \leftarrow \|w\|_2$
11 $v^{(k+1)} \leftarrow w/h_{k+1,k}$
12 **end**
13 Let $Z_m \leftarrow (z^{(1)}, \ldots, z^{(m)})$
14 Compute $y^{(m)} \leftarrow \arg\min_y \|\beta e_1 - H_m y\|_2$ with $e_1 = (1, 0, \ldots, 0)^T$
15 $x^{(m)} \leftarrow x^{(0)} + Z_m y^{(m)}$
16 **if** *not converged* **then**
17 Set $x^{(0)} \leftarrow x^{(m)}$ and go to line 2
18 **end**
19 **return** $x^{(m)}$

of the form

$$y^{(k)} = \sum_{j=0}^{k} \beta_j^{(k)} x^{(j)}$$

with weights $\{\beta_j^{(k)}\}_{j=0}^k$ that satisfy $\sum_{j=0}^k \beta_j^{(k)} = 1$. We now try to find an appropriate choice of weights that minimizes the norm of $x - y^{(k)}$, where x denotes the solution to $\mathcal{A}x = f$. It holds that

$$x - y^{(k)} = \sum_{j=0}^{k} \beta_j^{(k)} \mathcal{T}^j \left(x - x^{(0)}\right) = p_k(\mathcal{T}) \left(x - x^{(0)}\right), \tag{2.16}$$

where p_k is the polynomial deduced by the weights. Note that $p_k(1) = 1$ is equivalent to the condition $\sum_{j=0}^k \beta_j^{(k)} = 1$. Since $\mathcal{P}$ is SPD, we can rewrite

$$\mathcal{T} = I - \mathcal{P}^{-1}\mathcal{A} = \mathcal{P}^{-1/2}\left(I - \mathcal{P}^{-1/2}\mathcal{A}\mathcal{P}^{-1/2}\right)\mathcal{P}^{1/2},$$

i.e., the iteration matrix $\mathcal{T}$ is similar to a symmetric matrix. Due to its symmetry, the matrix $I - \mathcal{P}^{-1/2}\mathcal{A}\mathcal{P}^{-1/2}$ admits an eigenvalue decomposition $\mathcal{Z}\Lambda\mathcal{Z}^T$ with $\mathcal{Z}^T\mathcal{Z} = I$ and $\Lambda = \operatorname{diag}(\lambda_i)$ being the diagonal matrix with the eigenvalues of $\mathcal{T}$. This allows us to simplify monomials of $\mathcal{T}$ to

$$\mathcal{T}^j = \mathcal{P}^{-1/2}\left(I - \mathcal{P}^{-1/2}\mathcal{A}\mathcal{P}^{-1/2}\right)^j \mathcal{P}^{1/2} = \mathcal{P}^{-1/2}\mathcal{Z}\Lambda^j\mathcal{Z}^T\mathcal{P}^{1/2}.$$

Plugging this into (2.16), yields

$$x - y^{(k)} = \mathcal{P}^{-1/2}\mathcal{Z}p_k(\Lambda)\mathcal{Z}^T\mathcal{P}^{1/2}\left(x - x^{(0)}\right).$$

This way, we obtain an estimate for the residual

$$\begin{aligned}
\|x - y^{(k)}\|_{\mathcal{P}} &= \|\mathcal{P}^{1/2}(x - y^{(k)})\|_2 \\
&= \|\mathcal{P}^{1/2}\mathcal{P}^{-1/2}\mathcal{Z}p_k(\Lambda)\mathcal{Z}^T\mathcal{P}^{1/2}(x - x^{(0)})\|_2 \\
&= \|\mathcal{Z}p_k(\Lambda)\mathcal{Z}^T\mathcal{P}^{1/2}(x - x^{(0)})\|_2 \\
&\leq \|\mathcal{Z}p_k(\Lambda)\mathcal{Z}^T\|_2\|\mathcal{P}^{1/2}(x - x^{(0)})\|_2 \\
&= \|p_k(\Lambda)\|_2\|x - x^{(0)}\|_{\mathcal{P}} \\
&= \max_i |p_k(\lambda_i)|\|x - x^{(0)}\|_{\mathcal{P}}.
\end{aligned}$$

Hence, if one has knowledge of all eigenvalues of the iteration matrix, p_k can be chosen to minimize $\max_i |p_k(\lambda_i)|$ with the condition $p_k(1) = 1$. In practice, such information is rarely available. In some cases, however, the eigenvalues are known to be contained in a certain interval, i.e., $\lambda(\mathcal{P}^{-1}\mathcal{A}) \in [a, b]$, where a, $b > 0$. This means that $\lambda(\mathcal{T}) \in [\bar{a}, \bar{b}]$ with $\bar{a} = 1 - b$ and $\bar{b} = 1 - a$. Then, it holds that

$$\max_i |p_k(\lambda_i)| \leq \max_{t\in[\bar{a},\bar{b}]} |p_k(t)|,$$

and we aim to find a polynomial that minimizes the latter expression. It is known that upon scaling, the Chebyshev polynomial of degree k is a minimizer. More specifically, it holds that

$$\min_{p_k\in\Pi_k,\, p_k(1)=1} \max_{t\in[\bar{a},\bar{b}]} |p_k(t)| \leq 2\left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1}\right)^k.$$

By a detailed analysis, it can be shown that $y^{(k)}$ can be constructed iteratively with $\beta_j^{(k)}$ chosen as the coefficients of the Chebyshev polynomial without keeping track of all vectors $\{x^{(j)}\}_{j=0}^k$. This resuls in the procedure outlined in Algorithm 2.4.

The convergence of the method can be derived from the min–max property of Chebyshev polynomials stated above, and is bounded by

$$\frac{\|x - y^{(k)}\|_{\mathcal{P}}}{\|x - x^{(0)}\|_{\mathcal{P}}} \leq 2\left(\frac{\sqrt{\kappa(\mathcal{P}^{-1}\mathcal{A})} - 1}{\sqrt{\kappa(\mathcal{P}^{-1}\mathcal{A})} + 1}\right)^k, \tag{2.17}$$

where $\kappa(\mathcal{P}^{-1}\mathcal{A}) = b/a$.

Algorithm 2.4: Chebyshev semi-iteration method, cf. [34]

Function ChebyshevSemiIteration($\mathcal{A}$, f, $\mathcal{P}$, $x^{(0)}$, a, b)**:**

Input : Matrix $\mathcal{A} \in \mathbb{R}^{n\times n}$
Right-hand side $f \in \mathbb{R}^n$
Preconditioner $\mathcal{P} \in \mathbb{R}^{n\times n}$
Initial guess $x^{(0)} \in \mathbb{R}^n$
Spectral bounds $a, b \in \mathbb{R}$ such that $\lambda(\mathcal{P}^{-1}\mathcal{A}) \in [a, b]$

Output: Approximation x solving $\mathcal{A}x = f$ to a certain tolerance

Set $x^{(-1)} \leftarrow 0$

Set $w_0 \leftarrow 0$, $w \leftarrow \dfrac{a+b}{b-a}$

for $k = 0$ **to** *convergence* **do**

  if $k = 1$ **then**

    $w_{k+1} \leftarrow 1/\left(1 - \dfrac{1}{2w^2}\right)$

  else

    $w_{k+1} \leftarrow 1/\left(1 - \dfrac{w_k^2}{4w^2}\right)$

  Solve $\mathcal{P}z^{(k)} = f - \mathcal{A}x^{(k)}$

  $x^{(k+1)} \leftarrow w_{k+1}\left(\dfrac{2}{a+b}z^{(k)} + x^{(k)} - x^{(k-1)}\right) + x^{(k-1)}$

return $x^{(k+1)}$

Since the method depends on eigenvalue bounds of the preconditioned system, Chebyshev semi-iteration is particularly attractive for problems where the spectral bounds are known a priori and are tight. Specifically, a range of finite element mass matrices fulfill this requirement, which makes this method highly efficient for solving systems involving such matrices. As demonstrated in [32, 33], tight eigenvalue bounds can be derived analytically for a variety of finite elements when $\mathcal{P}$ is chosen as the diagonal of the mass matrix. The technique of only using the diagonal is commonly referred to as a form of *mass lumping*. In the specific example of P_1- and P_2-elements on a two-dimensional domain, upper bounds for b/a are 4 and 5.2496, respectively (see [32]). Substituting these bounds into (2.17) yields a guaranteed convergence rate of $\frac{1}{3}$ for P_1-elements, and approximately 0.3923 for P_2-elements. Consequently, Chebyshev semi-iteration improves the solution by about one digit of accuracy every third iteration. This efficiency makes the method both practical and computationally efficient for either solving systems of mass matrices or for its use as smoothers or preconditioners.

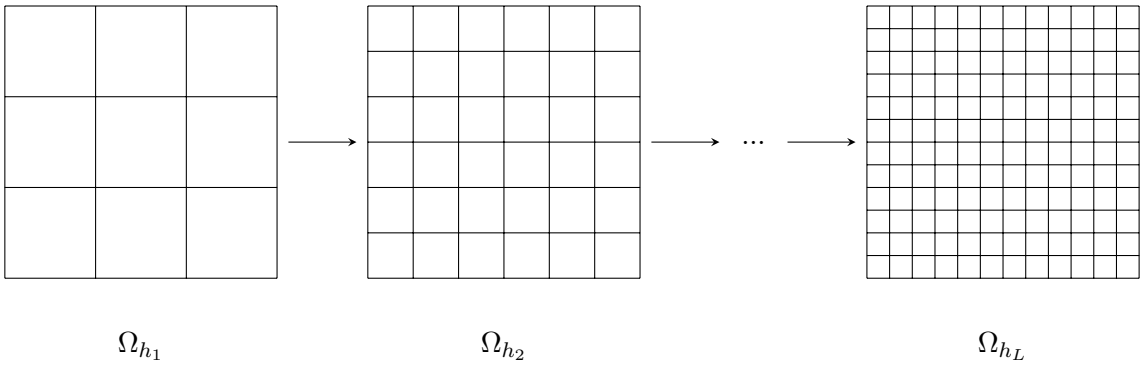
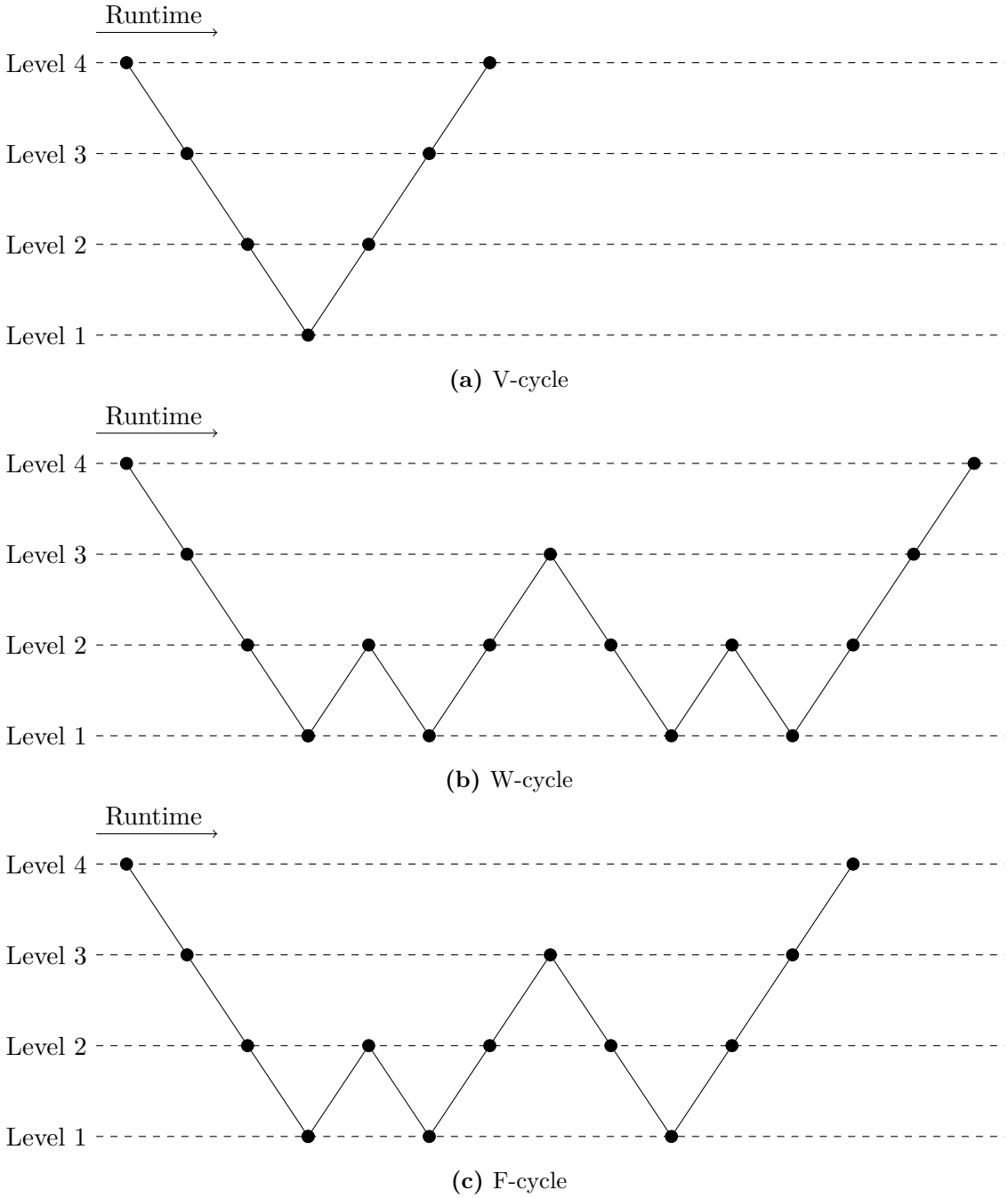


Figure 2.1: Sequence of refined meshes $\{\Omega_{h_j}\}_{j=1}^{L}$ for a two-dimensional domain Ω.

2.2.5 Multigrid Methods

Another prominent tool in the scope of finite element methods are *multigrid methods*. This introduction is based on [27, pp. 91 ff.], and we begin with the description of so-called *geometric multigrid methods*. Assume we seek the solution of a PDE on a domain $\Omega \subset \mathbb{R}^d$ of dimension d. To solve this problem using finite elements, we first generate a mesh Ω_h that approximates Ω. On this mesh, the PDE can be reformulated as a linear system $\mathcal{A}x = f$, where $\mathcal{A}$ and f arise from the discrete weak form of the differential operators, the right-hand side, and the boundary conditions. Multigrid methods exploit the fact that the problem can be represented on multiple mesh levels, each with different resolutions. Finer meshes yield more accurate solutions but also result in more challenging linear systems. The method constructs a hierarchy of meshes $\{\Omega_{h_j}\}_{j=1}^{L}$, ranging from coarse ($j = 1$) to fine ($j = L$) resolutions. This hierarchy can be built either by starting with a coarse mesh and refining it incrementally or by beginning with a fine mesh and coarsening it. An example of a sequence of refined meshes for a two-dimensional domain is shown in Figure 2.1. This process produces a sequence of linear systems represented by matrices $\{\mathcal{A}^{(j)}\}_{j=1}^{L}$ and restriction operators $\{\mathcal{R}^{(j)}\}_{j=2}^{L}$ that restrict solutions to coarser meshes and prolongate them to finer meshes via their transposes. We assume that the linear system at the coarsest level is so small that it can be solved directly. On finer levels, we assume the existence of smoothers—for example an inexpensive iterative solver such as Jacobi or Gauss–Seidel iteration—, whose aim is to damp high-frequency components in the residual. The core idea of multigrid is to compute the residual at a given level, restrict it to a coarser level, and repeat this process until reaching the coarsest level, where the problem is solved exactly. The solution is then prolongated back to finer grids and used to correct the approximate solutions. Thus, on finer meshes, only the smoothers are applied, while the complete solution is obtained only on the coarsest level, and, therefore, for significantly smaller systems. The sequence and frequency of these steps determine whether the multigrid method acts as a preconditioner with different cycles or as a standalone solver. While this sketches the key idea of the method, various assumptions on its components, such as the smoothers and the underlying PDE problem, have to be met in order to guarantee multigrid to become an efficient method, see, e.g., [27, 35]. However, a detailed discussion on the theory is omitted in this introduction.



(a) V-cycle

(b) W-cycle

(c) F-cycle

Figure 2.2: Different types of cycles for multigrid preconditioners.

Algorithm 2.5: Multigrid preconditioner with V-cycle, cf. [27, p. 105]

Function MGPrecondV($\{\mathcal{A}^{(j)}\}_{j=1}^{L}$, $\{\mathcal{R}^{(j)}\}_{j=2}^{L}$, $f^{(l)}$, l)**:**

Input : Linear operators on different levels $\{\mathcal{A}^{(j)}\}_{j=1}^{L}$
Restriction operators $\{\mathcal{R}^{(j)}\}_{j=2}^{L}$
Right-hand side on level on current level $f^{(l)}$
Current level $l \in \{1, \ldots, L\}$

Output: Approximation w to $\mathcal{A}^{(l)}w = f^{(l)}$

1 **if** $l = 1$ *(coarsest level)* **then**
2 $\quad w \leftarrow \left(\mathcal{A}^{(l)}\right)^{-1} f^{(l)}$
3 $\quad$ **return** w
4 **end**
5 $w \leftarrow 0$
6 $w \leftarrow$ PreSmooth($\mathcal{A}^{(l)}$, w, $f^{(l)}$)
7 $r \leftarrow \mathcal{R}^{(l)}\left(f^{(l)} - \mathcal{A}^{(l)}w\right)$ ▷ *Compute and restrict residual*
8 $d \leftarrow$ MGPrecondV($\{\mathcal{A}^{(j)}\}_{j=1}^{L}$, $\{\mathcal{R}^{(j)}\}_{j=2}^{L}$, r, $l-1$) ▷ *Compute correction*
9 $w \leftarrow w + \left(\mathcal{R}^{(l)}\right)^{T} d$ ▷ *Adjust solution with prolongated correction*
10 $w \leftarrow$ PostSmooth($\mathcal{A}^{(l)}$, w, $f^{(l)}$)
11 **return** w

When multigrid is employed as a preconditioner, the sequence of steps is typically fixed. The most common strategies are the V-cycle, W-cycle, and F-cycle. Algorithm 2.5 describes the multigrid method using a V-cycle. In this approach, the residual is smoothed at each level as it is transferred down to the coarsest level, where the linear system is solved exactly. Subsequently, the correction is propagated back up through the finer levels, with a post-smoothing step performed at each level. Figure 2.2a illustrates the path of the residual, where the levels are arranged vertically and the horizontal axis represents computational time. The resulting V-shaped trajectory gives this method its name. The W-cycle (see Figure 2.2b) is similar but revisits each coarser level, while the F-cycle (see Figure 2.2c) first descends to the coarsest level and then gradually ascends to the finest level. Under appropriate assumptions on the PDE problem and the used finite element method, multigrid preconditioners are optimal. This means that their runtime scales linearly in the number of degrees of freedom, and that they ensure a bounded number of iterations when embedded in solvers such as MINRES and GMRES.

Multigrid methods can serve as a standalone solvers if lines 6 through 10 of Algorithm 2.5 are iterated until the norm of $f^{(l)} - \mathcal{A}^{(l)}w$ falls below a specified tolerance. As a result, the procedure does not always follow patterns such as those illustrated in Figure 2.2. However, this setting does not allow multigrid to be embedded within MINRES or GMRES.

Geometric multigrid relies on explicit knowledge of the problem's geometry and the construction of associated meshes. In contrast, so-called *algebraic multigrid* (AMG) methods (cf. [35]) require only the system matrix $\mathcal{A}$ as input and automatically infer and exploit the graph structure directly from the linear system. The key advantage of AMG methods is their versatility as they can often serve as black-box preconditioners, provided $\mathcal{A}$ originates from problems with an underlying hierarchical or multigrid structure.

Chapter 3

Diagonalization-Based Parallel-in-Time Preconditioners for Instationary Fluid Flow Control Problems

As we have seen previously, an important class of optimal control problems is formed by PDE-constrained optimization problems, which have received great attention during the past few decades. This type of optimization problem arises widely in science and engineering, and has utility in many industrial processes. Among a wide range of applications of PDE-constrained optimization, our focus here lies on fluid flow control problems, see e.g., [36, 37, 38]. In this chapter, we will discuss the all-at-once solution of unsteady Stokes and Oseen control problems. Our approach emphasizes the exploitation of the problems' inherent block structure and factorized form, which in turn enables the development of fast and parallelizable solvers.

3.1 Introduction

As with many PDE problems, analytical solutions to unsteady Stokes and Oseen control problems are not known in general, which is why one is required to solve them numerically. However, their discretization often results in huge-scale systems of linear or linearized equations. The number of degrees of freedom sometimes scales poorly with the accuracy of the discretization. Off-the-shelf solvers, such as direct solvers for linear systems, often have storage requirements that are excessive for problems that are sufficiently finely-discretized to achieve high accuracy. In order to make a numerical solver feasible and effective, information about the linear system itself and the structure of the PDEs must frequently be taken into account when designing it. In recent years, preconditioned iterative methods have been successfully applied to PDE-constrained optimization problems [39, 31, 40, 41, 42, 43, 44], and, in particular, bespoke robust preconditioners for stationary and instationary Stokes and Oseen problems have enabled the fast and robust solution of flow control problems [45, 46, 47, 48]. Recent research has made it possible to tackle huge-scale nonlinear time-dependent problems and solve them to high accuracy, see e.g., [49, 50, 51, 52].

A key challenge in these fields is that there is a pressing need to model and simulate increasingly complicated and fine-scale physical phenomena, but in previous decades CPUs have not experienced a significant increase in speed and are also not expected to do so in the future [53]. However, processing architectures with increasing numbers of concurrent processing units have become more available. This in turn has increased the interest in parallelization of numerical methods in order to increase the range of problems they are capable of coping with, to resolve many of these numerical challenges. These include *parallel-in-time* methods, which are numerical methods for solving time-dependent problems that are designed to be parallelizable; a good overview of such methods can be found in [54]. A number of approaches within this family have been suggested. Methods such as Parareal [55] and ParaDiag [56] have been shown to accelerate the solution of evolution equations, but these must be adapted to become applicable when considering optimality systems, as they occur in time-dependent PDE-constrained optimization problems. Examples of parallel-in-time methods which have been successfully applied to optimization problems involving PDEs may be found in [57, 58, 59, 60, 61, 62].

In this work, we derive a new diagonalization-based parallel-in-time approach for fluid flow control problems, which allows their rapid solution by making use of the fast Fourier transform (FFT). With suitable permutations, we may then reformulate the problem to one of designing suitable inner solvers for systems of equations arising from individual time points, for which we derive bespoke, potent approximations. We make use of theory for saddle-point preconditioners, existing robust preconditioners for flow problems based on commutator arguments [45, 46], and tailored routines for individual submatrices, resulting in fast and robust preconditioners for the all-at-once solution of the Stokes and Oseen problems. To our knowledge, this is the first parallelizable-in-time preconditioner for either of these problems, which demonstrates very good (strong and weak) scaling properties in a practical parallel architecture.

The chapter is structured as follows. In Section 3.2, the flow control problems we consider are introduced. Their first-order optimality conditions are stated and the resulting systems of discretized equations are derived. Moreover, basic tools for the numerical analysis of saddle-point systems are summarized for the general case, and in the context of flow problems. Section 3.3 discusses the diagonalization of the system of equations. It is presented how adaptations of existing preconditioners for flow control problems can be embedded into this framework. Section 3.4 generalizes these approaches to the Oseen problem. In Section 3.5, we present numerical results, including results run in parallel on problems with more than 76 million degrees of freedom. Finally, concluding remarks are given in Section 3.6.

3.2 Problem Formulation

In this chapter, we develop a numerical method for solving instationary Stokes and Oseen flow control problems. Specifically, we consider an optimization problem that aims to find a distributed control minimizing the least-squares distance of the velocity field from a desired velocity governed by a Stokes or an Oseen flow. Additionally, the problem is regularized by penalizing the control in the cost functional. Since the Stokes flow is a special case of the Oseen flow, we will first introduce the problem for the latter and then highlight the difference between the two settings.

The Oseen flow control problem can be formulated more concretely as finding

$$\min_{v,p,u} \quad J(v,u) = \frac{1}{2}\int_0^T \int_\Omega \|v(x,t) - v_d(x,t)\|^2 \mathrm{d}x\mathrm{d}t + \frac{\beta}{2}\int_0^T \int_\Omega \|u(x,t)\|^2 \mathrm{d}x\mathrm{d}t, \tag{3.1}$$

subject to the equations

$$\begin{cases} \dfrac{\partial v}{\partial t} - \nu\Delta v + w(x)\cdot\nabla v + \nabla p = u + f(x,t) & \text{in } \Omega\times(0,T), \\ -\nabla\cdot v = 0 & \text{in } \Omega\times(0,T), \\ v(x,t) = h(x,t) & \text{on } \partial\Omega\times(0,T), \\ v(x,0) = v_0(x) & \text{in } \Omega, \end{cases} \tag{3.2}$$

where $\Omega \subset \mathbb{R}^d$ is a bounded domain of dimension $d = 2, 3$. The function v represents the vector velocity field, p the scalar pressure field, and u the control variable. The desired velocity state is denoted as v_d. The function w is referred to as the *wind*, which is a distinctive feature to the Oseen flow, and ν is the viscosity. We will assume that the wind is divergence-free, i.e., $\nabla\cdot w(x) = 0$ in $x \in \Omega$. The parameter $\beta > 0$ is the regularization parameter and determines the weight of the control costs. The Stokes flow control problem is achieved if no wind is present, i.e., $w = 0$.

Equation (3.2) is a set of partial differential equations in strong form. In order to construct finite element approximations, we will consider the weak formulation of these equations. This requires us to add the assumption that Ω is a Lipschitz domain and to consider the function spaces $v \in H^1(\Omega)^d$, $p \in L^2(\Omega)$, and $u \in L^2(\Omega)^d$. For a detailed discussion of the weak form for the Stokes problem, the reader is referred to [27, Chapter 3]. The Oseen flow can be treated similarly. For the reader's convenience, the following derivations and equations are presented in the strong form of the equations, however, these can be carried over to the weak form with only a few adaptations.

3.2.1 Optimality Conditions

The formulation (3.1)–(3.2) represents a convex optimization problem. This means it is sufficient to consider the first-order optimality conditions in order to solve the problem. These are given by the following set of equations:

$$\begin{cases} \dfrac{\partial v}{\partial t} - \nu\Delta v + w(x)\cdot\nabla v + \nabla p = \dfrac{1}{\beta}\lambda + f(x,t) & \text{in } \Omega\times(0,T), \\ -\nabla\cdot v = 0 & \text{in } \Omega\times(0,T), \\ v(x,t) = h(x,t) & \text{on } \partial\Omega\times(0,T) \\ v(x,0) = v_0(x) & \text{on } \Omega, \end{cases} \tag{3.3}$$

$$\begin{cases} -\dfrac{\partial \lambda}{\partial t} - \nu\Delta\lambda - w(x)\cdot\nabla\lambda + \nabla\mu = v_d - v & \text{in } \Omega\times(0,T), \\ -\nabla\cdot\lambda = 0 & \text{in } \Omega\times(0,T), \\ \lambda(x,t) = 0 & \text{on } \partial\Omega\times(0,T), \\ \lambda(x,T) = 0 & \text{on } \Omega. \end{cases} \tag{3.4}$$

Equations (3.3) and (3.4) are the *state* and *adjoint equations*, respectively. When deriving the optimality conditions, one would additionally obtain the gradient equation

$$\beta u - \lambda = 0,$$

which has already been substituted in to obtain the above state equations. Overall, the solution of the optimization problem has been narrowed down to solving the coupled PDEs (3.3) and (3.4).

3.2.2 Discretization

The optimality conditions are discretized with a finite element method in space. The function approximation spaces have to be chosen carefully to achieve stability of the numerical method. We will restrict ourselves to two stable choices of elements, which are P_2-P_1-elements, so-called Taylor–Hood elements, and Q_2-Q_1-elements. The properties of these discretizations are discussed in [27, pp. 133 ff. and pp. 149 ff.]. In time, the equations are discretized with the implicit Euler method. For the temporal discretization, it has to be taken into account that the state equation is to be considered forward in time and the adjoint equation backward in time. The number of time points is denoted as $n_t + 1$ for some $n_t \in \mathbb{N}$, which leads to a timestep of $\tau = T/n_t$. This

gives us the equations

$$\begin{aligned} M\underline{v}^{(j+1)} - M\underline{v}^{(j)} + \tau L\underline{v}^{(j+1)} + \tau B^T\underline{p}^{(j+1)} &= \frac{\tau}{\beta}M\underline{\lambda}^{(j+1)} + \underline{h}^{(j+1)} \quad \text{for } j \in \{0,\dots,n_t-1\}, \\ \tau B\underline{v}^{(j+1)} &= 0 \quad \text{for } j \in \{0,\dots,n_t-1\}, \\ M\underline{\lambda}^{(j-1)} - M\underline{\lambda}^{(j)} + \tau L^T\underline{\lambda}^{(j-1)} + \tau B^T\underline{\mu}^{(j-1)} &= \underline{g}^{(j-1)} + \tau M\left(\underline{v}_d^{(j-1)} - \underline{v}^{(j-1)}\right) \\ &\qquad \text{for } j \in \{1,\dots,n_t\}, \\ \tau B\underline{\lambda}^{(j-1)} &= 0 \quad \text{for } j \in \{1,\dots,n_t\}, \end{aligned}$$

where superscript j indicates the values at time $t = j\tau$. The variables $\underline{v}$, $\underline{p}$, $\underline{\lambda}$, $\underline{\mu}$, and $\underline{v}_d$ denote the vectors corresponding to the spatially discretized functions. The matrix M is the mass matrix in the velocity space, L is the discretization of the weak form of $-\nu\Delta + w(x)\cdot\nabla$ and B is the discretized negative divergence. The vectors $\underline{g}^{(\cdot)}$ and $\underline{h}^{(\cdot)}$ include boundary conditions and forcing terms. For $j = 0$ and $j = n_t$, we need to specify the initial and terminal conditions

$$\underline{v}^{(0)} = \underline{v}_0, \quad \underline{\lambda}^{(n_t)} = \underline{0}.$$

It is important to note that in the above discretization the pressure field is unique only up to a constant, which means that the resulting linear system is singular. In order to prevent potential difficulties caused by this singularity, we transform the system to a regular one by fixing the pressure at one pre-defined point in space. This way the divergence matrix B becomes full rank and the overall linear system invertible.

Assembling this into an all-at-once system leads to linear equations of the form

$$\bar{\mathcal{A}}\begin{pmatrix}\underline{v}^T & \underline{p}^T & \underline{\lambda}^T & \underline{\mu}^T\end{pmatrix}^T = \begin{pmatrix}\underline{g}^T & \underline{h}^T\end{pmatrix}^T,$$

where $\underline{v} = \begin{pmatrix}(\underline{v}^{(1)})^T & \cdots & (\underline{v}^{(n_t-1)})^T\end{pmatrix}^T$ is the concatenation of the discretized velocity at all time points (apart from the first and last). The vectors $\underline{p}$, $\underline{\lambda}$, and $\underline{\mu}$ are defined similarly. The matrix is defined as

$$\bar{\mathcal{A}} = \begin{pmatrix} \tau I_t \otimes M & 0 & E^T \otimes M + \tau I_t \otimes L^T & \tau I_t \otimes B^T \\ 0 & 0 & \tau I_t \otimes B & 0 \\ E \otimes M + \tau I_t \otimes L & \tau I_t \otimes B^T & -\frac{\tau}{\beta} I_t \otimes M & 0 \\ \tau I_t \otimes B & 0 & 0 & 0 \end{pmatrix}.$$

The matrix I_t is the identity matrix of size $n_t - 1$, and $E \in \mathbb{R}^{(n_t-1)\times(n_t-1)}$ arises from the implicit Euler discretization and has the form

$$E = \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix}.$$

The right-hand side $\begin{pmatrix} \underline{g}^T & \underline{h}^T \end{pmatrix}^T$ assembles all $\underline{g}^{(j)}$ and $\underline{h}^{(j)}$, and accounts for the forcing terms, boundary conditions, and the initial and terminal conditions. As we will only consider the discretized system in the following, the underlined notation for the discretized variables will be dropped for the sake of better readability.

While $\bar{\mathcal{A}}$ results from the direct assembly of the discretized equations maintaining the original ordering, we will consider a different permutation. The linear system can be permuted to obtain a matrix of the form

$$\mathcal{A} = \begin{pmatrix} \tau I_t & E^T \\ E & -\frac{\tau}{\beta} I_t \end{pmatrix} \otimes \begin{pmatrix} M & 0 \\ 0 & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & 0 \\ I_t & 0 \end{pmatrix} \otimes \begin{pmatrix} L & B^T \\ B & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & I_t \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} L^T & B^T \\ B & 0 \end{pmatrix}, \quad (3.5)$$

whose Kronecker product form makes upcoming manipulations clearer.

Remark 3.1. *The choice of the parameter T is determined by the application and the time frame in which one is interested in practice. The parameter n_t is also a design parameter and is determined by the targeted accuracy of the solution, as prescribed by the application. More specifically, the error of the resulting approximation is partially controlled by τ, with this component scaling asymptotically as $O(\tau) = O(T/n_t)$. Hence, the necessity of highly accurate solutions and large T require large numbers of n_t and, therefore, pose a numerically challenging problem. We develop methods that can tackle fluid flow control problems efficiently for all settings of T and n_t.*

3.2.3 Saddle-Point Systems

Given that several of the linear systems under consideration exhibit a saddle-point structure, this section will briefly present key properties of generalized saddle-point systems. Saddle-point systems have been extensively studied over the past decades, with some of their findings being the basis for key ideas of this work. We are especially interested in their efficient numerical solution and will outline relevant preconditioners in this section. A detailed discussion of the analysis and numerical treatment of these systems can be found in [63].

The (generalized) saddle-point systems discussed here are linear systems of the form

$$\underbrace{\begin{pmatrix} \Phi & \Psi^T \\ \Psi & -\Theta \end{pmatrix}}_{=\mathcal{H}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

It is assumed that the matrices $\Phi \in \mathbb{R}^{n\times n}$ and $\Theta \in \mathbb{R}^{m\times m}$ are symmetric positive definite and symmetric positive semi-definite, respectively. The Schur complement of $\mathcal{H}$ is defined by

$$S = \Theta + \Psi\Phi^{-1}\Psi^T$$

and forms the basis for a broad range of preconditioners for $\mathcal{H}$, since it enables the decomposition

$$\mathcal{H} = \begin{pmatrix} I & 0 \\ \Psi\Phi^{-1} & I \end{pmatrix} \begin{pmatrix} \Phi & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & \Phi^{-1}\Psi^T \\ 0 & -I \end{pmatrix} = \begin{pmatrix} \Phi & 0 \\ \Psi & S \end{pmatrix} \begin{pmatrix} I & \Phi^{-1}\Psi^T \\ 0 & -I \end{pmatrix},$$

which in turn justifies the following choice of preconditioner:

$$\mathcal{P} = \begin{pmatrix} \Phi & 0 \\ \Psi & S \end{pmatrix},$$

given invertibility of S. The resulting preconditioned system has only two distinct eigenvalues $\lambda(\mathcal{P}^{-1}\mathcal{H}) = \{-1, 1\}$, which means that preconditioned Krylov subspace solvers, such as MINRES [28] or GMRES [29], converge within two iterations. In order to make the application of $\mathcal{P}^{-1}$ practical, an efficient application of Φ^{-1} and S^{-1} is required, which is why instead of $\mathcal{P}$, we consider an approximation

$$\widehat{\mathcal{P}} = \begin{pmatrix} \widehat{\Phi} & 0 \\ \Psi & \widehat{S} \end{pmatrix}. \tag{3.6}$$

While a good approximation of Φ can often be achieved with approaches such as multigrid methods or iterative methods like Chebyshev semi-iteration [64, 33] when dealing with PDEs, the approximation of the Schur complement is a more delicate task. Since S is dense in general and also expensive to compute, it is infeasible to assemble S in most PDE-based applications. In certain cases, however, efficient approximations with good spectral properties can be achieved. Subsequently, we will introduce two of such approximations that will be leveraged later.

One Schur complement approximation used in this work was first presented in [41, 42] for systems that fulfill $n = m$ and $\Theta = \alpha\Phi$. The authors proposed to use the approximation

$$S \approx \widehat{S} = \left(\Psi + \sqrt{\alpha}\Phi\right)\Phi^{-1}\left(\Psi + \sqrt{\alpha}\Phi\right)^T. \tag{3.7}$$

It was shown that the spectral properties are robust with respect to the parameter α. More precisely, it was shown in [42] that if $\Psi + \Psi^T$ is positive semi-definite, then

$$\lambda\left(\widehat{S}^{-1}S\right) \in \left[\frac{1}{2}, 1\right].$$

The other case in which we are particularly interested here is the saddle-point system

$$\mathcal{H} = \begin{pmatrix} L & B^T \\ B & 0 \end{pmatrix}$$

with the finite element matrices defined in the preceding section. A good approximation of the Schur complement is based on the idea of the commutator. This idea and its application to flow problems were proposed in, e.g., [45, 46], and an introductory overview is given in [27, Section 9.2]. The starting point of the argument is the assumption that the operator

$$\mathcal{E} = \nabla \cdot (-\nu\Delta + w \cdot \nabla) - (-\nu\Delta + w \cdot \nabla)_p \nabla\cdot$$

is small, which allows us to reason that

$$S = BL^{-1}B^T \approx M_p L_p^{-1} K_p \approx \widehat{S}$$

forms a good approximation. The subscripts p denote the analogous differential operators and matrices in the pressure space and the matrix K_p represents the finite element stiffness matrix arising from the negative Laplacian. In the case of the Stokes problem, it can be shown that the eigenvalues of the preconditioned Schur complement are bounded from above by 1. Theoretical results on the lower bound relate to (the square of) an inf–sup constant, which is generally not known analytically. Experiments, however, have verified that in practice $\widehat{S}$ indeed approximates well (in a spectral sense) the Schur complement, cf. [27, p. 175]. Hence, real numbers $0 < a \leq 1 \leq b$ can be found such that

$$\lambda\left(\widehat{S}^{-1}S\right) \in [a, b],$$

where a and b are robust with respect to the discretization and the model parameters. Thus, the block triangular preconditioner (3.6) with an exact (1,1)-block and the above Schur complement approximation has the spectral property

$$\left|\lambda\left(\widehat{\mathcal{P}}^{-1}\mathcal{H}\right)\right| \in [a, b].$$

For the Oseen problem, such bounds cannot be established since the differential operator has complex eigenvalues in general. The preconditioner, nonetheless, provides good clustering of the eigenvalues of the preconditioned system.

3.3 Preconditioners for the Stokes Control Problem

We aim to solve equation (3.5) with a Krylov subspace iterative solver. To apply such methods effectively, one is required to provide efficient preconditioners. We are interested in how a preconditioner can be designed to allow for parallelization among the timesteps to make solving systems with large numbers of timesteps tractable. This would provide a so-called *parallel-in-time* preconditioner. Over the last decades, various parallel-in-time approaches have been developed for optimality systems, see e.g., [57, 58, 59]. We will now focus on diagonalization-based approaches for the systems under consideration.

3.3.1 Block Diagonalization with FFT

First, our goal is to obtain a diagonalized representation of $\mathcal{A}$ that can be applied cheaply. There is no inherent potential for diagonalizing (by blocks) the matrix $\mathcal{A}$ as it is, which is why we consider a slightly perturbed matrix $\mathcal{P}_C$. The preconditioner is achieved by replacing E with a low-rank perturbation C

$$\mathcal{P}_C = \begin{pmatrix} \tau I_t & C^T \\ C & -\frac{\tau}{\beta} I_t \end{pmatrix} \otimes \begin{pmatrix} M & 0 \\ 0 & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & 0 \\ I_t & 0 \end{pmatrix} \otimes \begin{pmatrix} L & B^T \\ B & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & I_t \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} L^T & B^T \\ B & 0 \end{pmatrix},$$

where C is given by

$$C = \begin{pmatrix} 1 & & & -1 \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix} \approx E.$$

An interpretation of this is that the original Stokes problem is replaced *for the purposes of deriving preconditioners* with its periodic-in-time analogue. In [48], this problem is analyzed and its diagonalization is discussed. In the following, we will outline the main idea and generalize it in order to cover the case of a non-symmetric L, i.e., the Oseen flow. The advantage of the approximation is that C is a circulant matrix, which can be diagonalized with the Fourier matrix

$$C = F^{-1} D F,$$

where F may be approximated using the forward Fourier transform, F^{-1} with the inverse Fourier transform, and $D = \mathrm{diag}(Fc_1)$ with c_1 being the first column of C. This approximation and the Kronecker product form allow us to diagonalize the whole system

$$\mathcal{P}_C = \mathcal{F}^{-1} \underbrace{\left(\begin{pmatrix} \tau I_t & D^* \\ D & -\frac{\tau}{\beta} I_t \end{pmatrix} \otimes \begin{pmatrix} M & 0 \\ 0 & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & 0 \\ I_t & 0 \end{pmatrix} \otimes \begin{pmatrix} L & B^T \\ B & 0 \end{pmatrix} + \tau \begin{pmatrix} 0 & I_t \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} L^T & B^T \\ B & 0 \end{pmatrix} \right)}_{=\bar{\mathcal{G}}} \mathcal{F},$$

where $\mathcal{F}$ is the block Fourier transform in time

$$\mathcal{F} = \begin{pmatrix} F & 0 \\ 0 & F \end{pmatrix} \otimes \begin{pmatrix} I_{n_v} & 0 \\ 0 & I_{n_p} \end{pmatrix}.$$

The integers n_v and n_p are the numbers of degrees of freedom in the velocity and pressure approximation spaces. The operator $\mathcal{F}$ can be applied in $O((n_v + n_p) n_t \log n_t)$ floating point operations. We restrict ourselves to the case of implicit Euler time integration, however it is noted that this procedure can be generalized to higher-order one-step methods by introducing perturbations of higher rank.

Upon permutation, $\bar{\mathcal{G}}$ takes a block diagonal form $\mathcal{G} = \mathrm{diag}(G_1, \ldots, G_{n_t - 1})$ with the blocks

$$G_j = \begin{pmatrix} \tau M & d_j^* M + \tau L^T & 0 & \tau B^T \\ d_j M + \tau L & -\frac{\tau}{\beta} M & \tau B^T & 0 \\ 0 & \tau B & 0 & 0 \\ \tau B & 0 & 0 & 0 \end{pmatrix}, \tag{3.8}$$

where d_j is the jth diagonal entry of D. The permutation can be represented as the matrix

$$R = \begin{pmatrix} I_{n_t} \otimes \begin{pmatrix} I_{n_v} & 0 & 0 & 0 \\ 0 & 0 & I_{n_v} & 0 \\ 0 & I_{n_p} & 0 & 0 \\ 0 & 0 & 0 & I_{n_p} \end{pmatrix} \end{pmatrix} \left(\left(I_{n_t} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad I_{n_t} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \otimes \begin{pmatrix} I_{n_v} & 0 \\ 0 & I_{n_p} \end{pmatrix} \right).$$

We can now solve for each block G_j separately, i.e., in parallel. Each of the blocks can be interpreted as a version of a time-independent Stokes problem. Hence, we have arrived at a preconditioner that can be applied in a parallel-in-time fashion.

Since $\mathcal{P}_C$ represents a low-rank perturbation of $\mathcal{A}$, we can identify the majority of the eigenvalues of the preconditioned system.

Theorem 3.1. *The preconditioner $\mathcal{P}_C$ fulfills*

$$\# \left\{ \mu \in \lambda \left(\mathcal{P}_C^{-1} \mathcal{A} \right) \mid \mu = 1 \right\} \geq 2(n_t - 1)(n_v + n_p) - 2n_v.$$

Thus, the preconditioner has the desirable property of eigenvalue clustering. Nevertheless, the application of $\mathcal{P}_C$ still requires solving for a large number of stationary problems G_j at each Krylov iteration, which can be expensive for even moderately sized problems. In that case, the overall preconditioner might become inefficient for its practical use. In the following, we will look at possible ways of solving the subblocks G_j arising from the diagonalization. First, we will discuss possible preconditioners for the subsystem, then we will look at approaches for embedding these within the preconditioner $\mathcal{P}_C$.

Remark 3.2. *While we consider linear autonomous fluid flow control problems, some fluids in practice are governed by nonlinear evolution equations, such as the Navier–Stokes equations. The assumption of linear dynamics is made because the block diagonalization of the linear system is the keystone of our method and relies on the Kronecker product form of the all-at-once system. The Kronecker product form is, however, compromised when nonlinearities are introduced. One possible way of applying our methods to nonlinear problems is to assume weak nonlinearities and use linear autonomous approximations of the dynamics around stationary flows for designing the preconditioner. In the example of the Navier–Stokes equations, the weakness of the nonlinearity relates to the Reynolds number and the deviation from the considered stationary flow.*

3.3.2 Preconditioning the Subsystem

In [48], a preconditioner was developed for (3.8). First, the block is decomposed as

$$G_j = T_j^{(l)} Z_j T_j^{(r)}, \tag{3.9}$$

with the matrices

$$T_j^{(l)} = \begin{pmatrix} \sqrt{\tau} I & 0 & 0 & 0 \\ \frac{d_{j,c}}{\sqrt{\tau}} iI & \frac{\sqrt{\tau}}{c_{j,2}} I & 0 & 0 \\ 0 & 0 & \sqrt{\tau} c_{j,2} I & 0 \\ 0 & 0 & \frac{d_{j,c} c_{j,2}}{\sqrt{\tau}} iI & \sqrt{\tau} I \end{pmatrix}, \quad Z_j = \begin{pmatrix} M & c_{j,1}M + c_{j,2}L & 0 & B^T \\ c_{j,1}M + c_{j,2}L & -M & B^T & 0 \\ 0 & B & 0 & 0 \\ B & 0 & 0 & 0 \end{pmatrix},$$

with $T_j^{(r)} = (T_j^{(l)})^H$ the conjugate transpose of $T_j^{(l)}$, and the constants

$$d_{j,r} = \mathrm{Re}(d_j), \quad d_{j,c} = \mathrm{Im}(d_j), \quad c_{j,1} = \frac{d_{j,r}}{\sqrt{\frac{\tau^2}{\beta} + d_{j,c}^2}}, \quad c_{j,2} = \frac{1}{\sqrt{\frac{1}{\beta} + \frac{d_{j,c}^2}{\tau^2}}}.$$

Since $T_j^{(l)}$ and $T_j^{(r)}$ involve only block row manipulations and scalings, their inverses can be applied efficiently. This allows us to focus on preconditioning the matrix Z_j. This decomposition is based on the symmetry of L, i.e., $L = L^T$, which holds true as we examine the Stokes flow control problem in the present Section 3.3.

The first attempt involves using

$$\widehat{P}_j = \begin{pmatrix} W_j & & & \\ & W_j & & \\ & & BW_j^{-1}B^T & \\ & & & BW_j^{-1}B^T \end{pmatrix} \tag{3.10}$$

as a preconditioner for Z_j, where $W_j = M + c_{j,1}M + c_{j,2}L$. In [48], it was shown that $\widehat{P}_j$ is symmetric positive definite and that the absolute values of the eigenvalues of the preconditioned matrix are contained in the following interval:

$$\left|\lambda(\widehat{P}_j^{-1}Z_j)\right| \subseteq \left[\frac{1}{\sqrt{12}}, \frac{1+\sqrt{5}}{2}\right]. \tag{3.11}$$

Even though $\widehat{P}_j$ leads to robust spectral properties for the preconditioned system and has a block diagonal form, the matrices $BW_j^{-1}B^T$ are still difficult to apply the inverses of, since in general they are dense and expensive to assemble. Thus, [48] proposed an approximation based on the commutator argument

$$BW_j^{-1}B^T \approx M_p \left(M_p + c_{j,1}M_p + c_{j,2}L_p\right)^{-1} K_p = \left(K_p^{-1} + c_{j,1}K_p^{-1} + \nu c_{j,2}M_p^{-1}\right)^{-1} = \widehat{S}_j.$$

Note that $L_p = \nu K_p$, because we consider the Stokes problem. Plugging this into (3.10) results in a preconditioner denoted by $\widetilde{P}_j$.

3.3.3 Constant Preconditioner

The most immediate idea for a preconditioner of the system (3.5) is to start with $\mathcal{P}_C$ and replace all Z_j by approximations. The structure of the resulting preconditioner, given some approximations $\bar{Z}_j$, is outlined in Algorithm 3.1. The above analysis suggests that $\widehat{P}_j$ and $\widetilde{P}_j$ are good candidates. We will identify the preconditioners obtained using these approximations at every time point by $\widehat{\mathcal{P}}_C$ and $\widetilde{\mathcal{P}}_C$. As these particular preconditioners define a fixed, constant operator (as opposed to other approaches), they are referred to as *constant* preconditioners.

Similar to $\mathcal{P}_C$, we can derive estimates for the eigenvalues of the preconditioned systems $\widehat{\mathcal{P}}_C^{-1}\mathcal{A}$ and $\widetilde{\mathcal{P}}_C^{-1}\mathcal{A}$. The estimates are centered around the fact that $\mathcal{P}_C$ is a low-rank perturbation of the original system $\mathcal{A}$. More precisely, we have

$$\mathcal{A} = \mathcal{P}_C + \mathcal{P}_R,$$

Algorithm 3.1: Constant preconditioner $\bar{\mathcal{P}}_C$ for all-at-once system given an approximation $\bar{Z}_j$ of Z_j

Function $\bar{\mathcal{P}}_C^{-1}(r)$**:**

$\hat{r} \leftarrow \mathcal{F}r$

Permute $\hat{r}$ to arrive at block-diagonal system

for $j \in \{1, \ldots, n_t - 1\}$ **do**

$\hat{s}_j \leftarrow \left(T_j^{(l)}\right)^{-1} \hat{r}_j$

$\hat{x}_j \leftarrow \bar{Z}_j^{-1} \hat{s}_j$

$\hat{y}_j \leftarrow \left(T_j^{(r)}\right)^{-1} \hat{x}_j$

end

Reverse permutation of $\hat{y}$

$y \leftarrow \mathcal{F}^{-1}\hat{y}$

return y

where the matrix $\mathcal{P}_R$ has rank $2n_v$. Because $\widehat{\mathcal{P}}_C$ is symmetric positive definite, we have that the eigenvalue analysis of the preconditioned system can be carried out on the following similar matrices:

$$\widehat{\mathcal{P}}_C^{-1}\mathcal{A} \sim \underbrace{\widehat{\mathcal{P}}_C^{-1/2}\mathcal{A}\widehat{\mathcal{P}}_C^{-1/2}}_{=\mathcal{H}} = \underbrace{\widehat{\mathcal{P}}_C^{-1/2}\mathcal{P}_C\widehat{\mathcal{P}}_C^{-1/2}}_{=\mathcal{K}} + \underbrace{\widehat{\mathcal{P}}_C^{-1/2}\mathcal{P}_R\widehat{\mathcal{P}}_C^{-1/2}}_{=\mathcal{L}}. \tag{3.12}$$

Now, let the sequences η_i, κ_i, and λ_i be the eigenvalues of the matrices $\mathcal{H}$, $\mathcal{K}$, and $\mathcal{L}$ in non-increasing order. Each eigenvalue occurs in the sequence according to its multiplicity. The matrix $\mathcal{K}$ has the same eigenvalues as $\widehat{\mathcal{P}}_C^{-1}\mathcal{P}_C$, whose eigenvalues are determined by the preconditioned subblocks, i.e.,

$$\lambda\left(\widehat{\mathcal{P}}_C^{-1}\mathcal{P}_C\right) = \bigcup_j \lambda\left(\widehat{P}_j^{-1}Z_j\right).$$

This allows us to establish bounds $\hat{a}, \hat{b} > 0$ such that

$$|\lambda(\mathcal{K})| = \left|\lambda\left(\widehat{\mathcal{P}}_C^{-1}\mathcal{P}_C\right)\right| \in [\hat{a}, \hat{b}].$$

Due to the specific saddle-point structure of $\mathcal{P}_C$, Sylvester's law of inertia implies that the eigenvalues can be divided into two equally sized chunks of negative and positive eigenvalues, i.e., it holds that

$$\kappa_i \in [\hat{a}, \hat{b}] \text{ for } i \in I_\kappa^+, \quad \kappa_i \in [-\hat{b}, -\hat{a}] \text{ for } i \in I_\kappa^-,$$

with the index sets

$$I_\kappa^+ = \left\{1, \ldots, \frac{N}{2}\right\}, \quad I_\kappa^- = \left\{\frac{N}{2} + 1, \ldots, N\right\}.$$

where $N = 2(n_t - 1)(n_v + n_p)$. Due to the low-rank structure of $\mathcal{P}_R$, almost all eigenvalues λ_i are zero. Specifically, there are $2n_v$ non-zero eigenvalues. Although we do not have bounds on the magnitude of these eigenvalues, we can use a similar procedure as before to show with Sylvester's law of inertia that

$$\lambda_i \geq 0 \text{ for } i \in I_\lambda^+, \quad \lambda_i = 0 \text{ for } i \in I_\lambda^0, \quad \lambda_i \leq 0 \text{ for } i \in I_\lambda^-,$$

where the index sets are given by

$$I_\lambda^+ = \{1, \ldots, n_v\}, \quad I_\lambda^0 = \{n_v + 1, \ldots, N - n_v\}, \quad I_\lambda^- = \{N - n_v + 1, \ldots, N\}.$$

With that, we may derive estimates for η_i, which are broadly based on the following result from [65]:

Lemma 3.1. *Let $\bar{A}$ and $\bar{B}$ be arbitrary but fixed real symmetric matrices of size $n \times n$ and set $\bar{C} = \bar{A} + \bar{B}$. Denote the eigenvalues of $\bar{A}$, $\bar{B}$, and $\bar{C}$ by the sequences α_i, β_i, and γ_i in non-increasing order. Each eigenvalue occurs in the sequence according to its multiplicity. Then, for all $i + j - 1 \leq n$, it holds that*

$$\gamma_{i+j-1} \leq \alpha_i + \beta_j.$$

This allows us to carry over the initial estimate for $\mathcal{K}$ to a subset of the eigenvalues of $\mathcal{H}$.

Theorem 3.2. *The preconditioner $\widehat{\mathcal{P}}_C$ fulfills*

$$\# \left\{ \mu \in \lambda \left(\widehat{\mathcal{P}}_C^{-1} \mathcal{A} \right) \mid |\mu| \in [\hat{a}, \hat{b}] \right\} \geq 2(n_t - 1)(n_v + n_p) - 4n_v.$$

Proof. Let the matrices $\mathcal{H}$, $\mathcal{K}$, and $\mathcal{L}$ and the corresponding eigenvalue sequences be defined as in (3.12). First, we establish an estimate from above by $\hat{b}$ for some of the eigenvalues η_k of $\widehat{\mathcal{P}}_C^{-1} \mathcal{A}$. The estimate of Lemma 3.1

$$\eta_{i+j-1} \leq \kappa_i + \lambda_j \text{ for } i + j - 1 \leq N$$

simplifies to

$$\eta_{i+n_v} \leq \kappa_i \text{ for } i \leq N - n_v$$

by setting $j = n_v + 1$, since $n_v + 1 \in I_\lambda^0$. The estimate $\kappa_i \leq \hat{b}$ holds for $i \in I_\kappa^+$. Thus, we want to restrict ourselves to the case $i \leq \frac{N}{2}$. This allows us to infer that

$$\eta_{i+n_v} \leq \kappa_i \leq \hat{b} \text{ for } i \in \left\{ j \in I_\kappa^+ \mid j + n_v \leq \frac{N}{2} \right\}.$$

By substituting the index of η, this implies

$$\eta_k \leq \hat{b} \text{ for } k \in I_\eta^{\hat{b}} = \left\{ n_v + 1, \ldots, \frac{N}{2} \right\}.$$

Next, we determine a set of eigenvalues for which the upper bound $-\hat{a}$ holds. The same theorem can be used for that, but instead of $i \in I_\kappa^+$, we are interested in indices $i \in I_\kappa^-$, since for this set we have the estimate $\kappa_i \leq -\hat{a}$. Hence,

$$\eta_{i+n_v} \leq \kappa_i \leq -\hat{a} \text{ for } i \in \left\{ j \in I_\kappa^- \mid j + n_v \leq N \right\},$$

which leads to the estimate

$$\eta_k \leq -\hat{a} \text{ for } k \in I_\eta^{-\hat{a}} = \left\{ \frac{N}{2} + n_v + 1, \ldots, N \right\}.$$

The matrix $\mathcal{H}^- := -\mathcal{K} - \mathcal{L}$ has eigenvalues $\eta_k^- = -\eta_{N-k+1}$. If we follow the same procedure as above for η_k^-, we receive the estimates

$$\begin{aligned} \eta_k^- &\leq \hat{b} \text{ for } k \in I_\eta^{\hat{b}}, \\ \eta_k^- &\leq -\hat{a} \text{ for } k \in I_\eta^{-\hat{a}}, \end{aligned}$$

which in turn leads to

$$\begin{aligned} \eta_k &\geq -\hat{b} \text{ for } k \in I_\eta^{-\hat{b}} = \left\{ \frac{N}{2} + 1, \ldots, N - n_v \right\}, \\ \eta_k &\geq \hat{a} \text{ for } k \in I_\eta^{\hat{a}} = \left\{ 1, \ldots, \frac{N}{2} - n_v \right\}. \end{aligned}$$

Hence,

$$\begin{aligned} \eta_k &\in [\hat{a}, \hat{b}] \qquad \text{for } k \in I_\eta^+ = I_\eta^{\hat{a}} \cap I_\eta^{\hat{b}} = \left\{ n_v + 1, \ldots, \frac{N}{2} - n_v \right\}, \\ \eta_k &\in [-\hat{b}, -\hat{a}] \text{ for } k \in I_\eta^- = I_\eta^{-\hat{a}} \cap I_\eta^{-\hat{b}} = \left\{ \frac{N}{2} + n_v + 1, \ldots, N - n_v \right\}. \end{aligned}$$

Finally, we get the result by determining the size of the set $|I_\eta^+ \cup I_\eta^-| = N - 4n_v$. □

The same idea can be applied to the matrix $\widetilde{\mathcal{P}}_C^{-1}\mathcal{A}$ by first applying the following lemma:

Lemma 3.2. *Assume that $\bar{A}$ and $\bar{B}$ are symmetric positive definite matrices and that $\bar{C}$ is a symmetric, invertible matrix, with the following properties:*

$$\left| \lambda(\bar{A}^{-1}\bar{B}) \right| \in [a, b], \quad \left| \lambda(\bar{B}^{-1}\bar{C}) \right| \in [c, d].$$

Then, the product $\bar{A}^{-1}\bar{C}$ fulfills

$$\left|\lambda(\bar{A}^{-1}\bar{C})\right| \in [ac, bd].$$

Proof. First, we consider the upper bound. It holds that

$$\begin{aligned}
\max\left|\lambda(\bar{A}^{-1}\bar{C})\right| &= \max\left|\lambda(\bar{A}^{-1/2}\bar{C}\bar{A}^{-1/2})\right| \\
&= \max\left|\lambda(\bar{A}^{-1/2}\bar{B}^{1/2}\bar{B}^{-1/2}\bar{C}\bar{B}^{-1/2}\bar{B}^{1/2}\bar{A}^{-1/2})\right| \\
&= \left\|\bar{A}^{-1/2}\bar{B}^{1/2}\bar{B}^{-1/2}\bar{C}\bar{B}^{-1/2}\bar{B}^{1/2}\bar{A}^{-1/2}\right\|_2 \\
&\leq \left\|\bar{A}^{-1/2}\bar{B}^{1/2}\right\|_2 \left\|\bar{B}^{-1/2}\bar{C}\bar{B}^{-1/2}\right\|_2 \left\|\bar{B}^{1/2}\bar{A}^{-1/2}\right\|_2 \\
&= \max\left|\lambda(\bar{A}^{-1}\bar{B})\right| \max\left|\lambda(\bar{B}^{-1}\bar{C})\right| \\
&\leq bd,
\end{aligned}$$

where we used that $\max\left|\lambda(\bar{M})\right| = \|\bar{M}\|_2$ for any symmetric matrix $\bar{M}$. The lower bound follows from proceeding as above but with $(\bar{A}^{-1}\bar{C})^{-1}$ instead of $\bar{A}^{-1}\bar{C}$. Since

$$\lambda((\bar{A}^{-1}\bar{C})^{-1}) = \{\lambda^{-1} \mid \lambda \in \lambda(\bar{A}^{-1}\bar{C})\},$$

we get the upper bound

$$\min\left|\lambda(\bar{A}^{-1}\bar{C})\right| \geq ac. \qquad \square$$

Since the commutator argument gives robust bounds $\widehat{S}_j^{-1}BW_j^{-1}B^T \in [a, b]$, it holds that

$$\left|\lambda\left(\widetilde{\mathcal{P}}_C^{-1}\mathcal{P}_C\right)\right| \in [\hat{a}a, \hat{b}b] =: [\tilde{a}, \tilde{b}],$$

which makes the following theorem a direct consequence using the same reasoning as in Theorem 3.2:

Theorem 3.3. *The preconditioner $\widetilde{\mathcal{P}}_C$ fulfills*

$$\#\left\{\mu \in \lambda\left(\widetilde{\mathcal{P}}_C^{-1}\mathcal{A}\right) \mid |\mu| \in [\tilde{a}, \tilde{b}]\right\} \geq 2(n_t - 1)(n_v + n_p) - 4n_v.$$

Theorems 3.2 and 3.3 show that the vast majority of the eigenvalues of the relevant preconditioned systems are tightly contained in positive or negative clusters bounded away from the origin, with at most $4n_v$ outlier eigenvalues due to the low-rank update used to enable a parallelizable-in-time block circulant approximation.

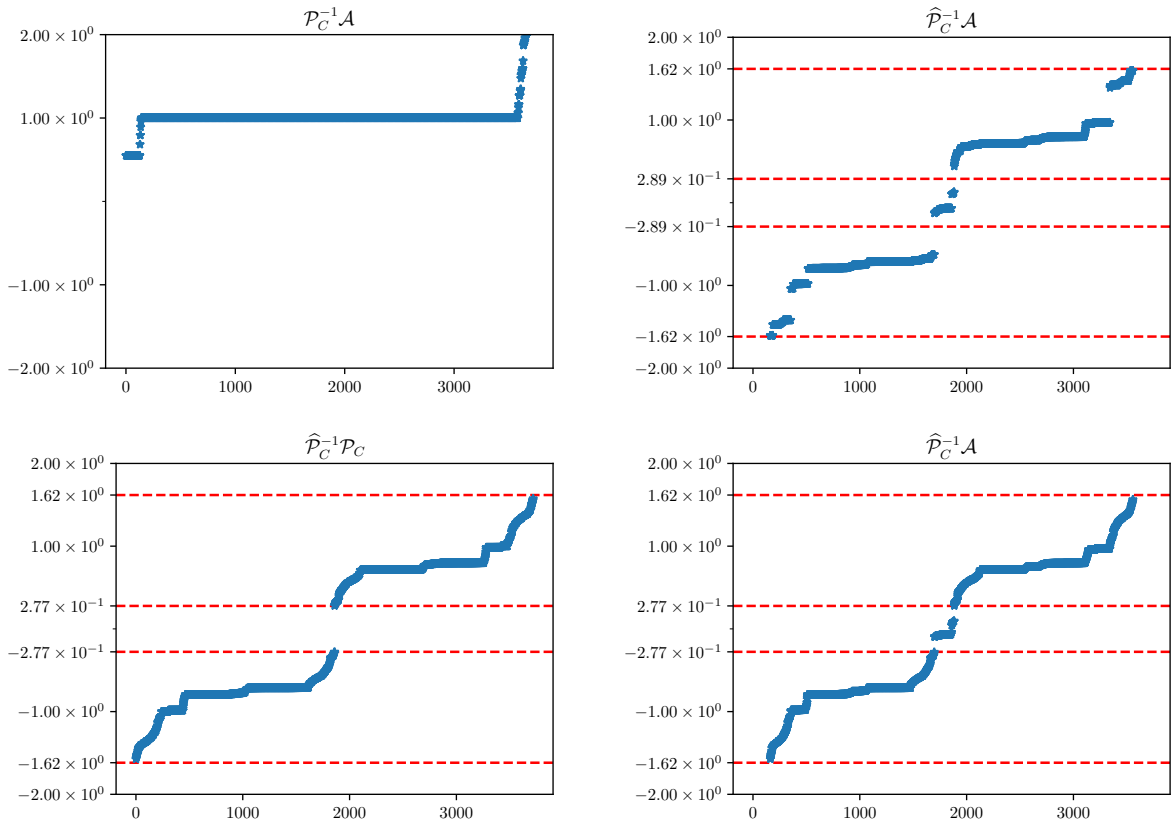


Figure 3.1: The (ordered) eigenvalues of the preconditioned matrices, with $n_t = 11$. The vertical axis presents the eigenvalues (or, in the top left plot, their real part) and the horizontal axis shows the index.

Theorems 3.1, 3.2, and 3.3 can be verified in practice. Figure 3.1 shows the real parts of the eigenvalues of some preconditioned systems. We restrict ourselves to rather small problems with $n_t = 11$ for this analysis due to the memory and computational requirements incurred by the computation of the eigenvalues. The top left plot shows that the majority of the eigenvalues of $\mathcal{P}_C^{-1}\mathcal{A}$ is 1 and the ratio of the remaining eigenvalues is less than $1/(n_t-1)$. In the top right figure, we see that the majority of the eigenvalues of $\widehat{\mathcal{P}}_C^{-1}\mathcal{A}$ are within the theoretical bounds (3.11), which are depicted as the red dashed lines in agreement with the results of Theorem 3.2. The bottom left plot verifies Lemma 3.2 in combination with the bounds arising from the application of the Schur complement approximation (3.10) and the commutator argument. Lastly, we can observe the implications of Theorem 3.3, in which case we can see that some of the eigenvalues are pushed outside of the bounds.

The potential to exploit parallelizability of $\widehat{\mathcal{P}}_C$ and $\widetilde{\mathcal{P}}_C$ is constrained by how often the preconditioners have to be applied. This limitation arises from the need for communication between each application of the preconditioners. If we assume that a more accurate approximation of the subblocks leads to a reduction in the number of iterations, then there exists a trade-off between

Algorithm 3.2: Non-constant version of preconditioner for all-at-once system

Function $\left(\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}\right)^{-1}(r)$**:**

$\hat{r} \leftarrow \mathcal{F}r$

Permute $\hat{r}$ to arrive at block-diagonal system

for $j \in \{1, \ldots, n_t - 1\}$ **do**

$\hat{s}_j \leftarrow \left(T_j^{(l)}\right)^{-1} \hat{r}_j$

$\hat{x}_j \leftarrow \mathrm{GMRES}(Z_j, \widehat{P}_j, \hat{s}_j, \mathrm{tol} = \varepsilon)$

$\hat{y}_j \leftarrow \left(T_j^{(r)}\right)^{-1} \hat{x}_j$

end

Reverse permutation of $\hat{y}$

$y \leftarrow \mathcal{F}^{-1}\hat{y}$

return y

the quality of the approximation of the subblocks G_j and the number of iterations required by the Krylov subspace solver. As we will see in the experiments, the above preconditioners do not provide a sufficient level of accuracy. Thus, we will look at preconditioner designs which use the approximations described above to enhance the performance of an inner solver.

3.3.4 Non-Constant Preconditioner

Theorem 3.3 suggests that the performance of $\widetilde{\mathcal{P}}_C$ might be improved by narrowing the bounds $\tilde{a}$ and $\tilde{b}$, which is equivalent to solving for G_j more accurately. Since we exclude the idea of using a direct solver due to its computational cost for very finely discretized problems, we propose to solve the subblocks to a certain relative tolerance only. As there is a good preconditioner $\widetilde{P}_j$ available for G_j, the subsystems can be solved with a preconditioned Krylov subspace method such as MINRES and GMRES, which will be called the *inner solver*. Accordingly, the tolerance $\varepsilon > 0$ of the subsystems will be called the *inner tolerance*. Since the preconditioner for the subsystems gives a good clustering of the eigenvalues, the inner solver will converge quickly and robustly with respect to the number of spatial degrees of freedom. This preconditioner will be denoted as $\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}$.

We will call the iteration for solving the all-at-once system the *outer iteration*. With the constant preconditioner, we would propose using MINRES or GMRES for the outer iteration, but this is no longer possible in this case. Since the inner solver is a Krylov method, it is a nonlinear operator, as opposed to a constant preconditioner which GMRES would require. A method that can cope with varying preconditioners at each iteration, is flexible GMRES (FGMRES), cf. [30]. The pseudocode for the preconditioner is presented in Algorithm 3.2.

It is important to remark that we should ideally not require many outer iterations, since the memory requirements of FGMRES grow linearly with the number of iterations. Let N be the number of degrees of freedom for the all-at-once optimal control problem. Then, with q denoting the number of outer iterations, we need to allocate memory in the order of $O(Nq)$. Considering

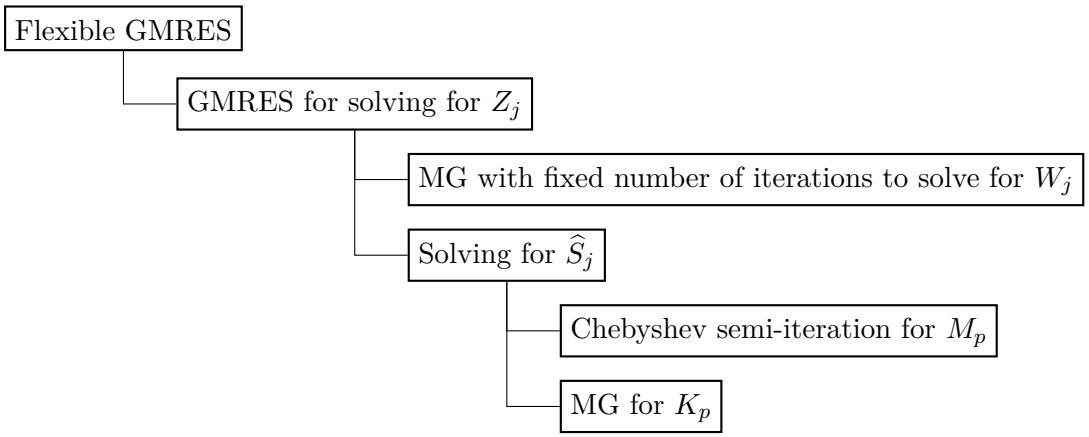


Figure 3.2: Overview of the solvers used for the application of $\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}$. Multigrid is denoted by 'MG'.

problems with possibly billions of degrees of freedom, it becomes clear that this reaches the limits of contemporary computing facilities rather quickly. Intuitively, we would expect that if the solution of the subsystems is better, i.e., if ε is smaller, fewer outer iterations are required. This would give us better control of the aforementioned bottleneck.

The non-constant preconditioner requires one to approximate the inverse of several finite element matrices. Since the exact solution of systems with these matrices can be expensive and can often be approximated without significant loss in the preconditioner's efficiency, we choose to use approximate solutions. One of the key methods here is the Chebyshev semi-iteration (cf. [64]), which shows fast convergence for mass matrices due to existing robust spectral estimates for the mass matrix, see e.g., [33] and [32]. The other method utilized is a geometric multigrid method (MG, see [66, 67]), of which various modes exist. In this work, we choose to apply such a multigrid method with successive over-relaxation for the smoothing. An overview of where these methods are applied within the preconditioner can be found in Figure 3.2.

3.4 Preconditioner for the Oseen Control Problem

Additionally to the Stokes problem, we would like to consider the more general Oseen flow control problem, i.e., the case when $w \neq 0$. Since we allowed L to be non-symmetric in Section 3.3.1, the idea of the circulant approximation of the system and its subsequent diagonalization can be transferred to the Oseen problem without any adaptations. In practice, preconditioners based on the idea of Algorithm 3.2 show promising results. We will apply the same technique to this problem and, thus, we are interested in developing an efficient preconditioner for G_j. Unfortunately, the transformation described by (3.9) does not apply to the Oseen problem, since it relies on the symmetry of L. This leaves us with developing preconditioners for G_j directly rather than for the simpler matrix Z_j.

First, we will partition G_j into a 2-by-2 block form, look at each of the blocks separately, and discuss their approximations. We define the (1,1)-block as

$$G_j^{(1,1)} = \begin{pmatrix} \tau M & d_j^* M + \tau L^T \\ d_j M + \tau L & -\frac{\tau}{\beta} M \end{pmatrix},$$

which is a saddle-point system. The special form of $G_j^{(1,1)}$ allows us to use (3.7) in order to achieve the following robust preconditioner:

$$P_j^{(1,1)} = \begin{pmatrix} \tau M & 0 \\ d_j M + \tau L & \frac{1}{\tau} Q_j M^{-1} Q_j^H \end{pmatrix}, \quad Q_j = d_j M + \tau L + \frac{\tau}{\sqrt{\beta}} M.$$

Although the eigenvalue bounds for (3.7) were originally established for real matrices only, it can be shown that these bounds also hold for the above complex matrices. Indeed, we can show the sufficient condition for the off-diagonal matrix blocks is

$$d_j M + \tau L + d_j^* M + \tau L^T = 2 d_{j,r} M + \tau \left(L + L^T \right) \succeq 0.$$

This is due to the property $d_{j,r} \geq 0$, seen by applying Gershgorin's theorem to C, and the fact that $L + L^T$ is a (scaled) discretized stiffness matrix, which is symmetric positive definite. The expected eigenvalue bounds for the preconditioned system were also verified numerically. The Schur complement of (3.8) has the form

$$S_j = \begin{pmatrix} 0 & \tau B \\ \tau B & 0 \end{pmatrix} \begin{pmatrix} \tau M & d_j^* M + \tau L^T \\ d_j M + \tau L & -\frac{\tau}{\beta} M \end{pmatrix}^{-1} \begin{pmatrix} 0 & \tau B^T \\ \tau B^T & 0 \end{pmatrix}.$$

Leveraging the commutator argument, we derive the following approximation:

$$S_j \approx \widehat{S}_j = \begin{pmatrix} 0 & \tau M_p \\ \tau M_p & 0 \end{pmatrix} \begin{pmatrix} \tau M_p & d_j^* M_p + \tau L_p^T \\ d_j M_p + \tau L_p & -\frac{\tau}{\beta} M_p \end{pmatrix}^{-1} \begin{pmatrix} 0 & \tau K_p \\ \tau K_p & 0 \end{pmatrix}.$$

Now, we have to put this together to efficiently precondition the overall system G_j. One idea would be to use the preconditioner (3.6) with the above approximations, which leads to

$$\widetilde{P}_j = \begin{pmatrix} P_j^{(1,1)} & 0 \\ G_j^{(2,1)} & \widehat{S}_j \end{pmatrix}. \tag{3.13}$$

In practice, however, this preconditioner does not perform well if the approximation of the (1,1)-block is not accurate enough. Indeed, if $P_j^{(1,1)}$ in (3.13) is replaced by its exact counterpart $G_j^{(1,1)}$ the performance was acceptable in experiments. This lets us conclude that a key contributor to the failure of P_j is that the approximation of the (1,1)-block does not suffice for a good overall performance. We, therefore, try to improve the approximation of this block.

Algorithm 3.3: Computation of the preconditioner $\widetilde{P}_j^{(\mathrm{UZ})}$

Function $(\widetilde{P}_j^{(\mathrm{UZ})})^{-1}(r)$**:**

Decompose $r \leftarrow \begin{pmatrix} r_1^T & r_2^T & r_3^T & r_4^T \end{pmatrix}^T$

Solve for $\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ according to (3.14) using initial guess $x_1^{(0)} = x_2^{(0)} = 0$, right-hand sides $b_1 = r_1$, $b_2 = r_2$

Compute $\begin{pmatrix} y_3 \\ y_4 \end{pmatrix} \leftarrow -\widehat{S}_j^{-1} \left(\begin{pmatrix} r_3 \\ r_4 \end{pmatrix} - \tau \begin{pmatrix} By_2 \\ By_1 \end{pmatrix} \right)$

return $\begin{pmatrix} y_1^T & y_2^T & y_3^T & y_4^T \end{pmatrix}^T$

One could follow the preceding procedure and nest a nonlinear Krylov subspace solver for the (1,1)-block in (3.13). Nevertheless, this would require us to use FGMRES instead of GMRES in Algorithm 3.2 and, therefore, lead to a threefold nested application of Krylov subspace solvers. The resulting additional complexity and the limited convergence theory of FGMRES gives reason to instead develop a linear approximation of $G_j^{(1,1)}$. A good candidate for that is the Uzawa iteration with a fixed number of iterations. More precisely, we use the inexact preconditioned Uzawa method or preconditioned Arrow–Hurwicz method, which is given by the following iterative form:

$$\begin{aligned} x_1^{(k+1)} &= x_1^{(k)} + \frac{1}{\tau}\widehat{M}^{-1}\left(b_1 - \tau M x_1^{(k)} - (d_j M + \tau L)^H x_2^{(k)}\right), \\ x_2^{(k+1)} &= x_2^{(k)} - \frac{1}{\mu}\widehat{S}^{-1}\left(b_2 - (d_j M + \tau L) x_1^{(k+1)} + \frac{\tau}{\beta} M x_2^{(k)}\right), \end{aligned} \tag{3.14}$$

for the solution of a saddle-point system of the form

$$G_j^{(1,1)} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where $\widehat{M}$ is an approximation of the mass matrix and $\mu > 0$ is a parameter of the method. For a detailed discussion of the Uzawa method, we refer to [34]. A suitable parameter choice for μ is

$$\mu = \frac{\lambda_{\min}(\widehat{S}^{-1}S) + \lambda_{\max}(\widehat{S}^{-1}S)}{2}.$$

In our case, the minimum and maximum eigenvalues of $\widehat{S}^{-1}S$ are $1/2$ and 1. Thus, we choose $\mu = 3/4$. The resulting preconditioner is denoted by $\widetilde{P}_j^{(\mathrm{UZ})}$ and the computation of its inverse is summarized in Algorithm 3.3. Since the number of iterations for the Uzawa method is fixed, $\widetilde{P}_j^{(\mathrm{UZ})}$ is constant over each iteration. Thus, it can be used as a preconditioner for the solver within the non-constant preconditioner design. The overall non-constant preconditioner will be denoted by $\widetilde{\mathcal{P}}_C^{(\mathrm{UZ})}$.

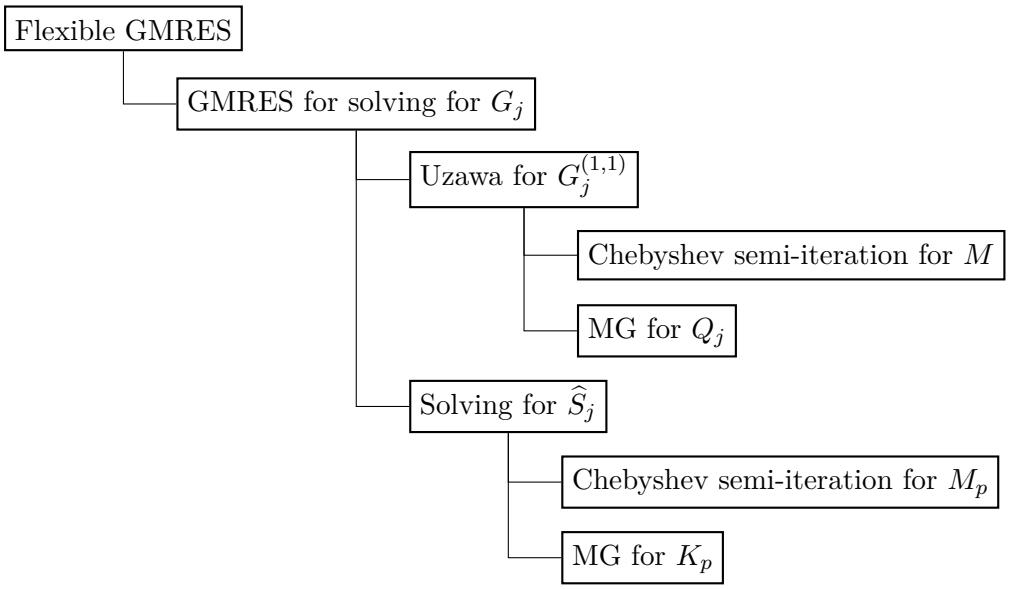


Figure 3.3: Overview of the solvers used for the application of $\widetilde{\mathcal{P}}_C^{(\mathrm{UZ})}$.

Similarly to the Stokes control problem, the final form of the preconditioner does not solve for the finite element matrices exactly but uses approximations instead. Figure 3.3 depicts the structure of the algorithm and where these approximations are used.

3.5 Numerical Experiments

The performance of the preconditioners motivated above will now be showcased with numerical experiments. We will first analyze the robustness of the preconditioner with respect to some model parameters and the number of timesteps, followed by a presentation of the potential computational speedup through parallelization. The experiments are conducted for two-dimensional Stokes and Oseen problems on the domain $\Omega = [-1, 1]^2$, where P_2-P_1-elements are used for its approximation. We restrict ourselves to two levels of spatial resolution, the coarse one having a maximum element size of $h_c = 1/32$ and the finer one having $h_f = 1/64$. Unless otherwise stated, the relative tolerance of the outer solver is set to $1\mathrm{e}-5$.

Within our implementation, we utilize the high-level finite element library DOLFINx [68] for spatial discretization. For miscellaneous numerical linear algebra, operations such as linear solvers, the libraries SciPy [69] and petsc4py [70] are used. Additionally to the functionalities for parallel computations shipped with petsc4py, the library mpi4py [71] is used for necessary non-standard communication within parallel infrastructures. The source code is publicly available under `https://github.com/bheinzelreiter/pintdiag-flow`. The experiments are run on varying numbers of processors of the model Intel(R) Xeon(R) CPU E7-4820 v2 @ 2.00GHz.

3.5.1 Stokes Problem

For the verification of the preconditioners discussed in Section 3.3, we consider the problem used in [72]. The exact solution of the problem is known and it is defined by the following desired velocity state, forcing term, and exact solution when $\nu = 1$:

$$
\begin{aligned}
v_d(x,t) = 4\beta \Big[& x_2 \left(2\left(3x_1^2 - 1\right)\left(x_2^2 - 1\right) + 3\left(x_1^2 - 1\right)^2 \right), \\
& -x_1 \left(3\left(x_2^2 - 1\right)^2 + 2\left(x_1^2 - 1\right)\left(3x_2^2 - 1\right) \right) \Big]^T \\
& + e^{T-t} \Big[20x_1x_2^3 + 2\beta x_2 \left(\left(x_1^2 - 1\right)^2 \left(x_2^2 - 7\right) - 4\left(3x_1^2 - 1\right)\left(x_2^2 - 1\right) + 2 \right), \\
& \quad 5\left(x_1^4 - x_2^4\right) - 2\beta x_1 \left(\left(x_2^2 - 1\right)^2 \left(x_1^2 - 7\right) - 4\left(x_1^2 - 1\right)\left(3x_2^2 - 1\right) - 2 \right) \Big]^T, \\
f(x,t) = e^{T-t} & \left[-20x_1x_2^3 - 2x_2\left(x_1^2 - 1\right)^2\left(x_2^2 - 1\right), 5\left(x_2^4 - x_1^4\right) + 2x_1\left(x_1^2 - 1\right)\left(x_2^2 - 1\right)^2 \right]^T \\
& + \left[2x_2\left(x_1^2 - 1\right)^2\left(x_2^2 - 1\right), -2x_1\left(x_1^2 - 1\right)\left(x_2^2 - 1\right)^2 \right]^T, \\
v(x,t) = e^{T-t} & \left[20x_1x_2^3, 5x_1^4 - 5x_2^4 \right]^T, \\
p(x,t) = e^{T-t} & \left(60x_1^2x_2 - 20x_2^3 \right) + \text{constant},
\end{aligned}
$$

where the initial and boundary conditions are given implicitly by $v(x,t)$. For our subsequent experiments, we retain the above choices of $v_d(x,t)$ and $f(x,t)$ for the desired state and forcing term.

We may use the known closed form of the exact solution to verify the implementation and observe the expected asymptotic scaling of the error when the space and time discretizations are refined. We report the errors of the solutions in Table 3.1. This shows the error measured in the $L^\infty(0,T;L^2(\Omega))$-norm for $T = 10$, $\nu = 1$, and values for β, h, n_t used in the following experiments. For $h = h_c$, we can observe that the error scales approximately as $O(\tau)$ for the lower values of n_t and stagnates for higher values. This indicates that, eventually, the error is dominated by the spatial discretization error as the time discretization is refined. For $h = h_f$, the approximate scaling $O(\tau)$ can be observed throughout the whole range of n_t. Furthermore, we can observe the expected scaling of the spatial discretization $O(h^2)$ for the maximal value of n_t. For this experiment only, the relative tolerance of the outer solver is set to $1\mathrm{e}-8$ so that the error from the Krylov solve is insignificant when calculating the errors reported in Table 3.1.

Table 3.1: Error of the numerical solution for the Stokes problem for different values of β, h, and n_t. The error is measured in the $L^\infty(0,T;L^2(\Omega))$-norm.

	$h = h_c$			$h = h_f$		
	$\beta = 1e{-}1$	$\beta = 1e{-}3$	$\beta = 1e{-}4$	$\beta = 1e{-}1$	$\beta = 1e{-}3$	$\beta = 1e{-}4$
n_t						
16	8.07e−4	5.70e−4	2.29e−4	8.06e−4	5.69e−4	2.26e−4
32	4.73e−4	3.40e−4	1.44e−4	4.71e−4	3.37e−4	1.35e−4
64	2.49e−4	1.87e−4	9.41e−5	2.42e−4	1.77e−4	7.35e−5
128	1.41e−4	1.09e−4	7.42e−5	1.28e−4	9.18e−5	3.80e−5
256	9.00e−5	7.90e−5	6.90e−5	6.61e−5	4.76e−5	2.08e−5
512	7.24e−5	7.01e−5	6.82e−5	3.43e−5	2.52e−5	1.28e−5
1024	6.90e−5	6.87e−5	6.83e−5	1.86e−5	1.45e−5	9.75e−6

Constant Preconditioner

We will now examine the constant preconditioner $\widetilde{\mathcal{P}}_C$, which is applied within a GMRES iterative solver. The iterative solver is configured to restart every 30 iterations. For this and all subsequent experiments, the viscosity is set to $\nu = 1e{-}2$, in each application of MG 4 V-cycles are used, and the number of Chebyshev semi-iterations for approximately applying the inverse of the mass matrix is set to 10. Unless otherwise stated, the final time is set to $T = 10$. Table 3.2 shows the numbers of GMRES iterations and CPU times (in seconds), for various numbers of timesteps and different values of the regularization parameter β. In this setting, all tests were run sequentially without leveraging the parallelizability in time. For certain large numbers of timesteps, the effective runtime would have exceeded the available resources. These experiments were computed in parallel and the runtime is reported as "$*$". The listed number of degrees of freedom (DOFs) includes all variables for the all-at-once formulation in space and time.

We can observe that the number of GMRES iterations grows significantly with the problem size for $\beta = 1e{-}1$, which is not the behaviour we are looking for in a preconditioner. For the other two values of β, the number of iterations remains in the same order independent of the problem size, which indicates that in this scenario $\widetilde{\mathcal{P}}_C$ shows some robustness concerning the number of timesteps. Furthermore, the computations are done sequentially, which means the runtime is expected to scale linearly with the overall problem size and the number of iterations of the solver. This aligns with the obtained data. The large number of iterations and the long runtime therefore incurred show the limitations in the applicability of this preconditioner.

Table 3.2: Solving the Stokes problem with $\widetilde{\mathcal{P}}_C$ for different parameter settings.

	$h = h_c$							$h = h_f$						
		$\beta = 1e{-}1$		$\beta = 1e{-}3$		$\beta = 1e{-}4$			$\beta = 1e{-}1$		$\beta = 1e{-}3$		$\beta = 1e{-}4$	
n_t	#DOFs	GMRES	CPU [s]	GMRES	CPU [s]	GMRES	CPU [s]	#DOFs	GMRES	CPU [s]	GMRES	CPU [s]	GMRES	CPU [s]
16	2.86e5	59	42	50	39	42	29	1.13e6	68	208	52	162	48	186
32	5.91e5	68	112	50	95	43	62	2.33e6	72	508	52	358	48	316
64	1.20e6	84	277	50	163	42	140	4.73e6	90	1245	52	732	48	601
128	2.42e6	134	1049	50	320	44	272	9.53e6	137	3788	54	1408	48	1347
256	4.86e6	252	2601	52	570	44	525	1.91e7	258	13440	54	2608	48	2676
512	9.75e6	522	$*$	54	$*$	44	$*$	3.83e7	796	$*$	66	$*$	48	$*$
1024	1.95e7	1277	$*$	81	$*$	46	$*$	7.67e7	$*$	$*$	$*$	$*$	$*$	$*$

Non-Constant Preconditioner

Next, we will consider the non-constant preconditioner $\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}$. As already mentioned, we have to use FGMRES as an outer iterative solver instead of GMRES. Given the substantial size of the problem and the increased memory requirements of FGMRES, we are required to use small numbers of iterations between each restart. Experiments suggest that 10 iterations are sufficient to ensure convergence for the considered problems. As an inner solver, we use GMRES. The inner tolerance is set to $\varepsilon = 1e{-}2$, which was found to be sufficient in numerical experiments, meaning that a smaller tolerance did not lead to a noticeable reduction of the outer iterations. All other parameters are chosen identically to those used in the experiments for the constant preconditioner. Tables 3.3 and 3.4 show convergence results for the two different levels of spatial discretization, specifically the number of outer FGMRES iterations, the average number of inner GMRES iterations (to the nearest integer), and the CPU time required. We are able to observe robustness of the number of outer iterations. Additionally, the number of inner iterations stays within the same order independent of the parameters, which ensures a linear scaling of the runtime in n_t. Additional experiments showed that the preconditioner can deal with smaller regimes of ν robustly. However, the MG iterations do not necessarily guarantee convergence for $\nu < 1e{-}3$, which is currently the bottleneck of the solver.

Table 3.3: Solving the Stokes problem with $\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}$, $h = h_c$.

		$\beta = 1\mathrm{e}{-1}$			$\beta = 1\mathrm{e}{-3}$			$\beta = 1\mathrm{e}{-4}$		
n_t	#DOFs	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]
16	2.86e5	4	32	59	3	26	49	3	33	52
32	5.91e5	5	34	226	4	26	127	3	26	96
64	1.20e6	5	29	276	4	23	173	4	25	239
128	2.42e6	6	29	932	4	21	446	4	21	395
256	4.86e6	6	26	1654	4	18	571	4	18	727
512	9.75e6	7	27	$*$	4	17	$*$	4	15	$*$
1024	1.95e7	7	26	$*$	5	23	$*$	5	21	$*$

Table 3.4: Solving the Stokes problem with $\widetilde{\mathcal{P}}_C^{(\mathrm{NC})}$, $h = h_f$.

		$\beta = 1\mathrm{e}{-1}$			$\beta = 1\mathrm{e}{-3}$			$\beta = 1\mathrm{e}{-4}$		
n_t	#DOFs	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]
16	1.13e6	4	34	360	3	35	252	3	39	301
32	2.33e6	4	32	764	4	28	641	3	31	474
64	4.73e6	4	30	1263	4	27	890	4	27	1140
128	9.53e6	5	33	3762	4	23	1735	4	25	2247
256	1.91e7	6	33	8483	4	21	3631	4	22	3324
512	3.83e7	6	30	$*$	4	17	$*$	4	19	$*$
1024	7.67e7	6	27	$*$	4	16	$*$	4	16	$*$

3.5.2 Oseen Problem

Next, we will analyze the performance of the preconditioner for the Oseen problem introduced in Section 3.4. The test problem is a lid-driven cavity problem inspired by the test problems for the Navier–Stokes equations in [72], with $T = 10$. The wind has two vortices and the desired state describes a velocity field with one vortex. It is given by the following data:

$$v_d(x,t) = \left[2x_2\left(1-x_1^4\right)\left(1-x_2^4\right), -2x_1\left(1-x_1^4\right)\left(1-x_2^4\right)\right]^T + \begin{cases} [5x_2-4,0]^T & \text{if } x_2 \geq \frac{4}{5}, \\ [0,0]^T & \text{otherwise,} \end{cases}$$

$$f(x,t) = \Big[-20x_1x_2^3 - 2x_2\left(x_1^2-1\right)^2\left(x_2^2-1\right), \\ 5\left(x_2^4-x_1^4\right) + 2x_1\left(x_1^2-1\right)\left(x_2^2-1\right)^2\Big]^T,$$

Table 3.5: Solving the Oseen problem with $\widetilde{\mathcal{P}}_C^{(\mathrm{UZ})}$, $h = h_c$.

		$\beta = 1\mathrm{e}{-1}$			$\beta = 1\mathrm{e}{-3}$			$\beta = 1\mathrm{e}{-4}$		
n_t	#DOFs	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]
16	2.86e5	5	7	114	3	3	35	3	5	51
32	5.91e5	6	6	260	4	4	116	3	4	98
64	1.20e6	8	6	1028	4	4	157	4	4	399
128	2.42e6	9	6	1562	4	4	467	4	4	502
256	4.86e6	9	5	2539	5	4	1384	4	4	1002
512	9.75e6	10	5	$*$	7	4	$*$	5	5	$*$
1024	1.95e7	10	5	$*$	8	4	$*$	6	6	$*$

$$v_0(x) = \begin{cases} [1,0]^T & \text{if } x_1 > -x_2 \text{ and } x_1 < x_2, \\ [0,0]^T & \text{otherwise,} \end{cases}$$

$$h(x,t) = \begin{cases} [1,0]^T & \text{if } x_2 = 1, \\ [0,0]^T & \text{otherwise,} \end{cases}$$

$$w(x) = \begin{cases} c_1(x)\left[\left(\frac{100}{99}\right)^2 x_2, -\left(\frac{100}{49}\right)^2 \left(x_1 - \frac{1}{2}\right)\right]^T & \text{if } c_1(x) \geq 0, \\ c_2(x)\left[\left(-\frac{100}{99}\right)^2 x_2, \left(\frac{100}{49}\right)^2 \left(x_1 - \frac{1}{2}\right)\right]^T & \text{if } c_2(x) \geq 0, \\ [0,0]^T & \text{otherwise,} \end{cases}$$

$$c_1(x) = 1 - \sqrt{\left(\frac{100}{49}\left(x_1 - \frac{1}{2}\right)\right)^2 + \left(\frac{100}{99} x_2\right)^2},$$

$$c_2(x) = 1 - \sqrt{\left(\frac{100}{49}\left(x_1 + \frac{1}{2}\right)\right)^2 + \left(\frac{100}{99} x_2\right)^2}.$$

Non-Constant Preconditioner

The preconditioner $\widetilde{\mathcal{P}}_C^{(\mathrm{UZ})}$ introduced in Section 3.4 is applied within the same iterative solver and associated settings as in Section 3.5.1. The number of iterations for the inner Uzawa method is set to 6. Tables 3.5 and 3.6 show a summary of the numerical results. Also for the Oseen problem, we observe robustness of the number of outer iterations as well as of the number of inner iterations. Accordingly, the total runtime scales linearly in n_t. Furthermore, convergence of the inner solver is achieved within few iterations, which suggests that $\widetilde{P}_j^{(\mathrm{UZ})}$ is a good candidate for preconditioning the subblocks.

Table 3.6: Solving the Oseen problem with $\widetilde{\mathcal{P}}_C^{(\mathrm{UZ})}$, $h = h_f$.

		$\beta = 1\mathrm{e}{-1}$			$\beta = 1\mathrm{e}{-3}$			$\beta = 1\mathrm{e}{-4}$		
n_t	#DOFs	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]	FGMRES	av. GMRES	CPU [s]
16	1.13e6	4	6	306	3	4	159	3	5	186
32	2.33e6	6	9	1263	4	5	503	3	5	281
64	4.73e6	6	7	2167	4	4	1097	3	5	870
128	9.53e6	7	7	4724	4	5	1999	4	4	1489
256	1.91e7	8	6	9507	5	5	4275	4	4	4158
512	3.83e7	9	5	$*$	6	4	$*$	4	5	$*$
1024	7.67e7	9	5	$*$	7	4	$*$	5	5	$*$

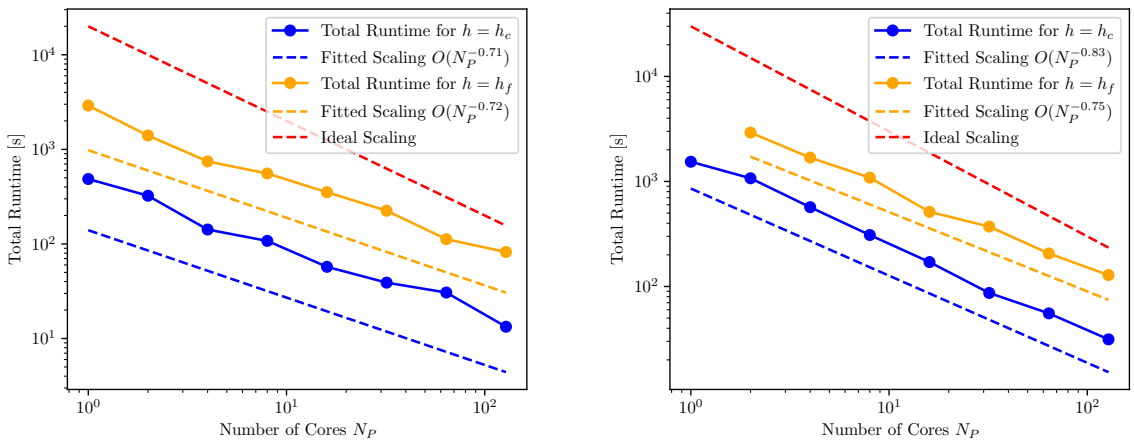


Figure 3.4: Strong runtime scaling with respect to available number of cores N_P for the Stokes (left) and Oseen problem (right) using the non-constant preconditioners.

3.5.3 Parallel Implementation

To test the preconditioners' potential for parallelizability, we present results showing how the runtime of a parallel implementation scales. Based on the preceding results, we will restrict ourselves to the discussed non-constant preconditioners only. The parallel implementation leverages petsc4py and mpi4py for communication, and the solution of each subblock is carried out in parallel. It is noted that the computation of the FFT is not parallelized in this implementation. Nevertheless, we are still able to observe consistent scaling in our experiments since the overall runtime is dominated by the solution of the individual subblocks, with the FFT contributing less than 2% of the total execution time. If larger problems are considered, this bottleneck significantly limits the method's scalability, making a fully parallel implementation essential.

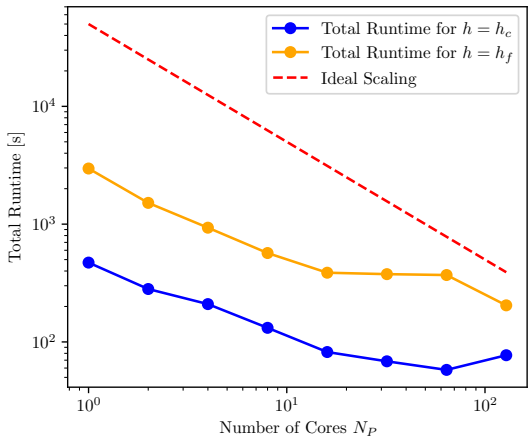


Figure 3.5: Strong runtime scaling with respect to available number of cores N_P for the Stokes problem using the constant preconditioner.

First, we will consider strong scaling. We set $n_t = 256$, $\beta = 1\text{e}{-3}$, and $T = 5$, and the inner tolerance is set to $\varepsilon = 1\text{e}{-2}$. Figure 3.4 shows how the runtime scales with the number of cores N_P available to the solver. For the Stokes problem, we achieve scalings of $O(N_P^{-0.71})$ and $O(N_P^{-0.72})$. The runtime scalings for the Oseen problem are slightly better with approximately $O(N_P^{-0.83})$ and $O(N_P^{-0.75})$. This can be explained by the increased complexity of the subblocks, resulting in a longer runtime for solving each of them. The loss in efficiency due to communication and the application of the FFT becomes less significant. In comparison, Figure 3.5 presents the strong scaling of the constant preconditioner for the Stokes problem, revealing that the performance gains from increased computational resources plateau much earlier than with the non-constant preconditioner, thereby demonstrating the superior scalability and parallelization potential of the non-constant preconditioning strategy. A snapshot of the solution to the considered benchmark problem is depicted in Figure 3.6. Therein, the timestamp is fixed to $t = T(n_t - 1)/n_t$ and the computations are carried out for $n_t = 32$.

Additionally, we consider weak scaling. The standard approach here would be to fix all model parameters except for n_t and N_P with the relation $n_t - 1 = \omega N_P$ for some constant $\omega \in \mathbb{N}$. However, we note it was observed that the first subblock requires significantly longer to solve than the others, so the workload for the first core is reduced by one subblock, i.e., the first core is assigned $\omega - 1$ subblocks and the others ω subblocks. Consequently, for our analysis we set $n_t = \omega N_P$. The remaining parameters are kept the same as in the previous experiments, with the regularization parameter set to $\beta = 1\text{e}{-3}$. In the case of ideal scaling, we would expect the runtime not to increase significantly with growing n_t. Table 3.7 shows the runtime of the solver for the Stokes and Oseen problem with coarse and fine spatial discretizations. With a factor of 2.87, the Stokes problem with $h = h_f$ shows the strongest relative increase in runtime from the smallest setting of n_t to the biggest. Hence, a 68.2 times bigger problem is solved at the cost of a 2.87 times increase in runtime in the worst-case scenario.

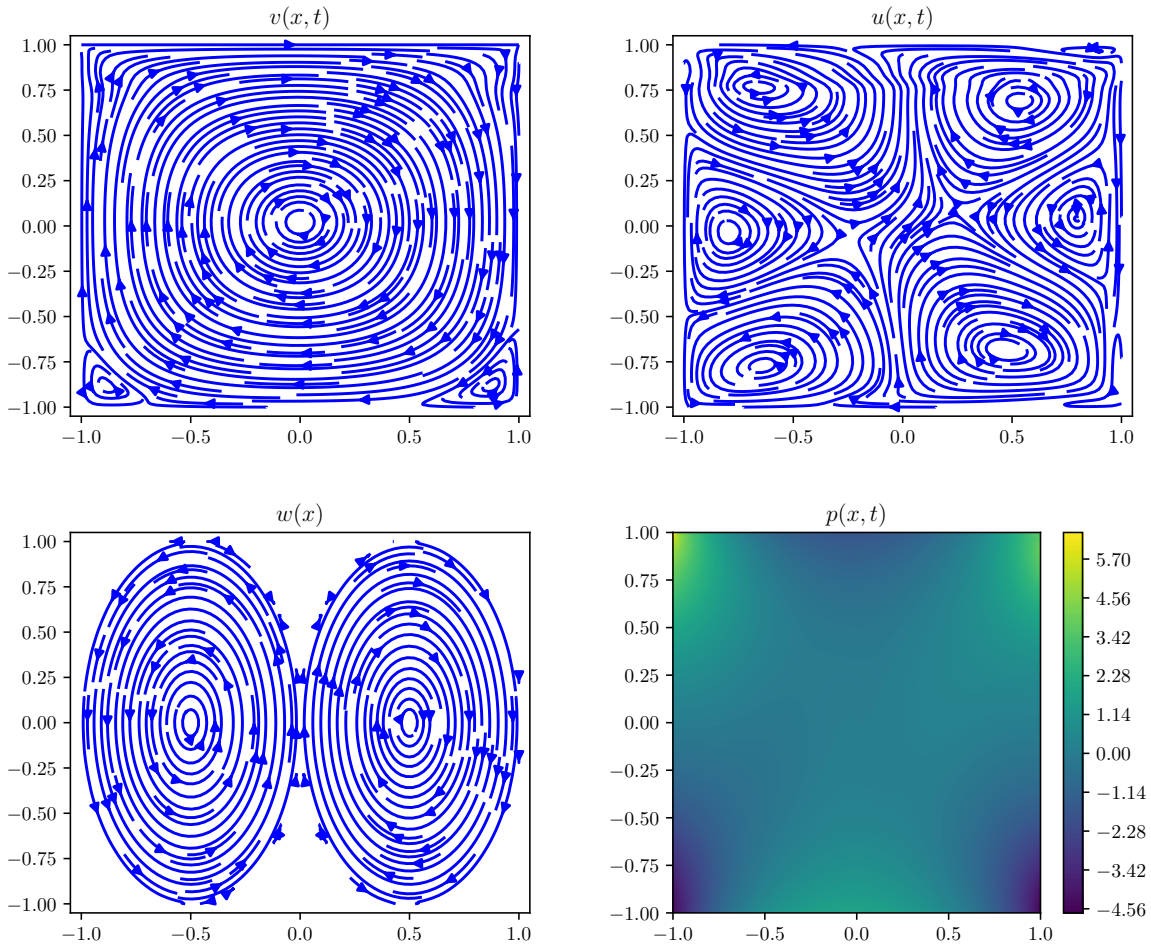


Figure 3.6: Snapshot of the solution to the problem used for benchmarking the strong scaling at timestamp $t = T(n_t - 1)/n_t$.

In summary, we have demonstrated that the preconditioners show good strong and weak scaling properties for the Stokes and Oseen problem. Thus, we can solve these large-scale problems efficiently with increasing computing power.

3.6 Conclusion

In this chapter, we introduced preconditioners for the efficient iterative solution of unsteady Stokes and Oseen control problems. The preconditioners approximated the original problem by its time-periodic equivalent, allowing us to perform a temporal diagonalization and achieve parallel-in-time solvers. The first preconditioner leveraged existing approximations of Stokes problems in order to create a constant preconditioner. To increase efficiency within parallel in-

Table 3.7: Runtime in seconds of solver for Stokes and Oseen problem. The number of processors is implicitly given by $n_t = 8N_P$.

n_t	16	32	64	128	256	512	1024
				$h = h_c$			
# DOFs	2.86e5	5.91e5	1.20e6	2.42e6	4.86e6	9.75e6	1.95e7
Stokes	30	30	32	36	37	56	78
Oseen	46	53	58	64	48	77	72
				$h = h_f$			
# DOFs	1.13e6	2.33e6	4.73e6	9.53e6	1.91e7	3.83e7	7.67e7
Stokes	135	141	162	206	295	307	388
Oseen	181	232	255	255	261	303	356

frastructures, we introduced additional preconditioners that made use of nested Krylov subspace solvers, and we achieved rapid convergence with these solvers, including for very large problems. Our approach also demonstrated significant robustness with respect to model parameters and the discretization level, as well as very good strong and weak scaling properties.

While these results are promising for linear and linearized flow control problems, extending our diagonalization-based approach to nonlinear PDEs remains an open challenge. The application of nonlinear solvers introduces time-varying problem structures that cannot be diagonalized using the methods presented here. A promising avenue for future research is examining how mildly nonlinear equations might still benefit from modified versions of our framework. One approach could employ averaging techniques as presented in [56] to handle mild nonlinearities. Additionally, advanced linearization techniques could enable the application of these methods to nonlinear autonomous flow control problems.

Chapter 4

Well-posedness, Convergence, and Efficient Numerical Methods for Carleman Linearization of Parabolic PDEs

Linear problems, as discussed in the previous chapter, often allow for effective use of their inherent special structure to develop potent numerical methods. However, these advantages are generally lost once nonlinearities are introduced into the equations. Many methods designed for linear problems become inapplicable, and new techniques specifically targeting nonlinearities must be developed—often at the cost of efficiency compared to the linear case. One common approach to managing the increased complexity due to nonlinearities is to linearize the system. Linearization techniques enable the application of established methods for linear problems to nonlinear cases. This process, however, introduces its own set of challenges and limitations. A fundamental example of linearization is Newton's method, which uses a first-order approximation around a reference state. This linearization provides accurate results only in a local neighborhood, and its quality can deteriorate rapidly as the system deviates from the reference state. Nevertheless, this is just one linearization technique; more advanced methods offer improved approximation properties, albeit at the cost of increased complexity in their analysis and computational resources. Given its potential benefits, the development and analysis of linearization methods for nonlinear dynamical systems remain significant and ongoing challenges in applied mathematics.

4.1 Introduction

Carleman linearization, introduced by Torsten Carleman in his seminal work [73], is a foundational technique for linearizing dynamical systems. A variant of this method, known as truncated Carleman linearization or finite-section approximation of the Carleman linearization, has garnered considerable attention. This approach effectively transforms nonlinear systems into approximate linear systems while retaining some of the original nonlinear behavior, thus making analysis and computation more tractable.

The relevance of the Carleman linearization is underscored by its diverse applications across multiple domains. In applied mathematics, it has been employed in optimal control to simplify the handling of nonlinear constraints, making it easier to derive control laws [74, 75, 76, 77]. In the field of model order reduction, the linearization has been utilized to extend established order-reducing methods to complex nonlinear systems, thus enabling more efficient simulations and analyses [78, 79]. It has also found application in system identification problems; see, e.g., [80]. More recently, the linearization has proven valuable for quantum computing applications as it facilitates the development of quantum algorithms to solve large-scale nonlinear dynamical systems [81, 82].

Since its generalization to nonlinear partial differential equations in [83], the Carleman linearization has been successfully applied in practice to approximate solutions to nonlinear PDEs. For instance, the method has been employed in a range of fluid flow applications, from simplified benchmark problems such as the Burgers' equation (see, e.g., [79]) to more complex fluid flow problems (see, e.g., [84, 85]), as well as reaction–diffusion problems [86]. These and related applications highlight the potential of the linearization technique as a powerful tool for addressing the complexities inherent in nonlinear infinite-dimensional dynamical systems.

A critical question in the application of the Carleman linearization is its accuracy in approximating the original nonlinear system, particularly the effect of truncation on this accuracy. This question is of paramount importance as it determines how well the linearized system can recover the nonlinear behavior of the original system. Error estimates of the truncated linearization have recently been established for finite-dimensional dynamical systems, i.e., nonlinear ordinary differential equations (ODEs). For instance, [87] provides comprehensive error estimates for quadratically nonlinear equations, while [88, 89] extend these estimates to general nonlinear systems. Although these theoretical estimates generally align well with experimental observations for ODEs, they fail to explain the behavior of the linearization when applied to certain infinite-dimensional dynamical systems, including a range of PDEs. A straightforward method to generalize the error estimates for PDEs involves first discretizing the equation to arrive at a finite-dimensional nonlinear dynamical system, which is then linearized, followed by applying the available error estimates from [89]. The primary issue in such case is that if the PDE is first discretized and then linearized, the existing error estimates depend on the properties of the discrete operators. For certain classes of nonlinearities, these bounds can be shown to be robust with respect to the discretization; see, e.g., [86] for nonlinear reaction terms. In the more general case, however, this dependency can lead to error estimates that are not robust, a problem first observed by [90] in the context of fluid flow problems. Despite this, it has been empirically observed that the linearization can approximate nonlinear behavior effectively, even in the context of infinite-dimensional systems and independently of the discretization. An additional challenge for infinite-dimensional systems that has not been addressed in existing literature is proving that the resulting linearization is well-posed.

In this chapter, we tackle the questions of well-posedness and convergence of the truncated Carleman linearization when applied to a class of infinite-dimensional parabolic systems. We establish both well-posedness and convergence in an infinite-dimensional and undiscretized setting, which allows for error estimates that are independent of any discretization of the system. This approach not only provides a more robust theoretical framework and better understanding of the linearization but also offers practical advantages in numerical applications. We demonstrate that this approach enables the separation of the linearization error from the discretization error through a so-called linearize-then-discretize strategy, ultimately allowing us to develop new numerical approximations that mitigate the curse of dimensionality associated with the linearization. This paves the way for a new class of efficient and accurate numerical methods, essential for the practical implementation of the linearization in real-world problems.

The structure of the chapter is as follows: In Section 4.2, we introduce the nonlinear dynamical system and outline the appropriate assumptions on the state space and the involved operators to guarantee the well-posedness of the nonlinear system. The problem setup covers parabolic second-order differential equations but is not limited to these. Section 4.3 offers a non-rigorous introduction to the concept of the truncated Carleman linearization, providing an intuitive understanding of the method. In Section 4.4, we equip the Carleman linearization with appropriate function spaces and demonstrate the well-posedness of the problem under additional assumptions. This section is crucial for establishing the theoretical validity of the linearization method in an infinite-dimensional setting. Section 4.5 shows how further assumptions ensure the convergence of the linearization and showcases certain standard systems that fulfill these assumptions, thereby illustrating the practical applicability of the theoretical results. Section 4.6 discusses how our approach motivates the development of efficient numerical methods. It is shown how the traditional Carleman linearization of PDEs, which is based on a discretization of the equation, is a special case of a greater class of numerical methods. Furthermore, this section highlights how our approach overcomes the shortcomings of the existing error bounds. The theoretical results are verified in Section 4.7 through a series of numerical experiments, which provide empirical evidence supporting the validity and effectiveness of the proposed convergence results and methods. Finally, Section 4.8 concludes the work and provides an outlook on potential future research directions, suggesting avenues for further exploration of efficient numerical methods.

4.2 Nonlinear Cauchy Problems in Hilbert Spaces

We are interested in the analysis of nonlinear Cauchy problems defined on an infinite-dimensional separable Hilbert space H. Specifically, we consider systems with quadratic nonlinearities, i.e., of the form

$$\begin{aligned} y'(t) + Ay(t) + B(y(t) \otimes y(t)) &= f(t) \quad \text{for a.e. } t \in (0,T), \\ y(0) &= y_0, \end{aligned} \tag{4.1}$$

where we seek a solution y, given some initial condition y_0 and forcing f. The final time can be bounded or unbounded, i.e., $T \in (0, \infty]$. The system is considered in the space H. We refer to A as the linear part of the dynamical system. It is assumed that $A : D(A) \to H$ is a closed invertible operator with a dense domain $D(A)$ continuously embedded in H. Moreover, A generates an analytic semigroup. The operator $B : D(A) \times D(A) \to H$ is assumed continuous and is referred to as the quadratic part of the system. The forcing lives in the space $L^2(0, T; H)$. If the equalities in system (4.1) are considered in H and $y_0 \in D(A)$, the dynamical system is called the strong form of the Cauchy problem.

4.2.1 Weak Formulation

Since our ultimate goal is the analysis of PDEs and their efficient numerical approximation, it is more suitable to consider the problem in its weak form. The following analysis presents the key ideas of the theory of parabolic systems and refines the assumptions on the present operators. For an in-depth discussion of the topic, the reader is referred to [91]. Let $V = [D(A), H]_{1/2}$ be the interpolation space between $D(A)$ and H, and V' be its topological dual. Let $(\cdot, \cdot)_H$ be the inner product of H and $\langle \cdot, \cdot \rangle_V$ the duality pairing of $V' \times V$. We assume that the embedding $V \hookrightarrow H \cong H' \hookrightarrow V'$ is continuous and dense, where each element $h \in H$ is identified with an element of H' via the mapping $g \mapsto \langle h, g \rangle_V = (h, g)_H$ for all $g \in H$. Then, (V, H, V') forms a Gelfand triple. Assume that A and B can be extended to $A : V \to V'$ and $B : V \times V \to V'$. Then, the weak form of the Cauchy problem reads as

$$\begin{aligned} y'(t) + Ay(t) + B(y(t) \otimes y(t)) = f(t) \quad \text{in } L^2(0, T; V'), \\ y(0) = y_0, \end{aligned} \tag{4.2}$$

where the equality is to be considered in the dual space $L^2(0, T; V')$ and $y_0 \in H$, and the solution is sought in the space $y \in W(0, T; V, V')$ with

$$\begin{aligned} W(0, T; X, Y) &= \left\{ y \in L^2(0, T; X) \mid y' \in L^2(0, T; Y) \right\}, \\ \|y\|^2_{W(0,T;X,Y)} &= \|y\|^2_{L^2(0,T;X)} + \|y'\|^2_{L^2(0,T;Y)} \end{aligned}$$

for any pair of Banach spaces X and Y. Due to the Lions–Magenes lemma [92], the space of continuous functions in H is continuously embedded in $W(0, T; V, V')$, i.e., $C(0, T; H) \hookrightarrow W(0, T; V, V')$, which makes point-wise evaluations well-posed. The weak formulation allows us to impose weaker regularity on the forcing $f \in L^2(0, T; V')$.

4.2.2 Linear Equation

We assume that V and V' admit an eigenvalue representation. For that, assume there exists a closed positive self-adjoint operator L with domain $D(L)$ such that $D(L^{\bar{\alpha}/2}) = D(A^{\bar{\alpha}/2})$ for some $\bar{\alpha} \geq 1$. This allows us to define the norm of V by the equivalent norm $\|\cdot\|_V := \|L^{1/2}\cdot\|_H$. Since L is self-adjoint, we can identify this norm and, hence, the space V with its eigenvectors and eigenvalues. Let $\{\varphi_i\}_{i\in\mathbb{N}}$ be eigenvectors of L forming an orthonormal basis of H and $\{\lambda_i\}_{i\in\mathbb{N}}$ its corresponding eigenvalues. Then, we can rewrite the V-norm as

$$\|v\|_V^2 = \sum_{i=1}^{\infty} \lambda_i \, (v, \varphi_i)_H^2 \, .$$

Assume that A is bounded, i.e.,

$$|\langle Av, w\rangle_V| \leq \beta \|v\|_V \|w\|_V. \tag{A1}$$

Furthermore, let A be V-H coercive (sometimes referred to as weakly coercive), which is achieved if there exist constants $\lambda \geq 0$ and $\gamma > 0$ with

$$\langle Av, v\rangle_V + \lambda \|v\|_H^2 \geq \gamma \|v\|_V^2 \quad \text{for all } v \in V. \tag{A2}$$

This implies that the linear equivalent of our Cauchy problem is well-posed.

Lemma 4.1. *Let $T < \infty$ and $A(t)$ be a family of V-H operators fulfilling* (A1) *and* (A2) *for $t \in (0,T)$ with constants β, γ, and λ independent of t. Then, for every $y_0 \in H$ and $f \in L^2(0,T;V')$, the linear Cauchy problem*

$$\begin{aligned} y'(t) + A(t)y(t) &= f(t) \quad \text{in } L^2(0,T;V'), \\ y(0) &= y_0 \end{aligned}$$

has a unique solution $y \in W(0,T;V,V')$. This solution satisfies the estimate

$$\|y(t)\|_H^2 + \gamma \int_0^t \exp(2\lambda(t-\tau))\|y(\tau)\|_V^2 \mathrm{d}\tau \leq \exp(2\lambda t)\|y_0\|_H^2 + \frac{1}{\gamma}\int_0^t \exp(2\lambda(t-\tau))\|f(\tau)\|_{V'}^2 \mathrm{d}\tau \tag{4.3}$$

for all $t \in [0,T)$. Moreover, there exists a constant $c_L \geq 1$, independent of y_0, f, and T, such that

$$\|y\|_{L^\infty(0,T;H)} + \|y\|_{W(0,T;V,V')} \leq c_L \exp(\lambda T)\left(\|y_0\|_H + \|f\|_{L^2(0,T;V')}\right). \tag{4.4}$$

If $\lambda = 0$, then one can choose $T = \infty$.

Proof. See Theorem II.2.1.1 in [91] for the uniqueness and existence, and [17, 93] for the estimate (4.3). The inequality (4.4) is a consequence of (4.3) and the properties of $A(t)$. Specifically, for $t \in [0, T]$, we have that

$$\begin{aligned}\|y(t)\|_H^2 &\leq \exp(2\lambda t)\|y_0\|_H^2 + \frac{1}{\gamma}\int_0^t \exp(2\lambda(t-\tau))\|f(\tau)\|_{V'}^2 \mathrm{d}\tau \\ &\leq \exp(2\lambda T)\|y_0\|_H^2 + \frac{1}{\gamma}\exp(2\lambda T)\int_0^T \|f(\tau)\|_{V'}^2 \mathrm{d}\tau \\ &\leq \exp(2\lambda T)\max\left(1, \frac{1}{\gamma}\right)\left(\|y_0\|_H^2 + \|f\|_{L^2(0,T;V')}^2\right),\end{aligned}$$

which implies the same upper bound for $\|y\|_{L^\infty(0,T;H)}$. Furthermore, we can bound the norm of the time derivative by

$$\begin{aligned}\|y'\|_{L^2(0,T;V')}^2 &= \int_0^T \|y'(\tau)\|_{V'}^2 \mathrm{d}\tau = \int_0^T \| - A(\tau)y(\tau) + f(\tau)\|_{V'}^2 \mathrm{d}\tau \\ &\leq \int_0^T \left(\|A(\tau)y(\tau)\|_{V'}^2 + \|f(\tau)\|_{V'}^2\right)\mathrm{d}\tau \leq \int_0^T \left(\beta\|y(\tau)\|_V^2 + \|f(\tau)\|_{V'}^2\right)\mathrm{d}\tau \\ &= \beta\|y\|_{L^2(0,T;V)}^2 + \|f\|_{L^2(0,T;V')}^2.\end{aligned}$$

Thus,

$$\|y\|_{W(0,T;V,V')}^2 = \|y\|_{L^2(0,T;V)}^2 + \|y'\|_{L^2(0,T;V')}^2 \leq \max(1, 1+\beta)\left(\|y\|_{L^2(0,T;V)}^2 + \|f\|_{L^2(0,T;V')}^2\right).$$

The first expression can then be bounded by using (4.3)

$$\begin{aligned}\gamma\|y\|_{L^2(0,T;V)}^2 &= \gamma\int_0^T \|y(\tau)\|_V^2 \mathrm{d}\tau \leq \gamma\int_0^T \exp(2\lambda(T-\tau))\|y(\tau)\|_V^2 \mathrm{d}\tau \\ &\leq \exp(2\lambda T)\|y_0\|_H^2 + \frac{1}{\gamma}\int_0^T \exp(2\lambda(T-\tau))\|f(\tau)\|_{V'}^2 \mathrm{d}\tau \\ &\leq \exp(2\lambda T)\max\left(1, \frac{1}{\gamma}\right)\left(\|y_0\|_H^2 + \|f\|_{L^2(0,T;V')}^2\right),\end{aligned}$$

which yields the estimate (4.4). □

While the linear operator A in the original Cauchy problem is time-invariant and fulfills all the assumptions of Lemma 4.1, the result covers time-dependent linear operators $A(t)$. This will be leveraged to prove the Carleman linearization's well-posedness and convergence. Additionally, the result applies not only to the spaces V and H, but also to other suitable spaces forming Gelfand triples with the same properties. In this case, the concepts of boundedness and coercivity should be understood in their respective spaces.

The system remains well-posed even under weak, time-dependent perturbations, as the following lemma shows.

Lemma 4.2. *Let $T < \infty$ and $A(t)$ be a family of operators fulfilling the same assumptions as in Lemma 4.1. Let $K(t)$ be a family of operators*

$$K(t) : V \to H \quad \text{for a.e. } t \in (0,T)$$

such that for all $v \in V$ the mapping $t \mapsto K(t)v$ belongs to $L^\infty(0,T;H)$. Then, $A(t) + K(t)$ is uniformly V-H coercive and the variational Cauchy problem

$$\begin{aligned} y'(t) + (A(t) + K(t))y(t) &= f(t) \quad \text{in } L^2(0,T;V'), \\ y(0) &= y_0 \end{aligned}$$

has a unique solution in $W(0,T;V,V')$ for all $y_0 \in H$ and $f \in L^2(0,T;V')$ that depends continuously on the data.

Proof. See Theorem II.2.1.2 in [91] with $\theta = 0$. □

4.2.3 Nonlinear Equation

Moving to the nonlinear equation, we have to add assumptions on B to show the well-posedness of the Cauchy problem. While there are various suitable types of assumptions, we focus on a particular choice that covers most common PDE problems. Assume there is a constant c_B such that

$$|\langle B(u \otimes v), w\rangle_V| \leq c_B \|u\|_H^{\frac{1}{2}} \|u\|_V^{\frac{1}{2}} \|v\|_H^{\frac{1}{2}} \|v\|_V^{\frac{1}{2}} \|w\|_V \quad \text{for all } u, v, w \in V. \tag{A3}$$

The following lemma is a modification of Lemma 5 in [94] and proves the local existence and uniqueness of solutions to the nonlinear Cauchy problem on finite time intervals for unstable A and on unbounded time intervals for the stable case. This adaptation also takes into account the explicit dependence of the constants on T, which will play a crucial role in the convergence behavior of the Carleman linearization.

Lemma 4.3. *Let $T < \infty$, and A and B fulfill* (A1), (A2), *and* (A3). *There exists a constant c_N independent of T and c_B such that for all $y_0 \in H$ and $f \in L^2(0,T;V')$ with*

$$\|y_0\|_H + \|f\|_{L^2(0,T;V')} \leq \rho(T) := \frac{1}{8c_N^2 \exp(2\lambda T) c_B},$$

there exists a unique solution $y \in W(0,T;V,V')$ to the nonlinear Cauchy problem (4.2). *Moreover, the estimate*

$$\|y_0\|_H + \|y\|_{W(0,T;V,V')} \leq 2c_N \exp(\lambda T) \left(\|y_0\|_H + \|f\|_{L^2(0,T;V')} \right)$$

holds. If the coercivity constant λ of the linear operator is zero, then the same result holds for $T = \infty$.

Proof. The proof is provided in Appendix A.1. □

4.3 Truncated Carleman Linearization

We start with an informal definition of the Carleman linearization, focusing on the concept and leaving a rigorous mathematical analysis for the subsequent sections. The main idea is to lift the dynamical system into higher-dimensional tensor products of the original solution space. While this increases the dimensionality of the spaces, the resulting system is linear. Thus, the linearization can be seen as a trade-off between the nonlinearity and the complexity of the underlying spaces.

We first introduce the operators necessary for the linearization. Define the Kronecker sum of an operator or function T by

$$\bigoplus^{k} T := \sum_{i=1}^{k} \left(\bigotimes^{i-1} I \right) \otimes T \otimes \left(\bigotimes^{k-i} I \right),$$

where I denotes the identity operator. This allows us to define

$$A_k = \bigoplus^{k} A, \quad B_k = \bigoplus^{k} B, \quad F_k(t) = \bigoplus^{k} (-f(t)).$$

To illustrate the structure and action of these operators, consider $u, v, w \in V$. Then, for $k = 2$, we obtain

$$\begin{aligned} A_2(u \otimes v) &= (Au) \otimes v + u \otimes (Av), \\ B_2(u \otimes v \otimes w) &= (B(u \otimes v)) \otimes w + u \otimes (B(v \otimes w)), \\ F_2(t)u &= (-f(t)) \otimes u + u \otimes (-f(t)). \end{aligned}$$

If a solution y to (4.1) is sufficiently regular, then its moments $y^{(k)}(t) := \bigotimes^k y(t)$ fulfill the equation

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} y^{(1)}(t) + A_1 y^{(1)}(t) + B_1 y^{(2)}(t) &= f(t), \\ \frac{\mathrm{d}}{\mathrm{d}t} y^{(k)}(t) + F_k(t) y^{(k-1)}(t) + A_k y^{(k)}(t) + B_k y^{(k+1)}(t) &= 0 && \text{for } k > 1, \\ y^{(k)}(0) &= y_0^{(k)} && \text{for } k \geq 1. \end{aligned}$$

These equations establish a linear relationship between moments of order $k-1$, k, and $k+1$. Instead of solving for y, we can seek a solution consisting of moments $y^{(k)}$ that fulfill the above dynamical system. While this erases the nonlinearity from the dynamical system, the approach has a caveat. To fully describe the equation for $y^{(k)}$, one needs $y^{(k+1)}$. This, in turn, requires one to add an equation for $y^{(k+1)}$, ultimately resulting in an infinite chain of equations, which is

called the Carleman linearization. It is not obvious that the infinite system of equations is well-posed; neither is it clear whether a solution of this chain solves the original nonlinear Cauchy problem. For self-adjoint positive A and $f = 0$ some of these questions were answered in, e.g., [95, 96, 97] in the scope of fluid flow problems, where the chain is referred to as the *Friedman–Keller chain of equations* and arises from a statistical analysis of Navier–Stokes equations. Here, we are interested in the computation and approximation aspects of the linearization and will, therefore, omit an analysis of the infinite system. Instead, we explore an approach to circumvent the infinite sequence, restricting our analysis to its finite equivalent.

In order to arrive at a computable dynamical system, we have to close the infinite sequence of equations, sometimes referred to as *moment closure*. For the Carleman linearization, the most common approach is to truncate the system and set $y^{(N+1)}(t) = 0$ for a fixed $N > 1$, decoupling the first N equations. The rationale behind this approach is that higher-order terms are expected to become negligible for the first moment. This results in the so-called *truncated Carleman linearization*, or sometimes also referred to as its *finite-section approximation*. Then, the dynamical system can be written as

$$\begin{aligned} y_N'(t) + \mathcal{A}_N(t) y_N(t) &= f_N(t) \quad \text{for } t \in (0,T), \\ y_N(0) &= y_{N,0} = \begin{pmatrix} y_0^{(1)} & y_0^{(2)} & \cdots & y_0^{(N)} \end{pmatrix}^T, \end{aligned} \tag{TC}$$

with the block operator matrix and right-hand side defined as

$$\mathcal{A}_N(t) = \begin{pmatrix} A_1 & B_1 & 0 & \cdots & 0 \\ F_2(t) & A_2 & B_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & F_{N-1}(t) & A_{N-1} & B_{N-1} \\ 0 & \cdots & 0 & F_N(t) & A_N \end{pmatrix}, \quad f_N(t) = \begin{pmatrix} f(t) \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

We denote the number N as the *truncation level*.

4.4 Well-Posedness of the Truncated Carleman Linearization

To rigorously define the system (TC), we introduce suitable function spaces and impose specific assumptions on B and f, which build upon and extend the approaches of [95, 96, 97]. These criteria will ultimately enable us to show existence and uniqueness of the dynamical system, i.e., its well-posedness. This is done by establishing a weak formulation of the truncated linearization.

4.4.1 Function Spaces

Let us define the family of interpolation spaces with the corresponding norms

$$V^\alpha := [D(L), H]_{1-\alpha/2}, \qquad \|v\|_{V^\alpha}^2 := \|L^{\alpha/2} v\|_H^2 = \sum_{i=1}^{\infty} \lambda_i^\alpha (v, \varphi_i)_H^2,$$
$$V^{-\alpha} := (V^\alpha)', \qquad \|v\|_{V^{-\alpha}}^2 := \sup_{0 \neq w \in V^\alpha} \frac{\langle v, w\rangle_{V^{-\alpha}}^2}{\|w\|_{V^\alpha}^2} = \sum_{i=1}^{\infty} \lambda_i^{-\alpha} \langle v, \varphi_i\rangle_V^2$$

for $\alpha \geq 0$. This means that $V^0 = H$ and $V^1 = V$. For example, in the case of the Sobolev spaces $H = L^2(\Omega)$ and $V = H_0^1(\Omega)$ for a bounded Lipschitz domain $\Omega \subset \mathbb{R}^n$, these spaces can be identified with (cf. [23, pp. 283 ff.])

$$V^\alpha = \begin{cases} H^\alpha(\Omega) & \text{for } 0 \leq \alpha < \frac{1}{2}, \\ H^\alpha(\Omega) \cap H_0^1(\Omega) & \text{for } \frac{1}{2} \leq \alpha \leq 2. \end{cases}$$

Furthermore, define the tensor product spaces for $k \in \mathbb{N}$ and $q \in [-1, 1]$

$$H(k) = \bigotimes_{j=1}^{k} H, \quad V^\alpha(k) = \bigotimes_{j=1}^{k} V^\alpha, \quad V_q^\alpha(k) = \bigcap_{i=1}^{k} \bigotimes_{j=1}^{k} V^{\alpha + \delta_{i,j} q},$$

where $V_q^\alpha(k)$ can be interpreted as the tensor product space $V^\alpha(k)$ with increased ($q > 0$) or decreased ($q < 0$) regularity in single directions. The space $H(k)$ is a Hilbert space endowed with the standard tensor product scalar product. Note that $V^\alpha(k) = V_0^\alpha(k)$ and $H(k) = V^0(k)$. The corresponding norms are defined as

$$\|v\|_{V_q^\alpha(k)}^2 := \sum_{\vec{i} \in \mathbb{N}^k} \pi(\lambda_{\vec{i}})^\alpha \sigma(\lambda_{\vec{i}})^q \langle v, \varphi_{\vec{i}}\rangle_V^2.$$

The function $\varphi_{\vec{i}}$ is defined as $\varphi_{\vec{i}} = \varphi_{\vec{i}_1} \otimes \cdots \otimes \varphi_{\vec{i}_k}$ for any multi-index $\vec{i} \in \mathbb{N}^k$ and the associated tuple of eigenvalues as $\lambda_{\vec{i}} = (\lambda_{\vec{i}_1}, \cdots, \lambda_{\vec{i}_k})$. Furthermore, we set $\pi(\lambda_{\vec{i}}) := \prod_{l=1}^k \lambda_{\vec{i}_l}$ and $\sigma(\lambda_{\vec{i}}) := \sum_{l=1}^k \lambda_{\vec{i}_l}$. In the special case of $k = 1$, we have that $V_q^\alpha(1) = V^{\alpha+q}$. The dual space $(V_1^0(k))'$ can be identified with $V_{-1}^0(k)$ and their norms coincide.

Finally, we provide a function space for a complete solution y_N to (TC), which assembles moments in $V_q^\alpha(k)$ into a vector of moments y_N. For that, define the direct sums of vector spaces

$$U_q^\alpha(N) := \bigoplus_{k=1}^N V_q^\alpha(k), \quad \|(v_1,\dots,v_N)\|^2_{U_q^\alpha(N)} = \sum_{k=1}^N \|v_k\|^2_{V_q^\alpha(k)},$$
$$Y(N) := \bigoplus_{k=1}^N H(k), \qquad \|(v_1,\dots,v_N)\|^2_{Y(N)} = \sum_{k=1}^N \|v_k\|^2_{H(k)}.$$

The space $Y(N)$ forms a Hilbert space with the scalar product $((u_1,\dots,u_N),(v_1,\dots,v_N))_{Y(N)} = \sum_{k=1}^N (u_k,v_k)_{H(k)}$. It is noted that $(U_1^0(N), Y(N), U_{-1}^0(N))$ forms a Gelfand triple as the dual space $(U_1^0(N))'$ can be identified with $U_{-1}^0(N)$.

4.4.2 Properties of A_k, B_k, and F_k

The operators A_k, B_k, and F_k are essential components of the truncated Carleman linearization. The construction of the operators through the Kronecker sum suggests that the operators' norms grow at least linearly in k. However, the dependence of the boundedness and the coercivity on k can be refined and, in certain cases, eliminated, which will play a crucial role in proving the convergence of the linearization.

The operator A_k is a closed operator with domain $D(A_k) = V_2^0(k)$. From the definition of the space $V_1^0(k)$, we know that A_k can be extended to $A_k : V_1^0(k) \to V_{-1}^0(k)$. Similarly to above, the function space of choice for a weak formulation is $D(A_k^{1/2})$. Theorem 13.1 in [92] combined with the properties of L implies that we can identify the interpolation space as $D(A_k^{1/2}) = V_1^0(k)$, and, therefore, $D(A_k^{1/2})' = V_{-1}^0(k)$. Thus, the operator $A_k : D(A_k^{1/2}) \to D(A_k^{1/2})'$ is well defined. The following lemma demonstrates that not only can the Kronecker sum of A be extended to the interpolation space, but it also maintains coercivity and boundedness.

Lemma 4.4. *Assume the operator A fulfills* (A1) *and* (A2) *with constants β, γ, and λ. Then, the operator $A_k : V_1^0(k) \to V_{-1}^0(k)$ is bounded and $V_1^0(k)$-$H(k)$ coercive with constants*

$$\langle A_k u, v\rangle_{V_1^0(k)} \le \beta \|u\|_{V_1^0(k)} \|v\|_{V_1^0(k)},$$
$$\langle A_k v, v\rangle_{V_1^0(k)} + k\lambda \|v\|^2_{H(k)} \ge \gamma \|v\|^2_{V_1^0(k)}$$

for all $u, v \in V_1^0(k)$.

Proof. The proof is provided in Appendix A.2. □

While boundedness and coercivity are pivotal properties for showing existence and uniqueness of parabolic problems on its own, the result can even be strengthened in the case $\lambda = 0$, in which case the constants for boundedness and coercivity are independent of k.

Turning to the operator B_k, assume that B is a symmetric and continuous bilinear mapping between the spaces $B : V_1^\alpha(2) \to V^{\alpha-1+\varepsilon}$ for fixed constants $\varepsilon \in [0,1]$ and $\alpha \geq 0$. Let $c_B(\alpha,\varepsilon) > 0$ be a constant such that

$$\|Bv\|_{V^{\alpha-1+\varepsilon}} \leq c_B(\alpha,\varepsilon)\|v\|_{V_1^\alpha(2)} \quad \text{for all } v \in V_1^\alpha(2). \tag{A4}$$

It is noted that property (A3) implies (A4) for $\alpha = 0$ and $\varepsilon = 0$ since

$$\begin{aligned}
\|B(u_1 \otimes u_2)\|_{V^{-1}}^2 &= \|B(u_1 \otimes u_2)\|_{V'}^2 = \sup_{w\in V,\, w\neq 0} \frac{|\langle B(u_1 \otimes u_2), w\rangle|^2}{\|w\|_V^2} \\
&\leq \sup_{w\in V,\, w\neq 0} \frac{c_B\|u_1\|_H\|u_1\|_V\|u_2\|_H\|u_2\|_V\|w\|_V^2}{\|w\|_V^2} \\
&= c_B\|u_1\|_H\|u_1\|_V\|u_2\|_H\|u_2\|_V \leq \frac{1}{2}c_B\left(\|u_1\|_V^2\|u_2\|_H^2 + \|u_1\|_H^2\|u_2\|_V^2\right) \\
&= \frac{1}{2}c_B\|u_1 \otimes u_2\|_{V_1^0(2)}^2 = c_B(0,0)\|u_1 \otimes u_2\|_{V_1^0(2)}^2.
\end{aligned}$$

for all $u_1, u_2 \in V$, where Young's inequality was used. However, the converse does not hold in general, i.e., (A3) cannot be deduced from (A4), and, therefore, is a stronger condition on B. The importance of the distinction between these assumptions becomes evident in the subsequent discussion of operator examples. The following result for B_k was shown in [96, 97].

Lemma 4.5. *Let B fulfill* (A4) *with constants α and ε. The operator $B_k : V_1^\alpha(k+1) \to V_{-1+\varepsilon}^\alpha(k)$ is well-defined and continuous. More specifically,*

$$\|B_k v\|_{V_{-1+\varepsilon}^\alpha(k)} \leq \sqrt{2}c_B(\alpha,\varepsilon)k^{\varepsilon/2}\|v\|_{V_1^\alpha(k+1)} \quad \textit{for all } v \in V_1^\alpha(k+1),$$

where $c_B(\alpha,\varepsilon)$ is the constant from (A4) *and does not depend on k.*

Proof. The proof is provided in Appendix A.2 and is a generalization of the proof of Lemma 2.4 in [96] (case $\varepsilon = 0$) to the more general result for $\varepsilon \in [0,1]$ stated in [97] (proof not provided therein). □

We can show similar bounds for F_k.

Lemma 4.6. *For any $\alpha \geq 0$, the operator $F_k(t) : V_1^\alpha(k) \to V_{-1+\varepsilon}^\alpha(k+1)$ is bounded for $f(t) \in V^\alpha$. More specifically, it holds that*

$$\|F_k(t)v\|_{V_{-1+\varepsilon}^\alpha(k+1)} \leq c_F(\varepsilon)\|f(t)\|_{V^\alpha}k^{\varepsilon/2}\|v\|_{V_1^\alpha(k)} \quad \textit{for all } v \in V_1^\alpha(k),$$

where the constant $c_F(\varepsilon)$ does not depend on f or k.

Proof. The proof is provided in Appendix A.2. □

4.4.3 Lifted Linearization and Well-Posedness

Intuitively, we would like to consider system (TC) in the space $U_1^0(N)$, meaning we seek a solution $y_N \in U_1^0(N)$ while the equality is to be considered in its dual space $U_{-1}^0(N)$. However, if assumption (A4) does not hold for $\alpha = 0$ but for some $\alpha > 0$, Lemma 4.5 does not allow us to infer well-posedness of the linearization through Lemma 4.2 in the classical weak formulation. One way to address this issue is to consider the equation directly in $U_1^\alpha(N)$ and $U_{-1}^\alpha(N)$ instead, as has been done in [96, 97] in the case of self-adjoint positive A and $f = 0$. Here, we choose a different approach: We first transform the original linearization into an alternative dynamical system, which restricts the solution to higher regularity. This, ultimately, enables us to apply Lemma 4.2.

First, define the operator $A_\lambda := A + \lambda I$ where I denotes the identity operator and λ the constant from (A2). Then, A_λ is V-H coercive with the same constant γ as A and $\lambda = 0$. Since A_λ is a positive operator, its fractional powers are well-defined (see, e.g., Section 2.6 in [98]), and we can define $\bar{A}_{\lambda,k}^\xi := \bigotimes^k A_\lambda^\xi$ for any $\xi \in \mathbb{R}$. Furthermore, A_λ^ξ commutes with A and, consequently, $\bar{A}_{\lambda,k}^\xi$ commutes with A_k.

Instead of seeking a solution y_N to (TC), we seek a solution $\tilde{y}_N(t) = \mathcal{D}_{\lambda,N}^\alpha y_N(t)$ with $\mathcal{D}_{\lambda,N}^\xi := \operatorname{diag}(\bar{A}_{\lambda,1}^{\xi/2}, \dots, \bar{A}_{\lambda,N}^{\xi/2})$ for $\xi \in \mathbb{R}$, where we are specifically interested in the case $\alpha \in [0, \bar{\alpha} - 1]$. Plugging this into the original linearization leads to the dynamical system

$$\begin{aligned} \tilde{y}_N'(t) &= \underbrace{\mathcal{D}_{\lambda,N}^\alpha \mathcal{A}_N(t) \mathcal{D}_{\lambda,N}^{-\alpha}}_{\widetilde{\mathcal{A}}_N(t)} \tilde{y}_N(t) + \underbrace{\mathcal{D}_{\lambda,N}^\alpha f_N(t)}_{\tilde{f}_N(t)}, \\ \tilde{y}_N(0) &= \underbrace{\mathcal{D}_{\lambda,N}^\alpha y_{N,0}}_{\tilde{y}_{N,0}}. \end{aligned} \tag{TC$_\alpha$}$$

The block operator matrix $\widetilde{\mathcal{A}}_N$, the forcing $\tilde{f}_N$, and the initial condition have a similar form as in (TC):

$$\widetilde{\mathcal{A}}_N(t) = \begin{pmatrix} A_1 & \widetilde{B}_1 & 0 & \cdots & 0 \\ \widetilde{F}_2(t) & A_2 & \widetilde{B}_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \widetilde{F}_{N-1}(t) & A_{N-1} & \widetilde{B}_{N-1} \\ 0 & \cdots & 0 & \widetilde{F}_N(t) & A_N \end{pmatrix},$$

$$\tilde{f}_N(t) = \begin{pmatrix} A_\lambda^{\alpha/2} f(t) & 0 & \cdots & 0 \end{pmatrix}^T, \quad \tilde{y}_{N,0} = \begin{pmatrix} \bigotimes^1 A_\lambda^{\alpha/2} y_0 & \cdots & \bigotimes^N A_\lambda^{\alpha/2} y_0 \end{pmatrix}^T.$$

The operators $\widetilde{B}_k$ and $\widetilde{F}_k(t)$ define the corresponding Kronecker sums of the operators $\widetilde{B} := A_\lambda^{\alpha/2} B (A_\lambda^{-\alpha/2} \otimes A_\lambda^{-\alpha/2})$ and $\tilde{f}(t) = A_\lambda^{\alpha/2} f(t)$ respectively. It is noted that the operators A_k remain unchanged under this transformation as the operator A_k commutes with the operators pre- and post-applied. The operator $\widetilde{B}$ fulfills (A4) for $\alpha = 0$ with an adjusted constant, as can

be seen by the estimate

$$\begin{aligned}\|\widetilde{B}\|_{\mathcal{L}(V_0^1(2),V^{-1+\varepsilon})} &= \|A_\lambda^{\alpha/2}B(A_\lambda^{-\alpha/2}\otimes A_\lambda^{-\alpha/2})\|_{\mathcal{L}(V_0^1(2),V^{-1+\varepsilon})}\\ &\leq \|A_\lambda^{\alpha/2}\|_{\mathcal{L}(V^{\alpha-1+\varepsilon},V^{-1+\varepsilon})}\|A_\lambda^{-\alpha/2}\otimes A_\lambda^{-\alpha/2}\|_{\mathcal{L}(V_1^0(2),V_1^\alpha(2))}\|B\|_{\mathcal{L}(V_1^\alpha(2),V^{\alpha-1+\varepsilon})}\\ &\leq c_{\lambda,\alpha}c_B(\alpha,\varepsilon),\end{aligned} \tag{4.5}$$

where $\alpha \in [0, \bar{\alpha}-1]$ and the constant $c_{\lambda,\alpha}$ only depends on A, L, λ, and α. If $\alpha = 0$, then $c_{\lambda,\alpha} = 1$. This allows us to apply Lemma 4.5 to $\widetilde{B}_k$, i.e., that $\widetilde{B}_k : V_1^0(k+1) \to V_{-1+\varepsilon}^0(k)$ is continuous. Moreover, we can use Lemma 4.6 to show that $\widetilde{F}_k(t) : V_1^0(k-1) \to V_{-1+\varepsilon}^0(k)$ is continuous provided that $f(t) \in V^\alpha$, which implies $A_\lambda^{\alpha/2}f(t) \in H$. Although Lemmas 4.5 and 4.6 are technically applied here only for the case $\alpha = 0$—specifically, for the transformed operator $\widetilde{B}$ and the transformed forcing, with constants adjusted to account for the lifting—we presented these results in the previous section for arbitrary $\alpha \geq 0$. This broader presentation highlights that the approach of [96, 97] can be readily adapted to our framework.

The operator $\widetilde{\mathcal{A}}_N(t)$ can be viewed as a weak perturbation of a $U_1^0(N)$-$Y(N)$ coercive operator. For that, we decompose the operator into

$$\widetilde{\mathcal{A}}_N(t) = \mathcal{A}_{N,\mathrm{diag}} + \widetilde{\mathcal{B}}_N + \widetilde{\mathcal{F}}_N(t),$$

where $\mathcal{A}_{N,\mathrm{diag}}$ is the diagonal part, $\widetilde{\mathcal{B}}_N$ the upper diagonal part, and $\widetilde{\mathcal{F}}_N(t)$ the lower diagonal part. The following propositions establish the foundation for demonstrating the weak perturbation property.

Proposition 4.1. *Let A fulfill assumptions* (A1) *and* (A2) *with constants β, γ, and λ. Then, $\mathcal{A}_{N,\mathrm{diag}}$ is $U_1^0(N)$-$Y(N)$ coercive with the constants*

$$\begin{aligned}\langle \mathcal{A}_{N,\mathrm{diag}}u, v\rangle_{U_1^0(N)} &\leq \beta\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)},\\ \langle \mathcal{A}_{N,\mathrm{diag}}v, v\rangle_{U_1^0(N)} + N\lambda\|v\|_{Y(N)}^2 &\geq \gamma\|v\|_{U_1^0(N)}^2\end{aligned}$$

for all $u, v \in U_1^0(N)$.

Proof. Let $u = (u_1, \ldots, u_N) \in U_1^0(N)$ and $v = (v_1, \ldots, v_N) \in U_1^0(N)$ be arbitrary but fixed. Using Lemma 4.4, we can show that

$$\begin{aligned}\langle \mathcal{A}_{N,\mathrm{diag}}u, v\rangle_{U_1^0(N)} &= \sum_{k=1}^N \langle A_k u_k, v_k\rangle_{V_1^0(k)} \leq \sum_{k=1}^N \beta\|u_k\|_{V_1^0(k)}\|v_k\|_{V_1^0(k)}\\ &\leq \beta\left(\sum_{k=1}^N \|u_k\|_{V_1^0(k)}^2\right)^{\frac{1}{2}}\left(\sum_{k=1}^N \|v_k\|_{V_1^0(k)}^2\right)^{\frac{1}{2}} = \beta\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)}.\end{aligned}$$

The coercivity property can be shown as follows:

$$\begin{aligned}\langle \mathcal{A}_{N,\mathrm{diag}} v, v\rangle_{U_1^0(N)} &= \sum_{k=1}^{N} \langle A_k v_k, v_k\rangle_{V_1^0(k)} \geq \sum_{k=1}^{N} \left(\gamma \|v_k\|^2_{V_1^0(k)} - k\lambda \|v_k\|^2_{H(k)}\right) \\ &\geq \gamma \|v\|^2_{U_1^0(N)} - N\lambda \|v\|^2_{Y(N)}. \end{aligned}$$ □

Proposition 4.2. *Let B fulfill assumption* (A4) *for some $\alpha \in [0, \bar{\alpha} - 1]$ and $\varepsilon \in [0, 1]$. Then, $\widetilde{\mathcal{B}}_N : U_1^0(N) \to U_{-1+\varepsilon}^0(N)$ is continuous with*

$$\|\widetilde{\mathcal{B}}_N v\|_{U_{-1+\varepsilon}^0(N)} \leq c_{\mathcal{B}}(\alpha, \varepsilon) N^{\varepsilon/2} \|v\|_{U_1^0(N)} \quad \text{for all } v \in U_1^0(N),$$

where the constant $c_{\mathcal{B}}(\alpha, \varepsilon)$ can be bounded in terms of $c_B(\alpha, \varepsilon)$ and does not depend on N.

Proof. Let $v = (v_1, \ldots, v_N) \in U_1^0(N)$ be arbitrary but fixed. Because of Lemma 4.5 and the estimate (4.5), it holds that

$$\begin{aligned}\|\widetilde{\mathcal{B}}_N v\|^2_{U_{-1+\varepsilon}^0(N)} &= \sum_{k=1}^{N-1} \|\widetilde{B}_k v_{k+1}\|^2_{V_{-1+\varepsilon}^0(k)} \leq 2c^2_{\lambda,\alpha} c_B(\alpha, \varepsilon)^2 \sum_{k=1}^{N-1} k^{\varepsilon} \|v_{k+1}\|^2_{V_1^0(k+1)} \\ &\leq 2c^2_{\lambda,\alpha} c_B(\alpha, \varepsilon)^2 N^{\varepsilon} \|v\|^2_{U_1^0(N)}. \end{aligned}$$

This implies the result with $c_{\mathcal{B}}(\alpha, \varepsilon) = \sqrt{2} c_{\lambda,\alpha} c_B(\alpha, \varepsilon)$. □

Proposition 4.3. *Let $f(t) \in V^{\alpha}$ for some $\alpha \in [0, \bar{\alpha} - 1]$ and $\varepsilon \in [0, 1]$. Then, $\widetilde{\mathcal{F}}_N(t) : U_1^0(N) \to U_{-1+\varepsilon}^0(N)$ is continuous with*

$$\|\widetilde{\mathcal{F}}_N(t) v\|_{U_{-1+\varepsilon}^0(N)} \leq c_{\mathcal{F}}(\alpha, \varepsilon) \|f(t)\|_{V^{\alpha}} N^{\varepsilon/2} \|v\|_{U_1^0(N)} \quad \text{for all } v \in U_1^0(N),$$

where the constant $c_{\mathcal{F}}(\alpha, \varepsilon)$ does not depend on f or N.

Proof. Let $v = (v_1, \ldots, v_N) \in U_1^0(N)$ be arbitrary but fixed. With Lemma 4.6, we can show that

$$\begin{aligned}\|\widetilde{\mathcal{F}}_N(t) v\|^2_{U_{-1+\varepsilon}^0(N)} &= \sum_{k=2}^{N} \|\widetilde{F}_k(t) v_{k-1}\|^2_{V_{-1+\varepsilon}^0(k)} \leq c_F(\varepsilon)^2 \|A_{\lambda}^{\alpha/2} f(t)\|^2_H \sum_{k=2}^{N} k^{\varepsilon} \|v_{k-1}\|^2_{V_1^0(k-1)} \\ &\leq \tilde{c}^2_{\lambda,\alpha} c_F(\varepsilon)^2 \|f(t)\|^2_{V^{\alpha}} N^{\varepsilon} \|v\|^2_{U_1^0(N)}, \end{aligned}$$

where $\tilde{c}_{\lambda,\alpha} = \|A_{\lambda}^{\alpha/2} L^{-\alpha/2}\|_{\mathcal{L}(H,H)}$. This implies the result with $c_{\mathcal{F}}(\alpha, \varepsilon) = \tilde{c}_{\lambda,\alpha} c_F(\varepsilon)$ □

The following lemma lets us identify solutions $\tilde{y}$ of (TC_{α}) as solutions to (TC).

Lemma 4.7. *The linear mapping $\mathcal{D}^{\xi}_{\lambda,N} : W(0, T; U_1^{\xi}(N), U_{-1}^{\xi}(N)) \to W(0, T; U_1^0(N), U_{-1}^0(N))$ with $w \mapsto (t \mapsto \mathcal{D}^{\xi}_{\lambda,N} w(t))$ is an isomorphism with the inverse $\mathcal{D}^{-\xi}_{\lambda,N}$ for any $\xi \in \mathbb{R}$.*

We are now in a position to show the well-posedness of equation (TC).

Theorem 4.1 (Well-posedness)**.** *Let $T < \infty$. Moreover, let A fulfill* (A1) *and* (A2)*, B fulfill* (A4) *for some $\alpha \in [0, \bar{\alpha} - 1]$ and $\varepsilon = 1$, $f \in L^\infty(0, T; V^\alpha)$, and $y_0 \in V^\alpha$. Then, the weak form of the truncated Carleman linearization* (TC)*, which reads as*

$$\begin{aligned} y_N'(t) + \mathcal{A}_N(t) y_N(t) &= f_N(t) \quad \text{in } L^2(0, T; \left(U_1^0(N)\right)'), \\ y_N(0) &= y_{N,0}, \end{aligned} \tag{TCW}$$

has a unique solution $y_N \in W(0, T; U_1^\alpha(N), U_{-1}^\alpha(N))$ that depends continuously on y_0 and f.

Proof. Proposition 4.1 implies that $\mathcal{A}_{N,\text{diag}}$ is $U_1^0(N)$-$Y(N)$ coercive. From Proposition 4.3, we know that for any $v \in U_1^0(N)$ the mapping $t \mapsto \widetilde{\mathcal{F}}_N(t)v$ fulfills

$$\|t \mapsto \widetilde{\mathcal{F}}_N(t)v\|_{L^\infty(0,T;Y(N))} \leq c_{\mathcal{F}}(\alpha, \varepsilon)\|f\|_{L^\infty(0,T;V^\alpha)}\sqrt{N}\|v\|_{U_1^0(N)}.$$

With Proposition 4.2, it follows that $\widetilde{\mathcal{B}}_N + \widetilde{\mathcal{F}}_N(t)$ as a perturbation of $\mathcal{A}_{N,\text{diag}}$ fulfills the conditions in Lemma 4.2. Thus, the weak form of (TC_α), i.e.,

$$\begin{aligned} \tilde{y}_N'(t) + \widetilde{\mathcal{A}}_N(t)\tilde{y}_N(t) &= \tilde{f}_N(t) \quad \text{in } L^2(0, T; \left(U_1^0(N)\right)'), \\ \tilde{y}_N(0) &= \tilde{y}_{N,0}, \end{aligned} \tag{TCW$_\alpha$}$$

has a unique solution $\tilde{y}_N \in W(0, T; U_1^0(N), U_{-1}^0(N))$. With Lemma 4.7 and $(\mathcal{D}_N^{-\alpha})^* U_1^0(N) \subseteq U_1^0(N)$, we can infer that $y_N = \mathcal{D}_N^{-\alpha}\tilde{y}_N \in W(0, T; U_1^\alpha(N), U_{-1}^\alpha(N))$ solves (TCW). To show uniqueness, assume that $z_N \in W(0, T; U_1^\alpha(N), U_{-1}^\alpha(N))$ solves (TCW). Then, $\tilde{z}_N(t) = \mathcal{D}_{N,\lambda}^\alpha z_N(t)$ solves (TCW_α), which implies that $\tilde{z}_N = \tilde{y}_N$. Lemma 4.7 lets us deduce that $z_N = y_N$. □

4.5 Convergence in the Case of Small Nonlinearities

Having established the existence and (partial) uniqueness of a solution to (TCW), the question arises as to how well the linearization approximates the original nonlinear Cauchy problem. In particular, we are interested in whether a solution of the truncated Carleman linearization converges to the true solution of the nonlinear system for $N \to \infty$, i.e., whether the nonlinear dynamics can be recovered if the truncation level is chosen large enough.

Let $y_e \in W(0, T; V, V')$ be the solution to problem (4.2), also referred to as the exact solution, and $y_N \in W(0, T; V^{1+\alpha}, V^{-1+\alpha})$ the solution to the truncated system (TCW). The specific quantity we are interested in is the error defined as the norm of $\eta(t) := y_N^{(1)}(t) - y(t)$. For further analysis, we define the error of all moments $\eta_N(t) := y_N(t) - y_{N,e}(t)$, where $y_{N,e}(t)$ denotes the vector of the moments $y_e^{(k)}(t)$ for $k \in \{1, \ldots, N\}$. The error function η_N fulfills the dynamical system

$$\begin{aligned} \eta_N'(t) + \mathcal{A}_N(t)\eta_N(t) &= r_N(t), \\ \eta_N(0) &= 0, \\ r_N(t) &= \begin{pmatrix} 0 & \cdots & 0 & -B_N y_e^{(N+1)}(t) \end{pmatrix}^T. \end{aligned} \tag{4.6}$$

Under the assumptions of Theorem 4.1, the operator $\widetilde{\mathcal{A}}_N(t)$ is uniformly $U_1^0(N)$-$Y(N)$ coercive with some constants γ_N and λ_N, which enables us to apply the estimate (4.3) from Lemma 4.1. This provides a bound of η_N in terms of the right-hand side r_N, which depends on the exact solution of the original problem. However, the quality of the estimate depends on the coercivity constants. The proof of Theorem 4.1 establishes the coercivity of the operator $\widetilde{\mathcal{A}}_N(t)$ by applying Lemma 4.2. Careful consideration of the proof of Lemma 4.2 reveals that these constants depend on N; specifically, $\gamma_N \to 0$ and $\lambda_N \to \infty$ as $N \to \infty$. This indicates that the Cauchy problem becomes ill-posed in the asymptotic limit and that the estimate from Lemma 4.1 deteriorates. In what follows, we will explore how we can obtain coercivity in a more controlled manner for the case of small nonlinearities and forcing, yielding coercivity constants that are more robust with respect to N. This provides a better understanding of the system's asymptotic behavior as $N \to \infty$.

Proposition 4.4. *Let A fulfill* (A1) *and* (A2) *with constants β, γ, and λ, let B fulfill* (A4) *for some $\alpha \in [0, \bar{\alpha} - 1]$ and $\varepsilon = 0$, and let $f \in L^\infty(0,T;V^\alpha)$. If*

$$c_{\mathcal{P}} := c_{\mathcal{B}}(\alpha, 0) + c_{\mathcal{F}}(\alpha, 0)\|f\|_{L^\infty(0,T;V^\alpha)} < \gamma,$$

then the operator $\widetilde{\mathcal{A}}_N(t)$ is bounded and coercive with constants

$$\begin{aligned} \langle \widetilde{\mathcal{A}}_N(t)u, v\rangle_{U_1^0(N)} &\leq (\beta + c_{\mathcal{P}})\, \|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)}, \\ \langle \widetilde{\mathcal{A}}_N(t)v, v\rangle_{U_1^0(N)} + N\lambda\|v\|^2_{Y(N)} &\geq (\gamma - c_{\mathcal{P}})\, \|v\|^2_{U_1^0(N)}, \end{aligned}$$

for almost every $t \in [0,T)$ and all $u, v \in U_1^0(k)$.

Proof. The boundedness and coercivity follow from Propositions 4.1, 4.2, and 4.3 for $\varepsilon = 0$. More specifically, let $u, v \in U_1^0(N)$ be arbitrary but fixed. Then, we obtain an upper bound through

$$\begin{aligned} \langle \widetilde{\mathcal{A}}_N(t)u, v\rangle_{U_1^0(N)} &= \langle \mathcal{A}_{N,\mathrm{diag}}u, v\rangle_{U_1^0(N)} + \langle \widetilde{\mathcal{B}}_N u, v\rangle_{U_1^0(N)} + \langle \widetilde{\mathcal{F}}_N(t)u, v\rangle_{U_1^0(N)} \\ &\leq \beta\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)} + \|\widetilde{\mathcal{B}}_N u\|_{U_{-1}^0(N)}\|v\|_{U_1^0(N)} + \|\widetilde{\mathcal{F}}_N(t)u\|_{U_{-1}^0(N)}\|v\|_{U_1^0(N)} \\ &\leq \beta\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)} + c_{\mathcal{B}}(\alpha,0)\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)} + c_{\mathcal{F}}(\alpha,0)\|f(t)\|_{V^\alpha}\|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)} \\ &= (\beta + c_{\mathcal{B}}(\alpha,0) + c_{\mathcal{F}}(\alpha,0)\|f(t)\|_{V^\alpha})\, \|u\|_{U_1^0(N)}\|v\|_{U_1^0(N)} \end{aligned}$$

for almost every $t \in [0,T)$. Similarly, the coercivity follows from

$$\begin{aligned} \langle \widetilde{\mathcal{A}}_N(t)v, v\rangle_{U_1^0(N)} &\geq \langle \mathcal{A}_{N,\mathrm{diag}}v, v\rangle_{U_1^0(N)} - \left|\langle \widetilde{\mathcal{B}}_N v, v\rangle_{U_1^0(N)}\right| - \left|\langle \widetilde{\mathcal{F}}_N(t)v, v\rangle_{U_1^0(N)}\right| \\ &\geq \gamma\|v\|^2_{U_1^0(N)} - N\lambda\|v\|^2_{Y(N)} - \|\widetilde{\mathcal{B}}_N v\|_{U_{-1}^0(N)}\|v\|_{U_1^0(N)} - \|\widetilde{\mathcal{F}}_N(t)v\|_{U_{-1}^0(N)}\|v\|_{U_1^0(N)} \\ &\geq (\gamma - c_{\mathcal{B}}(\alpha,0) - c_{\mathcal{F}}(\alpha,0)\|f(t)\|_{V^\alpha})\, \|v\|^2_{U_1^0(N)} - N\lambda\|v\|^2_{Y(N)}. \end{aligned}$$

□

The estimate (4.3) applied to (4.6) with the improved constants of Proposition 4.4 yields a bound of η_N that can be expressed in terms of the $L^2(0,T;V_1^0(N))$-norm of $\bigotimes^{N+1} y_e$. Thus, it is necessary to estimate the latter term.

Proposition 4.5. *Let $T \in (0,\infty]$. For all $z \in W(0,T;V,V')$ and $N \in \mathbb{N}$ it holds that*

$$\|\bigotimes^N z\|_{L^2(0,T;V_1^0(N))} \leq \sqrt{N}\left(\|z(0)\|_H + \|z\|_{W(0,T;V,V')}\right)^N.$$

Proof. Let $N \in \mathbb{N}$ and $z \in W(0,T;V,V')$ be arbitrary but fixed. With Lemma A.1.1, it follows that

$$\begin{aligned}
\|\bigotimes^N z\|^2_{L^2(0,T;V_1^0(N))} &= \int_0^T \|\bigotimes^N z(t)\|^2_{V_1^0(N)}\mathrm{d}t = \int_0^T \sum_{i=1}^N \prod_{j=1}^N \|z(t)\|^2_{V^{\delta_{i,j}}}\mathrm{d}t \\
&= N\int_0^T \|z(t)\|_V^2 \|z(t)\|_H^{2(N-1)}\mathrm{d}t \leq N(\|z\|_{L^\infty(0,T;H)})^{2(N-1)}\int_0^T \|z(t)\|_V^2\mathrm{d}t \\
&\leq N\left(\|z(0)\|_H + \|z(t)\|_{W(0,T;V,V')}\right)^{2N}.
\end{aligned}$$

□

Finally, we achieve convergence, provided the exact solution is small enough.

Theorem 4.2 (Convergence)**.** *Let $T < \infty$. Let A, B, and f fulfill all assumptions of Proposition 4.4 and $c_{\mathcal{P}} < \gamma$. The initial condition is assumed to fulfill $y_0 \in V^\alpha$. Moreover, assume that* (4.2) *admits a solution $y_e \in W(0,T;V^{1+\alpha},V^{-1+\alpha})$. Then, it holds that*

$$\begin{aligned}
&\|\eta\|_{L^\infty(0,T;V^\alpha)} + \|\eta\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \\
&\quad\leq c_1\sqrt{N+1}\left(c_2\exp(\lambda T)\left(\|y_0\|_{V^\alpha} + \|y_e\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})}\right)\right)^{N+1},
\end{aligned}$$

where $\eta(t) := y_N^{(1)}(t) - y_e(t)$ and $y_N \in W(0,T;U_1^\alpha(N),U_{-1}^\alpha(N))$ is the solution to (TCW)*. The constants $c_1, c_2 > 0$ do not depend on N, T, y_0, or y_e. If $\lambda = 0$, one can choose $T = \infty$.*

Proof. Due to Proposition 4.4, we know that (TCW) has a unique solution $y_N \in W(0,T;U_1^0(N),$ $U_{-1}^0(N))$. Based on the definition $\eta_N(t) = y_N(t) - y_{N,e}(t)$, we define $\tilde{\eta}_N(t) = \mathcal{D}_{\lambda,N}^\alpha \eta_N(t)$, which satisfies the transformed dynamical system

$$\begin{aligned}
\tilde{\eta}_N'(t) + \widetilde{\mathcal{A}}_N(t)\tilde{\eta}_N(t) = \tilde{r}_N(t) \quad &\text{in } L^2(0,T;\left(U_1^0(N)\right)'), \\
\tilde{\eta}_N(0) &= 0, \\
\tilde{r}_N(t) &= \begin{pmatrix} 0 & \cdots & 0 & -\widetilde{B}_N\tilde{y}_e^{(N+1)}(t)\end{pmatrix}^T
\end{aligned}$$

with $\tilde{y}_e^{(N+1)}(t) := \bigotimes^{N+1} A_\lambda^{\alpha/2} y_e(t)$. From Lemma 4.1 and Proposition 4.4, we know that the solution to this dynamical system fulfills the estimate

$$\begin{aligned}
&\|\tilde{\eta}_N\|^2_{L^\infty(0,T;Y(N))} + (\gamma - c_\mathcal{P}) \|\tilde{\eta}_N\|^2_{L^2(0,T;U_1^0(N))} \\
&\quad \leq \frac{1}{\gamma - c_\mathcal{P}} \int_0^T \exp(2N\lambda(T-t)) \|\tilde{r}_N(t)\|^2_{U_{-1}^0(N)} \mathrm{d}t \\
&\quad \leq \frac{1}{\gamma - c_\mathcal{P}} \exp(2N\lambda T) \|\tilde{r}_N\|^2_{L^2(0,T;U_{-1}^0(N))}.
\end{aligned} \tag{4.7}$$

Lemma 4.5 and Proposition 4.5 imply that

$$\begin{aligned}
\|\tilde{r}_N\|_{L^2(0,T;U_{-1}^0(N))} &= \|\widetilde{B}_N \tilde{y}_e^{(N+1)}\|_{L^2(0,T;V_{-1}^0(N))} \leq \sqrt{2} c_{\lambda,\alpha} c_B(\alpha,0) \|\tilde{y}_e^{(N+1)}\|_{L^2(0,T;V_1^0(N+1))} \\
&\leq \sqrt{2} c_{\lambda,\alpha} c_B(\alpha,0) \sqrt{N+1} \left(\|\tilde{y}_e(0)\|_H + \|\tilde{y}_e\|_{W(0,T;V,V')} \right)^{N+1}.
\end{aligned} \tag{4.8}$$

Moreover, we can relate the estimate of η_N' with η_N by

$$\begin{aligned}
\|\tilde{\eta}_N'\|^2_{L^2(0,T;U_{-1}^0(N))} &= \int_0^T \| - \widetilde{\mathcal{A}}_N(t) \tilde{\eta}_N(t) + \tilde{r}_N(t)\|^2_{U_{-1}^0(N)} \mathrm{d}t \\
&\leq \int_0^T \left((\beta + c_\mathcal{P}) \|\tilde{\eta}_N(t)\|^2_{U_1^0(N)} + \|\tilde{r}_N(t)\|^2_{U_{-1}^0(N)} \right) \mathrm{d}t \\
&= (\beta + c_\mathcal{P}) \|\tilde{\eta}_N\|^2_{L^2(0,T;U_1^0(N))} + \|\tilde{r}_N\|^2_{L^2(0,T;U_{-1}^0(N))}.
\end{aligned} \tag{4.9}$$

By simplifying the norms and transformed variables, this, finally, leads to the estimate

$$\begin{aligned}
&\|\eta\|_{L^\infty(0,T;V^\alpha)} + \|\eta\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} = \tilde{c}_{\lambda,\alpha} \left(\|\tilde{\eta}\|_{L^\infty(0,T;H)} + \|\tilde{\eta}\|_{W(0,T;V,V')} \right) \\
&\quad \leq \tilde{c}_{\lambda,\alpha} \left(\|\tilde{\eta}_N\|_{L^\infty(0,T;Y(N))} + \|\tilde{\eta}_N\|_{W(0,T;U_1^0(N),U_{-1}^0(N))} \right) \\
&\quad \underset{(4.9),(4.7)}{\leq} \tilde{c}_{\lambda,\alpha} \bar{c}(\gamma,\beta,c_\mathcal{P}) \exp(N\lambda T) \|\tilde{r}_N\|_{L^2(0,T;U_{-1}^0(N))} \\
&\quad \underset{(4.8)}{\leq} \tilde{c}_{\lambda,\alpha} \bar{c}(\gamma,\beta,c_\mathcal{P}) \sqrt{2} c_{\lambda,\alpha} c_B(\alpha,0) \exp(N\lambda T) \sqrt{N+1} \left(\|\tilde{y}_0\|_H + \|\tilde{y}_e\|_{W(0,T;V,V')} \right)^{N+1} \\
&\quad = c_1 \sqrt{N+1} \left(\exp(\lambda T) \left(\|\tilde{y}_0\|_H + \|\tilde{y}_e\|_{W(0,T;V,V')} \right) \right)^{N+1} \\
&\quad \leq c_1 \sqrt{N+1} \left(c_2 \exp(\lambda T) \left(\|y_0\|_{V^\alpha} + \|y_e\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \right) \right)^{N+1},
\end{aligned}$$

where $\tilde{c}_{\lambda,\alpha}$ is a constant depending only on $\|L^{\alpha/2} A_\lambda^{-\alpha/2}\|_{\mathcal{L}(H,H)}$, $\|L^{\alpha/2} A_\lambda^{-\alpha/2}\|_{\mathcal{L}(V,V)}$, and $\|L^{\alpha/2} A_\lambda^{-\alpha/2}\|_{\mathcal{L}(V',V')}$, while c_2 is defined analogously, but with $A_\lambda^{\alpha/2} L^{-\alpha/2}$. The constant $\bar{c}(\gamma,\beta,c_\mathcal{P}) > 0$ depends only on the parameters listed. In the special case $\alpha = 0$, we have $\tilde{c}_{\lambda,\alpha} = c_2 = 1$. □

Remark 4.1. *Theorem 4.2 assumes that the exact solution y_e exhibits increased regularity compared to the results discussed in Section 4.2. Proving improved regularity can be achieved with additional assumptions on the initial condition, the forcing, and the involved operators. In the linear case, this is broadly covered; see, e.g., [91, pp. 180 ff.] for general weak forms of Cauchy problems on Hilbert spaces and [16, pp. 410 ff.] specifically for PDEs. In the nonlinear case, this has also been explored, particularly through concepts like strong variational solutions*

to the three-dimensional Navier–Stokes equations; see, e.g., [99]. Alternatively, one could extend assumption (A3) *in such a way that it accounts for* α *similarly to* (A4)*. This would allow for an equivalent of Lemma 4.3 with improved regularity. However, a detailed discussion is deferred as it would exceed the scope of this work.*

Theorem 4.2 can be refined if we assume $\alpha = 0$ and leverage the results about solutions to the nonlinear Cauchy problem (4.2).

Corollary 4.1 (Convergence for $\alpha = 0$)**.** *Let* $T < \infty$*. Let* A *fulfill* (A1) *and* (A2) *with constants* β*,* γ*, and* λ*, let* B *fulfill* (A3)*, and let* $f \in L^2(0,T;V') \cap L^\infty(0,T;H)$ *and* $y_0 \in H$*. Additionally, assume that* B*,* f*, and* y_0 *are so small that* $c_{\mathcal{P}} < \gamma$ *and* $\|y_0\|_H + \|f\|_{L^2(0,T;V')} \leq \rho(T)$*, where* $c_{\mathcal{P}}$ *and* $\rho(T)$ *are the constants from Proposition 4.4 and Lemma 4.3. Let* $y_e \in W(0,T;V,V')$ *be the unique solution to* (4.2)*. Then, it holds that*

$$\|\eta\|_{L^\infty(0,T;H)} + \|\eta\|_{W(0,T;V,V')} \leq c_1\sqrt{N+1}\left(2c_N \exp(2\lambda T)\left(\|y_0\|_H + \|f\|_{L^2(0,T;V')}\right)\right)^{N+1},$$

where $\eta(t) = y_N^{(1)} - y_e(t)$ *and* $y_N \in W(0,T;U_1^0(N),U_{-1}^0(N))$ *is the unique solution to* (TCW) *for any* $N \in \mathbb{N}$*. The constant* $c_1 > 0$ *does not depend on* N*,* T*,* y_0*, or* f*, and* c_N *is the constant from Lemma 4.3. If* $\lambda = 0$*, one can choose* $T = \infty$*.*

Proof. Since assumption (A3) implies (A4) for $\alpha = 0$, the statement follows from Theorem 4.2 and Lemma 4.3. □

4.5.1 Assumptions and Implications of the Result

Subsequently, we interpret the assumptions of Theorem 4.2 and Corollary 4.1, and highlight their influence on the linearization's performance.

First, consider the assumptions $c_{\mathcal{P}} < \gamma$ and $\|y_0\|_H + \|f\|_{L^2(0,T;V')} \leq \rho(T)$. The nonlinear operator B is assumed to fulfill (A4). The constant $c_{\mathcal{B}}(\alpha,0)$ is bounded by $c_B(\alpha,\varepsilon)$, in particular, there is a $\mu > 0$ such that $c_{\mathcal{B}}(\alpha,\varepsilon) \leq \mu c_B(\alpha,\varepsilon)$. The constant $c_B(\alpha,\varepsilon)$, in turn, determines the size of the nonlinearity B. This becomes apparent if we assume that B is given as a scaled nonlinear term B_0, i.e., $B = \omega B_0$ with $\omega \in \mathbb{R}$. Then, one can choose $c_B(\alpha,\varepsilon) = \omega c_{B_0}(\alpha,\varepsilon)$. For example, in fluid flow problems, B might represent a convection term, and ω the Reynolds number. What this means for Theorem 4.2 is that ω and f have to be chosen sufficiently small that $c_{\mathcal{P}} < \gamma$, i.e., the Carleman linearization is only admissible for small nonlinearities and forcing. The coercivity constant γ can be interpreted as a measure of the stability of the operator A. Hence, more stable linear parts allow for larger nonlinearities. Similar implications hold for the assumption $\|y_0\|_H + \|f\|_{L^2(0,T;V')} \leq \rho(T)$ in Corollary 4.1. The constant $\rho(T)$ is inversely proportional to $c_B(\alpha,0)$, and, by extension, to ω. This means that the set of admissible y_0 and f restricted by this assumption grows as the nonlinearity gets smaller. Moreover, if $\lambda > 0$, i.e., if the linear part is asymptotically unstable, it holds that $\rho(T) \to 0$ exponentially in T. So, the convergence result can only be applied on finite time horizons if $\lambda > 0$.

Turning to the error bound of Theorem 4.2, we observe that convergence of the approximate solution is guaranteed if and only if

$$c_2 \exp(\lambda T) \left(\|y_0\|_{V^\alpha} + \|y_e\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \right) < 1.$$

Since $\lambda \geq 0$, this necessitates for the solution to fulfill $c_2(\|y_0\|_{V^\alpha} + \|y_e\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})}) < 1$. On the one hand, this enforces an upper bound on the initial condition. On the other hand, it restricts the size of the exact solution, which entails multiple restrictions on our dynamical system. Since $\|y_e\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \to 0$ as $T \to 0$, the condition might give an upper bound on feasible T and larger time horizons also lead to poorer approximation properties of the linearization. This part of the estimate, however, does not necessarily lead to an upper bound of feasible T in general since the solution y_e can be bounded on the unbounded time horizon $[0, \infty)$ in the case $\lambda = 0$. Additionally, it is suggested that larger nonlinearities may lead to solutions y_e with larger norms, resulting in poorer convergence or leaving one without a guarantee of convergence at all. Taking into account how Corollary 4.1 relates the norm of the solution to the forcing, the condition for convergence reads as

$$c_2 \exp(\lambda T) \left(\|y_0\|_H + \|f\|_{L^2(0,T;V')} \right) < 1$$

and, therefore, also implies an upper bound on f. In the unstable case $\lambda > 0$, we have that convergence is only guaranteed for certain bounded time horizons with fixed y_0 and f.

These observations coincide with what is generally known about the linearization in the finite-dimensional case. Theorem 4.2 can be seen as an equivalent result to Theorem 4.2 in [87] without forcing and Corollary 4.1 to Theorems 3.2, 3.5, and 3.6 in [89] including the case of forcing. A notable difference is that our estimates include an additional factor $\sqrt{N+1}$, which means that our results only show (sub-)exponential convergence.

Remark 4.2. *The result from Theorem 4.2 and Corollary 4.1 is designed for the truncated Carleman linearization, i.e., $y_{N+1} = 0$ is assumed in the chain of moment equations. As we have seen, there is an upper bound for admissible norms of $\|y_0\|$, and the linearization does not converge even locally in time if the initial condition is too large. However, if an estimate $\hat{y}_{N+1}(t) = \bigotimes^{N+1} \hat{y}(t) \approx y_{N+1}(t)$ is available, we can refine the linearization and the corresponding error bound. This estimate can be incorporated into the linearization by changing the right-hand side to $\hat{f}_N(t) = (f(t), 0, \ldots, 0, -B_N \hat{y}(t))^T$. By adapting Theorem 4.2 for this case, we end up with*

$$\begin{aligned} &\|\eta\|_{L^\infty(0,T;V^\alpha)} + \|\eta\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \\ &\quad \leq c_1 \sqrt{N+1} \left(c_2 \exp(\lambda T) \left(\|y_0 - \hat{y}(0)\|_{V^\alpha} + \|y_e - \hat{y}\|_{W(0,T;V^{1+\alpha},V^{-1+\alpha})} \right) \right)^{N+1}. \end{aligned}$$

Hence, if a guess $\hat{y}$ is close enough to y_e, we achieve convergence. Furthermore, this implies that if the assumptions of Theorem 4.2 are fulfilled and $\hat{y}(0) = y_0$, then there exists a (sufficiently small) $T > 0$ such that the linearization converges. This provides a local-in-time convergence result for arbitrarily large data y_0 and f.

4.5.2 Examples of Operators A and B

We examine specific cases of PDE problems where the assumptions (A3) and (A4) on B are fulfilled. We assume that A is a second-order elliptic differential operator on a bounded Lipschitz domain $\Omega \in \mathbb{R}^d$ for $d \in \mathbb{N}$. We restrict ourselves to the case of homogeneous Dirichlet boundary conditions and, therefore, choose $D(A) = H^2(\Omega) \cap H_0^1(\Omega)$ and $H = L^2(\Omega)$, or vectorized versions of them. In this context, $L = -\Delta + I$ with $\bar{\alpha} = 2$ serves as a suitable choice, and $V = H_0^1(\Omega)$. Consequently, the V-norm is equivalent to the standard $H^1(\Omega)$-norm.

Zeroth-order derivatives

First, we consider the case of B being a zeroth-order quadratic term, i.e., of the form

$$B(u, v)(x) := \psi(x)u(x)v(x)$$

for some $\psi \in L^\infty(\Omega)$. From Ladyzhenskaya's inequality, we know that $H^1(\Omega)$ is continuously embedded in $L^4(\Omega)$ for dimensions $d \leq 2$. So, there is a constant c_{LA} such that

$$\|u\|_{L^4(\Omega)} \leq c_{LA} \|u\|_{L^2(\Omega)}^{\frac{1}{2}} \|u\|_{H^1(\Omega)}^{\frac{1}{2}}$$

for all $u \in H^1(\Omega)$. Thus, for $d \leq 2$

$$\begin{aligned} |\langle B(u, v), w\rangle_V| &\leq \|\psi\|_{L^\infty(\Omega)} \|u\|_{L^4(\Omega)} \|v\|_{L^4(\Omega)} \|w\|_{L^2(\Omega)} \\ &\leq c_{LA}^2 \|\psi\|_{L^\infty(\Omega)} \|u\|_{L^2(\Omega)}^{\frac{1}{2}} \|u\|_{H^1(\Omega)}^{\frac{1}{2}} \|v\|_{L^2(\Omega)}^{\frac{1}{2}} \|v\|_{H^1(\Omega)}^{\frac{1}{2}} \|w\|_{L^2(\Omega)} \end{aligned}$$

for all $u, v \in V$ and $w \in H$. This proves the required bound for (A3) and (A4) for $\alpha = 0$ and $\varepsilon \in [0, 1]$. In the case $d = 3$, we can prove (A3). Using the Gagliardo–Nirenberg interpolation inequality in bounded domains

$$\|u\|_{L^3(\Omega)} \leq c_{GN} \|u\|_{L^2(\Omega)}^{\frac{1}{2}} \|u\|_{H^1(\Omega)}^{\frac{1}{2}},$$

we obtain the estimate

$$\begin{aligned} |\langle B(u, v), w\rangle_V| &\leq \|\psi\|_{L^\infty(\Omega)} \|u\|_{L^3(\Omega)} \|v\|_{L^3(\Omega)} \|w\|_{L^3(\Omega)} \\ &\leq c_{GN}^3 \|\psi\|_{L^\infty(\Omega)} \|u\|_{L^2(\Omega)}^{\frac{1}{2}} \|u\|_{H^1(\Omega)}^{\frac{1}{2}} \|v\|_{L^2(\Omega)}^{\frac{1}{2}} \|v\|_{H^1(\Omega)}^{\frac{1}{2}} \|w\|_{H^1(\Omega)} \end{aligned}$$

for all $u, v, w \in V$. Thus, assumption (A3) is fulfilled for $d = 3$, and, therefore, also (A4) with $\alpha = 0$ and $\varepsilon = 0$; however, it does not hold for $\varepsilon = 1$ in general.

Fluid flow problems

In the case of the incompressible Navier–Stokes equation, the original coupled problem can be rephrased as a parabolic Cauchy problem with a quadratic nonlinearity by the use of the Leray projection P and by restricting H to solenoidal vector fields (see, e.g., [99]), and, therefore, fits our framework. Then, the nonlinear term reads as

$$B(u,v) = \frac{1}{2}\left(\widehat{B}(u,v) + \widehat{B}(v,u)\right), \quad \widehat{B}(u,v) = P(u \cdot \nabla v).$$

In this case, assumption (A4) holds for $\alpha > d/2-1$ and $0 \leq \varepsilon < \alpha - d/2+1$, cf. [97]. In the case of $d = 1, 2$, assumption (A3) is fulfilled; see, e.g., [94, 1]. While this implies (A4) for $\alpha = 0$ and $\varepsilon = 0$ for $d = 1, 2$, it does not guarantee this property for $\varepsilon = 1$. This means that Theorem 4.1 cannot be applied to show well-posedness, but one has to use Theorem 4.2 coupled with assumptions of small nonlinearities, forcings, and initial conditions. This example underscores the importance of carefully examining the various assumptions on B.

Nonlocal interaction terms

Some problems modelled by PDEs can involve nonlocal and possibly nonlinear terms. Examples of such problems can be found in multiscale particle dynamics. Our framework also covers nonlocal nonlinear problems including low-order derivatives. For example, the so-called dynamic density functional theory [100, 101] tackles particle dynamics problems by approximating them by a PDE with a nonlocal term of the form

$$B(u,v) = \frac{1}{2}\left(\widehat{B}(u,v) + \widehat{B}(v,u)\right), \quad \widehat{B}(u,v)(x) = \nabla_x \cdot \int_\Omega u(x)v(x')K(x,x')\mathrm{d}x'$$

for some vector-valued kernel function $K(x,x')$. If we assume that $K \in (L^\infty(\Omega \times \Omega))^d$ and $\nabla_x \cdot K \in L^\infty(\Omega \times \Omega)$, we can show

$$\begin{aligned}\left|\langle \widehat{B}(u,v), w\rangle_V\right| &= \left|\int_\Omega w(x)\nabla_x \cdot \int_\Omega u(x)v(x')K(x,x')\mathrm{d}x'\mathrm{d}x\right| \\ &= \left|\int_\Omega \int_\Omega \left(\nabla_x w(x)\right) u(x)v(x')K(x,x')\mathrm{d}x'\mathrm{d}x\right| \\ &\leq \|K\|_{L^\infty(\Omega\times\Omega)}\|u\|_{L^2(\Omega)}\|v\|_{L^1(\Omega)}\|w\|_{H^1(\Omega)}\end{aligned}$$

by using integration by parts combined with the homogeneous boundary conditions. So, B fulfills (A3) and, therefore, also (A4) for $\alpha = 0$ and $\varepsilon = 0$. Furthermore, we obtain that

$$\begin{aligned}\left|\langle \widehat{B}(u,v), w\rangle_V\right| &= \left|\int_\Omega \int_\Omega w(x)v(x')\left[u(x)\nabla \cdot K(x,x') + (\nabla u(x)) \cdot K(x,x')\right]\mathrm{d}x'\mathrm{d}x\right| \\ &\leq \int_\Omega \int_\Omega \left[\left|w(x)v(x')u(x)\nabla \cdot K(x,x')\right| + \left|w(x)v(x')\left(\nabla u(x)\right) \cdot K(x,x')\right|\right]\mathrm{d}x'\mathrm{d}x \\ &\leq \left(\|\nabla \cdot K\|_{L^\infty(\Omega\times\Omega)} + \|K\|_{(L^\infty(\Omega\times\Omega))^d}\right)\left(\|u\|_{L^2(\Omega)}\|v\|_{H^1(\Omega)} + \|u\|_{H^1(\Omega)}\|v\|_{L^2(\Omega)}\right)\|w\|_{L^2(\Omega)},\end{aligned}$$

which implies (A4) for $\alpha = 0$ and $\varepsilon = 1$. It is noted that this holds true for any dimension $d \in \mathbb{N}$.

4.6 Numerical Approximations

Since we consider infinite-dimensional Hilbert spaces H, the presented parabolic Cauchy problems stemming from the Carleman linearization cannot be solved exactly in general. To address this, one must first discretize the equations and compute a numerical approximation. As we will see, the convergence analysis in Section 4.5 not only improves the understanding of the Carleman linearization for infinite-dimensional problems in a continuous setting but also clarifies its discretized counterpart. By pursuing a so-called linearize-then-discretize approach, the error analysis of the truncation is decoupled from the discretization error. This substantially differentiates our approach from traditional *discretize-then-linearize* methods, which analyze the Carleman linearization of a finite-dimensional approximation of the original Cauchy problem. Additionally, the linearize-then-discretize approach will enable alternative discretization approaches.

As shown before, the truncated Carleman linearization results in a parabolic problem. The numerical solution of parabolic problems is broadly covered in the literature; see, e.g., [17, 102]. Many discretization techniques can be organized into two separate topics: the spatial and the temporal discretization. In this section, we focus only on the spatial discretization, leaving aside the discussion of temporal discretizations as they are not relevant to this chapter. Specifically, we analyze the so-called semi-discrete problem, which is obtained by only approximating H but keeping the problem continuous in time. Subsequently, we outline the fundamental implications for the semi-discrete Cauchy problem based on the obtained results on the Carleman linearization.

We choose to use a conforming Galerkin method for the spatial discretization; see, e.g., [102]. It is assumed that all assumptions of Corollary 4.1 are fulfilled. Let $U_h(N) \subset U_1^0(N)$ be a family of finite-dimensional subspaces of $U_1^0(N)$ with the property

$$\inf_{u_h \in U_h(N)} \|u_h - u\|_{Y(N)} \to 0 \quad \text{as} \quad h \to 0 \quad \text{for all } u \in U_1^0(N). \tag{A5}$$

In the case of Galerkin finite element methods for PDEs, h denotes the maximum cell size, for instance. We then seek a function $y_{N,h} \in W(0,T;U_h(N),(U_h(N))')$ such that (TCW) is fulfilled, in which we only test against functions in $U_h(N)$. This equation is referred to as the semi-discretization of (TCW). The (semi-)discretization leads to an additional error source in our linearization. This becomes apparent when considering

$$\begin{aligned}\|y_{N,h}^{(1)} - y_e\|_{L^\infty(0,T;H)} &\leq \|y_N^{(1)} - y_e\|_{L^\infty(0,T;H)} + \|y_{N,h}^{(1)} - y_N^{(1)}\|_{L^\infty(0,T;H)} \\ &\leq \|y_N^{(1)} - y_e\|_{L^\infty(0,T;H)} + \|y_{N,h} - y_N\|_{L^\infty(0,T;Y(N))}.\end{aligned}$$

The first term of the right-hand side, the linearization error, is analyzed in Corollary 4.1, while the second term, the discretization error, purely measures the approximation error of the Galerkin discretization. We thereby established a separation of the two error sources. The same observation remains valid if the norms $L^\infty(0,T;H)$ and $L^\infty(0,T;Y(N))$ are replaced by the norms $L^2(0,T;V)$ and $L^2(0,T;U_1^0(N))$, respectively. The following lemma is a variation of Theorem 23.A in [17] and can be found in various standard references on the topic.

Lemma 4.8. *Let all assumptions of Corollary 4.1 be fulfilled. Moreover, let $N \in \mathbb{N}$ be fixed and $U_h(N) \subset U_1^0(N)$ be a family of subspaces fulfilling* (A5)*. Let $y_N \in W(0,T;U_1^0(N),U_{-1}^0(N))$ be the solution to* (TCW) *and $y_{N,h} \in W(0,T;U_h(N),(U_h(N))')$ its Galerkin approximation. Then, it holds that*

$$\begin{aligned} \|y_N(t) - y_{N,h}(t)\|_{L^\infty(0,T;Y(N))} &\to 0, \\ \|y_N - y_{N,h}\|_{L^2(0,T;U_1^0(N))} &\to 0, \end{aligned}$$

as $h \to 0$.

Hence, the assumptions ensure that the discretization error vanishes as $h \to 0$ for a fixed N. Informally, this means that the discretized Carleman linearization converges to the exact solution as $N \to \infty$ and $h \to 0$ simultaneously. However, careful selection of h is crucial for ensuring overall convergence behavior. As N increases, the dimensionality of problem (TCW) grows, possibly detoriating the discretization's approximation properties due to the curse of dimensionality. Although these factors strongly depend on the specific discretization method, this suggests that one has to choose h such that it decreases sufficiently fast as a function of N to gain overall convergence.

It remains an open question how one can choose $U_h(N)$ appropriately. The subsequent sections present a standard choice of discretization but also initiate the idea for nonstandard approaches that exploit the structure of the linearization.

4.6.1 Standard Discretization

We begin by considering a discretization that intuitively mirrors the construction of the spaces $V_1^0(k)$ in their discrete form. Let $V_h \subset V$ be a family of subspaces that fulfills

$$\inf_{v_h \in V_h} \|v - v_h\|_H \to 0 \quad \text{as} \quad h \to 0 \quad \text{for all } v \in V.$$

Then, we construct an approximation $V_h(k) \subset V_1^0(k)$ by assembling the tensor products of V_h, i.e., $V_h(k) = \bigotimes^k V_h$. This suggests $U_h(N) = \bigoplus_{k=1}^N V_h(k)$. It can be shown that this approximation space fulfills (A5). We refer to this discretization as the standard discretization since it coincides with the system one would obtain from a discretize-then-linearize approach.

As opposed to existing analyses of the discretize-then-linearize approach, our convergence radii and rates depend on y_0, f, and B but are independent of h. In some instances, similar results can be achieved using the discretize-then-linear approach; see, e.g., [86] for diffusion equations with nonlinear reaction terms. However, for problems in which B includes derivatives, like the problems in Sections 4.5.2 and 4.5.2, such an approach fails. This is due to the diverging scaling behaviors of the norms of the discretized operators A_h and B_h with respect to h. In particular, for fluid flow problems, the discrete coercivity constant γ_h of A_h remains $O(1)$ but $\|B_h\| = O(h^{-1})$ as $h \to 0$. This discrepancy ultimately results in a violation of the condition $c_{\mathcal{P}} < \gamma$ in Corollary 4.1 if h is chosen small enough. For the same reason, the error estimates of [87, 89] are compromised as $h \to 0$ and do not provide a theoretical guarantee for convergence for small h. This was observed in, e.g., [90]. By isolating the truncation error from the discretization error in our analysis, we bridge this knowledge gap and enhance the understanding of the convergence behavior of the linearization for general infinite-dimensional systems.

4.6.2 Non-Standard Discretizations

A significant drawback of the standard Carleman linearization is that it suffers from the curse of dimensionality. Using the approximation space $V_h(k)$ from the previous section leads to an exponential scaling of the degrees of freedom of $U_h(N)$ in N. This scaling is already problematic for finite-dimensional Cauchy problems of low dimension in combination with moderate regimes of N, but depending on the specific dynamical system, this effect can become more severe in the case of infinite-dimensional equations. For PDEs, for example, relatively small models can lead to dimensions in the order of $\dim V_h = O(10^5)$ to reach satisfactory accuracy, and therefore $\dim U_h(N) = O(10^{5N})$. Consequently, the Carleman linearization might not be numerically tractable even for $N = 2$.

Instead of using the tensor product spaces, we have more freedom in choosing different approximation spaces for each k. High-dimensional equations that involve tensor product and Kronecker sum structures, similar to those found in the Carleman linearization, have been extensively studied. This research provides a wide range of efficient methods specifically designed to address the curse of dimensionality. We refer to the application of such methods as non-standard discretizations. To demonstrate the flexibility and potential of the linearize-then-discretize approach for non-standard discretizations, we focus on one specific method known as *sparse grids* or *sparse tensor product spaces*. For more information on the general theory and approximation properties of sparse grids; see [103, 104, 105, 106]. For their application to PDEs and Galerkin methods, we refer to [107, 108]. In addition to showcasing the capability of non-standard discretizations, the use of sparse grids will enable numerical experiments on otherwise intractable problems.

Similarly to the previous discretization, we start with a discretization of V and, additionally, assume that it exhibits a nested sequence of spaces

$$V_{h_1} \subset V_{h_2} \subset \cdots \subset V_{h_j} \subset \cdots \subset V$$

for the decreasing sequence $h_j = 2^{-j}$ for $j \in \mathbb{N}$. Such a hierarchy can, for example, arise from successive mesh refinement when working with PDEs. For the sake of readability, we write $V_j := V_{h_j}$ for indices j. It is assumed that the approximation error decreases at the rate

$$\inf_{v_h \in V_j} \|v - v_h\|_H \leq c h_j \|v\|_V$$

for some $c > 0$, and the dimensions of the spaces scale as $\dim V_j \sim h_j^{-d}$ for some $d \in \mathbb{N}$. If we now want to roll out V_J for some fixed $J \in \mathbb{N}$ to a discretization of $V_1^0(k)$ and follow the same procedure as in the standard discretization, we would arrive at an exponentially growing space $V_J(k)$. Sparse grids, however, suggest that if a function $v \in V_1^0(k)$ exhibits increased regularity of the kind $v \in V_0^1(k)$, a large portion of the elements of $V_J(k)$ can be neglected without a significant loss in accuracy. In particular, within such a method one chooses the approximation space

$$\widehat{V}_J(k) := \operatorname{span}\left\{ \bigcup_{\vec{j} \in \mathbb{N}^k, |\vec{j}|_1 \leq J} \bigotimes_{i=1}^{k} V_{\vec{j}_i} \right\}.$$

This means that instead of coupling the finest discretization V_J in each dimension within the tensor product, we only couple fine discretizations with coarser ones. This translates to the condition $|\vec{j}|_1 \leq J$. The dimensionality of $\widehat{V}_J(k)$ can be bounded by $\dim \widehat{V}_J(k) \lesssim h_J^d J^{k-1}$ as opposed to $\dim V_J(k) \sim h_J^{kd}$. Finally, we choose $\widehat{U}_h(N) := \bigoplus_{k=1}^{N} \widehat{V}_J(k)$.

The additional regularity assumptions can be justified through the following consideration. If we assume that the assumptions of Corollary 4.1 are fulfilled and also that those of Theorem 4.2 hold for $\alpha = 1$, then the unique solution has regularirty $y_N \in W(0,T; V_1^1(N), V_{-1}^1(N)) \subseteq C(0,T; V_0^1(N))$. Increased regularity holds for larger values of α, which can be complemented with higher-order sparse grid methods as well.

Another example of structure-exploiting methods is the tensor train decomposition [109]. This method has been successfully applied to problems of Kronecker product form to achieve low-rank approximations. For example, if the underlying nonlinear Cauchy problem is a parabolic PDE, the Carleman linearization describes a system of high-dimensional PDEs. This becomes apparent if we take the example of $H = L^2(\Omega)$ and $V = H_0^1(\Omega)$. Then, it holds that $H(k) = L^2(\Omega^k)$ and $V_1^0(k) = H_0^1(\Omega^k)$, i.e., the tensorized spaces are equivalent to Sobolev spaces in higher-dimensional domains. A survey on the efficacy of tensor train decompositions for high-dimensional PDEs can be found in [110]. A detailed study of this and other non-standard discretizations would exceed the scope of this thesis and is a subject of future research.

4.7 Numerical Experiments

We now verify our theoretical findings with a series of numerical experiments. These experiments include a qualitative analysis of the recovery of nonlinear behavior via the linearization, as well as measuring the convergence properties reflecting the theoretical bounds. Additionally, we present the advantages of the non-standard discretization method employed. Although our primary focus is on a second-order parabolic PDE, it is important to note that the framework we propose is versatile and applicable to a broader range of equations.

The methods are implemented in Python. For the discretization of PDEs, the high-level finite element method library DOLFINx [68] is used. The storage and computation of dense and sparse tensors are done with the tensor compiler suite taco [111]. The library petsc4py [70] is used for various numerical linear algebra operations. The source code used for the experiments is publicy available under `https://github.com/bheinzelreiter/carleman-pde`.

4.7.1 Burgers' Equation and Discretization

As a model problem, we consider the one-dimensional Burgers' equation extended by a destabilizing linear term. The PDE in its strong form is given by

$$\begin{aligned} y'(t,x) - \nu\Delta y(t,x) - \lambda y(t,x) + y(t,x)\frac{\partial}{\partial x}y(t,x) &= f(t,x) \quad && \text{for } t \in [0,T) \text{ and } x \in [-1,1], \\ y(t,-1) = y(t,1) &= 0 && \text{for } t \in [0,T), \\ y(0,x) &= y_0(x), \end{aligned}$$

with the viscosity $\nu > 0$ and the coefficient $\lambda \geq 0$ of the destabilizing term. In our notation, the linear operator corresponds to the weak form of $Au = -\nu\Delta u - \lambda u$ and the quadratic operator to $B(u \otimes v) = 1/2(uv_x + vu_x)$. We use the initial condition

$$y_0(x) = \frac{2\pi b \sin(\pi x)}{a + \cos(\pi x)}. \tag{4.10}$$

for some constants $a > 1$ and $b > 0$. This initial condition is adopted from [112], wherein it is shown that the Burgers equation with initial condition (4.10) admits an exact solution if $\lambda = 0$, $b = \nu$, and $f = 0$:

$$y(t,x) = \frac{2\pi\nu \exp(-\pi^2\nu t)\sin(\pi x)}{a + \exp(-\pi^2\nu t)\cos(\pi x)}.$$

We used the exact solution to verify our implementation. In experiments including forcing, we take

$$f(t,x) = c(x^2 - 1)$$

for some constant $c \in \mathbb{R}$.

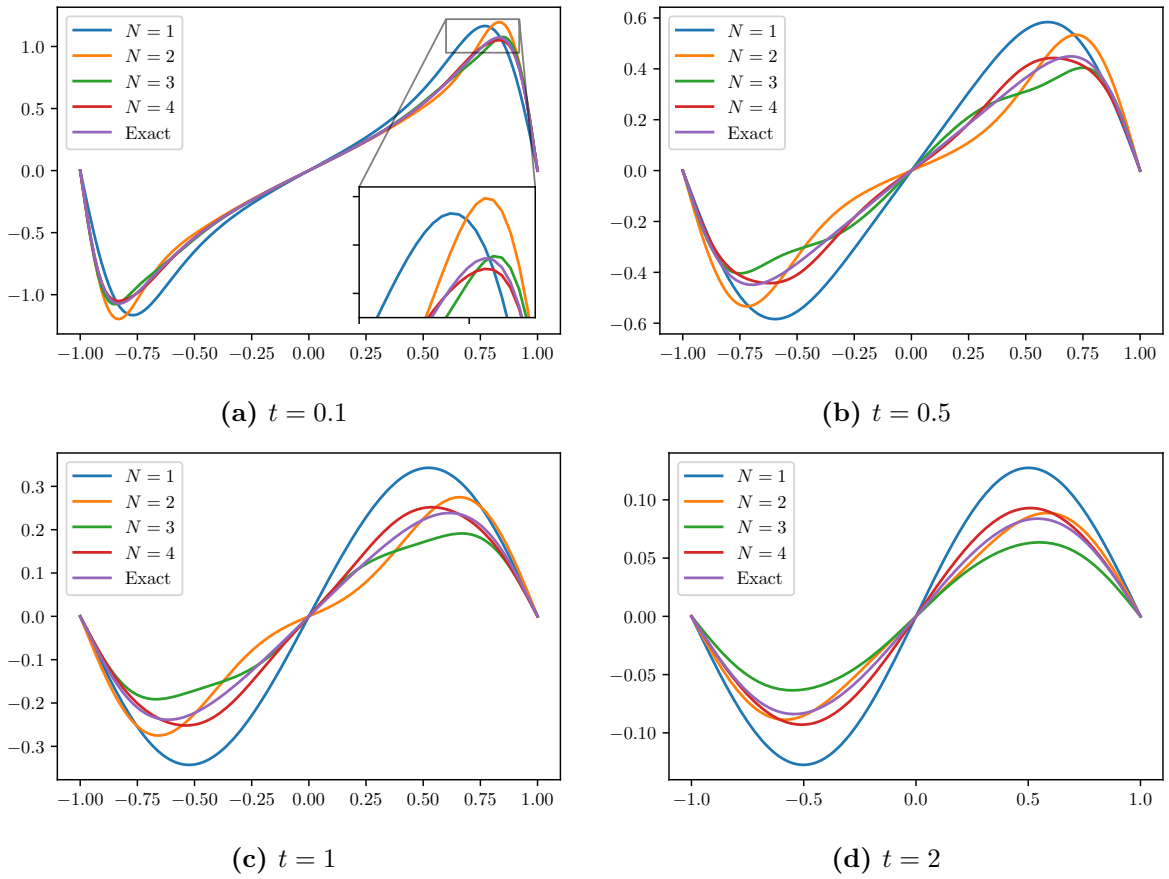
The Carleman linearization is discretized using continuous piecewise linear finite elements in space and the implicit Euler method in time. It is noted that the time discretization is set to a high accuracy in order to avoid possible interference of the linearization, finite element discretization, and time integration errors. To measure the errors of approximations when there is no known exact solution, we compute a baseline solution using a pseudospectral discretization of the equations. To accommodate high regimes of N, we approximate the higher-order terms using the sparse grids technique known as the combination technique [113]. In the case of no forcing, i.e., $c = 0$, the linear system arising from a single implicit Euler step has a block-triangular structure. This allows us to solve the system recursively, where each step involves solving an elliptic PDE linked to varying truncation levels, starting with the block associated with A_N, i.e., the right bottom block of $\mathcal{A}_N(t)$. In the case of $c \neq 0$, we have to turn to more elaborate methods since the linear system has a block-triangular structure. Theorem 4.2 and Corollary 4.1 imply a form of block-diagonal dominance of the block operator matrix $\mathcal{A}_N(t)$, which transfers to the linear system of a single implicit Euler step. This property led to the convergence of a block Gauss–Seidel method applied to the linear system. While this property holds in the undiscretized setting, it still has to be shown that it remains true upon discretization. This task is deferred to future research; however, it has been found to hold true empirically, and convergence of the block Gauss–Seidel method is achieved in our numerical experiments.

4.7.2 Snapshots of Solutions

First, we analyze snapshots of the exact solution to the Burgers equation in comparison to those obtained through the Carleman linearization. We set $\nu = 0.1$, $a = 1.05$, $b = 0.1$, and $c = 0$. Figure 4.1 shows the solutions at four timestamps for truncation levels up to $N = 4$. At $t = 0.1$, we observe that a higher-order linearization results in a better approximation of the dynamical system. However, it also shows that the quality of the approximation deteriorates over time. This is confirmed in Figure 4.2, which illustrates the normalized error given by $(y_N^{(1)}(t) - y_e(t))/\|y_e(t)\|_H$.

4.7.3 Approximation Error

Theorem 4.2 and Corollary 4.1 give an upper bound on the expected approximation error arising from the linearization, which suggests (sub-)exponential convergence with respect to N. In this section, we verify these convergence rates by analyzing how the error behaves for varying parameters T, c_B, y_0, f, and λ. Each of these parameters are examined in a separate subsection.



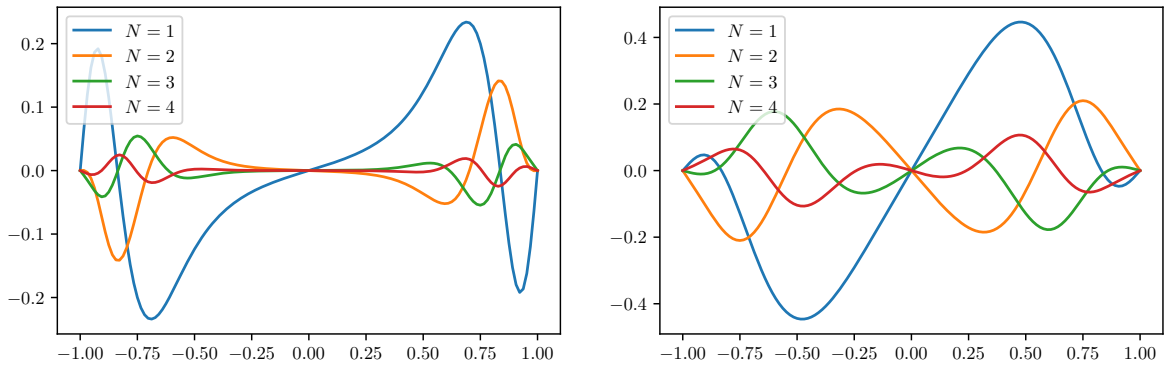
(a) $t = 0.1$ **(b)** $t = 0.5$

(c) $t = 1$ **(d)** $t = 2$

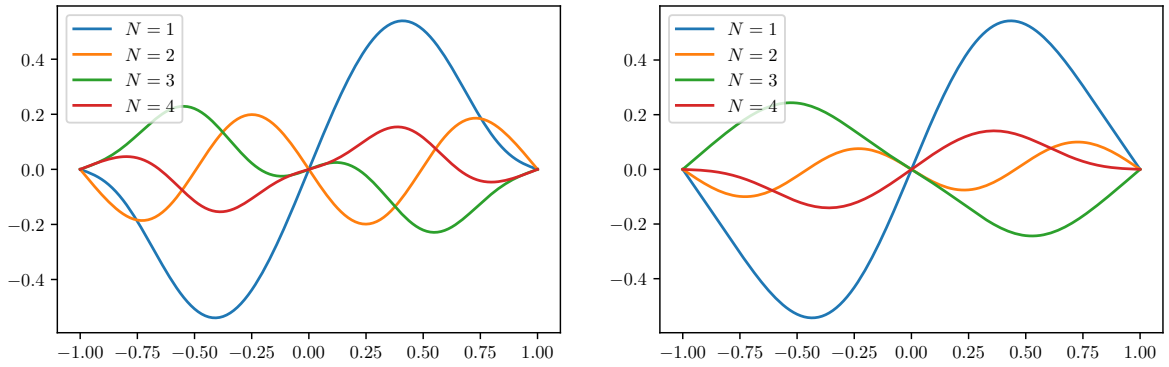
Figure 4.1: Snapshots of solutions to the linearization of the Burgers' equation $y_N^{(1)}(t)$ for different truncation levels N compared to the exact solution $y_e(t)$.

Dependence on T

The convergence rate given by Theorem 4.2 is partially governed by the $W(0,T;V^{1+\alpha},V^{-1+\alpha})$-norm of y_e. Since this expression grows as T is increased, this suggests that slower convergence is to be expected for larger time horizons. Figure 4.3a shows the approximation error as a function of N measured with $\|\eta\|_{L^\infty(0,T;H)}$ for $\nu = 0.01$, $a = 1.05$, $b = 0.01$, $c = 0$, $\lambda = 0$, and various final times T. It is observed that larger values of T lead to slower convergence. Furthermore, the error decreases exponentially fast with respect to N initially until a certain threshold is reached. This phenomenon reflects the two error source terms, the linearization and the discretization error. The linearization error decays exponentially until the discretization error dominates.



(a) $t = 0.1$ **(b)** $t = 0.5$



(c) $t = 1$ **(d)** $t = 2$

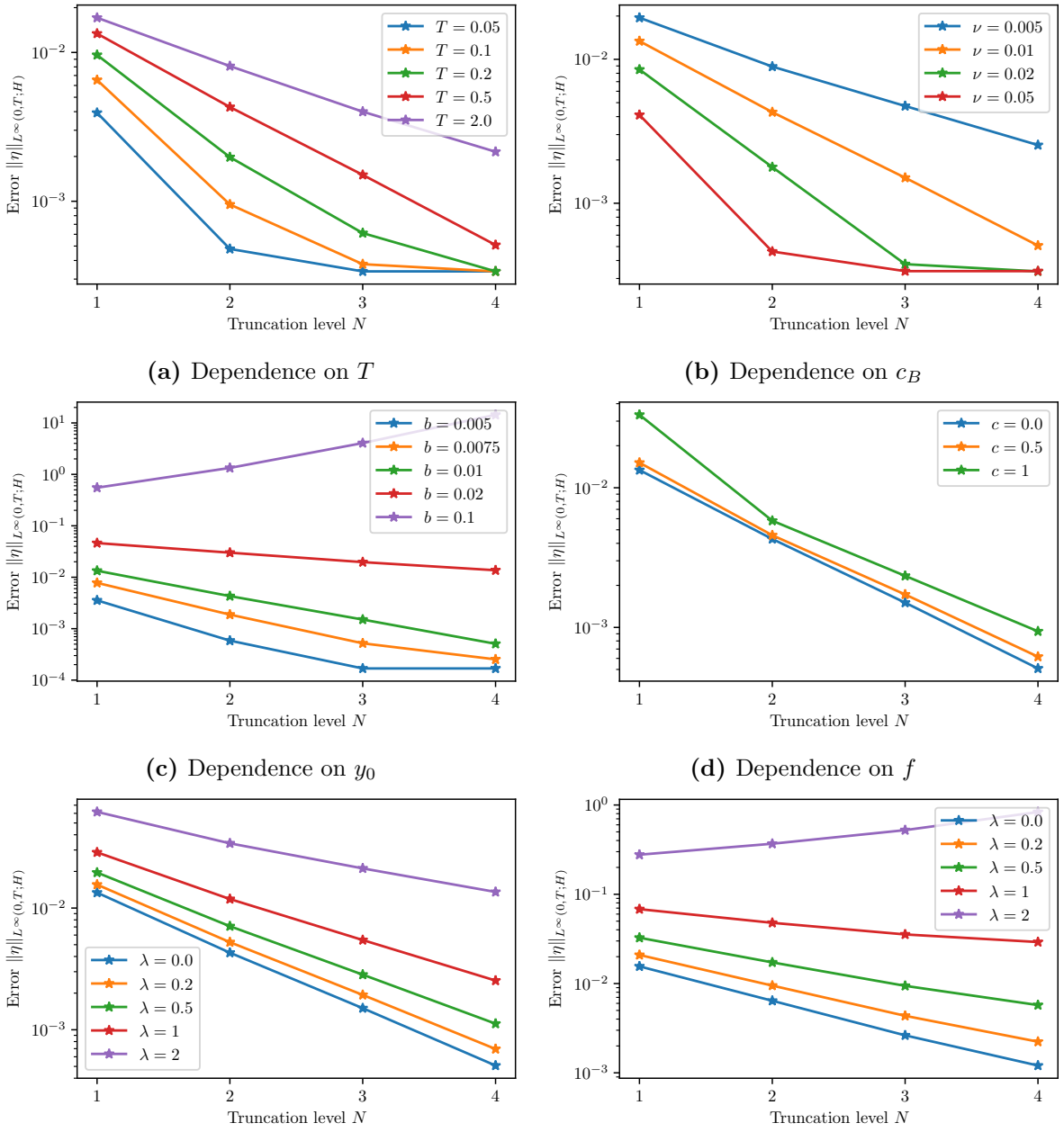
Figure 4.2: Error plots of snapshots of solutions to the linearization of the Burgers' equation $(y_N^{(1)}(t) - y_e(t))/\|y_e(t)\|_H$ for different truncation levels N.

Dependence on c_B

As observed in the preceding subsection, the convergence rate is determined by the size of the exact solution, which is also affected by the size of the nonlinearity c_B. In the case of the Burgers' equation, we have that $c_B \sim \nu$. Figure 4.3b depicts the error as a function of N for $T = 0.5$, $a = 1.05$, $b = 0.01$, $c = 0$, $\lambda = 0$, and various values for ν. This verifies the deterioration of the convergence rate as ν is decreased, as well as exponential convergence until the discretization error is reached.

Dependence on y_0

In addition to the norm of y_e, the convergence rate also depends on the initial value y_0. Similarly, we expect the convergence to worsen as y_0 becomes larger. In our model problem, the size of the initial condition is determined by the parameter b. Figure 4.3c shows the error as a function of N for $T = 0.5$, $\nu = 0.01$, $a = 1.05$, $c = 0$, $\lambda = 0$, and various values of b. The plot confirms the expected behavior and also demonstrates that the linearization diverges when the initial condition is too large.



(a) Dependence on T

(b) Dependence on c_B

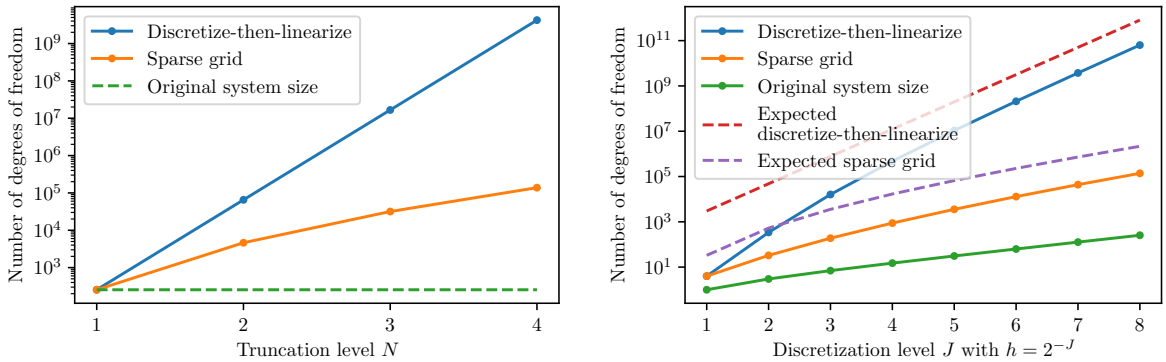
(c) Dependence on y_0

(d) Dependence on f

(e) Dependence on λ with $T = 0.5$

(f) Dependence on λ with $T = 1$

Figure 4.3: Convergence of the Carleman linearization with respect to the truncation level N measured by the error $\|\eta\|_{L^\infty(0,T;H)}$. Each plot indicates that the error behaves exponentially in N, whereas different sets of model parameters affect the error bounds in Theorem 4.2 and Corollary 4.1.



(a) DOFs as a function of N for $h = 2^{-8}$ **(b)** DOFs as a function of discret. levels for $N = 4$

Figure 4.4: Number of degrees of freedom (denoted DOFs) $\dim U_h(N)$ for different discretization methods for varying truncation levels N and refinement levels J (the corresponding maximum cell size of the mesh is given by $h = 2^{-J}$).

Dependence on f

As stated in Corollary 4.1, the forcing f has an immediate effect on the size of the solution y_e. We expect that larger f result in larger solutions y_e, which in turn leads to slower convergence of the linearization. Figure 4.3d shows the error as a function of N for $T = 0.5$, $\nu = 0.01$, $a = 1.05$, $b = 0.01$, $\lambda = 0$, and various values of c, which determines the size of f. We observe the expected behavior. It is noted that the size of f has only a mild effect on the convergence in the presented experiments.

Dependence on λ

Lastly, we examine the parameter λ. The established theoretical error bound includes a factor $\exp(\lambda T)$. This indicates that larger values of λ and T lead to poorer convergence. Moreover, the linearization is not guaranteed to converge for arbitrarily large T if $\lambda > 0$. Figures 4.3e and 4.3f show the error as a function of N for $\nu = 0.01$, $a = 1.05$, $b = 0.01$, $c = 0$, and various values of λ. The first plot shows the error for $T = 0.5$, and the second for $T = 1$. In both scenarios, exponential convergence is achieved, which worsens as λ is increased. While the linearization converges for all scenarios for $T = 0.5$, the linearization diverges in the case of $T = 1$ and $\lambda = 2$, confirming our error bounds.

4.7.4 Benefits of Non-Standard Discretization

Non-standard discretizations can offer advantages over traditional methods. In our model problem, the use of sparse grids improves the scaling of the required degrees of freedom. Figure 4.4a illustrates the spatial degrees of freedom $\dim U_h(N)$ for various values of N using the selected discretization. Meanwhile, Figure 4.4b shows the scaling (in some cases expected scaling) of

$\dim U_h(N)$ for different levels of discretization J, where the maximum cell size of the mesh is determined by $h = 2^{-J}$. These plots demonstrate the benefits of the non-standard discretization compared to the discretize-then-linearize approach, highlighting how this method helps alleviate the exponential increase in degrees of freedom.

4.8 Conclusion

In this chapter, we derived the well-posedness and the convergence of the truncated Carleman linearization for infinite-dimensional parabolic Cauchy problems under suitable assumptions. We achieved this in an undiscretized setting. This allowed us to separate the error arising from the truncated linearization from the discretization error stemming from the approximation of the infinite-dimensional equations. We thus justified the application of the linearization to PDE problems. We verified the theoretical findings with a series of numerical experiments, showing the expected convergence of the linearization. The theoretical findings motivate and support the application of non-standard discretization methods, which enable higher-order linearizations that were previously intractable. Numerical experiments showcase how such methods can reduce the number of degrees of freedom by orders of magnitude.

This chapter addresses the linearization of parabolic dynamical systems. Even though parabolic equations possess several favorable regularity properties and are generally better understood than hyperbolic equations, it is suggested that many of the concepts discussed here are transferable to second-order hyperbolic systems. Therefore, we propose that investigating the Carleman linearization of such systems is a promising direction for future research, and we briefly outline the necessary steps. Consider equations of the form

$$\begin{aligned} y''(t) + Ay(t) + B(y(t) \otimes y(t)) &= f(t) \quad \text{in } L^2(0,T;V'), \\ y(0) &= y_0, \\ y'(0) &= y_1. \end{aligned} \tag{4.11}$$

By assuming analogous properties for A, B, and the Hilbert spaces with $\alpha = 0$, Proposition 4.2, 4.3, and 4.4 remain valid. When combined with Theorem 9.4 in [92, p. 290], this ensures the existence of a unique solution to (4.11) for $f \in L^2(0,T;V')$, $y_0 \in H$, and $y_1 \in V'$. Furthermore, it guarantees the existence and uniqueness of a solution to the Carleman linearization if $f \in L^2(0,T;H)$. To demonstrate convergence, one could proceed as in the analysis of the residual equation (4.6). Completing the convergence proof requires an estimate for the hyperbolic case analogous to (4.3) in Lemma 4.1. However, these estimates must be derived from scratch, as it is essential to carefully track the constants and their explicit dependence on the coercivity and boundedness constants. Doing so would enable a rigorous theoretical analysis of hyperbolic nonlinear PDEs.

While this work introduces methods to mitigate the computational burden of the Carleman linearization, the practical performance and optimization of these methods remain open questions. A study on the computational aspects of the Carleman linearization could evaluate the effectiveness of sparse grid methods across various applications, explore their efficient implementation, and investigate the possibility of establishing theoretical bounds on the discretization error. Additionally, such a study could examine various structure-exploiting low-rank methods such as the tensor train method. The performance of the methods could be analyzed in various applications of the linearization, including model-order reduction and optimal control design.

Another potential branch of future research is concerned with the efficient solution of parabolic Cauchy problems, as well as associated optimal control problems. So-called parallel-in-time methods address the efficient solution of dynamical systems while enabling parallelization along the time axis. Among these methods, diagonalization-based approaches, such as those discussed in [56, 61, 13], have shown significant promise. These methods excel in handling (nearly) linear time-invariant problems. Recent studies have demonstrated the efficiency of diagonalization-based approaches from moderately sized problems to complex linear fluid flow problems. However, they encounter difficulties with nonlinear equations. Integrating the Carleman linearization with non-standard discretizations could potentially extend the applicability of diagonalization-based methods to nonlinear PDE problems, including the Navier–Stokes equations.

Lastly, a well-established method for solving large-scale nonlinear equations is Newton's method. For highly nonlinear equations, such as the Navier–Stokes equation with a high Reynolds number, Newton's method can struggle to converge unless a good initial guess is provided. Newton's method is based on a local second-order approximation of a nonlinear equation. The Carleman linearization could enhance Newton's method by generalizing it to a higher-order approach, thereby improving its convergence properties.

Chapter 5

A Framework for the Solution of Tree-Coupled Saddle-Point Systems

The preceding chapters are concerned with the numerical analysis of PDEs and PDE-constrained optimization problems. In this chapter, we remain in the scope of optimal control problems, however, we shift our focus to cases not necessarily governed by PDEs. Specifically, we address systems exhibiting a tree structure. While our assumptions are framed from a linear algebraic perspective, these problems are motivated by and stem from a range of optimal control problems. We demonstrate how the block and recursive structure of these problems can be leveraged to devise efficient algorithms that enable fast and parallelizable solutions. Additionally, we show how the proposed approaches can be integrated into methods capable of addressing highly nonlinear problems.

5.1 Introduction

As we have seen, saddle-point systems play an important role in numerical analysis, arising in various contexts such as optimization and coupled problems. Solving large-scale saddle-point systems typically demands specialized numerical methods that exploit the inherent structure of the problem for greater computational efficiency. Here, we focus on problems characterized by a set of saddle-point systems that are coupled through a tree-shaped topology, with an emphasis on their efficient numerical solution. We are particularly interested in problems with large numbers of saddle-point systems embedded within the overall system, where the coupling is limited to a relatively small subset of degrees of freedom, resulting in sparse coupling, also referred to as a tree-sparse system.

In practice, certain problems naturally exhibit such tree structure. This is seen, for example, in stochastic programming problems [114], where different scenarios are often largely, but not completely, independent. Tree structures also arise when a problem is defined on a physical domain with inherent arborescent connectivity. For instance, blood flow and pressure in systemic arteries can be modeled through fluid flow equations (see, e.g., [115, Chapter 5]), which results in tree-coupled saddle-point systems when the problem is decomposed in accordance with the arteries' topology. Gas transport network optimization [116] provides another example of physical

tree-shaped domains. While general network structures may exist in such cases, analyzing tree-coupled structures helps in tackling more complex network-structured systems. Additionally, certain domain decomposition methods—including overlapping and non-overlapping Schwarz methods [117, 118] for spatial decomposition, or parallel-in-time methods for optimal control such as multiple shooting [119]—can be phrased as tree-structured problems, as will be discussed later in this chapter.

As we have seen in previous chapters, the numerical solution of saddle-point systems is a well-established field of study. Saddle-point systems arise as subproblems in numerous optimization methods, such as *sequential quadratic programming* (SQP) [120], [121, Sec. 12.4], *interior point* [122, Ch. 19], and *sequential homotopy* [123] methods. They also appear in the context of PDEs, particularly following discretizations with *mixed finite element methods* [124], a prominent example being the discretized Stokes equations. Consequently, suitable preconditioning techniques have been thoroughly examined in the context of PDE discretizations (see [27, Ch. 4] for a summary). Common and effective approaches include block-diagonal [125, 126, 127] and block-triangular [128, 129] preconditioners, as seen in Section 3.2.3, which are designed to ensure convergence rates that remain robust with respect to parameters or problem sizes. Solution methods have also been developed for broader problem classes, not limited to PDEs. For instance, the class of *constraint preconditioners* [130, 131] can address a variety of general saddle-point systems. In nonlinear programming, additional efforts [132, 133] have focused on exploiting the specific block structure of these systems. More recently, the analysis of preconditioners has also been extended to both double [123, 134] and multiple [135] saddle-point systems. When it comes to the solution of saddle-point systems, we often follow recursive approaches to problem-solving. Such approaches for general linear systems have already been used [136, 137, 138] to great effect to derive recursively computed black-box preconditioners for general linear systems, which are more efficient than standard incomplete decompositions and more general than multigrid methods, for example. Although this overview is not exhaustive, it highlights the rich landscape of solution methods available for saddle-point systems.

While the above-mentioned methods can be effective, further computational gains may be achieved by exploiting more specific problem structures, such as tree-coupled systems—a subject that has already been explored in the literature. For example, a framework for interior-point methods named OOPS has been proposed [139], which addresses convex quadratic programs exhibiting nested structures. Furthermore, tree-sparse quadratic problems have been studied extensively in [140, 141]. Conventional domain decomposition methods such as overlapping Schwarz methods have been generalized to solve graph-based quadratic programs [142]. Interfaces to such problem-specific solvers are combined within a package called Plasmo.jl [143], allowing for the generic inclusion of network information at the modeling stage of nonlinear problems. Despite these advances, we have identified that a unified framework combining direct, recursive, structure-exploiting strategies with robust preconditioning techniques specifically for tree-coupled saddle-point systems remains an open challenge.

In this chapter, we provide a new mathematical framework for deriving and analyzing direct and preconditioned iterative methods for tree-coupled saddle-point systems. Specifically, we extend previous structure-exploiting approaches for saddle-point systems by incorporating a graph-based coupling structure, where interactions between individual and otherwise isolated subsystems are expressed via generic coupling constraints. We propose a range of novel solution algorithms. Aside from a parallelizable direct method, we demonstrate a range of structured preconditioners which may be embedded within suitable Krylov subspace methods, including block preconditioners, recursive preconditioners, and multi-level approaches. We prove a range of results relating to the convergence, complexity, and spectral properties of our algorithms.

The chapter is structured as follows. Section 5.2 formalizes the specific structure of the tree-coupled systems and imposes necessary assumptions on the system. In Section 5.3, an existing direct, recursive method from the literature is discussed within the scope of our framework. The key drawbacks of this method are addressed, and central concepts for the subsequent discussion, such as the Schur complement, are introduced. Section 5.4 presents preconditioners based on the recursive block structure of the problem that follow similar recursive patterns as the direct method. In Section 5.5, we explore a non-recursive approach to preconditioning and consider multi-level strategies, among others. Section 5.6 details numerical experiments that validate our theoretical results and demonstrate the efficacy and versatility of our mathematical approach. The implementations are carried out in a purely sequential fashion; while these methods are designed to be amenable to parallelization, achieving this would require a bespoke implementation, which exceeds the scope of this work. Lastly, we conclude and outline avenues for future research in Section 5.7.

5.2 Problem Formulation

Let $\mathbb{D} = (\mathbb{V}, \mathbb{A})$ be a directed tree (an arborescence), with N vertices $\mathbb{V} \coloneqq \{1, \dots, N\}$ and M arcs $\mathbb{A} \coloneqq \{a_1, \dots, a_M\} \subseteq \mathbb{V} \times \mathbb{V}$ directed away from a root $R \in \mathbb{V}$. Each vertex has associated variables $x_i \in \mathbb{R}^{n_i}$, for $n_i \in \mathbb{N}$, which are coupled along the arcs in $\mathbb{A}$. For each arc $a_k = (i, j)$, two matrices $C_k^+ \in \mathbb{R}^{l_k \times n_i}$ and $C_k^- \in \mathbb{R}^{l_k \times n_j}$ describe the coupling between variables x_i and x_j. Specifically, if we let $\delta^+(i)$ and $\delta^-(i)$ denote the outgoing and incoming arcs of $i \in \mathbb{V}$, respectively, the saddle-point system we shall investigate is defined by

$$\begin{aligned} B_i x_i + \sum_{a_k \in \delta^+(i)} (C_k^+)^T y_k - \sum_{a_k \in \delta^-(i)} (C_k^-)^T y_k = h_i \quad \text{for all } i \in \mathbb{V}, \\ C_k^+ x_i - C_k^- x_j - D_k y_k = f_k \quad \text{for all } a_k = (i, j) \in \mathbb{A}, \end{aligned} \tag{5.1}$$

where each $B_i = B_i^T \in \mathbb{R}^{n_i \times n_i}$ forms a saddle-point system and corresponds to conditions on x_i with a right-hand side $h_i \in \mathbb{R}^{n_i}$, and $y_k \in \mathbb{R}^{l_k}$ are coupling variables with conditions given by matrices $D_k = D_k^T \in \mathbb{R}^{l_k \times l_k}$ and right-hand sides $f_k \in \mathbb{R}^{l_k}$ (see Figure 5.1 for an example). This system is symmetric and overall forms a larger saddle-point system

$$\begin{pmatrix} \mathcal{B} & \mathcal{C}^T \\ \mathcal{C} & -\mathcal{D} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} h \\ f \end{pmatrix} \tag{5.2}$$

by setting

$$\begin{aligned} \mathcal{B} &\coloneqq \operatorname{diag}(B_1, \dots, B_N), \\ \mathcal{D} &\coloneqq \operatorname{diag}(D_1, \dots, D_M), \text{ and} \\ \mathcal{C} = (\mathcal{C}_{k,i}) &\coloneqq \begin{cases} C_k^+ & \text{if } a_k = (i,j), \\ -C_k^- & \text{if } a_k = (j,i), \\ 0 & \text{otherwise.} \end{cases} \quad \text{for } i \in \mathbb{V} \text{ and } k \in \{1, \dots, M\}, \end{aligned}$$

where the notation $\operatorname{diag}(B_1, \dots, B_N)$ denotes a matrix consisting of N blocks of rows and columns with diagonal blocks set to the matrices B_i and off-diagonal blocks set to zero matrices of appropriate dimensions (and similarly for other uses of the 'diag' notation). Systems of the form (5.1) arise from a broader class of problems. Specifically, consider a nonlinear optimization problem with a separable objective function $\sum_{i \in \mathbb{V}} \phi_i(\zeta_i)$, and constraints of the form

$$\begin{aligned} c_i(\zeta_i) = 0 \quad \forall i \in \mathbb{V} \qquad & \mid \cdot \nu_i \\ C_k^+ \zeta_i - C_k^- \zeta_j = 0 \quad \forall a_k = (i,j) \in \mathbb{A} \qquad & \mid \cdot y_k \end{aligned}$$

composed of (possibly nonlinear) internal constraints as well as linear coupling constraints on the graph $\mathbb{D}$, with corresponding Lagrange multipliers ν_i and y_k. If this problem has a quadratic objective $\phi_i(\zeta_i) = g_i^T \zeta_i + \frac{1}{2} \zeta_i^T H_i \zeta_i$ and linear constraints $c_i(\zeta_i) \coloneqq A_i \zeta_i - b_i \ (\stackrel{!}{=} 0)$, its optimality system, also known as the Karush–Kuhn–Tucker system, is of the form (5.1), where

$$B_i = \begin{pmatrix} H_i & A_i^T \\ A_i & 0 \end{pmatrix}, \quad x_i = \begin{pmatrix} \zeta_i \\ \nu_i \end{pmatrix}, \quad h_i = \begin{pmatrix} -g_i \\ b_i \end{pmatrix}, \quad D_k = 0, \quad f_k = 0.$$

For general nonlinear ϕ_i and c_i, we can employ an interior point method [122, Chapter 19], generating a sequence $(\zeta_i^{(l)}, \nu_i^{(l)}, y_k^{(l)})_l$ of primal–dual solutions based on a given starting point. At each iteration, a system of the form (5.1) is solved, where the internal systems have a matrix H_i corresponding to the Hessian of the Lagrangian $\phi_i + \nu_i^T c_i$ plus a barrier term, and A_i to the Jacobian of c_i evaluated at the current primal–dual iterate, with D_k diagonal. An alternative to an interior point method is the sequential homotopy method [144, 123], which also uses linear systems as an algorithmic backbone. The linear systems to be solved are again of the form (5.1)

with blocks given by

$$B_i = \begin{pmatrix} H_i + \lambda I & A_i^T \\ A_i & -\delta I \end{pmatrix} \quad \text{and} \quad D_k = \delta I,$$

where I denotes the identity matrix of appropriate dimension and $\lambda, \delta > 0$ are algorithmic parameters.

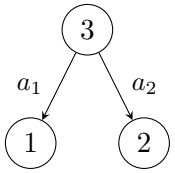
Lastly, note that we make no specific assumptions regarding the coupling matrices C_k^+ and C_k^-, as our algorithmic framework does not require any such assumption. In terms of modeling, a common choice for these matrices stems from the enforcement of *consensus constraints* or *matching constraints*, i.e., requiring certain entries of the variables x_i and x_j to coincide. For this particular case of coupling, C_k^+ and C_k^- then consist of rows of positive or negative unit vectors.

5.2.1 Notation and Definitions

A vertex $i \in \mathbb{V}$ is said to be a *leaf* of the tree $\mathbb{D}$ iff $\delta^+(i) = \emptyset$ and an *inner vertex* otherwise. The *inner subgraph*, denoted by $\mathbb{D}^\circ = (\mathbb{V}^\circ, \mathbb{A}^\circ)$ is the subgraph induced by the set $\mathbb{V}^\circ$ of inner vertices. We generally assume that the sets $\delta^+(i)$ and $\delta^-(i)$ are ordered consistently and let $\delta(i) \coloneqq \delta^+(i) \cup \delta^-(i)$. For each arc $a_k = (i, j) \in \mathbb{A}$ we set $\text{head}(a_k) \coloneqq j$ and $\text{tail}(a_k) \coloneqq i$. The *parent* of a vertex $i \in \mathbb{V}$, $i \neq R$, is the vertex $k \in \mathbb{V}$ such that $(k, i) \in \mathbb{A}$ and $k^-(i)$ denotes the index of the arc entering i, i.e., $k^-(i) \in \{1, \dots, M\}$ is such that $\delta^-(i) = \{a_{k^-(i)}\}$ and $a_{k^-(i)} = (k, i)$.

For each vertex $i \in \mathbb{V}$ the *children* of i are the head vertices of the arcs in $\delta^+(i)$. The *depth* of i, which we denote as $\text{depth}(i)$, is defined as the length of the (unique) (R, i)-path in $\mathbb{D}$. Similarly, the *height* of i, denoted $\text{height}(i)$, is defined to be zero if i is a leaf, and the maximum height of any child vertex in $\delta^+(i)$ plus one otherwise. The height of $\mathbb{D}$, $\text{height}(\mathbb{D})$, is defined as the height of R or, equivalently, as the maximum depth of any vertex in $\mathbb{V}$. The subtree rooted at vertex i is denoted by $\mathbb{D}_{\leq i} = (\mathbb{V}_{\leq i}, \mathbb{A}_{\leq i})$ and given by the union of the vertices and arcs on all (i, j)-paths in $\mathbb{D}$. We also let $l_i^+ \coloneqq \sum_{a_k \in \delta^+(i)} l_k$, $l_i^- \coloneqq \sum_{a_k \in \delta^-(i)} l_k$, and $l_i \coloneqq l_i^+ + l_i^-$ be the outgoing, incoming, and total number of variables coupled to $i \in \mathbb{V}$ respectively. An example for these definitions is given in Figure 5.1.

We also use lower case letters to denote vectors, upper case ones for provided matrices, and curly upper case letters for larger block matrices. Lastly, we present results regarding complexity in the usual O-notation [145, Chapter 1], where for functions $f, g : \mathbb{N} \to \mathbb{N}$ we say that $f \in O(g)$ if $\limsup_{n \to \infty} f(n)/g(n) < \infty$ and $f \in \Theta(g)$ if $f \in O(g)$ and $g \in O(f)$.



$$\begin{pmatrix} B_1 & & & -(C_1^-)^T & \\ & B_2 & & & -(C_2^-)^T \\ & & B_3 & (C_1^+)^T & (C_2^+)^T \\ -C_1^- & & C_1^+ & -D_1 & \\ & -C_2^- & C_2^+ & & -D_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ f_1 \\ f_2 \end{pmatrix}$$

Figure 5.1: Example of a tree-coupled system, based on a tree with $N = 3$ vertices $\mathbb{V} = \{1, 2, 3\}$ and $M = 2$ arcs $\mathbb{A} = \{a_1 = (3, 1), a_2 = (3, 2)\}$. Vertices 1 and 2 are leaves each having a height of zero, a depth of one, and 3 as their parent. Vertex $R = 3$ is an inner vertex with a height of one (equal to the height of $\mathbb{D}$), a depth of zero, and 1 and 2 as its children. The inner subgraph $\mathbb{D}^\circ$ consists of vertex 3 without containing any arcs. The arcs entering 1 and 2 are a_1 and a_2 respectively, i.e., it holds that $k^-(1) = 1$ and $k^-(2) = 2$. Both arcs have 3 as their tail while $\text{head}(a_1) = 1$ and $\text{head}(a_2) = 2$. The subtree rooted at 3 is equal to the graph itself, whereas $\mathbb{D}_{\leq 1}$ and $\mathbb{D}_{\leq 2}$ consist of only the vertices 1 and 2 respectively without any arcs.

5.2.2 Assumptions

Besides the symmetry of the matrices B_i and D_k, we make the following additional assumption in order to ensure the non-singularity of (5.2):

Assumption 5.1.

1. *The matrix $\mathcal{B}$ is invertible.*
2. *The Schur complement of* (5.2)*, given by*

$$\mathcal{S} \coloneqq \mathcal{C}\mathcal{B}^{-1}\mathcal{C}^T + \mathcal{D}, \tag{5.3}$$

is positive definite.

Lemma 5.1. *Under Assumption 5.1, system* (5.2) *is invertible.*

Proof. This follows from the fact that (5.2) can be factorized into a product of $\text{diag}(\mathcal{B}, -\mathcal{S})$ and two other invertible matrices [63, Eq. (3.1)]. □

Regarding Assumption 5.1 it is apparent from Lemma 5.1 that non-singularity of $\mathcal{S}$ is sufficient to ensure that system (5.2) is invertible. We do, however, rely on positive definiteness of $\mathcal{S}$, in particular in Section 5.5. While we have verified that this stronger assumption is satisfied for a number of problems, our numerical experiments indicate that our methods work well even if $\mathcal{S}$ is merely invertible.

5.3 Direct Method

In order to solve the system (5.1) we make use of a Schur complement approach rather than a complete sparse decomposition, which has two advantages: First, a decomposition may be unnecessary in particularly if only a few variables are coupled, i.e., $l_k \ll \min(n_i, n_j)$. In this case the systems involving the matrices B_i are largely independent, the corresponding Schur complements are small in size, and a substantial portion of the computations may be carried out in parallel in order to improve performance and scale to larger systems. Second, our approach is highly flexible in how systems involving the matrices B_i are solved. Thus, any structure-exploiting solution methods for solving these systems can be easily incorporated into our computational framework.

5.3.1 Structure of Algorithm

We begin by giving a direct method (see Algorithm 5.1) inspired by the exploitation of a 'symmetric bordered block-diagonal structure' introduced in [133]. Using a symmetric permutation of the blocks constituting system (5.2), we obtain a nested sequence of systems with this exploitable structure for each $i \in \mathbb{V}$. The system associated with i then depends recursively on all children of i, thereby corresponding to the submatrix of (5.2) associated with the subtree of $\mathbb{D}$ rooted at i.

If i is a leaf, we let $\mathcal{B}_{\leq i} \coloneqq B_i$, $\mathcal{C}_i^- \coloneqq C_{k^-(i)}^-$, and $h_{\leq i} \coloneqq h_i$. Otherwise, we let $\delta^+(i) = (a_{k_1} = (i, j_1), \ldots, a_{k_{r_i}} = (i, j_{r_i}))$ be the outgoing arcs of vertex i and set

$$
\begin{aligned}
x_{\leq i}^T &\coloneqq (x_{\leq j_1}^T, \ldots, x_{\leq j_{r_i}}^T, x_i^T, y_{k_1}^T, \ldots, y_{k_{r_i}}^T),\\
h_{\leq i}^T &\coloneqq (h_{\leq j_1}^T, \ldots, h_{\leq j_{r_i}}^T, h_i^T, f_{k_1}^T, \ldots, f_{k_{r_i}}^T),\\
\mathcal{D}_{\leq i} &\coloneqq \operatorname{diag}\left(D_{k_1}, \ldots, D_{k_{r_i}}\right),\\
\mathcal{C}_{\leq i}^+ &\coloneqq \left(\left(C_{k_1}^+\right)^T, \ldots, \left(C_{k_{r_i}}^+\right)^T\right)^T,\\
\mathcal{C}_i^- &\coloneqq \left(0, \ldots, 0, C_{k^-(i)}^-, 0\right) \text{ for } i \neq R, \text{ and}\\
\mathcal{C}_{\leq i}^- &\coloneqq \operatorname{diag}\left(\mathcal{C}_{j_1}^-, \ldots, \mathcal{C}_{j_{r_i}}^-\right),
\end{aligned}
$$

where $a_{k^-(i)}$ is the unique arc in $\delta^-(i)$ as defined above and the zero matrices in the definition of $\mathcal{C}_i^-$ are of appropriate dimensions to align the columns of $C_{k^-(i)}^-$ with those of B_i in the following definition of

$$\mathcal{B}_{\leq i} \coloneqq \begin{pmatrix} \mathcal{B}_{\leq j_1} & & & & \\ & \ddots & & & -\left(\mathcal{C}_{\leq i}^-\right)^T \\ & & \mathcal{B}_{\leq j_{r_i}} & & \\ & & & B_i & \left(\mathcal{C}_{\leq i}^+\right)^T \\ & -\mathcal{C}_{\leq i}^- & & \mathcal{C}_{\leq i}^+ & -\mathcal{D}_{\leq i} \end{pmatrix}.$$

To obtain the solution associated with the subtree rooted at i, we solve the system $\mathcal{B}_{\leq i} x_{\leq i} = h_{\leq i}$. We call the structure of $\mathcal{B}_{\leq i}$ (lower-right pointing) *arrowhead* structure rather than bordered-block diagonal as in [133]. Since the diagonal blocks $\mathcal{B}_{\leq j_l}$ have arrowhead structure themselves, $\mathcal{B}_{\leq i}$ has a *nested (lower-right pointing) arrowhead* structure in general. Since $\mathcal{B}_{\leq R}$ is a symmetric permutation of (5.2), it is invertible under Assumption 5.1. Our approach necessitates stronger assumptions, however. Specifically, we want to be able to solve the nested arrowhead matrices $\mathcal{B}_{\leq i}$ using a recursive approach based on Schur complements. To this end, we need the following additional assumption:

Assumption 5.2.

1. *The matrices $\mathcal{B}_{\leq i}$ are invertible.*
2. *The Schur complements*

$$\mathcal{S}_{\leq i} \coloneqq \begin{pmatrix} -\mathcal{C}_{\leq i}^- & \mathcal{C}_{\leq i}^+ \end{pmatrix} \begin{pmatrix} \mathcal{B}_{\leq j_1}^{-1} & & & \\ & \ddots & & \\ & & \mathcal{B}_{\leq j_{r_i}}^{-1} & \\ & & & B_i^{-1} \end{pmatrix} \begin{pmatrix} -\left(\mathcal{C}_{\leq i}^-\right)^T \\ \left(\mathcal{C}_{\leq i}^+\right)^T \end{pmatrix} + \mathcal{D}_{\leq i}$$

   are positive definite for all inner vertices $i \in \mathbb{V}$.

Based on Assumption 5.2, we propose the direct method laid out in Algorithm 5.1, solving a system with matrix $\mathcal{B}_{\leq i}$ using the Schur complement $\mathcal{S}_{\leq i}$ and recursing into the subtree rooted at i. The algorithm consists of three steps: First, the right-hand sides $\hat{x}_l$ for the Schur complement are computed recursively. Then the system $\mathcal{S}_{\leq i} y = \hat{x}$ is solved, yielding the portion of the solution associated with the coupling variables y_{k_l}. Finally, the remaining parts of the solution, x_i and $x_{\leq j_l}$, are recursively computed based on the variables y_{k_l}. The first and third steps involve a recursion into the subtrees rooted at the head vertices of the arcs in $\delta^+(i)$. To solve (5.2) in its entirety, we employ the algorithm at the root R of $\mathbb{D}$.

Algorithm 5.1: Direct method to solve system involving $\mathcal{B}_{\leq i}$

Function SolveDirectSchur(i, $(\mathcal{B}_{\leq j})_{j\in\mathbb{V}_{\leq i}}$, $(\mathcal{S}_{\leq j})_{j\in\mathbb{V}_{\leq i}}$, $h_{\leq i}$)**:**

Input : Vertex $i \in \mathbb{V}$
Subtree systems $(\mathcal{B}_{\leq j})_{j\in\mathbb{V}_{\leq i}}$
Subtree Schur complements of $\mathcal{S}_{\leq j}$ for $j \in \mathbb{V}_{\leq i}$
Right-hand side $h_{\leq i}$

Output: Solution $x_{\leq i} = \mathcal{B}_{\leq i}^{-1} h_{\leq i}$

if *Vertex i is a leaf in $\mathbb{D}$* **then** ▷ *Base case: $x_{\leq i} = x_i$*

  return $x_i = B_i^{-1} h_i$

foreach $l \in \{1, \ldots, r_i\}$ **do** ▷ *Compute right-hand sides*

  $\hat{y}_l \leftarrow$ SolveDirectSchur(j_l, $(\mathcal{B}_{\leq k})_{k\in\mathbb{V}_{\leq j_l}}$, $(\mathcal{S}_{\leq k})_{k\in\mathbb{V}_{\leq j_l}}$, $h_{\leq j_l}$)

  $\hat{x}_l \leftarrow -\mathcal{C}_{j_l}^{-} \hat{y}_l + C_{k_l}^{+} B_i^{-1} h_i - f_{k_l}$

Solve Schur complement system: $(y_{k_1}, \ldots, y_{k_{r_i}}) \leftarrow \mathcal{S}_{\leq i}^{-1} (\hat{x}_1, \ldots, \hat{x}_{r_i})$

Compute solution: $x_i \leftarrow B_i^{-1} \left(h_i - \sum_{l=1}^{r_i} (C_{k_l}^{+})^T y_{k_l} \right)$

foreach $l \in \{1, \ldots, r_i\}$ **do** ▷ *Compute solutions $x_{\leq j_l}$*

  $\hat{z}_l \leftarrow h_{\leq j_l} + (\mathcal{C}_{j_l}^{-})^T y_{k_l}$

  $x_{\leq j_l} \leftarrow$ SolveDirectSchur(j_l, $(\mathcal{B}_{\leq k})_{k\in\mathbb{V}_{\leq j_l}}$, $(\mathcal{S}_{\leq k})_{k\in\mathbb{V}_{\leq j_l}}$, $\hat{z}_l$)

return $x_{\leq i}^T = \left(x_{\leq j_1}^T, \ldots, x_{\leq j_{r_i}}^T, x_i^T, y_{k_1}^T, \ldots, y_{k_{r_i}}^T \right)$

To apply Algorithm 5.1, we need a method to compute the Schur complements $\mathcal{S}_{\leq i}$. Simple calculations show that $\mathcal{S}_{\leq i}$ is given by blocks

$$(\mathcal{S}_{\leq i})_{l,l'} = \begin{cases} C_{k_l}^{+} B_i^{-1} (C_{k_l}^{+})^T + \mathcal{C}_{j_l}^{-} \mathcal{B}_{\leq j_l}^{-1} (\mathcal{C}_{j_l}^{-})^T + D_{k_l} & \text{if } l = l', \text{ and} \\ C_{k_l}^{+} B_i^{-1} (C_{k_{l'}}^{+})^T & \text{otherwise,} \end{cases}$$

for $l, l' \in \{1, \ldots, r_i\}$. The occurrence of the inverses of the matrices $\mathcal{B}_{\leq j_l}^{-1}$ suggests that the Schur complements can be computed from the leaves of the tree up towards its root.

At a leaf $i \in \mathbb{V}$, no Schur complement is required and Algorithm 5.1 only solves a system with matrix B_i. To explain the computation higher up in the tree, we put ourselves in the position where at vertex $i \in \mathbb{V}$, $\mathcal{S}_{\leq i}$ has already been computed. At this stage, all that remains to be done at vertex i is the computation of the product $\mathcal{C}_i^{-} \mathcal{B}_{\leq i}^{-1} (\mathcal{C}_i^{-})^T$, required for some diagonal block of the Schur complement of the parent vertex of i (assuming of course that $i \neq R$, in which case all Schur complements are by now computed). To compute this product, we could simply use Algorithm 5.1 repeatedly to solve systems with the matrix $\mathcal{B}_{\leq i}$ and right-hand sides defined using the rows of $\mathcal{C}_i^{-}$. However, the structure of the product makes the computations significantly easier:

First, the only non-zero block $\mathcal{C}_i^{-}$ is at the position of B_i with respect to $\mathcal{B}_{\leq i}$. Thus, the right-hand sides $h_{\leq i}$ for the computation have the property that $h_{\leq j_l} = 0$ and $f_{k_l} = 0$ for all $l \in \{1, \ldots, r_i\}$. Second, since we only require the product of the solution with $\mathcal{C}_i^{-}$, we only have to compute the solution value of x_i and can skip the computation of $x_{\leq j_l}$ and y_{k_l} for all $l \in \{1, \ldots, r_i\}$. As a

Algorithm 5.2: Algorithm to compute Schur complement $\mathcal{S}_{\leq i}$

Function ComputeArrowheadSchur(i)**:**

Input : Vertex $i \in \mathbb{V}$

Output: Schur complements $(\mathcal{S}_{\leq j})_{j \in \mathbb{V}_{\leq i}}$

if *Vertex i is a leaf in $\mathbb{D}$* **then**

  return $\emptyset$

Compute subtree Schur complements:

$$(\mathcal{S}_{\leq k})_{k \in \mathbb{V}_{\leq j}} \leftarrow \text{ComputeArrowheadSchur}(j)$$
$$\forall\, j \in V_i = \{j \in \mathbb{V}_{\leq i} \mid j \neq i\}$$

$(\mathcal{S}_{\leq i})_{l,l'} \leftarrow 0 \quad \forall\, l, l' \in \{1, \ldots, r_i\}$

forall $l \in \{1, \ldots, r_i\}$ **do**

  $Z_l \leftarrow B_i^{-1}(C_{k_l}^+)^T$

forall $l \in \{1, \ldots, r_i\}$ **do**

  forall $l' \in \{1, \ldots, r_i\}$ **do**

    $(\mathcal{S}_{\leq i})_{l,l'} \leftarrow (\mathcal{S}_{\leq i})_{l,l'} + C_{k_l}^+ Z_{l'}$

  $S_l \leftarrow \text{OutgoingSchurBlock}(j_l, \mathcal{B}_{\leq j_l}, \mathcal{S}_{\leq j_l})$

  $(\mathcal{S}_{\leq i})_{l,l} \leftarrow (\mathcal{S}_{\leq i})_{l,l} + S_l + D_{k_l}$

return $(\mathcal{S}_{\leq j})_{j \in \mathbb{V}_{\leq i}}$

consequence, no recursion is required at all in order to determine the required product, greatly accelerating the computations. The specific computations to be carried out are summarized in Algorithms 5.2 and 5.3 with an example of the application of the algorithms being shown in Appendix B.1.

5.3.2 Complexity

In the following, we examine the running time of both Algorithms 5.1 and 5.2 used to solve (5.1) and determine the required Schur complements $\mathcal{S}_{\leq i}$, respectively.

In general, the complexity of the algorithms depends on how the systems with matrices B_i and $\mathcal{S}_{\leq i}$ are solved, which we have not specified so far. Since we make no assumptions regarding the matrices B_i other than non-singularity, any appropriate method may be used in practice. On the other hand, since the Schur complements $\mathcal{S}_{\leq i}$ are assumed to be positive definite, Cholesky decompositions or the Conjugate Gradient method [146] are likely to be used.

To perform an analysis of the complexity independently of these details we make two assumptions: First, since l_k is small compared to the sizes n_i of the matrices B_i, we assume that the computational costs are dominated by the solutions of systems $B_i x_i = y_i$ with different right-hand sides y_i, ignoring other parts such as matrix–vector products with C_k^- and C_k^+ and the solution of systems involving $\mathcal{S}_{\leq i}$. Second, since the methods used to solve the systems B_i are arbitrary, we simply assume that the solution of a single system involving B_i can be carried out in constant time and count the total number of such solves, finding the following:

Algorithm 5.3: Algorithm to compute outgoing Schur complement block term $\mathcal{C}_i^- \mathcal{B}_{\leq i}^{-1} (\mathcal{C}_i^-)^T$

Function OutgoingSchurBlock(i, $\mathcal{B}_{\leq i}$, $\mathcal{S}_{\leq i}$)**:**

Input : Vertex $i \in \mathbb{V}$
System $\mathcal{B}_{\leq i}$
Schur complement $\mathcal{S}_{\leq i}$
Output: Schur block $\mathcal{C}_i^- \mathcal{B}_{\leq i}^{-1} (\mathcal{C}_i^-)^T$

1 **if** *Vertex i is a leaf in $\mathbb{D}$* **then**
2 **return** $C^-_{k^-(i)} B_i^{-1} (C^-_{k^-(i)})^T$
3 $(h_1, \ldots, h_{l_{k^-(i)}}) \leftarrow$ Columns of matrix $(C^-_{k^-(i)})^T$
4 **forall** $l \in \{1, \ldots, l_{k^-(i)}\}$ **do**
5 Solve system to obtain $z_l \leftarrow B_i^{-1} h_l$
6 Compute right-hand sides for each $l' \in \{1, \ldots, r_i\}$: $\hat{x}^l_{l'} \leftarrow C^+_{k_{l'}} z_l$
7 Solve Schur complement system: $(y^l_{k_1}, \ldots, y^l_{k_{r_i}}) \leftarrow \mathcal{S}_{\leq i}^{-1} (\hat{x}^l_1, \ldots, \hat{x}^l_{r_i})$
8 Compute solution: $x_i^l \leftarrow B_i^{-1} \left(h_l - \sum_{l'=1}^{r_i} (C^+_{k_{l'}})^T y^l_{k_{l'}} \right)$
9 $Z_i \leftarrow$ Matrix consisting of columns x_i^l for $l \in \{1, \ldots, l_{k^-(i)}\}$
10 **return** $C^-_{k^-(i)} Z_i$

Lemma 5.2. *For each vertex $i \in \mathbb{V}$ the following holds:*

1. *The solution of system* (5.1) *using Algorithm 5.1 with precomputed Schur complements requires $\Theta(2^{\text{depth}(i)})$ solutions of systems involving B_i.*
2. *The computation of the Schur complements using Algorithm 5.2 requires $O(l_i)$ solutions of systems involving B_i.*

Proof. 1. Let us examine a vertex $i \in \mathbb{V}$ at a depth of $l \geq 0$. Let $R = i_1, i_2, \ldots, i_{l+1} = i$ be the vertices on the (R, i)-path in $\mathbb{D}$. During the execution of Algorithm 5.1 at the parent vertex i_l, the algorithm recurses into vertex i_{l+1} exactly twice: First during the computation of the right-hand sides $\hat{x}_l$ in line 4, and then in line 10 to assemble the solution $x_{\leq i_l}$ based on the solution of the Schur complement. Thus, each time Algorithm 5.1 is executed at vertex i_l, it recurses into vertex i twice. The same holds for i_l and its parent vertex i_{l-1} and so on up to the root R where the algorithm is executed once. In total, Algorithm 5.1 is executed 2^l times for vertex i. Each time the algorithm is executed at vertex i, a system involving B_i is solved once at line 2 if i is a leaf and twice at lines 5 and 7 respectively if i is an inner vertex, yielding a total of $\Theta(2^l)$ solves.

2. The result follows from the structure of Algorithm 5.2: The computation of the Schur complement $\mathcal{S}_{\leq i}$ does not require a recursion beyond the child vertices of i itself. Consequently, both functions in the algorithm are executed exactly once for each vertex i. At vertex i a system involving B_i is solved once for each of the rows of $C^+_{k_l}$ for all $l \in \{1, \ldots, r_i\}$ at line 6 and twice for each row in $C^-_{k^-(i)}$ at lines 5 and 8 of Algorithm 5.3 respectively, yielding a total number of $O(l_i)$ solves. □

Based on these bounds it is clear that the total number solved systems involving B_i summed over all vertices during an application of Algorithm 5.1 at R is in $\Theta(2^{\text{height}(\mathbb{D})})$ where systems at the leaves of $\mathbb{D}$ are solved most often. This running time is exponential rather than polynomial for general graphs $\mathbb{D}$, for which $\text{height}(\mathbb{D}) \in \Theta(N)$. Specifically, a worst-case instance for the algorithm consists of a tree composed of a single path having a height of $N-1$. Conversely, if the height is sufficiently small compared to the input size of the problem, the complexity remains polynomially bounded. This holds in particular for full proper trees, where height grows logarithmically in the number of vertices. In contrast to the solution of the system based on Algorithm 5.1, the computation of the Schur blocks using Algorithm 5.2 is polynomial regardless of the height of $\mathbb{D}$, which is due to the fact that the recursion is much more limited during the computation of the Schur complement blocks.

Lastly, we would like to put these results in the context of those obtained in [133]: Firstly, the authors consider the 'symmetric bordered block-diagonal structure' as one of several exploitable structures, another one being 'rank-k correcting matrices'. They examine these structures from a software design framework and show how they can be seen as a set of data structures with common functionality which can be mapped onto a class hierarchy in object-oriented software. Consequently, since these data structures are expected to behave like interchangeable black-box components, the corresponding analysis of their interplay is limited. For instance, the complexity laid out here is not examined in [133] and the computation of the required Schur complements according to their framework would also incur an exponential running time, as opposed to the polynomial running time of Algorithm 5.2.

5.4 Nested Arrowhead Structure

In this section, we continue the investigation of the nested arrowhead structure introduced in Section 5.3. While this section was based on a thorough examination of the method introduced in [133], we now proceed to introduce several entirely new approaches based on problem-specific preconditioners, which we define recursively from the leaves of the graph $\mathbb{D}$ towards its root.

5.4.1 A Block-Diagonal Preconditioner

First, we let $\mathcal{P}_{\leq i}$ be the preconditioner associated with the subtree of $\mathbb{D}$ rooted at $i \in \mathbb{V}$, where we denote by $\mathcal{P}$ the preconditioner $\mathcal{P}_{\leq R}$ for the entire tree. If a vertex $i \in \mathbb{V}$ is a leaf, we always let $\mathcal{P}_{\leq i} \coloneqq B_i$. If i is a inner vertex we let $\delta^+(i) = (a_{k_1} = (i, j_1), \ldots, a_{k_{r_i}} = (i, j_{r_i}))$ be the outgoing arcs of i and define $\mathcal{P}_{\leq i}$ recursively based on the preconditioners $\mathcal{P}_{\leq j_l}$. Our first preconditioner is called the *block-diagonal preconditioner* defined by

$$\mathcal{P}_{\leq i}^{\text{diag}} \coloneqq \text{diag}\left(\mathcal{P}_{\leq j_1}^{\text{diag}}, \ldots, \mathcal{P}_{\leq j_{r_i}}^{\text{diag}}, B_i, -\mathcal{D}_{\leq i}\right). \tag{5.4}$$

This preconditioner is fairly simple in its structure, approximating the matrices B_i while disregarding the coupling matrices $\mathcal{C}^-_{\leq i}$ and $\mathcal{C}^+_{\leq i}$ completely. This preconditioner can, however, only be applied if all matrices D_k are invertible, which holds upon application of the sequential homotopy or interior point methods but not for KKT systems arising from quadratic programs of an SQP method. In some problems, the use of the Schur complement in place of $\mathcal{D}_{\leq i}$ within the preconditioner is recommended. However, we observed no performance improvement when doing so. Since the Schur complement is more difficult to compute we chose to exclude this variant from further consideration.

5.4.2 The Hook Preconditioner

A class of preconditioners can be derived based on the Schur complements $\mathcal{S}_{\leq i}$ introduced above. Recall that as opposed to the direct method in Algorithm 5.1, these preconditioners can be efficiently computed using Algorithm 5.2. We, of course, again work under Assumption 5.2 to ensure the Schur complements exist and are positive definite. The *hook preconditioner* $\mathcal{P}^{\text{hook}}$ is defined as

$$\mathcal{P}^{\text{hook}}_{\leq i} := \begin{pmatrix} \mathcal{P}^{\text{hook}}_{\leq j_1} & & & & \\ & \ddots & & & \\ & & \mathcal{P}^{\text{hook}}_{\leq j_{r_i}} & & \\ & & & B_i & \\ -\mathcal{C}^-_{\leq i} & & & \mathcal{C}^+_{\leq i} & -\mathcal{S}_{\leq i} \end{pmatrix}. \tag{5.5}$$

The matrix $\mathcal{P}^{\text{hook}}_{\leq i}$ is lower block-triangular and belongs to the class of saddle-point preconditioners considered in [125, 128]. For any inner vertex i the application of the inverse of $\mathcal{P}^{\text{hook}}_{\leq i}$ to a given right-hand side $h^T_{\leq i} = (h^T_{\leq j_1}, \ldots, h^T_{\leq j_{r_i}}, h^T_i, f^T_{k_1}, \ldots, f^T_{k_{r_i}})$ is carried out by conducting the following steps:

1. Recursively compute products $x_{\leq j_l} \leftarrow (\mathcal{P}^{\text{hook}}_{\leq j_l})^{-1} h_{\leq j_l}$ for $l = 1, \ldots, r_i$ and $x_i \leftarrow B_i^{-1} h_i$.
2. Compute right-hand side

$$\hat{z} \leftarrow -\mathcal{C}^-_{\leq i} \begin{pmatrix} x^T_{\leq j_1} & \ldots & x^T_{\leq j_{r_i}} \end{pmatrix}^T + \mathcal{C}^+_{\leq i} x_i - \begin{pmatrix} f^T_{k_1} & \ldots & f^T_{k_{r_i}} \end{pmatrix}^T.$$

3. Compute solution

$$\begin{pmatrix} y^T_{k_1} & \ldots & y^T_{k_{r_i}} \end{pmatrix}^T \leftarrow \mathcal{S}^{-1}_{\leq i} \hat{z}.$$

Let us now turn to the theoretical properties of $\mathcal{P}^{\text{hook}}_{\leq i}$. To begin, we observe that if i is a leaf of $\mathbb{D}$, then $\mathcal{P}^{\text{hook}}_{\leq i} = B_i$ is a perfect preconditioner, which yields the correct solution after a single preconditioned step. If i has a height of one rather than being a leaf, we see that

$$\mathcal{P}^{\text{hook}}_{\leq i} = \begin{pmatrix} B_{j_1} (= \mathcal{B}_{\leq j_1}) & & & & \\ & \ddots & & & \\ & & B_{j_{r_i}} (= \mathcal{B}_{\leq j_{r_i}}) & & \\ & & & B_i & \\ -\mathcal{C}^-_{\leq i} & & & \mathcal{C}^+_{\leq i} & -\mathcal{S}_{\leq i} \end{pmatrix}.$$

Simple calculations show that the preconditioned matrix is then given by

$$(\mathcal{P}_{\leq i}^{\text{hook}})^{-1}\mathcal{B}_{\leq i} = \begin{pmatrix} I & & & & \\ & \ddots & & & -\mathcal{Z}_{\leq i}^{-} \\ & & I & & \\ & & & I & \mathcal{Z}_{\leq i}^{+} \\ & & & & I \end{pmatrix}, \tag{5.6}$$

where $\mathcal{Z}_{\leq i}^{-} \coloneqq \text{diag}(B_{j_1}^{-1}(C_{k_1}^{-})^T, \ldots, B_{j_{r_i}}^{-1}(C_{k_{r_i}}^{-})^T)$ and $\mathcal{Z}_{\leq i}^{+} \coloneqq B_i^{-1}(\mathcal{C}_{\leq i}^{+})^T$. Consequently, we can make the following observation:

Lemma 5.3. *If the height of vertex i is one, then the minimal polynomial of the preconditioned matrix* (5.6) *is given by*

$$\mu_{(\mathcal{P}_{\leq i}^{\text{hook}})^{-1}\mathcal{B}_{\leq i}}(x) = (1-x)^2.$$

Proof. The minimal polynomial of the preconditioned matrix (5.6) is given by $(1-x)^2$ since the square of the matrix

$$I - (\mathcal{P}_{\leq i}^{\text{hook}})^{-1}\mathcal{B}_{\leq i} = \begin{pmatrix} 0 & & & & \\ & \ddots & & & \mathcal{Z}_{\leq i}^{-} \\ & & 0 & & \\ & & & 0 & -\mathcal{Z}_{\leq i}^{+} \\ & & & & 0 \end{pmatrix}$$

evaluates to zero. □

Corollary 5.1. *If the height of vertex i is one, then the GMRES method applied to $\mathcal{B}_{\leq i}$ with preconditioner $\mathcal{P}_{\leq i}^{\text{hook}}$ converges in at most two iterations in exact arithmetic.*

Proof. See [29, Proposition 2]. □

5.4.3 Recursive Preconditioning

Naturally, for vertices with larger heights, the performance of the hook preconditioner can be assumed to degrade. However, we can make use of the preconditioner within an iterative scheme in order to solve the system $\mathcal{B}_{\leq i}x_{\leq i} = h_{\leq i}$. To this end, we

generalize the hook preconditioner to the concept of a *recursive preconditioner*, where a recursive preconditioner $\mathcal{P}_{\leq i}^{\text{rec}}$ is defined by the following conditions:

1. If i is a leaf of $\mathbb{D}$, then it holds that $\mathcal{P}_{\leq i}^{\text{rec}} = B_i$.

2. If i is an inner vertex, then $\mathcal{P}_{\leq i}^{\text{rec}}$ follows the recursive construction

$$\mathcal{P}_{\leq i}^{\text{rec}} \coloneqq \begin{pmatrix} \mathcal{P}_{\leq j_1}^{\text{rec}} & & & \\ & \ddots & & \\ & & \mathcal{P}_{\leq j_{r_i}}^{\text{rec}} & \\ & & & B_i \\ -\mathcal{C}_{\leq i}^{-} & & \mathcal{C}_{\leq i}^{+} & -\mathcal{S}_{\leq i} \end{pmatrix}. \tag{5.7}$$

Based on a recursive preconditioner $\mathcal{P}_{\leq i}^{\text{rec}}$ and an initial solution $x_{\leq i}^{(0)}$, we consider a linear fixed-point iteration based on the approximation $(\mathcal{P}_{\leq i}^{\text{rec}})^{-1} \approx \mathcal{B}_{\leq i}^{-1}$, which is given by

$$x_{\leq i}^{(k+1)} \leftarrow x_{\leq i}^{(k)} + (\mathcal{P}_{\leq i}^{\text{rec}})^{-1} \left(h_{\leq i} - \mathcal{B}_{\leq i} x_{\leq i}^{(k)} \right). \tag{5.8}$$

We can observe that recursive preconditioners can be made recursively exact when moving from the leaves of a tree towards its root by means of the linear fixed-point iteration:

Lemma 5.4. *Let i be a vertex of $\mathbb{D}$. If the recursive preconditioner $\mathcal{P}_{\leq i}^{\text{rec}}$ is exact on the child vertices $j_1, \ldots, j_{r_i}$ of i, i.e.,*

$$\mathcal{P}_{\leq j_l}^{\text{rec}} = \mathcal{B}_{\leq j_l} \quad \forall\, l = 1, \ldots, r_i,$$

then the iterative method given by (5.8) *converges in at most two iterations in exact arithmetic, independently of the initial iterate $x_{\leq i}^{(0)}$.*

Proof. If i is a leaf, then $\mathcal{P}_{\leq i}^{\text{rec}} = B_i$ must be exact. Otherwise, simple calculations show that the preconditioned matrix is given by

$$(\mathcal{P}_{\leq i}^{\text{rec}})^{-1} \mathcal{B}_{\leq i} = \begin{pmatrix} I & & & & \\ & \ddots & & & -\mathcal{Y}_{\leq i}^{-} \\ & & I & & \\ & & & I & \mathcal{Z}_{\leq i}^{+} \\ & & & & I \end{pmatrix},$$

where $\mathcal{Y}_{\leq i}^{-} \coloneqq \operatorname{diag}((\mathcal{P}_{\leq j_1}^{\text{rec}})^{-1}(\mathcal{C}_{j_1}^{-})^T, \ldots, (\mathcal{P}_{\leq j_{r_i}}^{\text{rec}})^{-1}(\mathcal{C}_{j_{r_i}}^{-})^T)$ and $\mathcal{Z}_{\leq i}^{+}$ is defined as above. As was the case before, the minimal polynomial of the preconditioned matrix is given by

$$\mu_{(\mathcal{P}_{\leq i}^{\text{rec}})^{-1}\mathcal{B}_{\leq i}}(x) = (1-x)^2,$$

which of course implies that $(I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})^2 = 0$. Consequently, the first two computed iterates are given by

$$\begin{aligned} x^{(1)}_{\leq i} &= (I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})x^{(0)}_{\leq i} + (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}h_{\leq i}, \\ x^{(2)}_{\leq i} &= (I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})^2 x^{(0)}_{\leq i} + (2I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})(\mathcal{P}^{\text{rec}}_{\leq i})^{-1}h_{\leq i} \\ &= (2I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})(\mathcal{P}^{\text{rec}}_{\leq i})^{-1}h_{\leq i}. \end{aligned}$$

Note that the value of $x^{(2)}_{\leq i}$ is independent of the initial solution $x^{(0)}_{\leq i}$. This holds in particular if we set $x^{(0)}_{\leq i} = \mathcal{B}^{-1}_{\leq i}h_{\leq i} =: x^*$, in which case it follows that $x^{(k)}_{\leq i} \equiv x^*$ for all $k \in \mathbb{N}$ and specifically for $k = 2$. However, due to the independence of $x^{(2)}_{\leq i}$ it must hold that $x^{(2)}_{\leq i} = x^*$ for *all* choices of $x^{(0)}_{\leq i}$. □

5.4.4 Exact Preconditioning

Based on the results obtained previously, we can derive an exact polynomial preconditioner $\mathcal{P}^{\text{exact}}$ as an alternative to the direct method introduced in Section 5.3. Specifically, our goal is to derive a preconditioner, which, as opposed to $\mathcal{P}^{\text{hook}}$, satisfies the conditions of Lemma 5.4 at every vertex, thereby ensuring exactness. We build this preconditioner bottom up, starting with single vertices while maintaining the conditions of Lemma 5.3. For each leaf $i \in \mathbb{V}$, we set $\mathcal{P}^{\text{exact}}_{\leq i} := B_i$. For each vertex of height one, we consider the recursive preconditioner $\mathcal{P}^{\text{rec}}_{\leq i}$ following the construction (5.7), using $\mathcal{P}^{\text{exact}}_{\leq j_l}$ for $l = 1, \dots, r_i$. This preconditioner satisfies the conditions of Lemma 5.4 yielding the exact solution of $\mathcal{B}_{\leq i}x_{\leq i} = h_{\leq i}$ by applying (5.8) twice. This recursion is linear in the right-hand side $h_{\leq i}$ and can be used to define $\mathcal{P}^{\text{exact}}_{\leq i}$:

$$x^{(2)}_{\leq i} = (2I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})(\mathcal{P}^{\text{rec}}_{\leq i})^{-1}h_{\leq i} =: (\mathcal{P}^{\text{exact}}_{\leq i})^{-1}h_{\leq i}.$$

For vertices of height two, we again perform two iterations using the recursive construction (5.7) based on the newly defined exact preconditioners for vertices of height one, yielding an exact preconditioner $\mathcal{P}^{\text{exact}}_{\leq i}$ for each vertex $i \in \mathbb{V}$ of height two. We follow this process until we reach the root R, obtaining an exact (but no longer recursive) preconditioner for the nested formulation of the original problem (5.2):

Corollary 5.2. *$\mathcal{P}^{\text{exact}}_{\leq i}$ is exact for every vertex $i \in \mathbb{V}$.*

5.4.5 Complexity

In the following, we consider the complexity of applying $(\mathcal{P}^{\text{hook}})^{-1}$ and $(\mathcal{P}^{\text{exact}})^{-1}$ in terms of the number of solutions of systems involving B_i analogous to Section 5.3. We exclude the computation of the Schur complements $\mathcal{S}_{\leq i}$ because their complexity has been established in Lemma 5.2 and find the following results:

Lemma 5.5. *1. The application of the inverse of $\mathcal{P}^{\text{hook}}$ requires exactly one solution of a system involving B_i for each vertex $i \in \mathbb{V}$.*

2. *The application of the inverse of* $\mathcal{P}^{\text{exact}}$ *requires* $\Theta(2^{\text{depth}(i)})$ *solutions of systems involving* B_i *for each vertex* $i \in \mathbb{V}$.

Proof. 1. To apply $(\mathcal{P}^{\text{hook}}_{\leq i})^{-1}$ at a vertex $i \in \mathbb{V}$, we need to solve exactly one system involving B_i and recurse into each child vertex of i exactly once. Therefore, to apply $\mathcal{P}^{\text{hook}}$ at the root vertex R, we have to solve exactly one system involving B_i.

2. Our reasoning is analogous to the proof of Lemma 5.2: To apply $(\mathcal{P}^{\text{exact}}_{\leq i})^{-1}$ at a leaf $i \in \mathbb{V}$, we solve a system involving B_i exactly once. At an inner vertex i, we need to compute the product of

$$(\mathcal{P}^{\text{exact}}_{\leq i})^{-1} = (2I - (\mathcal{P}^{\text{rec}}_{\leq i})^{-1}\mathcal{B}_{\leq i})(\mathcal{P}^{\text{rec}}_{\leq i})^{-1}$$

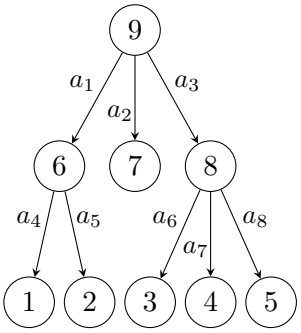
with some right-hand side. The computation of the product involves two solves of systems involving B_i itself as well as two recursions into each of the subtrees rooted at the child vertices $j_1, \ldots, j_{r_i}$ of i in order to apply the inverses of the respective preconditioners $\mathcal{P}^{\text{exact}}_{\leq j_l}$. Consequently, when moving up the tree from i towards R, the number of solves of systems involving B_i doubles each time we move from a vertex to its parent, leading to the stated complexity. □

It is apparent from these results that the application of the inverse of $\mathcal{P}^{\text{exact}}$ has the same asymptotic complexity as the direct method introduced in Section 5.3, which is exponential in the height of $\mathbb{D}$ and therefore only polynomial for trees with logarithmic heights. Conversely, the inverse of $\mathcal{P}^{\text{hook}}$ can always applied in polynomial time.

5.5 Non-nested Arrowhead Structure

In the following, we examine the original problem (5.2) more closely, particularly focusing on the Schur complement (5.3), which we assume to be positive definite. We derive a preconditioner $\mathcal{P}$ based on the non-nested Schur complement and focus on the efficient application of an approximate inverse of this preconditioner. We first note that the Schur complement $\mathcal{S}$ consists of blocks of sizes $l_k \times l_{k'}$ for each pair $(a_k, a_{k'})$ of arcs, where a block is non-zero iff the arcs share a vertex. It is however advantageous to instead examine $\mathcal{S}$ using vertex-based blocks, each of which combines the variables of the outgoing arcs of the respective vertices. Consequently, the only non-trivial vertex-based blocks appear for inner vertices of the inner subgraph $\mathbb{D}^\circ$.

Each vertex-based block $(\mathcal{S}_{i,j})_{i,j \in \mathbb{V}^\circ}$ has size $l_i^+ \times l_j^+$. An illustration of the resulting block structure is shown in Figure 5.2. We choose this partitioning of the Schur complement as it provides better performance with the subsequent methods. The following theory holds in both settings nevertheless. The individual blocks $\mathcal{S}_{i,j}$—subject to block permutations to account for the ordering of nodes and arcs—can be defined in terms of $\delta^+(i) = (a_{k_1} = (i, j_1), \ldots, a_{k_{r_i}} = (i, j_{r_i}))$ as follows:



$$
\begin{array}{c}
\\
a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \end{array} \\
\left(\begin{array}{ccc|cc|ccc}
\mathcal{S}^a_{1,1} & \mathcal{S}^a_{1,2} & \mathcal{S}^a_{1,3} & \mathcal{S}^a_{1,4} & \mathcal{S}^a_{1,5} & & & \\
\mathcal{S}^a_{2,1} & \mathcal{S}^a_{2,2} & \mathcal{S}^a_{2,3} & & & & & \\
\mathcal{S}^a_{3,1} & \mathcal{S}^a_{3,2} & \mathcal{S}^a_{3,3} & & & \mathcal{S}^a_{3,6} & \mathcal{S}^a_{3,7} & \mathcal{S}^a_{3,8} \\
\hline
\mathcal{S}^a_{4,1} & & & \mathcal{S}^a_{4,4} & \mathcal{S}^a_{4,5} & & & \\
\mathcal{S}^a_{5,1} & & & \mathcal{S}^a_{5,4} & \mathcal{S}^a_{5,5} & & & \\
\hline
 & & \mathcal{S}^a_{6,3} & & & \mathcal{S}^a_{6,6} & \mathcal{S}^a_{6,7} & \mathcal{S}^a_{6,8} \\
 & & \mathcal{S}^a_{7,3} & & & \mathcal{S}^a_{7,6} & \mathcal{S}^a_{7,7} & \mathcal{S}^a_{7,8} \\
 & & \mathcal{S}^a_{8,3} & & & \mathcal{S}^a_{8,6} & \mathcal{S}^a_{8,7} & \mathcal{S}^a_{8,8}
\end{array}\right)
\end{array}
$$

Figure 5.2: A graph $\mathbb{D}$ together with the Schur complement $\mathcal{S}$. The eight arc-based blocks (denoted here by $\mathcal{S}^a_{k,k'}$) can be combined into three vertex-based blocks corresponding to the outgoing arcs of the inner vertices, namely $\delta^+(9) = \{a_1, a_2, a_3\}$, $\delta^+(6) = \{a_4, a_5\}$, and $\delta^+(8) = \{a_6, a_7, a_8\}$.

1. For $i = j$ the matrix $\mathcal{S}_{i,i}$ consists of blocks

$$
\begin{cases}
C^+_{k_l} B_i^{-1} (C^+_{k_l})^T + C^-_{k_l} B_{j_l}^{-1} (C^-_{k_l})^T + D_{k_l} & \text{if } l = l', \\
C^+_{k_l} B_i^{-1} (C^+_{k_{l'}})^T & \text{otherwise,}
\end{cases}
$$

   for $l, l' = 1, \ldots, r_i$.
2. For $i \neq j$ such that $(i, j) \in \mathbb{A}^\circ$, we let $\delta^+(j) = (a_{k'_1}, \ldots, a_{k'_{r_j}})$ be the outgoing arcs of j, noting that $\mathcal{S}_{i,j}$ consists of $r_i \times r'_j$ blocks. The block for $l = 1, \ldots, r_i$ and $l' = 1, \ldots, r'_j$ is given by $-C^-_{k_l} B_j^{-1} (C^+_{k'_{l'}})^T$ if $a_{k_l} = (i, j)$ and a zero matrix otherwise.
3. For $i \neq j$ with $(j, i) \in \mathbb{A}^\circ$, it holds that $\mathcal{S}_{j,i} = \mathcal{S}^T_{i,j}$.
4. All other blocks are zero.

Note that while $\mathcal{S}$ does not coincide with $\mathcal{S}_{\leq R}$, the complexity required to compute $\mathcal{S}$ is on par with the complexity of computing all Schur complements $\mathcal{S}_{\leq i}$ using Algorithm 5.2 which was established in Lemma 5.2. Specifically, for each $j \in \mathbb{V}^\circ$ the computation of the block $\mathcal{S}_{j,j}$ requires l^+_j solutions of systems involving B_j plus an additional l^-_j for the block $\mathcal{S}_{i,j}$ if $(i, j) \in \mathbb{A}^\circ$. As a result, $O(l_j)$ solutions of systems involving B_j are required in total.

We proceed to investigate the application of the block-triangular preconditioner [128] in the context of tree-coupled systems. This preconditioner is given as

$$
\mathcal{P} \coloneqq \begin{pmatrix} \mathcal{B} & 0 \\ \mathcal{C} & \mathcal{S} \end{pmatrix} \quad \Longrightarrow \quad \mathcal{P}^{-1} \begin{pmatrix} \mathcal{B} & \mathcal{C}^T \\ \mathcal{C} & -\mathcal{D} \end{pmatrix} = \begin{pmatrix} I & \mathcal{B}^{-1}\mathcal{C}^T \\ 0 & -I \end{pmatrix}.
$$

Note that due to a difference in notation, we use $\mathcal{S}$ rather than $-\mathcal{S}$ in order to obtain this preconditioner. Moreover, we examine the preconditioner in a block-upper rather than block-lower form. This, however, affects neither its spectral properties nor any computational results in a significant way. It is apparent from inspection that the preconditioned matrix has the two eigenvalues ± 1 and its minimal polynomial is $(x-1)(x+1)$, leading to two-step convergence of a preconditioned GMRES method. The application of $\mathcal{P}^{-1}$ to a right-hand side (h, f), yielding a solution of (x, y), is a three-step process:

1. Compute x by solving $\mathcal{B}x = h$.
2. Compute the right-hand side $f_x \leftarrow f - \mathcal{C}x$.
3. Compute y by solving $\mathcal{S}y = f_x$.

The first step is straightforward, involving a solution of a system involving B_i for each $i \in \mathbb{V}$. The solution of the system involving $\mathcal{S}$ is computationally more challenging, since $\mathcal{S}$ has a size equal to the total number of coupled variables, therefore being substantially larger than each of the Schur complements $\mathcal{S}_{\leq i}$ from Section 5.3. Of course, since $\mathcal{S}$ is positive definite by Assumption 5.1, we can use a Cholesky decomposition to compute y, for example. We can, however, alternatively employ an approach that is more tailored to the structure of (5.2). To this end, we note that the positive definiteness of $\mathcal{S}$ implies that all of its diagonal blocks $\mathcal{S}_{i,i}$ must be positive definite as well. Consequently, the diagonal approximation $\mathcal{S}_{\text{diag}} \coloneqq \text{diag}(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{N,N})$ of $\mathcal{S}$ is also positive definite and therefore invertible. We can therefore use $\mathcal{S}_{\text{diag}}$ within the fixed-point iteration

$$y^{(k+1)} \leftarrow y^{(k)} + \mathcal{S}_{\text{diag}}^{-1}(f_x - \mathcal{S}y^{(k)}), \tag{5.9}$$

based on an initial solution $y^{(0)}$. A sufficient condition for the convergence of the iteration is what we refer to as the *smoothing property* established by the following bound on the spectral radius ρ of the iteration matrix:

Theorem 5.1. *It holds that* $\rho(I - \mathcal{S}_{\text{diag}}^{-1}\mathcal{S}) < 1$.

Proof. Recall that $\mathcal{S}$ as well as the blocks $\mathcal{S}_{i,i}$ and $\mathcal{S}_{\text{diag}}$ are positive definite by Assumption 5.1. Therefore, $\mathcal{S}_{\text{diag}}^{1/2}$ and $\mathcal{S}_{\text{diag}}^{-1/2}$ exist and the matrix $I - \mathcal{S}_{\text{diag}}^{-1}\mathcal{S}$ is similar to

$$\mathcal{S}_{\text{diag}}^{1/2}\left(I - \mathcal{S}_{\text{diag}}^{-1}\mathcal{S}\right)\mathcal{S}_{\text{diag}}^{-1/2} = I - \mathcal{S}_{\text{diag}}^{-1/2}\mathcal{S}\mathcal{S}_{\text{diag}}^{-1/2}.$$

Since this matrix is symmetric, all eigenvalues of $I - \mathcal{S}_{\text{diag}}^{-1}\mathcal{S}$ are real. To show that all of these real eigenvalues are contained in $(-1, 1)$ we have to show that $\det((1-\lambda)I - \mathcal{S}_{\text{diag}}^{-1}\mathcal{S}) \neq 0$ for all $\lambda \notin (-1, 1)$, or, equivalently that $\det(\mu\mathcal{S}_{\text{diag}} - \mathcal{S}) \neq 0$ for all $\mu \notin (0, 2)$.

For $\mu \leq 0$ the case is clear, since we add a positive multiple of a negative definite matrix to another negative definite matrix, preserving negative definiteness and therefore non-singularity.

To handle the case $\mu \geq 2$, we show that $\hat{\mathcal{S}} \coloneqq 2\mathcal{S}_{\text{diag}} - \mathcal{S}$ is positive definite: This is sufficient to ensure non-singularity for all $\mu > 2$, since we could then simply add a positive multiple of $(\mu - 2)$ of the positive definite $\mathcal{S}_{\text{diag}}$ to the positive definite matrix $\hat{\mathcal{S}}$. To show that $\hat{\mathcal{S}}$ is positive definite, consider the matrix $\mathcal{Z} = \text{diag}(Z_1, \ldots, Z_N)$, where $Z_i \coloneqq (-1)^{\text{depth}(i)}I$ for each $i \in \mathbb{V}^\circ$. It

holds that $\mathcal{Z}^T\mathcal{Z} = I$, implying that $\mathcal{Z}$ is invertible and $\tilde{\mathcal{S}} \coloneqq \mathcal{Z}\mathcal{S}\mathcal{Z}^T$ is positive definite. Due to the cancellation of negative signs, the block diagonal entries of $\tilde{\mathcal{S}}$ and $\mathcal{S}$ coincide. Furthermore, for each arc $a_k = (i,j) \in \mathbb{A}^\circ$, we have that $(-1)^{\mathrm{depth}(i)} \cdot (-1)^{\mathrm{depth}(j)} = -1$, ensuring that the off-diagonal blocks in $\tilde{\mathcal{S}}$ and $\mathcal{S}$ have opposing signs. Consequently, we have that $\tilde{\mathcal{S}} = 2\mathcal{S}_{\mathrm{diag}} - \mathcal{S} = \hat{\mathcal{S}}$, proving that $\hat{\mathcal{S}}$ is positive definite. □

Corollary 5.3. *The iterates $y^{(k)}$ produced by (5.9) converge to $y = \mathcal{S}^{-1}f_x$ for any initial $y^{(0)}$.*

Proof. This is an elementary result in iterative linear algebra (see e.g., [147, Theorem 4.1]), directly following from the smoothing property established in Theorem 5.1. □

5.5.1 Two-level Method

In the following we expand on the iterative method introduced above by applying multigrid (MG) ideas in order to solve the Schur complement, obtaining a multi-level method. The origins of algebraic multigrid (AMG) methods date back to the seminal works [148, 149], with an introduction given in [35]. We apply the exact two-level method [35, Sec. 5] to the problem of solving systems involving $\mathcal{S}$. To this end, we let $n_f \coloneqq \sum_{i\in\mathbb{V}^\circ} l_i^+$ be the dimension of $\mathcal{S}$ and consider a *coarse* approximation of the solution space $\mathbb{R}^{n_f}$ with a dimension of $n_c < n_f$. The *prolongation* matrix $P \in \mathbb{R}^{n_f \times n_c}$ is used to map solutions from the coarse to the original (fine) space whereas its transpose *restricts* solutions to the coarse space. The *smoothing* matrix $G \in \mathbb{R}^{n_f \times n_f}$ is chosen, as its name suggests, to satisfy the smoothing property with respect to $\mathcal{S}$. Furthermore, we let $\mathcal{S}_c \coloneqq P^T\mathcal{S}P$ denote the restriction of $\mathcal{S}$ to $\mathbb{R}^{n_c}$, which we assume to be invertible. The multi-level (ML) approximation $\Psi \in \mathbb{R}^{n_f \times n_f}$ of the inverse of $\mathcal{S}$ is defined by its action on a right-hand side $g \in \mathbb{R}^{n_f}$, which is computed based on the following steps:

1. Restrict g to $\mathbb{R}^{n_f}$ via $g_c \leftarrow P^T g$.
2. Compute coarse-space correction $w_c \leftarrow \mathcal{S}_c^{-1} g_c$.
3. Prolongate correction $w \leftarrow P w_c$.
4. Compute post-smoothing step $\Psi(g) \coloneqq w + G(g - \mathcal{S}w)$.

To ensure convergence, it is once again necessary to bound the spectral radius of the iteration matrix $I - \Psi\mathcal{S}$:

Lemma 5.6 ([35, Lemma 5.2])**.** *The iteration matrix $(I - \Psi\mathcal{S})$ is given by*

$$(I - G\mathcal{S})(I - \Pi_c),$$

where Π_c is the $(\cdot,\cdot)_{\mathcal{S}}$-orthogonal projection on $\mathbb{R}^{n_f}$, given by $\Pi_c \coloneqq P\mathcal{S}_c^{-1}P^T\mathcal{S}$.

Since the matrix $(I-\Pi_c)$ defines a $(\cdot,\cdot)_{\mathcal{S}}$-orthogonal projection, we have the estimate $\|I-\Pi_c\|_{\mathcal{S}} \leq 1$, where $\|\cdot\|_{\mathcal{S}}$ denotes the norm induced by the corresponding scalar product. Moreover, we have that $\|T\|_{\mathcal{S}} = \|\mathcal{S}^{1/2}T\mathcal{S}^{-1/2}\|_2$ for any compatible square matrix T. Along with the symmetry of G, we can bound the spectral radius of the ML iteration from above by

$$\begin{aligned}\rho\left((I-G\mathcal{S})\left(I-\Pi_c\right)\right) &\leq \left\|(I-G\mathcal{S})\left(I-\Pi_c\right)\right\|_{\mathcal{S}} \leq \|I-G\mathcal{S}\|_{\mathcal{S}} \\ &= \|I-\mathcal{S}^{1/2}G\mathcal{S}^{1/2}\|_2 = \rho\left(I-\mathcal{S}^{1/2}G\mathcal{S}^{1/2}\right) = \rho\left(I-G\mathcal{S}\right).\end{aligned}$$

Hence, the ML iteration itself satisfies the smoothing property with respect to $\mathcal{S}$ as long as G does. Since the smoothing property holds for $\mathcal{S}_{\text{diag}}$, setting $G = \mathcal{S}_{\text{diag}}^{-1}$ is an obvious choice. Readers familiar with multigrid methods will note that this iteration consists of a single post-smoothing step without any pre-smoothing applied. This is for ease of notation, and in practice may be adapted to improve convergence speed.

It remains to choose a suitable coarse prolongation matrix P. An important requirement for the choice is that the coarse-space correction w_c can be computed efficiently. Recall that a reason for why we cannot easily solve the matrix $\mathcal{S}$ in the first place is the presence of the off-diagonal blocks $\mathcal{S}_{i,j}$. If those blocks were not present, we could simply decompose the diagonal blocks. What is more, to apply the smoother $\mathcal{S}_{\text{diag}}^{-1}$, we may want to decompose the diagonal blocks anyway. We therefore propose to consider as a coarse space the restriction of $\mathcal{S}$ to a *conflict-free* set of vertices. Two vertices $i, j \in \mathbb{V}^\circ$ are said to be *in conflict* if $(i,j) \in \mathbb{A}^\circ$ or $(j,i) \in \mathbb{A}^\circ$ and *conflict-free* otherwise. A set $\mathbb{V}_s^\circ$ of vertices is *conflict-free* iff all of its vertices are conflict-free. Notable examples of conflict-free sets are given by the (inclusion-wise maximal) sets of even/odd vertices:

$$\mathbb{V}_{\text{even}}^\circ \coloneqq \{i \in \mathbb{V}^\circ \mid \text{depth}(i) \text{ even}\} \quad \text{and} \quad \mathbb{V}_{\text{odd}}^\circ \coloneqq \{i \in \mathbb{V}^\circ \mid \text{depth}(i) \text{ odd}\}.$$

To define a prolongation and restriction, consider two (ordered) subsets $\mathbb{V}_c^\circ \subseteq \mathbb{V}_f^\circ$ of vertices where $\mathbb{V}_f^\circ \coloneqq (i_1, \ldots, i_p)$ and $\mathbb{V}_c^\circ \coloneqq (j_1, \ldots, j_q)$. The *subset operator* $\mathrm{S}_{\mathbb{V}_c^\circ,\mathbb{V}_f^\circ}$ mapping from $\mathbb{R}^{l_{i_1}^+} \times \cdots \times \mathbb{R}^{l_{i_p}^+}$ to $\mathbb{R}^{l_{j_1}^+} \times \cdots \times \mathbb{R}^{l_{j_q}^+}$ is defined by its action on a vector $(y_{i_1}, \ldots, y_{i_p})$ as

$$\mathrm{S}_{\mathbb{V}_c^\circ,\mathbb{V}_f^\circ}((y_{i_1}, \ldots, y_{i_p})) \coloneqq (y_{j_1}, \ldots, y_{j_q}).$$

For a given conflict-free set $\mathbb{V}_s^\circ = \{i_1, \ldots, i_l\}$, we can therefore obtain a two-level method by defining the prolongation matrix to be $P \coloneqq \mathrm{S}_{\mathbb{V}_s^\circ,\mathbb{V}^\circ}^T$. As a consequence, the restriction $P^T = \mathrm{S}_{\mathbb{V}_s^\circ,\mathbb{V}^\circ}$ removes from a vector y the entries not belonging to $\mathbb{V}_s^\circ$, and

$$\mathcal{S}_c = \text{diag}(\mathcal{S}_{1,1}, \ldots, \mathcal{S}_{l,l})$$

is block-diagonal and invertible, enabling the efficient computation of the coarse-space correction.

5.5.2 Multi-level Methods

A different choice for the prolongation matrix P may be used to derive multi-level methods aiding in solving for the Schur complement $\mathcal{S}$. A *multi-level method* with K levels consists of K increasingly fine approximations of dimensions $n_c = n_1 \leq n_2 \leq \ldots \leq n_f$ of the original solution space together with smoothing and prolongation matrices $G_j \in \mathbb{R}^{n_j \times n_j}$, $P_j \in \mathbb{R}^{n_{j+1} \times n_j}$ yielding restrictions $\mathcal{S}_j \coloneqq P_j^T \cdots P_{K-1}^T \mathcal{S} P_{K-1} \cdots P_j$ for $j = 1, \ldots, K-1$, where we set $\mathcal{S}_1 =: \mathcal{S}_c$. The *V-cycle* consists of computing corrections on increasingly coarse subspaces up to $\mathbb{R}^{n_f}$ (where $\mathcal{S}_c$ is solved exactly), followed by a post-smoothing step. The corresponding ML matrix $\Psi_j \in \mathbb{R}^{n_j \times n_j}$ at level j can be described recursively by its action on a right-hand side $g_j \in \mathbb{R}^{n_j}$:

1. If $j = 1$, return $\mathcal{S}_c^{-1} g_j$.
2. Restrict g_j to $\mathbb{R}^{n_{j-1}}$ via $g_{j-1} \leftarrow P_{j-1}^T g_j$.
3. Compute coarse-space correction $w_{j-1} \leftarrow \Psi_{j-1}(g_{j-1})$.
4. Prolongate correction $w_j \leftarrow P_{j-1} w_{j-1}$.
5. Compute post-smoothing step $\Psi_j(g_j) \coloneqq w_j + G_j(g_j - \mathcal{S}_j w_j)$.

To define the matrices, we consider a nested family of sets of vertices $\mathbb{V}_c^\circ = \mathbb{V}_1^\circ \subset \cdots \subset \mathbb{V}_K^\circ = \mathbb{V}^\circ$. For each level $j = 1, \ldots, K$ we pick as $\mathbb{R}^{n_j}$ the space associated with the variables corresponding to $\mathbb{V}_j^\circ$. Correspondingly, the prolongation matrix for $j \in \{1, \ldots, K-1\}$ is given by $P_j \coloneqq \mathrm{S}^T_{\mathbb{V}_j^\circ, \mathbb{V}_{j+1}^\circ}$ Furthermore, we let the finest smoothing matrix be the original diagonal part of $\mathcal{S}$, i.e., $G_K \coloneqq \mathcal{S}_{\mathrm{diag}}^{-1}$, and set $G_j \coloneqq P_j^T \cdots P_{K-1}^T G_K P_{K-1} \cdots P_j$, which is to say the non-singular matrix consisting of the blocks of $\mathcal{S}_{\mathrm{diag}}^{-1}$ corresponding to $\mathbb{V}_j^\circ$. Finally, to ensure that we can solve $\mathcal{S}_c$ efficiently, we ask that $\mathbb{V}_c^\circ$ be conflict-free. To define nested families of sets of arcs, we once again make use of the topology of the graph by setting

$$\mathbb{V}_{\leq k}^\circ \coloneqq \{i \in \mathbb{V}^\circ \mid \mathrm{depth}(i) \leq k\} \quad \text{and}$$
$$\mathbb{V}_{\geq k}^\circ \coloneqq \{i \in \mathbb{V}^\circ \mid \mathrm{depth}(i) \geq k\},$$

for $k = 0, \ldots, \mathrm{height}(\mathbb{D}^\circ)$. Clearly it holds that $\mathbb{V}_{\leq j}^\circ \subset \mathbb{V}_{\leq j+1}^\circ$, and in particular $\mathbb{V}_{\leq \mathrm{height}(\mathbb{D}^\circ)}^\circ = \mathbb{V}^\circ$. Furthermore, the set $\mathbb{V}_{\leq 0}^\circ = \{R\}$ is conflict-free. Thus, we can construct a multi-level method with $K = \mathrm{height}(\mathbb{D}^\circ)$ levels based on these arc sets by setting $\mathbb{V}_j^\circ \coloneqq \mathbb{V}_{\leq j}^\circ$, where the coarsest solution space consists of the arcs incident to the root R. Symmetrically, we can consider the sets $\mathbb{V}_{\geq j+1}^\circ \subset \mathbb{V}_{\geq j}^\circ$, where $\mathbb{V}_{\geq 1}^\circ = \mathbb{V}^\circ$ and $\mathbb{V}_{\geq \mathrm{height}(\mathbb{D}^\circ)}^\circ$ consist of the (conflict-free) set of leaves of $\mathbb{D}^\circ$. Thus, we can set $\mathbb{V}_j^\circ \coloneqq \mathbb{V}_{\geq \mathrm{height}(\mathbb{D}) - j + 1}^\circ$ to obtain another multi-level method. We call these methods the *Bottom-Up* and *Top-Down* multi-level methods, respectively.

As for the convergence of these methods, we can apply Lemma 5.6 repeatedly going from the coarsest space (where the restricted system is solved exactly) to the finest space. To this end, the smoothing matrix G_j has to satisfy the smoothing property with respect to $\mathcal{S}_j$ for all $j = 2, \ldots, K$. It is, however, easy to see that these smoothing properties are always satisfied, since they correspond to the original smoothing property established in Theorem 5.1, applied either to a single subtree in the Bottom-Up method or several times to different subtrees in the Top-Down method.

Finally, several alternative iteration schemes, known as W- and F-cycles [67] have been introduced and can be easily adapted into corresponding multi-level methods. The notable difference between V- and W-cycles consists of the fact that the latter iteration performs two coarse-correction steps rather than a single one in each level. Consequently, the running time of the W-cycles grows exponentially in $K = \text{height}(\mathbb{D})$, whereas the running times of V- and F-cycles remain polynomial in K.

5.5.3 Super-Node Smoothing

The matrix $\mathcal{S}_{\text{diag}}$ introduced above has the smoothing property and is therefore a suitable choice as a smoother, while having the disadvantage of not capturing the adjacency in $\mathbb{D}^\circ$. An alternative is to incorporate the adjacency of a subset of the vertices in $\mathbb{V}^\circ$. To capture some of this information, we can merge a subset of the vertices in $\mathbb{D}$ into a single super-vertex. Specifically, we let $i \in \mathbb{V}^\circ$ and $\delta^+(i) = (a_{k_1} = (i, j_1), \ldots, a_{k_{r_i}} = (i, j_{r_i}))$ and treat the submatrix of $\mathcal{S}$ composed of the blocks for $i, j_1, \ldots, j_{r_i}$ as a single larger block. In terms of the underlying graph structure, the creation of the super-vertex is equivalent to the contraction of the arcs $a_{k_1}, \ldots, a_{k_{r_i}}$ in any order, resulting in a smaller directed tree corresponding to the change in the blocks of $\mathcal{S}$. Consequently, Theorem 5.1 still applies and we obtain another smoother to be used in our multi-level method. However, to apply the inverse of this *super-node* smoother to a right-hand side, we have to solve the system corresponding to the newly created super-vertex, possibly requiring an additional factorization. The approach can be extended by creating multiple super vertices based on different vertices. Furthermore, given a subset $\mathbb{V}_j^\circ \subseteq \mathbb{V}^\circ$ associated with a specific level j in a multi-level method, we can pick these super vertices depending on $\mathbb{V}_j^\circ$ (provided that $i, j_1, \ldots, j_{r_i}$ are contained in $\mathbb{V}_j^\circ$) in order to obtain a level-specific smoother. Since the intuitive importance of arcs decreases with decreasing height, we propose to turn all vertices in $\mathbb{V}_j^\circ$ with maximum height into super vertices, yielding the smoother $\mathcal{S}_{\text{super}}(\mathbb{V}_j^\circ)$.

5.6 Computational Experiments

In this section, we showcase the performance of the presented preconditioners with a set of numerical experiments. We examine the convergence of the Krylov solver GMRES equipped with our preconditioners and its dependence on the problem parameters and preconditioner settings. Table 5.1 provides an overview of the preconditioners that are analyzed subsequently, along with their abbreviated identifiers and definitions. The tests are implemented in Python 3.9.18 [150], complemented with the libraries SciPy 1.10.0 [69] for various numerical linear algebra routines, and DOLFINx for finite element method functionalities [68]. All experiments were conducted on an Intel® Core™ i7-10710U processor.

Table 5.1: Overview of the tested preconditioners and their definitions.

Formulation	Preconditioner	Definition
Nested	Nested Block-Diagonal	Equation (5.4) in Section 5.4.1
	Hook	Equation (5.5) in Section 5.4.2
	Recursive	Equation (5.7) in Section 5.4.3
	Direct method	Section 5.3
Non-nested	Non-nested Block-Diagonal	Equation (5.9) in Section 5.5
	Multi-level (ML)	Section 5.5.2

The following experiments serve as a proof of concept regarding the convergence behavior and the scaling of the computational cost of the iterative methods. We showcase our ability to compete with widely used methods by considering a specific example in Section 5.6.2. It is noted that most of the following problems can be solved efficiently with direct linear solvers, including sparse LU factorizations such as SuperLU [24]. However, our methods demonstrate better scaling and, therefore, can enable an improvement over direct methods for large problem sizes, so a specific parallel implementation to solve problems of huge scale is a key avenue of future work.

5.6.1 Scenario Tree NMPC

The first test problem stems from optimal control of systems under uncertainty. A well-established method for robustly tackling (deterministic) optimal control problems is *nonlinear model predictive control* (NMPC, see [151]). Uncertainties of systems can be modeled by scenario trees, which, when integrated with NMPC, leads to so-called *scenario tree NMPC* (see e.g., [152]). How scenario tree NMPC can be applied in practice is presented in [153]. The control problem is reduced to an optimization problem which leads to linear systems fitting into the framework discussed here. We consider an example based on [153], which models masses coupled through spring packets containing redundant arrays of springs, with each spring having a certain fault probability. The nonlinearity is handled by the sequential homotopy method, yielding a sequence of linear systems with tree-coupled structure, where we only consider the first system of the sequence.

We fix the time step for the scenario tree generation to 0.1. The underlying system contains 4 spring packets each of which contains 6 springs. The spring fault probability is set to 0.003. The scenario tree is chosen to model 15 time steps, corresponding to a tree of depth 15. An example resulting scenario tree is depicted in Figure 5.3. The overall system has 12 432 degrees of freedom and 76 nodes. The resulting linear system is then solved with GMRES equipped with the preconditioners introduced above. The iterative solver is run without restarts and a maximum of 100 iterations. To allow a better comparison of various preconditioners, we use right preconditioning. That is, to solve $\mathcal{A}x = b$, GMRES is applied to the system

$$\widetilde{\mathcal{A}}\widetilde{x} = b \quad \text{with} \quad \widetilde{\mathcal{A}} = \mathcal{A}\mathcal{P}^{-T} \text{ and } x = \mathcal{P}^{-T}\widetilde{x}.$$

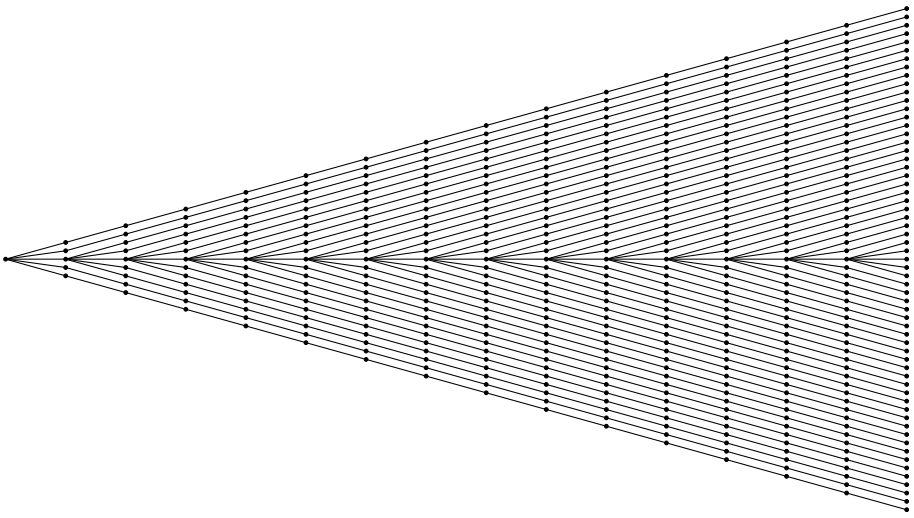
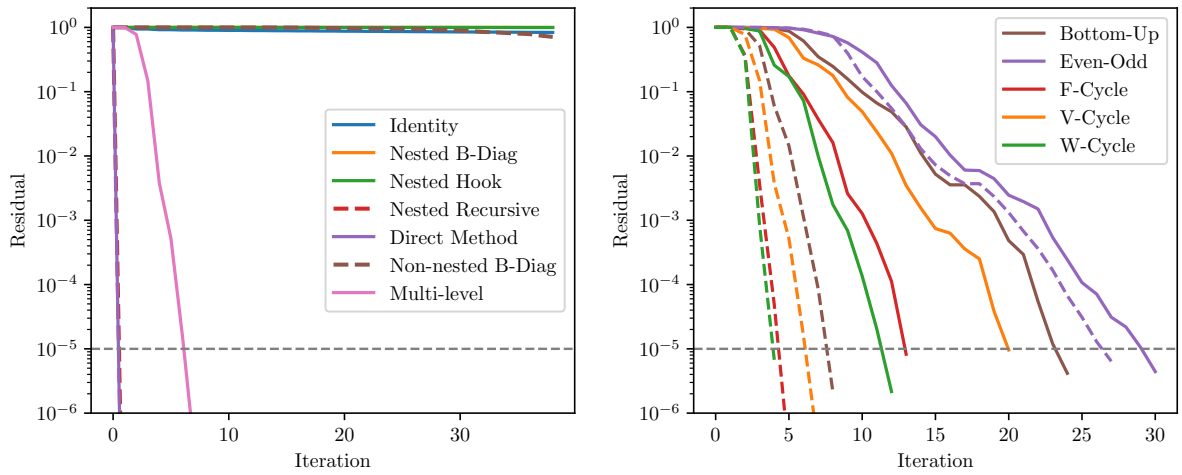


Figure 5.3: Example tree resulting from the application of the scenario tree NMPC approach [153] to a problem of connected spring packets. Each level corresponds to a decision point of the scenarios. Each path from the root to a leaf represents a single scenario. The preconditioners are based on the topology of this scenario tree.



(a) All preconditioners **(b)** ML preconditioners

Figure 5.4: Convergence for the scenario tree NMPC problem with different preconditioners for a tree depth of 15. The plots show the relative residual of the solution at each GMRES iteration: all preconditioners with one representative ML preconditioner (left), various ML preconditioners (right). The right plot shows the convergence for the block-diagonal (solid lines) and the super-node smoother (dashed lines). The direct method, nested recursive, and ML preconditioners show fast convergence, while the others do not provide a significant improvement over the unpreconditioned method. The residual lines of the nested hook and block-diagonal preconditioners overlap.

It is noted that the matrices $\mathcal{P}^{-1}\mathcal{A}$ and $\mathcal{A}\mathcal{P}^{-T}$ are spectrally equivalent and possess the same minimal polynomial. Thus, the above theoretical results also apply to right preconditioning with the transpose. Figure 5.4 shows the number of GMRES iterations for different preconditioners and the 2-norm of the associated relative residuals

$$r_{r,k} \coloneqq \frac{r_k}{\|r_0\|},$$

where r_k denotes the absolute residual at the kth iteration. We initialize the iteration with an initial guess of $x_0 = 0$, yielding an initial residual of $r_0 = b$. As found in experiments, different settings for the ML preconditioner lead to similar convergence results. Thus, Figure 5.4a only shows one representative for the ML preconditioner (the V-cycle with super-node smoother introduced in Section 5.5.3), while Figure 5.4b provides a detailed insight into the behavior of different settings of the ML method.

As expected, the direct method and the recursive preconditioners lead to convergence within one GMRES iteration. This aligns with the results of Lemma 5.4 and Corollary 5.2. The nested block-diagonal preconditioner does not result in a significant reduction in the iteration numbers. The nested hook preconditioner shows the same convergence as the nested-block-diagonal preconditioner and was found to be effective only for shallow trees. The discussion of the non-nested block-diagonal preconditioner refers to (5.9), and the experiment suggests that the preconditioner does not resemble the system well enough to serve as an effective preconditioner. Finally, we have the ML preconditioner, giving a significant reduction in the number of iterations. Figure 5.4b depicts variations of the ML approach (indicated by color) paired with either the block-diagonal smoother (solid lines) or the super-node smoother (dashed lines). The super-node smoother performs better than the block-diagonal smoother in general.

When increasing the tree depth to 20, most preconditioners failed due to excessive runtime or the exhaustion of the maximum number of iterations. On the one hand, this failure can be explained by the increased runtime for each application of some of the preconditioners due to their exponentially increasing complexity with respect to the tree depth. While the nested block-diagonal, the nested hook, and the non-nested block-diagonal preconditioners do not have such runtime scaling, these do not provide a sufficient performance in order to converge within the prescribed maximum number of iterations. The ML preconditioner using a V-cycle, however, still showed good convergence behavior, yielding convergence within 9 iterations in the case of the super-node smoother and within 26 iterations in the case of the block-diagonal smoother.

If the preconditioned system matrix is not too far from normality, the convergence characteristics of GMRES are primarily influenced by the eigenvalue distribution of the preconditioned system matrix. Figure 5.5 shows the absolute values of the eigenvalues of the preconditioned matrices, specifically $|\lambda(\mathcal{A}\mathcal{P}^{-T})|$. To facilitate the eigenvalue computation, a tree depth of 5 is selected with 2 spring packets, each containing 3 springs. For improved presentation, the plot does not show the eigenvalues of the direct method and the nested recursive preconditioner since these map all eigenvalues to 1 as they function as exact solvers. The distributions of the eigenvalues align with

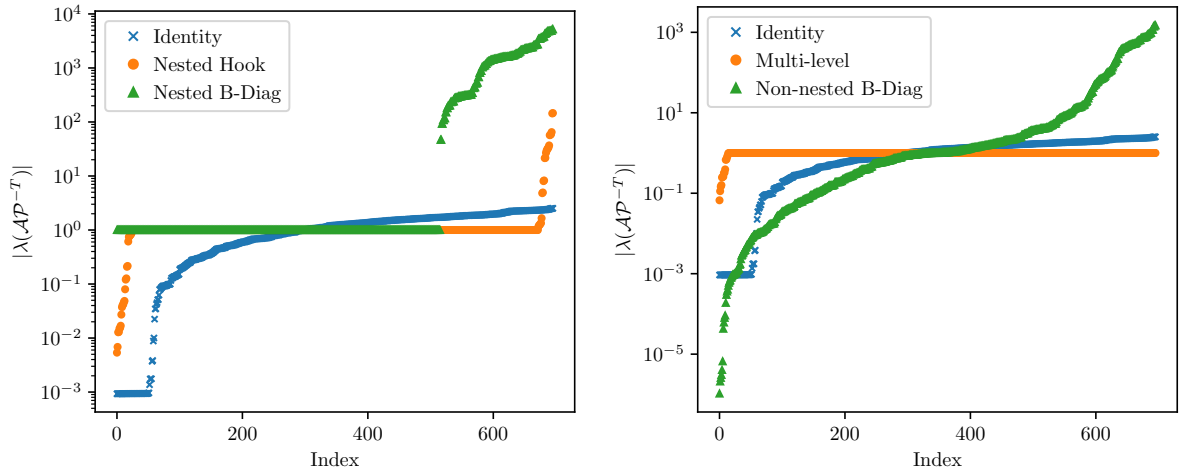


Figure 5.5: Eigenvalues of preconditioned matrices for different preconditioners. The example used is the scenario tree NMPC problem for tree depth of 5 with 2 spring packets, each containing 3 springs. The eigenvalue distributions reflect the observed convergence behavior.

the observed convergence behavior. In contrast, the hook preconditioner and the two block-diagonal preconditioners barely show any improvement, exhibiting a significant scattering of several eigenvalues. The good performance of the multi-level method is reflected in the eigenvalue distribution, with the majority of eigenvalues being 1 and only a few scattered not far from it.

5.6.2 Multiple Shooting for Optimal Control

We consider the solution of optimal control problems with ordinary differential equations, more specifically, problems of the form

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2}\int_0^T \|y(t) - y_d(t)\|^2 \,\mathrm{d}t + \frac{\beta}{2}\int_0^T \|u(t) - \bar{u}\|^2 \,\mathrm{d}t, \\ \text{s.t.} \quad & \dot{y}(t) = f(y(t), u(t)), \\ & y(0) = y_0. \end{aligned}$$

When considering large time horizons, such problems can lead to numerically challenging problems since the optimality conditions lead to equations both forward and backward in time. Parallel-in-time methods (see e.g., [54]) tackle this problem by allowing parallelization along the time axis. Among various types of such methods, we focus on a *multiple shooting* method [119, 154]. The time interval $[0, T]$ is separated into smaller intervals $[t_k, t_{k+1}]$ and, where necessary, appropriate matching constraints are enforced at the interfaces. This results in a series of subsystems, each of which can be considered independently. The subintervals can be arranged into sparsely coupled trees (see Figure 5.6). This tree structure is not just restricted to binary trees but can be generalized to arbitrary numbers of children.

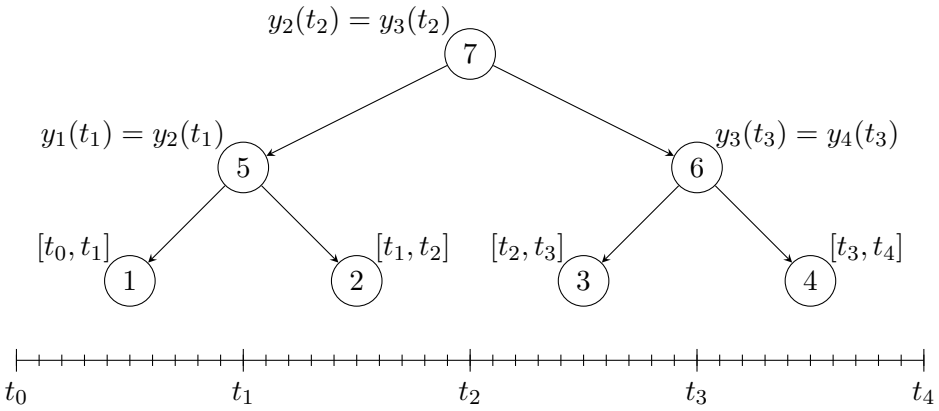
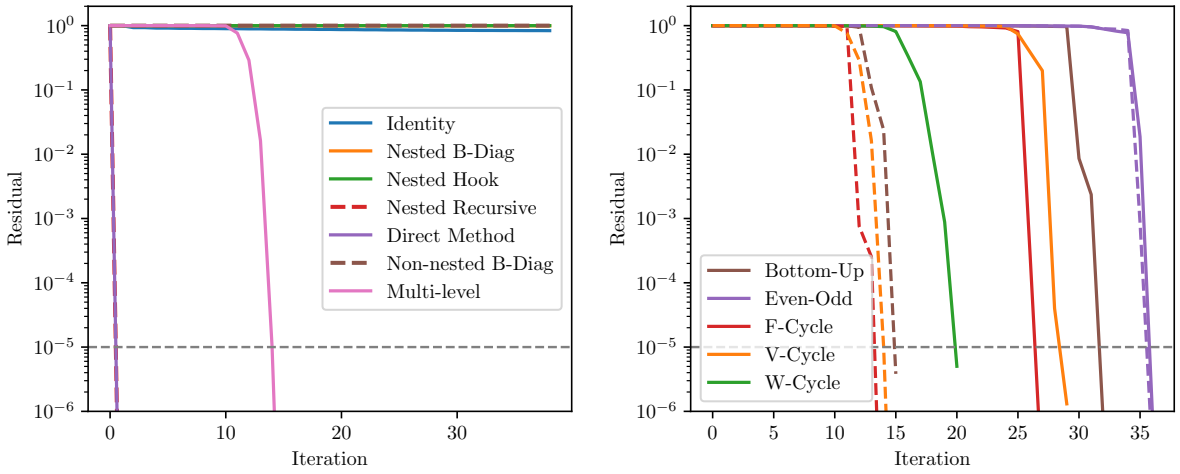
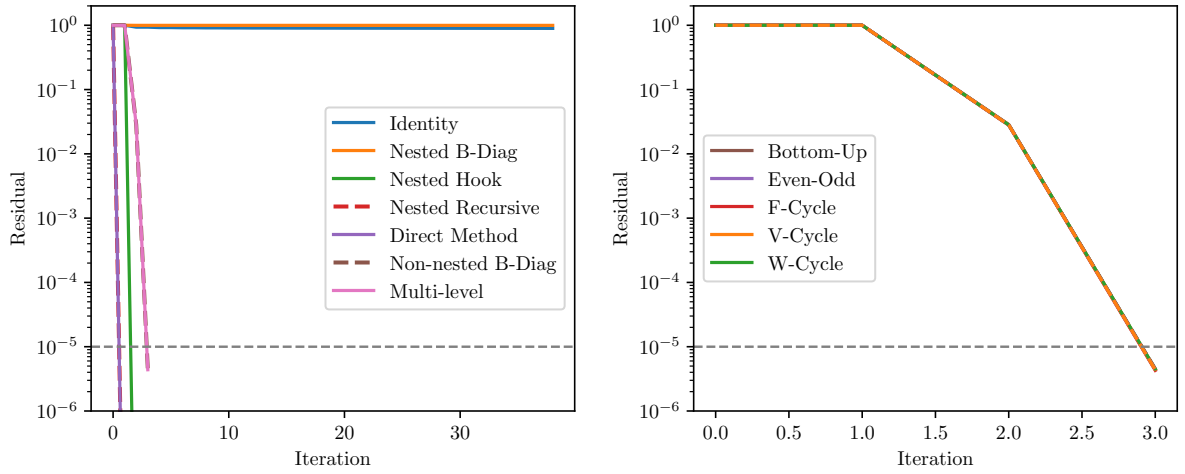


Figure 5.6: Rearranging matching constraints in multiple shooting.



(a) All preconditioners **(b)** ML preconditioners

Figure 5.7: Convergence for the multiple shooting problem with a binary tree. The plots follow the same structure as in Figure 5.4. The direct method, nested recursive, and ML preconditioners admit fast convergence, while the others do not provide a significant improvement over the unpreconditioned method. The residual lines of the nested hook and block-diagonal preconditioners overlap.



(a) All preconditioners **(b)** ML preconditioners

Figure 5.8: Convergence for the multiple shooting problem with a shallow tree. The plots follow the same structure as in Figure 5.4. All preconditioners admit fast convergence except for the nested block-diagonal preconditioner.

We look at a modified Lotka–Volterra problem

$$f(y,u) = \begin{pmatrix} y_1 - y_1 y_2 - c_1 y_1 u \\ -y_2 + y_1 y_2 - c_2 y_2 u \end{pmatrix},$$

with $y_d = (1,1)^T$, $\bar{u} = 0.5$, $T = 12$, $c_1 = 0.4$, $c_2 = 0.2$, $\beta = 0.1$, and the initial state $y_0 = (0.5, 0.7)^T$. First, the time horizon is separated into 8 subintervals and arranged in a binary tree with depth 4. Each of the subintervals is discretized with 20 timesteps. The nonlinear problem is linearized by the sequential homotopy method, where the resulting linear system is of size 12 166. Figure 5.7 shows the performance of the preconditioners for this setting. We can, in fact, observe similar behavior of the preconditioners as in the scenario tree NMPC problem except for the ML preconditioner. The first few iterations of the ML preconditioners do not lead to a significant improvement in the residual, but it eventually achieves rapid convergence.

In a second setting, we consider a shallow tree with the root node having 64 children. The resulting linear system is of size 47 242. The results are depicted in Figure 5.8. In contrast to the preceding experiments, we can observe that the hook preconditioner convergences within two iterations, which aligns with Corollary 5.1. All ML preconditioners are found to converge within three GMRES iterations. The non-nested block-diagonal preconditioner shows the same behavior as the V-cycle. The remaining convergence results overlap with what we have seen in the previous experiments.

Comparison with Condensing

We now compare the iterative solvers we have derived to an established problem-specific method, called *condensing* (cf. [154] and [155, pp. 560 ff.]). The condensing approach recursively constructs an explicit expression for the state as a function of the control, making use of the block-sparse structure of the underlying matrices. This way, the state variable is eliminated in the optimal control problem, and the problem is cast as an equivalent unconstrained quadratic program. While this is usually carried out on the shooting grid, it is more reasonable here to perform the condensing on the (much finer) time discretization grid due to the discretization of the control, which aligns with the time grid and not with the multiple shooting grid. The method is known to scale quadratically in the number of timesteps for constructing the explicit form of the state. Even though the resulting quadratic program is dense and the runtime of its solution would scale cubically, it is described in [155] how a Cholesky decomposition of the resulting linear system can be constructed with linear runtime scaling, reducing the method's overall runtime behavior to quadratic scaling. In the following analysis, the efficient computation of the Cholesky decomposition is not implemented. Since the runtime of condensing is dominated by the construction of the necessary dense matrices anyway, it is sufficient to measure the runtime of this step only to get an insight into how the methods compare.

Figure 5.9 compares the runtimes of condensing and the multiple shooting approach coupled with our ML-preconditioned iterative solver. The ML preconditioner is configured to use a V-cycle with the block-diagonal smoother. The domain is separated into sub-intervals for multiple shooting, each containing 40 equidistant forward Euler timesteps, and is arranged into a shallow tree. The horizontal axis shows the number of timesteps, and the vertical axis denotes the runtime in seconds for assembling the necessary matrices and solving the system (except for the condensing method). The dashed lines represent linear and quadratic scaling. The condensing approach has a quadratic scaling, consistent with the theoretical results. In contrast, the preconditioned iterative method exhibits a linear scaling and outperforms condensing for larger problem sizes. Additionally, the ML approach has the potential for parallelization, which would result in an even greater speedup. The same asymptotic scaling would be expected if the condensing was carried out on the coarse shooting grid.

5.6.3 Domain Decomposition for PDEs

The tree-sparse approach to multiple shooting in Section 5.6.2 can be transferred to domain decomposition methods for elliptic PDEs. Given a PDE on a domain Ω, we can dissect Ω into an arbitrary number of subdomains. This allows us to solve the PDE on each subdomain separately while enforcing consensus constraints along the interfaces, also known as *non-overlapping domain decomposition* [118, pp. 74 ff.]. State-of-the-art domain decompositions methods for elliptic PDEs are, e.g., as Finite Element Tearing and Interfacing (FETI) methods [156, 157]. Since these

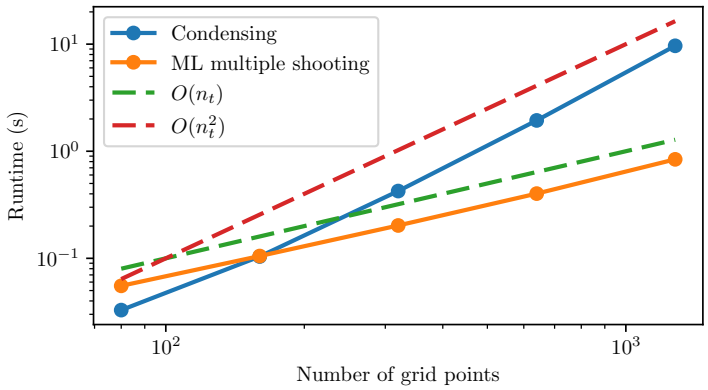
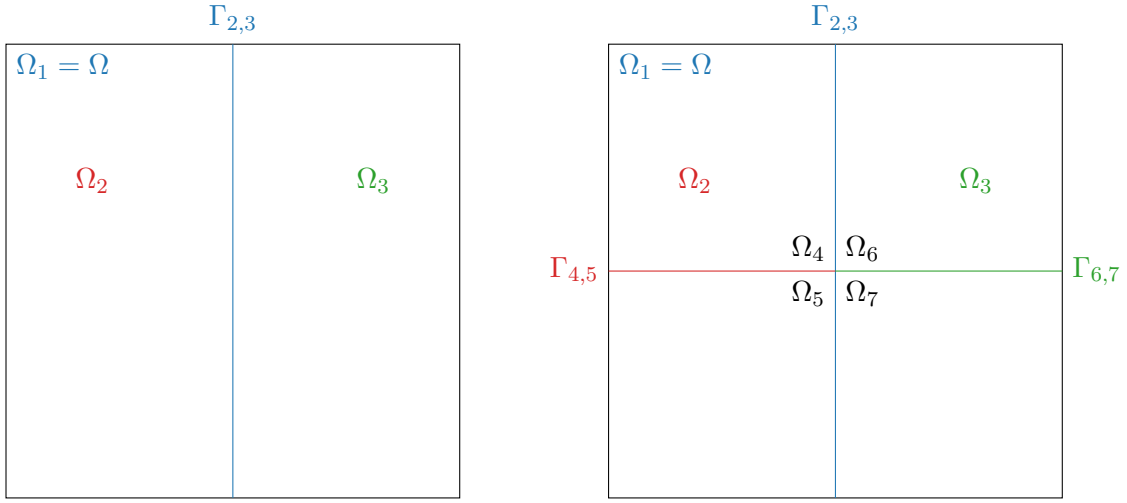
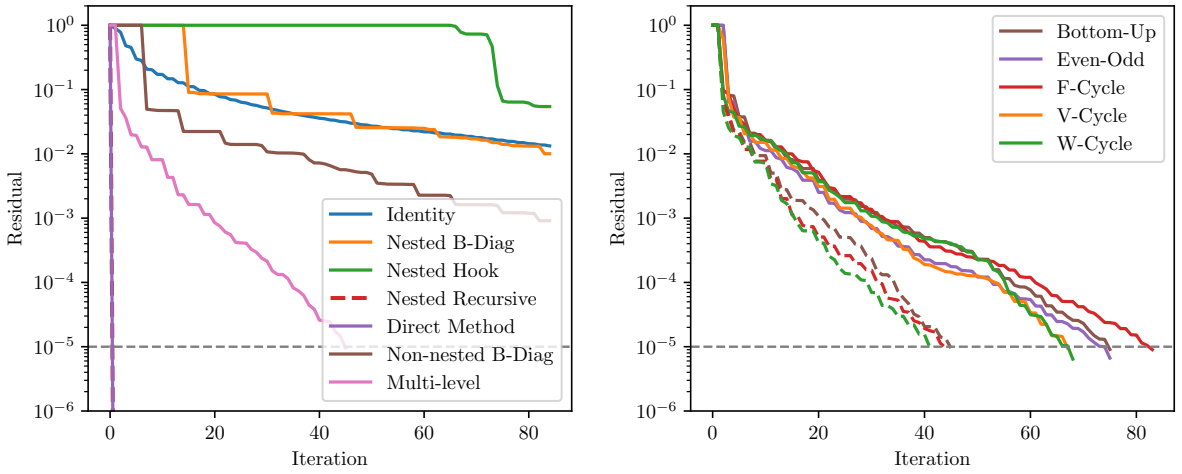


Figure 5.9: Runtime comparison of condensing and the ML preconditioner for solving the multiple shooting problem for various numbers of timesteps n_t. The linear runtime behavior of the ML-preconditioned solver provides better scaling than the quadratic scaling of the condensing approach.



(a) Dissection of Ω **(b)** Dissection of subdomains Ω_2 and Ω_3

Figure 5.10: Construction of subdomains by recursive dissection of the domain Ω. The dissecting lines of two domains Ω_i and Ω_j are labeled by $\Gamma_{i,j}$, along which consensus constraints are enforced.



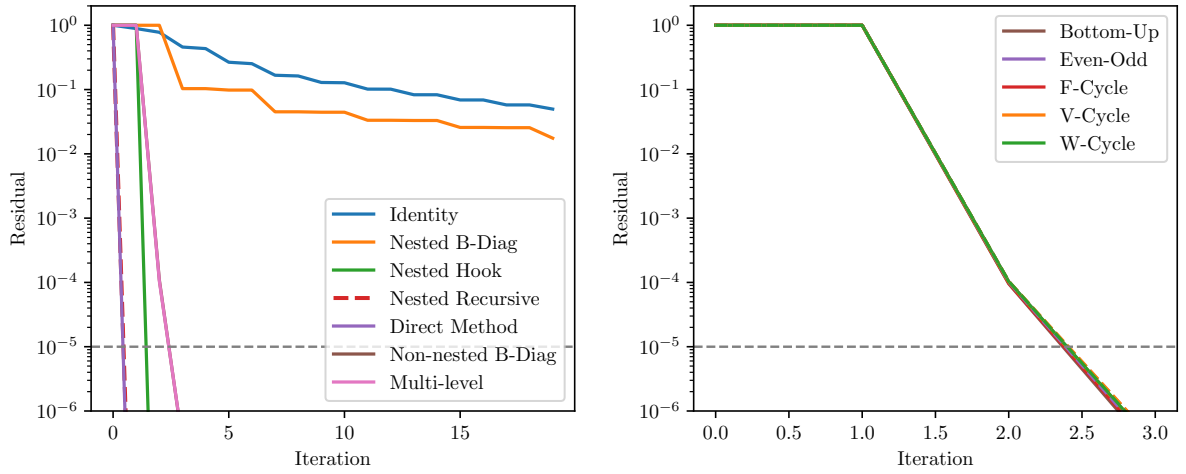
(a) All preconditioners **(b)** ML preconditioners

Figure 5.11: Convergence for the domain decomposition problem arranged in a binary tree. The plots follow the same structure as in Figure 5.4. The direct method, nested recursive, and ML preconditioners show fast convergence. The non-nested block-diagonal preconditioner provides slightly improved convergence compared to the unpreconditioned method.

methods are specifically tailored for the solution of elliptic PDEs, we found that they clearly outperform our proposed approach. We have opted not to include any direct comparisons here, because our framework addresses a much larger problem class and must be expected to perform worse than methods targeted at narrower classes of problems.

To apply our framework, we recursively carry out non-overlapping domain decompositions, resulting in a hierarchical decomposition of the domain, which can then be arranged into a tree structure (see Figure 5.10). The novel aspect of this approach is that the interfaces are interconnected across a tree, resulting in a more structured Schur complement. While the primary challenge of non-overlapping domain decomposition lies in approximating the Schur complement, we assume that complete information is available and demonstrate that the tree-sparse ordering coupled with our methods still provides good performance. This encourages future work to seek more efficient approximations of the structured and simpler Schur complement and to make the approach practicable. For the experiment, we consider the mixed formulation of the Poisson problem (see e.g., [158]). The solution of the Poisson problem within this formulation can be viewed as an optimization problem, hence allowing us to fit these into the above framework. A more detailed discussion of the theoretical background can be found in Appendix B.2.

We discretize the solution with a non-uniform triangulation of the domain $\Omega = [0, 1]^2$ and a maximum cell size of $h = 0.0125$. The discretization conforms with the predefined interfaces. In our first setting, the domain is decomposed into eight subdomains, which are then arranged in a binary tree of depth four. The resulting linear system is of size 223 365. The convergence of the preconditioners is shown in Figure 5.11 following the same structure as before. The direct method and the recursive preconditioners converge within one iteration. The hook preconditioner



(a) All preconditioners **(b)** ML preconditioners

Figure 5.12: Convergence for the domain decomposition problem arranged in a tree with each inner node having 8 children. The plots follow the same structure as in Figure 5.4. All preconditioners admit fast convergence except for the nested block-diagonal preconditioner.

is again found to be ineffective for deep trees. The nested block-diagonal preconditioner does not result in improved convergence when compared to the identity preconditioner. As opposed to the previous experiments, the non-nested block-diagonal preconditioner leads to a reduction in iteration numbers, but is still outperformed by the ML preconditioners.

Similarly to Section 5.6.2, the tree can be arranged as a shallow tree with 8 leaves directly connected to the root. The size of the linear system is 220 601. In Figure 5.12, we see the same behavior as in the case of the shallow tree in the multiple shooting test problem.

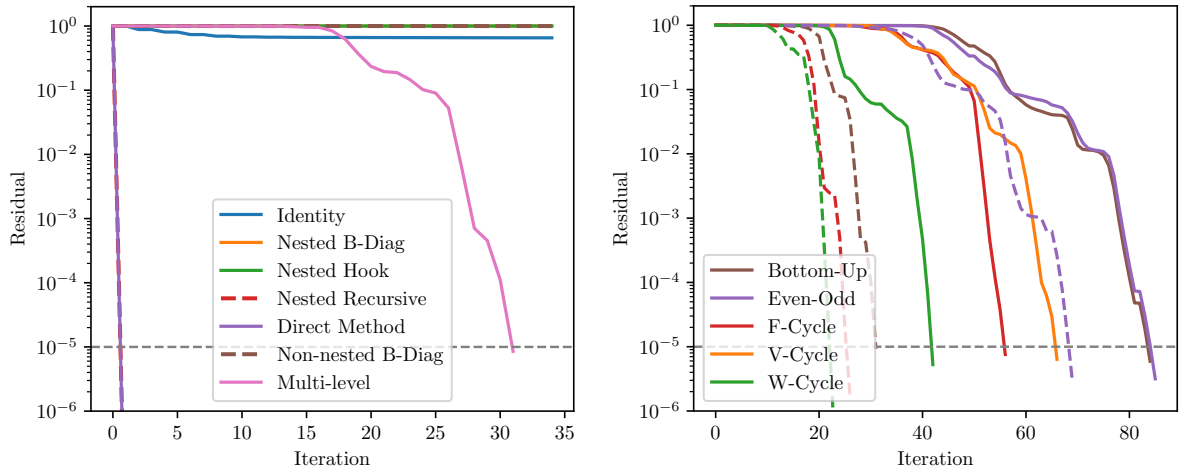
This approach can be extended to PDE-constrained optimization problems. For an overview of such problems, the reader is referred to [21]. We consider a Poisson control problem with a distributed control, i.e.,

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2}\int_\Omega |y(x) - y_d(x)|^2 \,\mathrm{d}x + \frac{\beta}{2}\int_\Omega u(x)^2 \,\mathrm{d}x, \\ \text{s.t.} \quad & -\Delta y(x) = u(x) \quad \text{in } \Omega, \\ & y(x) = 0 \qquad \text{on } \partial\Omega, \end{aligned}$$

where u denotes a distributed control and $\beta > 0$ is a regularization parameter. The domain decomposition is carried out the same way as for the Poisson problem.

Again, we use a triangulation of the domain $\Omega = [0,1]^2$ with a maximum cell size of 0.0125, and the regularization parameter is set to $\beta = 10^{-4}$. The desired state y_d is set to

$$y_d(x) = \sin(\pi x_1)\sin(\pi x_2).$$



(a) All preconditioners **(b)** ML preconditioners

Figure 5.13: Convergence for the domain decomposition problem for a PDE-constrained optimization problem arranged in a binary tree. The plots follow the same structure as in Figure 5.4. The direct method, nested recursive, and ML preconditioners show fast convergence. The residual lines of the nested hook and block-diagonal preconditioners overlap.

If we arrange the domain into a binary tree of depth four, we obtain the convergence behavior depicted in Figure 5.13. The linear system is of size 446 730. The convergence behavior aligns with what has been observed in the previous examples with deep trees. For the ML preconditioners, the residual first remains on a plateau, but eventually achieves rapid convergence.

We note that, unlike the previous experiments, the Schur complements for the PDE problems remain relatively small in size but are still too large to construct explicitly in practice. For such PDE-related problems, inexact Schur complement methods have been widely studied in the literature, see e.g., [159, 160]. While most of our preconditioners rely on an explicit Schur complement, the proposed multi-level approaches provide a way to circumvent its construction. In particular, one only needs to apply the smoother $\mathcal{S}_{\text{diag}}^{-1}$, which involves solving smaller sub-blocks rather than the full Schur complement. This substantially reduces computational cost and enables more efficient approximation strategies. Therefore, in these scenarios, multi-level preconditioners can effectively reduce the complexity associated with Schur complements and can support the use of inexact methods in a scalable manner. A detailed analysis of these approaches is left to future research.

Table 5.2: Number of solved subsystems involving B_i when solving the scenario tree NMPC problem for different tree depths and preconditioners.

Preconditioner			Tree Depth and #DOFs		
			5	10	15
			1 972	6 102	12 432
Direct Method			1 024	23 198	721 916
Nested Recursive			1 024	23 198	721 916
ML	V-Cycle	Block-diagonal	924	2 028	2 420
		Super-Node	508	1 059	1 508
	W-Cycle	Block-Diagonal	768	1 926	2 192
		Super-Node	456	957	1 432
	F-Cycle	Block-Diagonal	690	1 722	1 964
		Super-Node	456	906	1 356
	Bottom-Up	Block-Diagonal	1 002	2 436	2 800
		Super-Node	508	1 161	1 584
	Even-Odd	Block-Diagonal	1 184	3 099	3 256
		Super-Node	1 028	2 997	3 028

5.6.4 Computational Cost

Since the computational complexity varies among the preconditioners, the performance of a preconditioner is not determined by its iteration numbers alone. Hence, we analyze the running time of the preconditioners in this section. It is important to note that in these experiments no potential for parallelizability has been exploited as this is subject to future work. Additionally, the focus of the implementation is on the qualitative performance of the preconditioners rather than optimizing for computational efficiency. In order to provide insights into the running time behavior, the number of solved subsystems involving B_i (including the setup of the preconditioner) is considered as a measure of running time, as the overall computational burden is dominated by this. All subsystems are of the same size. We consider the scenario tree NMPC problem from Section 5.6.1 for the tree depths 5, 10, and 15. Table 5.2 shows the results for preconditioners that converged in under 100 iterations, and Figure 5.14 visualizes the same data. The direct method and nested recursive preconditioner have the same number of solved subsystems. Even though these preconditioners lead to convergence in one iteration, the computational complexity grows exponentially with the tree depth, resulting in 721 916 solved subsystems. On the other hand, the ML preconditioners are more robust with respect to the tree depth, only requiring between 1 356 and 3 256 subsystems to be solved for the deepest tree. Thus, the ML preconditioners are more suitable for large-scale problems, while for shallow trees the difference between the preconditioners becomes less significant. We have observed that these results transfer to the remaining numerical examples. In general, we observe the expected polynomial and exponential runtime scaling with respect to the tree depth for the respective preconditioners.

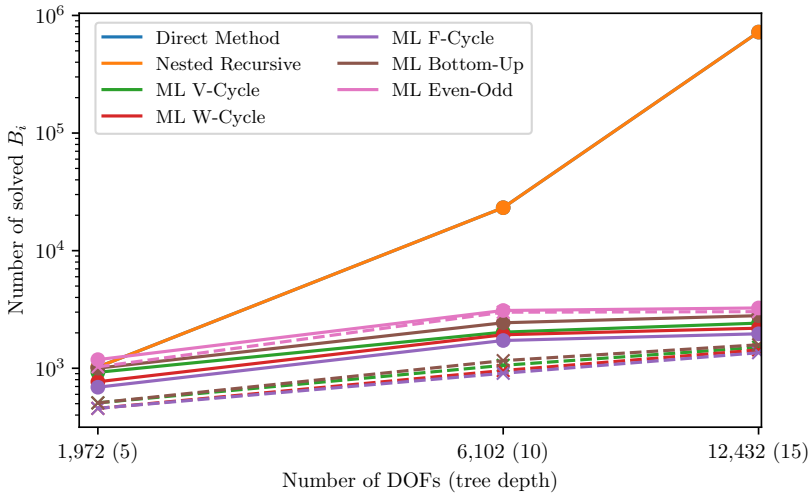


Figure 5.14: Number of solved subsystems involving B_i as a function of the number of DOFs when solving the scenario tree NMPC problem for different tree depths and preconditioners. Solid and dashed lines correspond to the block-diagonal and super-node smoothers, respectively.

5.7 Conclusion

We proposed and examined several algorithmic approaches to solve saddle-point systems with a tree-based block structure. Apart from the direct method, these approaches are based on preconditioned iterative linear solvers. Several of the problem-specific preconditioners have notable theoretical properties warranting their utilization in large-scale problem instances. This holds in particular for the ML methods obtained by applying multigrid approaches to tree-coupled systems, where much of the theory of multigrid methods carries over. The corresponding algorithms work very well in our numerical experiments, with the more accurate super-node smoothing combined with a V- or F-cycle iteration being particularly efficient.

Valuable future work would include the study of the effects of parallelization, the analysis of coupled systems with an even more general graph-based structure including cyclic dependencies, and the derivation of methods to automatically detect this exploitable structure within given linear systems.

Appendix A

Technical Proofs for Carleman Linearization of Parabolic PDEs

This appendix contains technical proofs of results presented in Chapter 4.

A.1 Proof of Lemma 4.3

Before we proceed to the proof of Lemma 4.3, we introduce a few results on B. Most of these are adaptations of results proved in [94] with a focus on the explicit dependence of the involved constants on T. In the following, we adopt the notation $B(x, y)$ in place of $B(x \otimes y)$ to emphasize the bilinear structure of the operator.

Lemma A.1.1. *Let $T \in (0, \infty]$. For all $z \in W(0, T; V, V')$, it holds that*

$$\|z\|_{L^\infty(0,T;H)} \leq \|z(0)\|_H + \|z\|_{W(0,T;V,V')}.$$

Proof. Let $z \in W(0, T; V, V')$. Then, we obtain the following upper bound:

$$\begin{aligned}\|z(t)\|_H^2 &= \|z(0)\|_H^2 + 2\int_0^t \langle z'(s), z(s)\rangle_V \mathrm{d}s \leq \|z(0)\|_H^2 + 2\int_0^T \left|\langle z'(s), z(s)\rangle_V\right| \mathrm{d}s \\ &\leq \|z(0)\|_H^2 + 2\|z'\|_{L^2(0,T;V')}\|z\|_{L^2(0,T;V)} \leq \|z(0)\|_H^2 + \|z'\|_{L^2(0,T;V')}^2 + \|z\|_{L^2(0,T;V)}^2\end{aligned}$$

for $t \in [0, T)$, cf. Lemma III.1.2 in [1]. □

Lemma A.1.2 (Lemma 2 in [94])**.** *Let $T \in (0, \infty]$ and B fulfill* (A3)*. Then, for all $y, z, w \in W(0, T; V, V')$, it holds that*

$$\begin{aligned}&\left|\langle B(y, z), w\rangle_{L^2(0,T;V'),L^2(0,T;V)}\right| \\ &\quad \leq c_B \|y\|_{L^\infty(0,T;H)}^{\frac{1}{2}} \|y\|_{L^2(0,T;V)}^{\frac{1}{2}} \|z\|_{L^\infty(0,T;H)}^{\frac{1}{2}} \|z\|_{L^2(0,T;V)}^{\frac{1}{2}} \|w\|_{L^2(0,T;V)}.\end{aligned}$$

Lemma A.1.3 (Refinement of Corollary 3 in [94])**.** *Let $T \in (0, \infty]$ and B fulfill* (A3)*. For all $y, z \in W(0, T; V, V')$, it holds that*

$$\|B(y, z)\|_{L^2(0,T;V')} \leq c_B(\|y(0)\|_H + \|y\|_{W(0,T;V,V')})(\|z(0)\|_H + \|z\|_{W(0,T;V,V')}).$$

Proof. The result follows from Lemma A.1.1 and Lemma A.1.2. □

Lemma A.1.4 (Refinement of Lemma 4 in [94])**.** *Let $T \in (0,\infty]$ and B fulfill* (A3)*. For all $\delta \in [0,1]$ and for all $y,z \in W(0,T;V,V')$ with $\|y(0)\|_H + \|y\|_{W(0,T;V,V')} \leq \delta$ and $\|z(0)\|_H + \|z\|_{W(0,T;V,V')} \leq \delta$, it holds that*

$$\|B(y,y) - B(z,z)\|_{L^2(0,T;V')} \leq 2\delta c_B \left(\|y(0)-z(0)\|_H + \|y-z\|_{W(0,T;V,V')}\right).$$

Proof. Let $y,z \in W(0,T;V,V')$. With Lemma A.1.3, it follows that

$$\begin{aligned}\|B(y,y) - B(z,z)\|_{L^2(0,T;V')} &\leq \|B(y,y-z)\|_{L^2(0,T;V')} + \|B(y-z,z)\|_{L^2(0,T;V')} \\ &\leq 2\delta c_B \left(\|y(0)-z(0)\|_H + \|y-z\|_{W(0,T;V,V')}\right).\end{aligned}$$ □

This enables us to prove the well-posedness of the nonlinear Cauchy problem.

Proof of Lemma 4.3. Let $y_0 \in H$ and $g \in L^2(0,T;V')$. Then, the system

$$\begin{aligned} z'(t) + Az(t) &= g(t) \quad \text{in } L^2(0,T;V'), \\ z(0) &= y_0 \end{aligned} \tag{A.1}$$

has a unique solution $z \in W(0,T;V,V')$ with $\|z(0)\|_H + \|z\|_{W(0,T;V,V')} \leq c_L \exp(\lambda T)(\|y_0\|_H + \|g\|_{L^2(0,T;V')})$ with the constant $c_L \geq 1$ from Lemma 4.1. Let $\mu := \|y_0\|_H + \|f\|_{L^2(0,T;V')}$. We set $c_N = \max(1/(4c_B \exp(\lambda T)), c_L) \geq 1$. Define the set

$$M = \{y \in W(0,T;V,V') \mid \|y(0)\|_H + \|y\|_{W(0,T;V,V')} \leq 2c_N \exp(\lambda T)\mu, \quad y(0) = y_0\}.$$

Due to $c_N \geq c_L$ and estimate (4.4), the solution to (A.1) with $g = f$ belongs to M. Thus, M is not empty. Due to the continuous embedding $C(0,T;H) \hookrightarrow W(0,T;V,V')$, the set is closed under the $W(0,T;V,V')$–norm. Define the map $Z : M \to W(0,T;V,V')$, where $z = Z(y)$ maps the function y to the solution of the system

$$\begin{aligned} z'(t) + Az(t) + B(y(t),y(t)) &= f(t), \\ z(0) &= y_0. \end{aligned}$$

Since $c_N \geq c_L$ and by using Lemma A.1.4 with $\delta = 2c_N \exp(\lambda T)\mu \leq \frac{1}{4c_N \exp(\lambda T) c_B} \leq 1$ and one argument being zero, we obtain

$$\begin{aligned}\|z(0)\|_H + \|z\|_{W(0,T;V,V')} &\leq c_N \exp(\lambda T)\left(\|y_0\|_H + \|B(y,y)\|_{L^2(0,T;V')} + \|f\|_{L^2(0,T,V')}\right) \\ &\leq c_N \exp(\lambda T)\left(\mu + 2\delta c_B \left(\|y_0\| + \|y\|_{W(0,T;V,V')}\right)\right) \\ &\leq c_N \exp(\lambda T)\left(\mu + \frac{1}{2c_N \exp(\lambda T) c_B} 2c_B c_N \exp(\lambda T)\mu\right) \\ &= 2c_N \exp(\lambda T)\mu.\end{aligned}$$

Hence, $Z(M) \subseteq M$. Next, we show that Z is a contraction. For $y_1, y_2 \in M$, let $z = Z(y_1) - Z(y_2)$, which solves

$$\begin{aligned} z'(t) + Az(t) + B(y_1(t), y_1(t)) - B(y_2(t), y_2(t)) = 0 \quad \text{in } L^2(0,T;V'), \\ z(0) = 0. \end{aligned}$$

From Lemma A.1.4, we obtain that

$$\begin{aligned} \|Z(y_1) - Z(y_2)\|_{W(0,T;V,V')} = \|z\|_{W(0,T;V,V')} &\leq c_N \exp(\lambda T) \left(\|B(y_1, y_1) - B(y_2, y_2)\|_{L^2(0,T;V')} \right) \\ &\leq c_N \exp(\lambda T) 2\delta c_B \|y_1 - y_2\|_{W(0,T;V,V')} \leq \frac{1}{2} \|y_1 - y_2\|_{W(0,T;V,V')}. \end{aligned}$$

Due to Banach's fixed-point theorem, there is a unique solution $y \in M$ to $Z(y) = y$, which proves the existence of a solution to the nonlinear Cauchy problem.

The uniqueness of the solution in $W(0,T;V,V')$ can be proven the same way as in [94]. □

A.2 Proofs of Properties of A_k, B_k, and F_k

For the following proofs, we introduce the notation $\vec{j}^{(l)} := (\vec{j}_1, \dots, \vec{j}_{l-1}, \vec{j}_{l+1}, \dots, \vec{j}_k) \in \mathbb{N}^{k-1}$ for $k \geq 2$ and $\vec{j}^{(l;p_1,\dots,p_m)} := (\vec{j}_1, \dots, \vec{j}_{l-1}, p_1, \dots, p_m, \vec{j}_{l+1}, \dots, \vec{j}_k) \in \mathbb{N}^{k+m-1}$ for $k \geq 1$, where $\vec{j} \in \mathbb{N}^k$ is a multi-index with $k \in \mathbb{N}$, and $p_n \in \mathbb{N}$ for $n \in \{1, \dots, m\}$ and $m \in \mathbb{N}$. Furthermore, the notation $\sum_{\vec{j}^{(l)} \in \mathbb{N}^{k-1}}$ for $l \in \{1, \dots, k\}$ translates to the sum over $\mathbb{N}^{k-1}$ with the indices $\vec{j}^{(l)} = (\vec{j}_1, \dots, \vec{j}_{l-1}, \vec{j}_{l+1}, \dots, \vec{j}_{k-1})$. The multi-index $\vec{j}^{(l;p_1,\dots,p_m)}$ is defined as above in such case. We will use $\langle \cdot, \cdot \rangle$ to denote the duality mappings $\langle \cdot, \cdot \rangle_V$ and $\langle \cdot, \cdot \rangle_{V_1^0(k)}$, where the respective spaces are to be inferred from the context.

Proof of Lemma 4.4. First, we show the operator's boundedness. Let $u, v \in V_1^0(k)$ be arbitrary but fixed. It holds that

$$A_k v = \sum_{\vec{i} \in \mathbb{N}^k} \langle A_k v, \varphi_{\vec{i}} \rangle \varphi_{\vec{i}}.$$

The functions u and v admit the representation $u = \sum_{\vec{i} \in \mathbb{N}^k} \hat{u}(\vec{i}) \varphi_{\vec{i}}$ with $\hat{u}(\vec{i}) = \langle u, \varphi_{\vec{i}} \rangle$, and a similar expression for v. It holds that

$$\begin{aligned} \langle A_k u, v \rangle &= \sum_{l=1}^{k} \sum_{\vec{i} \in \mathbb{N}^k} \sum_{\vec{j} \in \mathbb{N}^k} \hat{u}(\vec{i}) \hat{v}(\vec{j}) \left\langle \left(\bigotimes_{m=1}^{l-1} \varphi_{\vec{i}_m} \right) \otimes A \varphi_{\vec{i}_l} \otimes \left(\bigotimes_{m=1}^{k-l} \varphi_{\vec{i}_m} \right), \varphi_{\vec{j}} \right\rangle \\ &= \sum_{l=1}^{k} \sum_{\vec{i}^{(l)} \in \mathbb{N}^{k-1}} \sum_{p,r \in \mathbb{N}} \hat{u}(\vec{i}^{(l;p)}) \hat{v}(\vec{i}^{(l;r)}) \langle A \varphi_p, \varphi_r \rangle. \end{aligned}$$

Define the functions $g_{\vec{i},l} = \sum_{s\in\mathbb{N}} \hat{u}(\vec{i}^{(l;s)})\varphi_s \in V$ and $w_{\vec{i},l} = \sum_{s\in\mathbb{N}} \hat{v}(\vec{i}^{(l;s)})\varphi_s \in V$. Then, by leveraging the Cauchy–Schwarz inequality, we obtain that

$$\begin{aligned}
\langle A_k u, v\rangle &= \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \langle A g_{\vec{i},l}, w_{\vec{i},l}\rangle \underset{(\mathrm{A1})}{\leq} \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \beta \|g_{\vec{i},l}\|_V \|w_{\vec{i},l}\|_V \\
&\leq \beta \left[\sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \|g_{\vec{i},l}\|_V^2\right]^{\frac{1}{2}} \left[\sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \|w_{\vec{i},l}\|_V^2\right]^{\frac{1}{2}} \\
&= \beta \left[\sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \sum_{s\in\mathbb{N}} \lambda_s \hat{u}(\vec{i}^{(l;s)})^2\right]^{\frac{1}{2}} \left[\sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \sum_{s\in\mathbb{N}} \lambda_s \hat{v}(\vec{i}^{(l;s)})^2\right]^{\frac{1}{2}} \\
&= \beta \left[\sum_{l=1}^{k} \sum_{\vec{i}\in\mathbb{N}^{k}} \lambda_{\vec{i}_l} \hat{u}(\vec{i})^2\right]^{\frac{1}{2}} \left[\sum_{l=1}^{k} \sum_{\vec{i}\in\mathbb{N}^{k}} \lambda_{\vec{i}_l} \hat{v}(\vec{i})^2\right]^{\frac{1}{2}} = \beta \|u\|_{V_1^0(k)} \|v\|_{V_1^0(k)},
\end{aligned}$$

which shows the boundedness of A_k.

Next, we show the coercivity of A_k by

$$\begin{aligned}
\langle A_k v, v\rangle &= \sum_{l=1}^{k} \sum_{\vec{i}\in\mathbb{N}^k} \sum_{\vec{j}\in\mathbb{N}^k} \hat{v}(\vec{i})\hat{v}(\vec{j}) \left\langle \left(\bigotimes_{m=1}^{l-1} \varphi_{\vec{i}_m}\right) \otimes A\varphi_{\vec{i}_l} \otimes \left(\bigotimes_{m=1}^{k-l} \varphi_{\vec{i}_m}\right), \varphi_{\vec{j}} \right\rangle \\
&= \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \sum_{p,r\in\mathbb{N}} \hat{v}(\vec{i}^{(l;p)})\hat{v}(\vec{i}^{(l;r)}) \langle A\varphi_p, \varphi_r\rangle = \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \langle A w_{\vec{i},l}, w_{\vec{i},l}\rangle \\
&\underset{(\mathrm{A2})}{\geq} \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \left(\gamma \|w_{\vec{i},l}\|_V^2 - \lambda \|w_{\vec{i},l}\|_H^2\right) \\
&= \gamma \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \sum_{s\in\mathbb{N}} \lambda_s \hat{v}(\vec{i}^{(l;s)})^2 - \lambda \sum_{l=1}^{k} \sum_{\vec{i}^{(l)}\in\mathbb{N}^{k-1}} \sum_{s\in\mathbb{N}} \hat{v}(\vec{i}^{(l;s)})^2 \\
&= \gamma \sum_{l=1}^{k} \sum_{\vec{i}\in\mathbb{N}^k} \lambda_{\vec{i}_l} \hat{v}(\vec{i})^2 - \lambda \sum_{l=1}^{k} \sum_{\vec{i}\in\mathbb{N}^k} \hat{v}(\vec{i})^2 = \gamma \|v\|_{V_0^1(k)}^2 - k\lambda \|v\|_{H(k)}^2.
\end{aligned}$$

□

Proof of Lemma 4.5. Let $B_m^{p,r} = \langle B(\varphi_p \otimes \varphi_r), \varphi_m\rangle$ for $p, r, m \in \mathbb{N}$. Any $v \in V_1^\alpha(k+1)$ can be represented as $v = \sum_{\vec{i}\in\mathbb{N}^{k+1}} \hat{v}(\vec{i})\varphi_{\vec{i}}$. Then,

$$B_k v = \sum_{\vec{j}\in\mathbb{N}^k} \langle B_k v, \varphi_{\vec{j}}\rangle \varphi_{\vec{j}}.$$

Moreover,

$$\langle B_k v, \varphi_{\vec{j}}\rangle = \sum_{l=1}^{k} \left\langle \left(\bigotimes I\right) \otimes B \otimes \left(\bigotimes I\right) v, \varphi_{\vec{j}} \right\rangle = \sum_{l=1}^{k} \sum_{p,r\in\mathbb{N}} B_{\vec{j}_l}^{p,r} \hat{v}(\vec{j}^{(l;p,r)}).$$

Using the Cauchy–Schwarz and Hölder inequalities, we can conclude that

$$\begin{aligned}
\|B_k v\|^2_{V^\alpha_{-1+\varepsilon}(k)} &= \sum_{\vec{j}\in\mathbb{N}^k} \pi(\lambda_{\vec{j}})^\alpha \sigma(\lambda_{\vec{j}})^{-1+\varepsilon} \left| \sum_{l=1}^k \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 \\
&\leq \sum_{\vec{j}\in\mathbb{N}^k} \pi(\lambda_{\vec{j}})^\alpha \underbrace{\sigma(\lambda_{\vec{j}})^{-1+\varepsilon} \left(\sum_{l=1}^k \lambda_{\vec{j}_l}^{1-\varepsilon} \right)}_{\leq k^\varepsilon} \sum_{l=1}^k \frac{1}{\lambda_{\vec{j}_l}^{1-\varepsilon}} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 \\
&\leq k^\varepsilon \sum_{\vec{j}\in\mathbb{N}^k} \pi(\lambda_{\vec{j}})^\alpha \sum_{l=1}^k \frac{1}{\lambda_{\vec{j}_l}^{1-\varepsilon}} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 .
\end{aligned}$$

Hölder's inequality was used with $p = 1/(1-\varepsilon)$ and $q = 1/\varepsilon$ for $\varepsilon \in [0,1)$. Then, $1/p + 1/q = 1$ and

$$\sum_{l=1}^k \lambda_{\vec{j}_l}^{1-\varepsilon} \leq \left(\sum_{l=1}^k \lambda_{\vec{j}_l}^{(1-\varepsilon)p} \right)^{\frac{1}{p}} \left(\sum_{l=1}^k 1 \right)^{\frac{1}{q}} = \left(\sum_{l=1}^k \lambda_{\vec{j}_l} \right)^{1-\varepsilon} k^\varepsilon.$$

The same estimate can be shown directly for $\varepsilon = 1$. For $\vec{j} \in \mathbb{N}^k$ fixed, let

$$w_{\vec{j},l} := \sum_{p,r\in\mathbb{N}} \hat{v}(\vec{j}^{(l;p,r)}) \varphi_p \otimes \varphi_r \in V_1^\alpha(2).$$

From the regularity of the operator B, we have that

$$\begin{aligned}
&\sum_{\vec{j}_l\in\mathbb{N}} \lambda_{\vec{j}_l}^{\alpha-1+\varepsilon} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 = \|B w_{\vec{j},l}\|^2_{V^{\alpha-1+\varepsilon}} \\
&\quad \underset{\text{(A4)}}{\leq} c_B(\alpha,\varepsilon) \|w_{\vec{j},l}\|^2_{V_1^\alpha(2)} = c_B(\alpha,\varepsilon) \sum_{p,r\in\mathbb{N}} \hat{v}(\vec{j}^{(l;p,r)})^2 \lambda_p^\alpha \lambda_r^\alpha (\lambda_p + \lambda_r).
\end{aligned}$$

Plugging this into the above estimate, we obtain

$$\begin{aligned}
&\sum_{\vec{j}\in\mathbb{N}^k} \pi(\lambda_{\vec{j}})^\alpha \sum_{l=1}^k \frac{1}{\lambda_{\vec{j}_l}^{1-\varepsilon}} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 \\
&\quad = \sum_{l=1}^k \sum_{\vec{j}^{(l)}\in\mathbb{N}^{k-1}} \sum_{\vec{j}_l\in\mathbb{N}} \pi(\lambda_{\vec{j}})^\alpha \frac{1}{\lambda_{\vec{j}_l}^\alpha} \lambda_{\vec{j}_l}^{\alpha-1+\varepsilon} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 \\
&\quad = \sum_{l=1}^k \sum_{\vec{j}^{(l)}\in\mathbb{N}^{k-1}} \left(\prod_{m\neq l} \lambda_{\vec{j}_m}^\alpha \right) \sum_{\vec{j}_l\in\mathbb{N}} \lambda_{\vec{j}_l}^{\alpha-1+\varepsilon} \left| \sum_{p,r\in\mathbb{N}} B^{p,r}_{\vec{j}_l} \hat{v}(\vec{j}^{(l;p,r)}) \right|^2 \\
&\quad \leq c_B(\alpha,\varepsilon) \sum_{l=1}^k \sum_{\vec{j}^{(l)}\in\mathbb{N}^{k-1}} \left(\prod_{m\neq l} \lambda_{\vec{j}_m}^\alpha \right) \sum_{p,r\in\mathbb{N}} \hat{v}(\vec{j}^{(l;p,r)})^2 \lambda_p^\alpha \lambda_r^\alpha (\lambda_p + \lambda_r)
\end{aligned}$$

$$= c_B(\alpha,\varepsilon)\sum_{l=1}^{k}\sum_{\vec{i}\in\mathbb{N}^{k+1}}\pi(\lambda_{\vec{i}})^\alpha\hat{v}(\vec{i})^2(\lambda_{\vec{i}_l}+\lambda_{\vec{i}_{l+1}}) = c_B(\alpha,\varepsilon)\sum_{\vec{i}\in\mathbb{N}^{k+1}}\pi(\lambda_{\vec{i}})^\alpha\hat{v}(\vec{i})^2\underbrace{\sum_{l=1}^{k}(\lambda_{\vec{i}_l}+\lambda_{\vec{i}_{l+1}})}_{\leq 2\sigma(\lambda_{\vec{i}})}$$

$$\leq 2c_B(\alpha,\varepsilon)\sum_{\vec{i}\in\mathbb{N}^{k+1}}\pi(\lambda_{\vec{i}})^\alpha\sigma(\lambda_{\vec{i}})\hat{v}(\vec{i})^2 = 2c_B(\alpha,\varepsilon)\|v\|^2_{V_1^\alpha(k+1)}.$$

Thus,

$$\|B_k v\|^2_{V^\alpha_{-1+\varepsilon}(k)} \leq 2c_B(\alpha,\varepsilon)k^\varepsilon\|v\|^2_{V_1^\alpha(k+1)}.$$ □

Proof of Lemma 4.6. For all $v \in V_1^0(k-1)$ it holds that

$$F_k(t)v = \sum_{\vec{j}\in\mathbb{N}^k}\langle F_k(t)v,\varphi_{\vec{j}}\rangle\varphi_{\vec{j}} = \sum_{\vec{j}\in\mathbb{N}^k}\varphi_{\vec{j}}\sum_{l=1}^{k}\langle -f,\varphi_{\vec{j}_l}\rangle\hat{v}(\vec{j}^{(l)}).$$

With $\lambda_{\min}$ being the smallest eigenvalue of L, we have that

$$\begin{aligned}
\|F_k(t)v\|^2_{V^\alpha_{-1+\varepsilon}(k)} &= \sum_{\vec{j}\in\mathbb{N}^k}\pi(\lambda_{\vec{j}})^\alpha\sigma(\lambda_{\vec{j}})^{-1+\varepsilon}\left|\sum_{l=1}^{k}\langle -f,\varphi_{\vec{j}_l}\rangle\hat{v}(\vec{j}^{(l)})\right|^2\\
&\leq \sum_{\vec{j}\in\mathbb{N}^k}\pi(\lambda_{\vec{j}})^\alpha\sigma(\lambda_{\vec{j}})^{-1+\varepsilon}\left[\sum_{l=1}^{k}\lambda_{\vec{j}_l}^\alpha\langle f,\varphi_{\vec{j}_l}\rangle^2\right]\left[\sum_{l=1}^{k}\lambda_{\vec{j}_l}^{-\alpha}\hat{v}(\vec{j}^{(l)})^2\right]\\
&= \sum_{\vec{j}\in\mathbb{N}^k}\underbrace{\sigma(\lambda_{\vec{j}})^{-1+\varepsilon}}_{\leq\lambda_{\min}^{-1+\varepsilon}k^{-1+\varepsilon}}\left[\sum_{l=1}^{k}\lambda_{\vec{j}_l}^\alpha\langle f,\varphi_{\vec{j}_l}\rangle^2\right]\left[\sum_{l=1}^{k}\pi(\lambda_{\vec{j}})^\alpha\lambda_{\vec{j}_l}^{-\alpha}\hat{v}(\vec{j}^{(l)})^2\right]\\
&\leq \lambda_{\min}^{-1+\varepsilon}k^{-1+\varepsilon}\left[\sum_{j\in\mathbb{N}}\sum_{l=1}^{k}\lambda_j^\alpha\langle f,\varphi_j\rangle^2\right]\left[\sum_{\vec{i}\in\mathbb{N}^{k-1}}\sum_{l=1}^{k}\pi(\lambda_{\vec{i}})^\alpha\hat{v}(\vec{i})^2\right]\\
&= \lambda_{\min}^{-1+\varepsilon}k^{1+\varepsilon}\left[\sum_{j\in\mathbb{N}}\lambda_j^\alpha\langle f,\varphi_j\rangle^2\right]\left[\sum_{\vec{i}\in\mathbb{N}^{k-1}}\pi(\lambda_{\vec{i}})^\alpha\hat{v}(\vec{i})^2\right]\\
&\leq \lambda_{\min}^{-1+\varepsilon}k^{1+\varepsilon}\|f(t)\|^2_{V^\alpha}\sum_{\vec{i}\in\mathbb{N}^{k-1}}\underbrace{\sigma(\lambda_{\vec{i}})^{-1}}_{\leq\lambda_{\min}^{-1}\frac{1}{k-1}}\sigma(\lambda_{\vec{i}})\pi(\lambda_{\vec{i}})^\alpha\hat{v}(\vec{i})^2\\
&\leq \lambda_{\min}^{-2+\varepsilon}\frac{k^{1+\varepsilon}}{k-1}\|f(t)\|^2_{V^\alpha}\|v\|^2_{V_1^\alpha(k)}\\
&\leq 2\lambda_{\min}^{-2+\varepsilon}k^\varepsilon\|f(t)\|^2_{V^\alpha}\|v\|^2_{V_1^\alpha(k)},
\end{aligned}$$

since $k \geq 2$. This concludes the proof. □

Appendix B

Supplementary Material for Tree-Coupled Saddle-Point Systems

The following sections provide a more detailed examination of some topics discussed in Chapter 5.

B.1 Example Application of the Direct Method

To illustrate the direct method introduced in Section 5.3, we examine its application to the solution of the example in Figure 5.1. Following the notation for the nested systems we find that $\mathcal{B}_{\leq 1} = B_1$, $\mathcal{B}_{\leq 2} = B_2$, and

$$\mathcal{B}_{\leq 3} = \begin{pmatrix} \mathcal{B}_{\leq 1} & & -(\mathcal{C}^-_{\leq 3})^T \\ & \mathcal{B}_{\leq 2} & \\ & B_3 & (\mathcal{C}^+_{\leq 3})^T \\ -\mathcal{C}^-_{\leq 3} & \mathcal{C}^+_{\leq 3} & -\mathcal{D}_{\leq 3} \end{pmatrix},$$

where

$$\mathcal{D}_{\leq 3} = \begin{pmatrix} D_1 & \\ & D_2 \end{pmatrix}, \quad \mathcal{C}^+_{\leq 3} = \begin{pmatrix} C_1^+ \\ C_2^+ \end{pmatrix}, \quad \text{and} \quad \mathcal{C}^-_{\leq 3} = \begin{pmatrix} C_1^- & \\ & C_2^- \end{pmatrix},$$

yielding the same linear system as in the example. We wish to solve a system with this matrix and a right-hand side $(h_{\leq 1}, h_{\leq 2}, h_3, f_1, f_2)$ for variables $(x_{\leq 1}, x_{\leq 2}, x_3, y_1, y_2)$.

Recall that the direct method is based on the Schur complement of this system with respect to the block made up by $\operatorname{diag}(\mathcal{B}_{\leq 1}, \mathcal{B}_{\leq 2}, B_3)$. The block is inverted to provide a solution for $(x_{\leq 1}, x_{\leq 2}, x_3)$ based on y_1 and y_2. To obtain values for the latter variables, this solution is substituted into the remaining rows, yielding the two systems

$$\mathcal{S}_{\leq 3} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -\mathcal{C}^-_{\leq 3} & \mathcal{C}^+_{\leq 3} \end{pmatrix} \begin{pmatrix} \mathcal{B}_{\leq 1} & & \\ & \mathcal{B}_{\leq 2} & \\ & & B_3 \end{pmatrix}^{-1} \begin{pmatrix} h_{\leq 1} \\ h_{\leq 2} \\ h_3 \end{pmatrix} - \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \text{ and}$$

$$\begin{pmatrix} \mathcal{B}_{\leq 1} & & \\ & \mathcal{B}_{\leq 2} & \\ & & B_3 \end{pmatrix} \begin{pmatrix} x_{\leq 1} \\ x_{\leq 2} \\ x_3 \end{pmatrix} = \begin{pmatrix} h_{\leq 1} \\ h_{\leq 2} \\ h_3 \end{pmatrix} - \begin{pmatrix} -(\mathcal{C}^-_{\leq 3})^T \\ (\mathcal{C}^+_{\leq 3})^T \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix},$$

where the Schur complement $\mathcal{S}_{\leq 3}$ is given by

$$\mathcal{S}_{\leq 3} = \begin{pmatrix} -\mathcal{C}^-_{\leq 3} & \mathcal{C}^+_{\leq 3} \end{pmatrix} \begin{pmatrix} \mathcal{B}_{\leq 1} & & \\ & \mathcal{B}_{\leq 2} & \\ & & B_3 \end{pmatrix}^{-1} \begin{pmatrix} -(\mathcal{C}^-_{\leq 3})^T \\ (\mathcal{C}^+_{\leq 3})^T \end{pmatrix} + \mathcal{D}_{\leq 3}.$$

In order to solve the systems we first need to compute $\mathcal{S}_{\leq 3}$. By multiplying out the products in the definition and rearranging terms we obtain

$$\begin{aligned} \mathcal{S}_{\leq 3} = &\begin{pmatrix} C^-_1 B_1^{-1} (C^-_1)^T & \\ & C^-_2 B_2^{-1} (C^-_2)^T \end{pmatrix} \\ &+ \begin{pmatrix} C^+_1 B_3^{-1} (C^+_1)^T & C^+_1 B_3^{-1} (C^+_2)^T \\ C^+_2 B_3^{-1} (C^+_1)^T & C^+_2 B_3^{-1} (C^+_2)^T \end{pmatrix} \\ &+ \begin{pmatrix} D_1 & \\ & D_2 \end{pmatrix}. \end{aligned}$$

These blocks are precisely the ones computed in Algorithms 5.2 and 5.3: The blocks D_1 and D_2 are added to the diagonal of $\mathcal{S}_{\leq 3}$, while the blocks in the second term with blocks $C^+_* B_3^{-1} (C^+_*)^T$ corresponds to the products $C^+_{kl} Z_{l'}$ computed in Algorithm 5.2. The remaining first term is computed by two calls to Algorithm 5.3, each of which consists of the base case, since vertices 1 and 2 are leaves.

Once $\mathcal{S}_{\leq 3}$ is computed, we can use it to solve the two systems above one after the other. First, to obtain values for the variables y_1 and y_2 based on the first system, the direct method in Algorithm 5.1 recurses into the leaves 1 and 2 to obtain $\hat{y}_1$ and $\hat{y}_2$ respectively. These vectors are then used to compute the right-hand side for the system involving $\mathcal{S}_{\leq 3}$, which is subsequently solved, yielding y_1 and y_2. Second, once these values have been computed, we have all of the information required to compute the right-hand side and then the solution of the second system, yielding the values of the remaining variables $x_{\leq 1}$, $x_{\leq 2}$, and x_3, where the computation of $x_{\leq 1}$ and $x_{\leq 2}$ requires a second recursion into the respective leaves of the tree.

B.2 Domain Decomposition in the Mixed Formulation

In this section, we give a more detailed description of how domain decomposition can be applied to elliptic PDEs in the presented framework. We consider the Poisson problem on a given Lipschitz domain Ω. The PDE is given by

$$\begin{aligned} -\Delta u(x) &= f(x) \quad \text{in } x \in \Omega, \\ u(x) &= u_0(x) \quad \text{on } x \in \Gamma_D, \\ \frac{\partial u}{\partial n}(x) &= g(x) \quad \text{on } x \in \Gamma_N, \end{aligned} \tag{B.1}$$

where $\Omega \subset \mathbb{R}^d$ with $d \in \mathbb{N}$, $\partial\Omega = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$, $f \in L^2(\Omega)$, $u_0 \in H^{1/2}(\Gamma_D)$, and $g \in H^{-1/2}(\Gamma_N)$. Our aim is to solve the PDE on separate domains in order to break down the overall problem into smaller pieces. For that purpose, we decompose Ω into non-overlapping closed subsets $\Omega_i \subset \Omega$ for $i \in \{1, \dots, n\}$ for some $n \in \mathbb{N}$, known as non-overlapping domain decomposition (see [118]). We then have that $\Omega = \bigcup_{i=1}^{n} \Omega_i$ and $\Omega_i \cap \Omega_j$ is a set of measure zero for $i \neq j$. We denote the interfaces between the subdomains by $\Gamma_{i,j} \coloneqq \partial\Omega_i \cap \partial\Omega_j$ for $i, j \in \{1, \dots, n\}$ with $i \neq j$. One can then solve the PDE on each subdomain separately, while enforcing continuity of u and of the flux on the interfaces, i.e.,

$$\begin{aligned} u_i(x) &= u_j(x) && \text{on } x \in \Gamma_{i,j}, \\ \nabla u_i(x) \cdot n &= -\nabla u_j(x) \cdot n && \text{on } x \in \Gamma_{i,j}, \end{aligned}$$

where u_i denotes the solution on the subdomain Ω_i. If the continuity at the interfaces is fulfilled while all u_i solve the PDE (B.1) on each subdomain, the overall solution can be achieved by joining the partial solutions u_i. While this visualizes the basic idea of non-overlapping domain decomposition, we choose to use the mixed formulation of the Poisson problem instead of (B.1), since this formulation allows us to impose the continuity conditions through matching of degrees of freedom in magnitude (given an appropriate choice of approximation function spaces). One further advantage of using the mixed formulation is that it allows lower regularity of the data and solution. The mixed formulation is given by

$$\begin{aligned} \sigma - \nabla u &= 0 && \text{in } \Omega, \\ \nabla \cdot \sigma &= -f && \text{in } \Omega, \\ \sigma \cdot n &= g && \text{on } \Gamma_N, \\ u &= u_0 && \text{on } \Gamma_D. \end{aligned} \tag{B.2}$$

Under appropriate assumptions, it can be shown that (B.2) is equivalent to (B.1). We use finite element methods to solve the PDE, which requires us to consider the weak formulation of the PDE:

$$\int_\Omega \left(\tau \cdot \sigma + (\nabla \cdot \tau)\, u + v\, (\nabla \cdot \sigma)\right) \mathrm{d}x = -\int_\Omega f v \mathrm{d}x + \int_{\Gamma_D} (\tau \cdot n)\, u_0 \mathrm{d}s \quad \text{for all } (\tau, v) \in \Sigma \times U,$$

where $(\sigma, u) \in \Sigma \times U$, $\Sigma = \{\sigma \in H(\mathrm{div}, \Omega) \mid \sigma \cdot n = g \text{ on } \Gamma_N\}$ and $U = L^2(\Omega)$. Now, the continuity of the flux at the interface $\Gamma_{i,j}$ translates to

$$\sigma_i \cdot n = -\sigma_j \cdot n \quad \text{on } \Gamma_{i,j}.$$

In order to give this meaning, we have to consider a weak formulation. One might be tempted to choose

$$\int_{\Gamma_{i,j}} \sigma_i \cdot n \mathrm{d}s = -\int_{\Gamma_{i,j}} \sigma_j \cdot n \mathrm{d}s \quad \text{for all } v \in L^2(\Omega).$$

However, one has to be aware that v is not well-defined on $\Gamma_{i,j}$ and the trace of σ is an operator $\gamma_{\Sigma,i,j} \in H^{-1/2}(\Gamma_{i,j})$. In [158], it is discussed how this problem can be circumvented. They introduce an additional test function for the interfaces. First, the following spaces are introduced:

$$\begin{aligned} M &\coloneqq \prod_{i,j} L^2(\Gamma_{i,j}), \\ \Sigma &\coloneqq \left\{ q \in \left(L^2(\Omega)\right)^d \mid q|_{\Omega_i} \in H(\mathrm{div}, \Omega_i), [q \cdot n] \in M \right\}, \\ U &\coloneqq L^2(\Omega). \end{aligned}$$

Let Γ^I denote the set of all interfaces. Let $u, v \in U$, $\sigma, \tau \in \Sigma$, and $\phi, \psi \in M$. The mixed formulation is then given by

$$\begin{aligned} \int_\Omega (\sigma \cdot \tau + v \nabla \cdot \sigma + u \nabla \cdot \tau) \,\mathrm{d}x + \int_{\Gamma^I} [\sigma \cdot n] \psi \mathrm{d}s - \int_{\Gamma^I} [\tau \cdot n] \phi \mathrm{d}s \\ = -\int_\Omega f v \mathrm{d}x \quad \text{for all } \; v \in U, \; \tau \in \Sigma, \; \psi \in M. \end{aligned} \tag{B.3}$$

Boundary terms have been omitted for the sake of simplicity. Another way to look at this is to enforce the continuity of the flux in a weak sense, but with trial and test functions that have higher regularity. The higher regularity of Σ at the interfaces increases the regularity of the trace from $H^{-1/2}(\Gamma_{i,j})$ to $L^2(\Gamma_{i,j})$. This formulation enforces both the continuity of u_i as well as the continuity of the flux at the interfaces, and yields matrices fitting the framework presented in this chapter.

In [158, Sec. 5], conditions for conforming subspaces are stated under which convergence of solutions of (B.3) to solutions to (B.2) is guaranteed. We make use of the more specific case discussed in [158, Eq. (5.16)]. It states that the space spanned by the traces of the discretization Σ_h of Σ at the interfaces has to be the same space as the discretization M_h of M. The conforming subspaces

$$\begin{aligned} M_h &\coloneqq \prod_{i,j} P_0(\Gamma_{i,j,h}) \subset M, \\ \Sigma_h &\coloneqq RT_0(\Omega_h) \subset \Sigma, \\ U_h &\coloneqq P_0(\Omega_h), \end{aligned}$$

fulfill this condition, where $RT_0(\Omega_h)$ denotes the Raviart–Thomas finite element function space of first order. The traces of $RT_0(\Omega_h)$ are constant at each facet, which spans the space M_h. This also shows that the traces of Σ_h are in L^2. Moreover, we have that

$$\operatorname{div} \Sigma_h \subseteq U_h.$$

If the decomposition is carried out recursively one can arrange the continuity constraints in a tree form, as described in the numerical experiments.

Bibliography

[1] Roger Temam. *Navier–Stokes Equations: Theory and Numerical Analysis*. 2nd ed. Studies in Mathematics and its Applications, Vol. 2. Amsterdam: North-Holland Publishing Company, 1979. ISBN: 0-444-85307-3 (cit. on pp. 1, 77, 126).

[2] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge: Cambridge University Press, 2000. DOI: 10.1017/CBO9780511800955 (cit. on p. 1).

[3] Geoffrey K. Vallis. *Atmospheric and Oceanic Fluid Dynamics: Fundamentals and Large-Scale Circulation*. 2nd ed. Cambridge: Cambridge University Press, 2017. DOI: 10.1017/9781107588417 (cit. on p. 1).

[4] James D. Murray. *Mathematical Biology I. An Introduction*. 3rd ed. New York, NY: Springer, 2003. ISBN: 978-0-387-95223-9. DOI: 10.1007/b98868 (cit. on p. 1).

[5] James D. Murray. *Mathematical Biology II. Spatial Models and Biomedical Applications*. 3rd ed. New York, NY: Springer, 2003. ISBN: 978-0-387-95228-4. DOI: 10.1007/b98869 (cit. on p. 1).

[6] Leah Edelstein-Keshet. *Mathematical Models in Biology*. Society for Industrial and Applied Mathematics, 2005. ISBN: 978-0-89871-554-5. DOI: 10.1137/1.9780898719147 (cit. on p. 1).

[7] James Keener and James Sneyd. *Mathematical physiology. I: Cellular Physiology*. Ed. by James Keener and James Sneyd. 2nd ed. Interdisciplinary applied mathematics. New York, NY: Springer, 2008. ISBN: 978-0-387-75846-6. DOI: 10.1007/978-0-387-75847-3 (cit. on p. 1).

[8] James Keener and James Sneyd. *Mathematical physiology. II: Systems Physiology*. Ed. by James Keener and James Sneyd. 2nd ed. Interdisciplinary applied mathematics. New York, NY: Springer, 2008. ISBN: 978-0-387-79387-0. DOI: 10.1007/978-0-387-79388-7 (cit. on p. 1).

[9] Fred Brauer, Carlos Castillo-Chavez, and Zhilan Feng. *Mathematical Models in Epidemiology*. 1st ed. New York, NY: Springer, 2019. ISBN: 978-1-4939-9828-9. DOI: 10.1007/978-1-4939-9828-9 (cit. on p. 1).

[10] Benoît Perthame. "PDE models for chemotactic movements: Parabolic, hyperbolic and kinetic". In: *Applications of Mathematics* 49.6 (2004). ISSN: 08627940. DOI: 10.1007/s10492-004-6431-9 (cit. on p. 1).

[11] Ricky T.Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. "Neural ordinary differential equations". In: *Advances in Neural Information Processing Systems*. Vol. 2018-December. Neural information processing systems foundation, 2018, pp. 6571–6583 (cit. on p. 1).

[12] Martin P. Bendsøe and Ole Sigmund. *Topology Optimization: Theory, Methods, and Applications*. 2nd ed. Berlin, Heidelberg: Springer, 2004. ISBN: 978-3-540-42992-0. DOI: 10.1007/978-3-662-05086-6 (cit. on p. 1).

[13] Bernhard Heinzelreiter and John W. Pearson. "Diagonalization-based parallel-in-time preconditioners for instationary fluid flow control problems". In: *IMA Journal of Numerical Analysis* (2025). DOI: 10.1093/imanum/draf088 (cit. on pp. 2, 89).

[14] Bernhard Heinzelreiter and John W. Pearson. "Carleman linearization of parabolic PDEs: Well-posedness, convergence, and efficient numerical methods". arXiv:2510.00722 [math.NA]. 2025 (cit. on pp. 2, 3).

[15] Christoph Hansknecht, Bernhard Heinzelreiter, John W. Pearson, and Andreas Potschka. "A framework for the solution of tree-coupled saddle-point systems". In: *Numerical Linear Algebra with Applications* (2025). DOI: 10.1002/nla.70038 (cit. on pp. 2, 3).

[16] Lawrence C. Evans. *Partial Differential Equations*. 2nd ed. Providence, RI: American Mathematical Society, 2010. ISBN: 9780821849743. DOI: 10.1090/gsm/019 (cit. on pp. 5, 73).

[17] Eberhard Zeidler. *Nonlinear Functional Analysis and its Applications. II/ A: Linear Monotone Operators*. New York, NY: Springer, 1990. ISBN: 978-1-4612-6971-7. DOI: 10.1007/978-1-4612-0985-0 (cit. on pp. 5, 60, 78, 79).

[18] Fredi Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*. American Mathematical Society, 2009. ISBN: 978-1-4704-7644-1 (cit. on p. 5).

[19] Jacques-Louis Lions. *Optimal Control of Systems Governed by Partial Differential Equations*. Berlin, Heidelberg: Springer, 1971. ISBN: 978-3-642-65026-0. DOI: 10.1007/978-3-642-65024-6 (cit. on pp. 5, 10).

[20] Jerzy Zabczyk. *Mathematical Control Theory*. Boston, MA: Birkhäuser, 2008. ISBN: 978-0-8176-4732-2. DOI: 10.1007/978-0-8176-4733-9 (cit. on pp. 5, 10).

[21] Michael Hinze, Rene Pinnau, Michael Ulbrich, and Stefan Ulbrich. *Optimization with PDE Constraints*. Dordrecht: Springer, 2009. ISBN: 978-1-4020-8838-4. DOI: 10.1007/978-1-4020-8839-1 (cit. on pp. 5, 6, 9, 10, 122).

[22] Matthias Heinkenschloss and Dmitriy Leykekhman. "Local Error Estimates for SUPG Solutions of Advection-Dominated Elliptic Linear-Quadratic Optimal Control Problems". In: *SIAM Journal on Numerical Analysis* 47.6 (2010), pp. 4607–4638. DOI: 10.1137/090759902 (cit. on p. 9).

[23] Irena Lasiecka and Roberto Triggiani. *Control Theory for Partial Differential Equations*. Vol. 1: Abstract Parabolic Systems. Cambridge University Press, 2000. ISBN: 978-0-5214-3408-9. DOI: 10.1017/CBO9781107340848 (cit. on pp. 10, 64).

[24] Xiaoye S. Li. "An overview of SuperLU: Algorithms, implementation, and user interface". In: *ACM Transactions on Mathematical Software* 31.3 (2005), pp. 302–325. DOI: 10.1145/1089014.1089017 (cit. on pp. 11, 113).

[25] Timothy A Davis. "A column pre-ordering strategy for the unsymmetric-pattern multifrontal method". In: *ACM Transactions on Mathematical Software (TOMS)* 30.2 (2004), pp. 165–195. DOI: 10.1145/992200.992205 (cit. on p. 11).

[26] P.R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. "A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling". In: *SIAM Journal on Matrix Analysis and Applications* 23.1 (2001), pp. 15–41. DOI: 10.1137/S0895479899358194 (cit. on p. 11).

[27] Howard Elman, David Silvester, and Andy Wathen. *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. 2nd ed. Oxford University Press, 2014. ISBN: 978-0-199-67879-2. DOI: 10.1093/acprof:oso/9780199678792.001.0001 (cit. on pp. 11–14, 20, 22, 26, 27, 31, 91).

[28] Christopher C. Paige and Michael A. Saunders. "Solution of sparse indefinite systems of linear equations". In: *SIAM Journal on Numerical Analysis* 12.4 (1975), pp. 617–629. ISSN: 0036-1429. DOI: 10.1137/0712047 (cit. on pp. 12, 30).

[29] Youcef Saad and Martin H. Schultz. "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pp. 856–869. ISSN: 0196-5204. DOI: 10.1137/0907058 (cit. on pp. 14, 30, 103).

[30] Youcef Saad. "A flexible inner-outer preconditioned GMRES algorithm". In: *SIAM Journal on Scientific Computing* 14.2 (1993), pp. 461–469. ISSN: 1064-8275. DOI: 10.1137/0914028 (cit. on pp. 15–17, 41).

[31] Tyrone Rees, H. Sue Dollar, and Andrew J. Wathen. "Optimal solvers for PDE-constrained optimization". In: *SIAM Journal on Scientific Computing* 32.1 (2010), pp. 271–298. ISSN: 1064-8275. DOI: 10.1137/080727154 (cit. on pp. 16, 24).

[32] Andrew J. Wathen. "Realistic eigenvalue bounds for the Galerkin mass matrix". In: *IMA Journal of Numerical Analysis* 7.4 (1987), pp. 449–457. ISSN: 0272-4979. DOI: 10.1093/imanum/7.4.449 (cit. on pp. 19, 42).

[33] Andy Wathen and Tyrone Rees. "Chebyshev semi-iteration in preconditioning for problems including the mass matrix". In: *Electronic Transactions on Numerical Analysis* 34 (2008), pp. 125–135. ISSN: 10689613 (cit. on pp. 19, 30, 42).

[34] Tyrone Rees. "Preconditioning Iterative Methods for PDE Constrained Optimization". PhD thesis. Oxford, UK: University of Oxford, 2010 (cit. on pp. 19, 44).

[35] Jinchao Xu and Ludmil Zikatanov. "Algebraic multigrid methods". In: *Acta Numerica* 26 (2017), pp. 591–721. DOI: 10.1017/S0962492917000083 (cit. on pp. 20, 23, 109).

[36] Max D. Gunzburger. *Perspectives in Flow Control and Optimization*. Society for Industrial and Applied Mathematics, 2002. ISBN: 978-0-89871-527-9. DOI: 10.1137/1.9780898718720 (cit. on p. 24).

[37] Lorenz T. Biegler, Omar Ghattas, Matthias Heinkenschloss, David Keyes, and Bart van Bloemen Waanders. *Real-Time PDE-Constrained Optimization*. Ed. by Lorenz T Biegler, Omar Ghattas, Matthias Heinkenschloss, David Keyes, and Bart van Bloemen Waanders. Society for Industrial and Applied Mathematics, 2007. ISBN: 978-0-89871-621-4. DOI: 10.1137/1.9780898718935 (cit. on p. 24).

[38] Subhendu Bikash Hazra. *Large-Scale PDE-Constrained Optimization in Applications*. Vol. 49. Berlin, Heidelberg: Springer, 2010. ISBN: 978-3-642-01501-4. DOI: 10.1007/978-3-642-01502-1 (cit. on p. 24).

[39] Joachim Schöberl and Walter Zulehner. "Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems". In: *SIAM Journal on Matrix Analysis and Applications* 29.3 (2007), pp. 752–773. ISSN: 0895-4798. DOI: 10.1137/060660977 (cit. on p. 24).

[40] Joachim Schöberl, René Simon, and Walter Zulehner. "A robust multigrid method for elliptic optimal control problems". In: *SIAM Journal on Numerical Analysis* 49.4 (2011), pp. 1482–1503. ISSN: 0036-1429. DOI: 10.1137/100783285 (cit. on p. 24).

[41] John W. Pearson and Andrew J. Wathen. "A new approximation of the Schur complement in preconditioners for PDE-constrained optimization". In: *Numerical Linear Algebra with Applications* 19.5 (2012), pp. 816–829. ISSN: 1070-5325. DOI: 10.1002/nla.814 (cit. on pp. 24, 30).

[42] John W. Pearson and Andrew J. Wathen. "Fast iterative solvers for convection-diffusion control problems". In: *Electronic Transactions on Numerical Analysis* 40 (2013), pp. 294–310. ISSN: 10689613 (cit. on pp. 24, 30, 31).

[43] John W. Pearson and Martin Stoll. "Fast iterative solution of reaction-diffusion control problems arising from chemical processes". In: *SIAM Journal on Scientific Computing* 35.5 (2013), B987–B1009. ISSN: 1064-8275. DOI: 10.1137/120892003 (cit. on p. 24).

[44] Eleanor McDonald, Jennifer Pestana, and Andy Wathen. "Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations". In: *SIAM Journal on Scientific Computing* 40.2 (2018), A1012–A1033. ISSN: 10957197. DOI: 10.1137/16M1062016 (cit. on p. 24).

[45] David Silvester, Howard Elman, David Kay, and Andrew Wathen. "Efficient preconditioning of the linearized Navier–Stokes equations for incompressible flow". In: *Journal of Computational and Applied Mathematics* 128.1–2 (2001), pp. 261–279. ISSN: 03770427. DOI: 10.1016/S0377-0427(00)00515-X (cit. on pp. 24, 25, 31).

[46] David Kay, Daniel Loghin, and Andrew Wathen. "A preconditioner for the steady-state Navier–Stokes equations". In: *SIAM Journal on Scientific Computing* 24.1 (2002), pp. 237–256. ISSN: 1064-8275. DOI: 10.1137/S106482759935808X (cit. on pp. 24, 25, 31).

[47] Wolfgang Krendl, Valeria Simoncini, and Walter Zulehner. "Stability estimates and structural spectral properties of saddle point problems". In: *Numerische Mathematik* 124.1 (2013), pp. 183–213. ISSN: 0029-599X. DOI: 10.1007/s00211-012-0507-3 (cit. on p. 24).

[48] Wolfgang Krendl, Valeria Simoncini, and Walter Zulehner. "Efficient preconditioning for an optimal control problem with the time-periodic Stokes equations". In: *Lecture Notes in Computational Science and Engineering*. Vol. 103. 2015, pp. 479–487. DOI: 10.1007/978-3-319-10705-9_47 (cit. on pp. 24, 32, 34, 35).

[49] Federico Danieli, Ben S. Southworth, and Andrew J. Wathen. "Space-time block preconditioning for incompressible flow". In: *SIAM Journal on Scientific Computing* 44.1 (2022), A337–A363. ISSN: 10957197. DOI: 10.1137/21M1390773 (cit. on p. 24).

[50] Santolo Leveque and John W. Pearson. "Fast iterative solver for the optimal control of time-dependent PDEs with Crank–Nicolson discretization in time". In: *Numerical Linear Algebra with Applications* 29.2 (2022). ISSN: 1070-5325. DOI: 10.1002/nla.2419 (cit. on p. 24).

[51] Michael Hinze, Michael Köster, and Stefan Turek. "A Space-Time Multigrid Method for Optimal Flow Control". In: *Constrained Optimization and Optimal Control for Partial Differential Equations*. Basel: Birkhäuser, 2012, pp. 147–170. ISBN: 978-3-0348-0132-4. DOI: 10.1007/978-3-0348-0133-1_8 (cit. on p. 24).

[52] Nick Janssens and Johan Meyers. "Parallel-in-time multiple shooting for optimal control problems governed by the Navier–Stokes equations". In: *Computer Physics Communications* 296 (2024), p. 109019. ISSN: 00104655. DOI: 10.1016/j.cpc.2023.109019 (cit. on p. 24).

[53] Herb Sutter. "The free lunch is over: A fundamental turn toward concurrency in software". In: *Dr. Dobb's Journal* 30.3 (2005). ISSN: 18681751 (cit. on p. 25).

[54] Martin J. Gander. "50 years of time parallel time integration". In: *Multiple Shooting and Time Domain Decomposition Methods*. Ed. by Thomas Carraro, Michael Geiger, Stefan Körkel, and Rolf Rannacher. Vol. 9. Contributions in Mathematical and Computational Sciences. Cham: Springer, 2015, pp. 69–113. ISBN: 978-3-319-23320-8. DOI: 10.1007/978-3-319-23321-5 (cit. on pp. 25, 116).

[55] Jacques-Louis Lions, Yvon Maday, and Gabriel Turinici. "Résolution d'EDP par un schéma en temps 'pararéel'". In: *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics* 332.7 (2001), pp. 661–668. ISSN: 07644442. DOI: 10.1016/S0764-4442(00)01793-6 (cit. on p. 25).

[56] Martin J. Gander, Jun Liu, Shu-Lin Wu, Xiaoqiang Yue, and Tao Zhou. "ParaDiag: parallel-in-time algorithms based on the diagonalization technique". arXiv:2005.09158 [math.NA]. 2021 (cit. on pp. 25, 54, 89).

[57] Sebastian Götschel and Michael L. Minion. "An efficient parallel-in-time method for optimization with parabolic PDEs". In: *SIAM Journal on Scientific Computing* 41.6 (2019), pp. C603–C626. ISSN: 1064-8275. DOI: 10.1137/19M1239313 (cit. on pp. 25, 32).

[58] Martin J. Gander, Felix Kwok, and Julien Salomon. "ParaOpt: A parareal algorithm for optimality systems". In: *SIAM Journal on Scientific Computing* 42.5 (2020), A2773–A2802. ISSN: 1064-8275. DOI: 10.1137/19M1292291 (cit. on pp. 25, 32).

[59] Arne Bouillon, Giovanni Samaey, and Karl Meerbergen. "On generalized preconditioners for time-parallel parabolic optimal control". In: *SIAM Journal on Scientific Computing* 46.4 (2024), A2298–A2323. DOI: 10.1137/23M1553194 (cit. on pp. 25, 32).

[60] Shu-Lin Wu and Jun Liu. "A parallel-in-time block-circulant preconditioner for optimal control of wave equations". In: *SIAM Journal on Scientific Computing* 42.3 (2020). ISSN: 10957197. DOI: 10.1137/19M1289613 (cit. on p. 25).

[61] Shu-Lin Wu and Tao Zhou. "Diagonalization-based parallel-in-time algorithms for parabolic PDE-constrained optimization problems". In: *ESAIM: Control, Optimisation and Calculus of Variations* 26 (2020), Art. 88. ISSN: 12623377. DOI: 10.1051/cocv/2020012 (cit. on pp. 25, 89).

[62] Shu Lin Wu, Zhiyong Wang, and Tao Zhou. "PinT preconditioner for forward-backward evolutionary equations". In: *SIAM Journal on Matrix Analysis and Applications* 44.4 (2023). ISSN: 10957162. DOI: 10.1137/22M1516476 (cit. on p. 25).

[63] Michele Benzi, Gene H. Golub, and Jörg Liesen. "Numerical solution of saddle point problems". In: *Acta Numerica* 14 (2005), pp. 1–137. ISSN: 0962-4929. DOI: 10.1017/S0962492904000212 (cit. on pp. 29, 95).

[64] Gene H. Golub and Richard S. Varga. "Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods Part I". In: *Numerische Mathematik* 3.1 (1961), pp. 147–156. ISSN: 0029599X. DOI: 10.1007/BF01386013 (cit. on pp. 30, 42).

[65] Hermann Weyl. "Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung)". In: *Mathematische Annalen* 71.4 (1912), pp. 441–479. ISSN: 0025-5831. DOI: 10.1007/BF01456804 (cit. on p. 37).

[66] Wolfgang Hackbusch. *Multi-Grid Methods and Applications*. Vol. 4. Berlin, Heidelberg: Springer, 1985. ISBN: 978-3-642-05722-9. DOI: 10.1007/978-3-662-02427-0 (cit. on p. 42).

[67] William L. Briggs, Van Emden Henson, and Steve F. McCormick. *A Multigrid Tutorial*. 2nd ed. Society for Industrial and Applied Mathematics, 2000. ISBN: 978-0-89871-462-3. DOI: 10.1137/1.9780898719505 (cit. on pp. 42, 112).

[68] Igor A. Baratta et al. *DOLFINx: The next generation FEniCS problem solving environment*. Zenodo. 2023. DOI: 10.5281/zenodo.10447666 (cit. on pp. 45, 82, 112).

[69] Pauli Virtanen et al. "SciPy 1.0: Fundamental algorithms for scientific computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. ISSN: 1548-7105. DOI: 10.1038/s41592-019-0686-2 (cit. on pp. 45, 112).

[70] Lisandro D. Dalcin, Rodrigo R. Paz, Pablo A. Kler, and Alejandro Cosimo. "Parallel distributed computing using Python". In: *Advances in Water Resources* 34.9 (2011), pp. 1124–1139. ISSN: 03091708. DOI: 10.1016/j.advwatres.2011.04.013 (cit. on pp. 45, 82).

[71] Lisandro Dalcin and Yao Lung L. Fang. "mpi4py: Status update after 12 years of development". In: *Computing in Science and Engineering* 23.4 (2021), pp. 47–54. ISSN: 1558366X. DOI: 10.1109/MCSE.2021.3083216 (cit. on p. 45).

[72] Santolo Leveque and John W. Pearson. "Parameter-robust preconditioning for Oseen iteration applied to stationary and instationary Navier–Stokes control". In: *SIAM Journal on Scientific Computing* 44.3 (2022), B694–B722. ISSN: 1064-8275. DOI: 10.1137/21M1436531 (cit. on pp. 46, 49).

[73] Torsten Carleman. "Application de la théorie des équations intégrales linéaires aux systèmes d'équations différentielles non linéaires". In: *Acta Mathematica* 59 (1932), pp. 63–87. ISSN: 0001-5962. DOI: 10.1007/BF02546499 (cit. on p. 55).

[74] Andreas Rauh, Johanna Minisini, and Harald Aschemann. "Carleman linearization for control and for state and disturbance estimation of nonlinear dynamical processes". In: *IFAC Proceedings Volumes* 42.13 (2009), pp. 455–460. ISSN: 14746670. DOI: 10.3182/20090819-3-PL-3002.00079 (cit. on p. 56).

[75] Arash Amini, Qiyu Sun, and Nader Motee. "Carleman state feedback control design of a class of nonlinear control systems". In: *IFAC-PapersOnLine* 52.20 (2019), pp. 229–234. ISSN: 24058963. DOI: 10.1016/j.ifacol.2019.12.163 (cit. on p. 56).

[76] Arash Amini, Qiyu Sun, and Nader Motee. "Approximate optimal control design for a class of nonlinear systems by lifting Hamilton–Jacobi–Bellman equation". In: *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 2717–2722. ISBN: 978-1-5386-8266-1. DOI: 10.23919/ACC45564.2020.9147576 (cit. on p. 56).

[77] Jishnudeep Kar, He Bai, and Aranya Chakrabortty. "Reinforcement learning based approximate optimal control of nonlinear systems using Carleman linearization". In: *2023 American Control Conference (ACC)*. IEEE, 2023, pp. 3362–3367. ISBN: 979-8-3503-2806-6. DOI: 10.23919/ACC55779.2023.10156057 (cit. on p. 56).

[78] Pawan Goyal, Mian Ilyas Ahmad, and Peter Benner. "Model reduction of quadratic-bilinear descriptor systems via Carleman bilinearization". In: *2015 European Control Conference (ECC)*. 2015, pp. 1177–1182. ISBN: 978-3-9524-2693-7. DOI: 10.1109/ECC.2015.7330699 (cit. on p. 56).

[79] Peter Benner and Tobias Breiten. "Interpolation-based $\mathcal{H}_2$-model reduction of bilinear control systems". In: *SIAM Journal on Matrix Analysis and Applications* 33.3 (2012), pp. 859–885. ISSN: 0895-4798. DOI: 10.1137/110836742 (cit. on p. 56).

[80] Moad Abudia, Joel A. Rosenfeld, and Rushikesh Kamalapurkar. "Carleman lifting for nonlinear system identification with guaranteed error bounds". In: *2023 American Control Conference (ACC)*. IEEE, 2023, pp. 929–934. ISBN: 979-8-3503-2806-6. DOI: 10.23919/ACC55779.2023.10155924 (cit. on p. 56).

[81] Amit Surana, Abeynaya Gnanasekaran, and Tuhin Sahai. "An efficient quantum algorithm for simulating polynomial dynamical systems". In: *Quantum Information Processing* 23.3 (2024), Art. 105. ISSN: 1573-1332. DOI: 10.1007/s11128-024-04311-2 (cit. on p. 56).

[82] Hsuan-Cheng Wu, Jingyao Wang, and Xiantao Li. "Quantum algorithms for nonlinear dynamics: Revisiting Carleman linearization with no dissipative conditions". In: *SIAM Journal on Scientific Computing* 47.2 (2025), A943–A970. ISSN: 1064-8275. DOI: 10.1137/24M1665799 (cit. on p. 56).

[83] S. P. Banks. "Infinite-dimensional Carleman linearization, the Lie series and optimal control of non-linear partial differential equations". In: *International Journal of Systems Science* 23.5 (1992), pp. 663–675. ISSN: 14645319. DOI: 10.1080/00207729208949241 (cit. on p. 56).

[84] C. Sanavio, R. Scatamacchia, C. de Falco, and S. Succi. "Three Carleman routes to the quantum simulation of classical fluids". In: *Physics of Fluids* 36.5 (2024), Art. 057143. ISSN: 1070-6631. DOI: 10.1063/5.0204955 (cit. on p. 56).

[85] Claudio Sanavio and Sauro Succi. "Lattice Boltzmann–Carleman quantum algorithm and circuit for fluid flows at moderate Reynolds number". In: *AVS Quantum Science* 6.2 (2024), Art. 023802. ISSN: 2639-0213. DOI: 10.1116/5.0195549 (cit. on p. 56).

[86] Jin-Peng Liu et al. "Efficient quantum algorithm for nonlinear reaction–diffusion equations and energy estimation". In: *Communications in Mathematical Physics* 404.2 (2023), pp. 963–1020. ISSN: 0010-3616. DOI: 10.1007/s00220-023-04857-9 (cit. on pp. 56, 80).

[87] Marcelo Forets and Amaury Pouly. "Explicit error bounds for Carleman linearization". arXiv:1711.02552 [math.NA]. 2017 (cit. on pp. 56, 75, 80).

[88] Arash Amini, Qiyu Sun, and Nader Motee. "Error bounds for Carleman linearization of general nonlinear systems". In: *SIAM Conference on Control and its Applications, CT 2021*. 2021, pp. 1–8. DOI: 10.1137/1.9781611976847.1 (cit. on p. 56).

[89] Arash Amini, Cong Zheng, Qiyu Sun, and Nader Motee. "Carleman linearization of nonlinear systems and its finite-section approximations". In: *Discrete and Continuous Dynamical Systems - Series B* 30.2 (2025), pp. 577–603. ISSN: 1531-3492. DOI: 10.3934/dcdsb.2024102 (cit. on pp. 56, 75, 80).

[90] Javier Gonzalez-Conde, Dylan Lewis, Sachin S. Bharadwaj, and Mikel Sanz. "Quantum Carleman linearization efficiency in nonlinear fluid dynamics". In: *Physical Review Research* 7 (2025), Art. 023254. ISSN: 2643-1564. DOI: 10.1103/PhysRevResearch.7.023254 (cit. on pp. 56, 80).

[91] Alain Bensoussan, Giuseppe Da Prato, Michel C. Delfour, and Sanjoy K. Mitter. *Representation and Control of Infinite Dimensional Systems*. 2nd ed. Boston, MA: Birkhäuser, 2007. ISBN: 978-0-8176-4461-1. DOI: 10.1007/978-0-8176-4581-6 (cit. on pp. 58, 60, 61, 73).

[92] Jacques L. Lions and Enrico Magenes. *Non-Homogeneous Boundary Value Problems and Applications*. Vol. 1. Berlin, Heidelberg: Springer, 1972. ISBN: 978-3-642-65163-2. DOI: 10.1007/978-3-642-65161-8 (cit. on pp. 58, 65, 88).

[93] Alfio Quarteroni and Alberto Valli. *Numerical Approximation of Partial Differential Equations*. Berlin, Heidelberg: Springer, 1994. ISBN: 978-3-540-85267-4. DOI: 10.1007/978-3-540-85268-1 (cit. on p. 60).

[94] Tobias Breiten, Karl Kunisch, and Laurent Pfeiffer. "Feedback stabilization of the two-dimensional Navier–Stokes equations by value function approximation". In: *Applied Mathematics & Optimization* 80.3 (2019), pp. 599–641. ISSN: 0095-4616. DOI: 10.1007/s00245-019-09586-x (cit. on pp. 61, 77, 126–128).

[95] Mark J. Vishik and Andrei Vladimirovich Fursikov. *Mathematical Problems of Statistical Hydromechanics*. Dordrecht: Springer, 1988. ISBN: 978-94-010-7137-6. DOI: 10.1007/978-94-009-1423-0 (cit. on pp. 63, 64).

[96] Andrei Vladimirovich Fursikov. "On uniqueness of the solution of the chain of moment equations corresponding to the three-dimensional Navier–Stokes system". In: *Mathematics of the USSR-Sbornik* 62.2 (1989), pp. 465–490. ISSN: 21695288. DOI: 10.1070/SM1989v062n02ABEH003249 (cit. on pp. 63, 64, 66–68).

[97] Andrei Vladimirovich Fursikov. "Moment theory for the Navier–Stokes equations with a random right side". In: *Russian Academy of Sciences: Izvestiya Mathematics* 41.3 (1993), pp. 515–555. ISSN: 1064-5632. DOI: 10.1070/im1993v041n03abeh002274 (cit. on pp. 63, 64, 66–68, 77).

[98] Amnon Pazy. *Semigroups of Linear Operators and Applications to Partial Differential Equations*. New York, NY: Springer, 1983. ISBN: 978-1-4612-5563-5. DOI: 10.1007/978-1-4612-5561-1 (cit. on p. 67).

[99] Viorel Barbu. *Stabilization of Navier–Stokes Flows*. London: Springer, 2011. ISBN: 978-0-85729-042-7. DOI: 10.1007/978-0-85729-043-4 (cit. on pp. 74, 77).

[100] Umberto Marini Bettolo Marconi and Pedro Tarazona. "Dynamic density functional theory of fluids". In: *Journal of Physics: Condensed Matter* 12.8A (2000), A413–A418. ISSN: 0953-8984. DOI: 10.1088/0953-8984/12/8A/356 (cit. on p. 77).

[101] Garnet Kin-Lic Chan and Reimar Finken. "Time-dependent density functional theory of classical fluids". In: *Physical Review Letters* 94.18 (2005), Art. 183001. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.94.183001 (cit. on p. 77).

[102] Vidar Thomée. *Galerkin Finite Element Methods for Parabolic Problems*. 2nd ed. Berlin, Heidelberg, New York: Springer, 2006. ISBN: 978-3-540-33121-6. DOI: 10.1007/3-540-33122-0 (cit. on p. 78).

[103] Hans-Joachim Bungartz and Michael Griebel. "Sparse grids". In: *Acta Numerica* 13 (2004), pp. 147–269. ISSN: 0962-4929. DOI: 10.1017/S0962492904000182 (cit. on p. 80).

[104] Jochen Garcke. "Sparse grids in a nutshell". In: *Sparse Grids and Applications*. Ed. by Jochen Garcke and Michael Griebel. Berlin, Heidelberg: Springer, 2013, pp. 57–80. ISBN: 978-3-642-31703-3. DOI: 10.1007/978-3-642-31703-3_3 (cit. on p. 80).

[105] Reinhard Hochmuth, Stephan Knapek, and Gerhard Zumbusch. *Tensor products of Sobolev spaces and applications*. Tech. rep. 685, SFB 256. Universität Bonn, 2000 (cit. on p. 80).

[106] Michael Griebel and Helmut Harbrecht. "On the construction of sparse tensor product spaces". In: *Mathematics of Computation* 82.282 (2012), pp. 975–994. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-2012-02638-X (cit. on p. 80).

[107] Hans-Joachim Bungartz and Thomas Dornseifer. "Sparse grids: Recent developments for elliptic partial differential equations". In: *Multigrid Methods V*. Ed. by Wolfgang Hackbusch and Gabriel Wittum. Berlin, Heidelberg: Springer, 1998, pp. 45–70. ISBN: 978-3-642-58734-4. DOI: 10.1007/978-3-642-58734-4_3 (cit. on p. 80).

[108] Viet Ha Hoang and Christoph Schwab. "High-dimensional finite elements for elliptic problems with multiple scales". In: *Multiscale Modeling & Simulation* 3.1 (2005), pp. 168–194. ISSN: 1540-3459. DOI: 10.1137/030601077 (cit. on p. 80).

[109] I. V. Oseledets. "Tensor-train decomposition". In: *SIAM Journal on Scientific Computing* 33.5 (2011), pp. 2295–2317. ISSN: 1064-8275. DOI: 10.1137/090752286 (cit. on p. 81).

[110] Boris N. Khoromskij. "Tensor numerical methods for multidimensional PDEs: theoretical analysis and initial applications". In: *ESAIM: Proceedings and Surveys* 48 (2015), pp. 1–28. ISSN: 2267-3059. DOI: 10.1051/proc/201448001 (cit. on p. 81).

[111] Fredrik Kjolstad, Shoaib Kamil, Stephen Chou, David Lugato, and Saman Amarasinghe. "The tensor algebra compiler". In: *Proceedings of the ACM on Programming Languages* 1.OOPSLA (2017), Art. 77. DOI: 10.1145/3133901 (cit. on p. 82).

[112] W. L. Wood. "An exact solution for Burger's equation". In: *Communications in Numerical Methods in Engineering* 22.7 (2006), pp. 797–798. ISSN: 10698299. DOI: 10.1002/cnm.850 (cit. on p. 82).

[113] Michael Griebel and Helmut Harbrecht. "On the convergence of the combination technique". In: *Sparse Grids and Applications - Munich 2012*. Ed. by Jochen Garcke and Dirk Pflüger. Lecture Notes in Computational Science and Engineering. Cham: Springer, 2014, pp. 55–74. ISBN: 978-3-319-04537-5. DOI: 10.1007/978-3-319-04537-5_3 (cit. on p. 83).

[114] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. 2nd. Springer New York, 2011. ISBN: 978-1-461-40236-7. DOI: 10.1007/978-1-4614-0237-4 (cit. on p. 90).

[115] Johnny T. Ottesen, Mette S. Olufsen, and Jesper K. Larsen. *Applied Mathematical Models in Human Physiology*. Society for Industrial and Applied Mathematics, 2004. DOI: 10.1137/1.9780898718287 (cit. on p. 90).

[116] Martin Schmidt and Falk M. Hante. "Gas Transport Network Optimization: PDE-Constrained Models". In: *Encyclopedia of Optimization*. Ed. by Panos M. Pardalos and Oleg A. Prokopyev. Cham: Springer International Publishing, 2020, pp. 1–7. ISBN: 978-3-030-54621-2. DOI: 10.1007/978-3-030-54621-2_872-1 (cit. on p. 90).

[117] Andreas Frommer and Daniel B. Szyld. "An algebraic convergence theory for restricted additive Schwarz methods using weighted max norms". In: *SIAM Journal on Numerical Analysis* 39.2 (2001), pp. 463–479. DOI: 10.1137/S0036142900370824 (cit. on p. 91).

[118] Tony F. Chan and Tarek P. Mathew. "Domain decomposition algorithms". In: *Acta Numerica* 3 (1994), pp. 61–143. DOI: 10.1017/S0962492900002427 (cit. on pp. 91, 119, 134).

[119] Martin Kiehl. "Parallel multiple shooting for the solution of initial value problems". In: *Parallel Computing* 20.3 (1994), pp. 275–295. ISSN: 0167-8191. DOI: 10.1016/S0167-8191(06)80013-X (cit. on pp. 91, 116).

[120] Roger Fletcher and Sven Leyffer. "Nonlinear programming without a penalty function". In: *Mathematical Programming* 91.2 (2002), pp. 239–269. DOI: 10.1007/s101070100244 (cit. on p. 91).

[121] Roger Fletcher. *Practical Methods of Optimization*. 2nd. John Wiley & Sons, 2000. ISBN: 978-0-471-49463-8. DOI: 10.1002/9781118723203 (cit. on p. 91).

[122] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. 2nd. Springer New York, 2006. ISBN: 978-0-387-30303-1. DOI: 10.1007/978-0-387-40065-5 (cit. on pp. 91, 93).

[123] John W. Pearson and Andreas Potschka. "Double saddle-point preconditioning for Krylov methods in the inexact sequential homotopy method". In: *Numerical Linear Algebra with Applications* 31.4 (2024), e2553. DOI: 10.1002/nla.2553 (cit. on pp. 91, 93).

[124] Daniele Boffi, Franco Brezzi, and Michel Fortin. *Mixed Finite Element Methods and Applications*. Vol. 44. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2013. ISBN: 978-3-642-36518-8. DOI: 10.1007/978-3-642-36519-5 (cit. on p. 91).

[125] Malcolm F Murphy, Gene H Golub, and Andrew J Wathen. "A note on preconditioning for indefinite linear systems". In: *SIAM Journal on Scientific Computing* 21.6 (2000), pp. 1969–1972. DOI: 10.1137/S1064827599355153 (cit. on pp. 91, 102).

[126] David J. Silvester and Andrew J. Wathen. "Fast & robust solvers for time-discretised incompressible Navier–Stokes equations". In: *Numerical Analysis 1995*. Ed. by David F. Griffiths and G. Alistair Watson. Longman Scientific, 1996, pp. 154–168. ISBN: 978-0-582-27633-8 (cit. on p. 91).

[127] Catherine Powell and David Silvester. "Black-box preconditioning for mixed formulation of self-adjoint elliptic PDEs". In: *Challenges in Scientific Computing – CISC 2002: Proceedings of the Conference Challenges in Scientific Computing Berlin, October 2–5, 2002*. Springer Berlin Heidelberg. 2003, pp. 268–285. DOI: 10.1007/978-3-642-19014-8_13 (cit. on p. 91).

[128] Valeria Simoncini. "Block triangular preconditioners for symmetric saddle-point problems". In: *Applied Numerical Mathematics* 49.1 (2004), pp. 63–80. DOI: 10.1016/j.apnum.2003.11.012 (cit. on pp. 91, 102, 107).

[129] Axel Klawonn. "Block-triangular preconditioners for saddle point problems with a penalty term". In: *SIAM Journal on Scientific Computing* 19.1 (1998), pp. 172–184. DOI: 10.1137/S1064827596303624 (cit. on p. 91).

[130] Carsten Keller, Nicholas I. M. Gould, and Andrew J. Wathen. "Constraint preconditioning for indefinite linear systems". In: *SIAM Journal on Matrix Analysis and Applications* 21.4 (2000), pp. 1300–1317. DOI: 10.1137/S0895479899351805 (cit. on p. 91).

[131] H. Sue Dollar, Nicholas I. M. Gould, Wil H. A. Schilders, and Andrew J. Wathen. "Using constraint preconditioners with regularized saddle-point problems". In: *Computational Optimization and Applications* 36.2–3 (2007), pp. 249–270. DOI: 10.1007/s10589-006-9004-x (cit. on p. 91).

[132] Jordi Castro and Jordi Cuesta. "Quadratic regularizations in an interior-point method for primal block-angular problems". In: *Mathematical Programming, Series A* 130.2 (2011), pp. 415–445. DOI: 10.1007/s10107-010-0341-2 (cit. on p. 91).

[133] Jacek Gondzio and Andreas Grothey. "Parallel interior-point solver for structured quadratic programs: Application to financial planning problems". In: *Annals of Operations Research* 152 (2007), pp. 319–339. DOI: 10.1007/s10479-006-0139-z (cit. on pp. 91, 96, 97, 101).

[134] Luca Bergamaschi, Ángeles Martínez, John W. Pearson, and Andreas Potshcka. "Spectral analysis of block preconditioners for double saddle-point linear systems with application to PDE-constrained optimization". In: *Computational Optimization and Applications* 91.2 (2025), pp. 423–455. DOI: 10.1007/s10589-024-00623-2 (cit. on p. 91).

[135] John W. Pearson and Andreas Potschka. "On symmetric positive definite preconditioners for multiple saddle-point systems". In: *IMA Journal of Numerical Analysis* 44.3 (2024), pp. 1731–1750. DOI: 10.1093/imanum/drad046 (cit. on p. 91).

[136] Yousef Saad and Brian Suchomel. "ARMS: an algebraic recursive multilevel solver for general sparse linear systems". In: *Numerical Linear Algebra with Applications* 9.5 (2002), pp. 359–378. DOI: 10.1002/nla.279 (cit. on p. 91).

[137] Yousef Saad, Azzeddine Soulaimani, and Ridha Touihri. "Variations on algebraic recursive multilevel solvers (ARMS) for the solution of CFD problems". In: *Applied Numerical Mathematics* 51.2–3 (2004), pp. 305–327. DOI: 10.1016/j.apnum.2004.06.017 (cit. on p. 91).

[138] Scott MacLachlan and Yousef Saad. "Greedy coarsening strategies for nonsymmetric problems". In: *SIAM Journal on Scientific Computing* 29.5 (2007), pp. 2115–2143. DOI: 10.1137/060660928 (cit. on p. 91).

[139] Jacek Gondzio and Andreas Grothey. "Exploiting structure in parallel implementation of interior point methods for optimization". In: *Computational Management Science* 6.2 (2009), pp. 135–160. DOI: 10.1007/s10287-008-0090-3 (cit. on p. 91).

[140] Marc C. Steinbach. "Tree-sparse convex programs". In: *Mathematical Methods of Operations Research* 56.3 (2003), pp. 347–376. DOI: 10.1007/s001860200227 (cit. on p. 91).

[141] Marc C. Steinbach. "Recursive direct algorithms for multistage stochastic programs in financial engineering". In: *Operations Research Proceedings 1998*. Springer Berlin Heidelberg. 1999, pp. 241–250. DOI: 10.1007/978-3-642-58409-1_24 (cit. on p. 91).

[142] Sungho Shin, Victor M. Zavala, and Mihai Anitescu. "Decentralized schemes with overlap for solving graph-structured optimization problems". In: *IEEE Transactions on Control of Network Systems* 7.3 (2020), pp. 1225–1236. DOI: 10.1109/TCNS.2020.2967805 (cit. on p. 91).

[143] Jordan Jalving, Sungho Shin, and Victor M. Zavala. "A graph-based modeling abstraction for optimization: concepts and implementation in Plasmo.jl". In: *Mathematical Programming Computation* 14.4 (2022), pp. 699–747. DOI: 10.1007/s12532-022-00223-3 (cit. on p. 91).

[144] Andreas Potschka and Hans Georg Bock. "A sequential homotopy method for mathematical programming problems". In: *Mathematical Programming, Series A* 187.1–2 (2021), pp. 459–486. DOI: 10.1007/s10107-020-01488-z (cit. on p. 93).

[145] Bernhard H. Korte and Jens Vygen. *Combinatorial Optimization: Theory and Applications*. 6th edition. Springer Berlin Heidelberg, 2018. ISBN: 978-3-5407-1843-7 (cit. on p. 94).

[146] Magnus R. Hestenes and Eduard Stiefel. "Methods of conjugate gradients for solving linear systems". In: *Journal of Research of the National Bureau of Standards* 49.6 (1952), pp. 409–436. DOI: 10.6028/jres.049.044 (cit. on p. 99).

[147] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. 2nd. Society for Industrial and Applied Mathematics, 2003. ISBN: 978-0-89871-534-7. DOI: 10.1137/1.9780898718003 (cit. on p. 109).

[148] Achi Brandt, Steve McCormick, and John Ruge. "Algebraic multigrid (AMG) for sparse matrix equations". In: *Sparsity and its Applications*. Ed. by David J. Evans. Cambridge University Press, 1985, pp. 257–284. ISBN: 978-0-521-26272-9 (cit. on p. 109).

[149] John W. Ruge and Klaus Stüben. "Algebraic multigrid". In: *Multigrid Methods*. Ed. by Stephen F. McCormick. Society for Industrial and Applied Mathematics, 1987, pp. 73–130. ISBN: 978-1-61197-105-7. DOI: 10.1137/1.9781611971057.ch4 (cit. on p. 109).

[150] Guido Van Rossum et al. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009 (cit. on p. 112).

[151] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Springer London, 2011. ISBN: 978-0-85729-501-9. DOI: 10.1007/978-3-319-46024-6 (cit. on p. 113).

[152] K. Dadhe and S. Engell. "Robust nonlinear model predictive control: A multi-model non-conservative approach". In: *International Workshop on Assessment and Future Directions on Nonlinear Model Predictive Control*. Pavia, Italy, 2008, p. 24 (cit. on p. 113).

[153] Andreas Potschka. *apotschka/markov-chain-scenario-trees: First release*. Version v1.0.0. 2019 (cit. on pp. 113, 114).

[154] Hans Georg Bock and Karl-Josef Plitt. "A multiple shooting algorithm for direct solution of optimal control problems". In: *IFAC Proceedings Volumes* 17.2 (1984), pp. 1603–1608. ISSN: 1474-6670. DOI: 10.1016/S1474-6670(17)61205-9 (cit. on pp. 116, 119).

[155] James B. Rawlings, David Q. Mayne, and Moritz M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. 2nd edition, 5th printing. Nob Hill Publishing, 2017. ISBN: 978-0-97593-773-0 (cit. on p. 119).

[156] Charbel Farhat. "A Lagrange multiplier based divide and conquer finite element algorithm". In: *Computing Systems in Engineering* 2.2–3 (1991), pp. 149–156. DOI: 10.1016/0956-0521(91)90015-W (cit. on p. 119).

[157] Charbel Farhat, Michel Lesoinne, Patrick LeTallec, Kendall Pierson, and Daniel Rixen. "FETI-DP: a dual–primal unified FETI method—part I: A faster alternative to the two-level FETI method". In: *International Journal for Numerical Methods in Engineering* 50.7 (2001), pp. 1523–1544. DOI: 10.1002/nme.76 (cit. on p. 119).

[158] Patrick Ciarlet Jr., Erell Jamelot, and Félix D. Kpadonou. "Domain decomposition methods for the diffusion equation with low-regularity solution". In: *Computers & Mathematics with Applications* 74.10 (2017), pp. 2369–2384. DOI: 10.1016/j.camwa.2017.07.017 (cit. on pp. 121, 135).

[159] Maya Neytcheva. "On element-by-element Schur complement approximations". In: *Linear Algebra and its Applications* 434.11 (2011), pp. 2308–2324. ISSN: 0024-3795. DOI: 10.1016/j.laa.2010.03.031 (cit. on p. 123).

[160] Yang Cao and Maya Neytcheva. "Cell-by-cell approximate Schur complement technique in preconditioning of meshfree discretized piezoelectric equations". In: *Numerical Linear Algebra with Applications* 28.4 (2021), e2362. DOI: 10.1002/nla.2362 (cit. on p. 123).