Fast Iterative Solvers for PDE-Constrained Optimization Problems

by
John Pearson
Worcester College

Thesis submitted for the degree of

Doctor of Philosophy

Hilary Term 2013

Abstract

In this thesis, we develop preconditioned iterative methods for the solution of matrix systems arising from PDE-constrained optimization problems. In order to do this, we exploit saddle point theory, as this is the form of the matrix systems we wish to solve. We utilize well-known results on saddle point systems to motivate preconditioners based on effective approximations of the $(1,1)$-block and Schur complement of the matrices involved. These preconditioners are used in conjunction with suitable iterative solvers, which include MINRES, non-standard Conjugate Gradients, GMRES and BiCG. The solvers we use are selected based on the particular problem and preconditioning strategy employed.

We consider the numerical solution of a range of PDE-constrained optimization problems, namely the distributed control, Neumann boundary control and subdomain control of Poisson's equation, convection-diffusion control, Stokes and Navier-Stokes control, the optimal control of the heat equation, and the optimal control of reaction-diffusion problems arising in chemical processes. Each of these problems has a special structure which we make use of when developing our preconditioners, and specific techniques and approximations are required for each problem. In each case, we motivate and derive our preconditioners, obtain eigenvalue bounds for the preconditioners where relevant, and demonstrate the effectiveness of our strategies through numerical experiments. The goal throughout this work is for our iterative solvers to be feasible and reliable, but also robust with respect to the parameters involved in the problems we consider.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor Andy Wathen. I have been hugely fortunate to have been able to work with a knowledgeable, inspiring and supportive researcher. I am very grateful to him for introducing me to such an interesting area of research, and for all the help and encouragement he has given me over the past three and a bit years. The comments he has made on initial drafts of this thesis have also been invaluable.

I would like to thank Martin Stoll, with whom I have had the pleasure to work on a number of projects, including some described in this thesis. I am grateful to him for inviting me to visit the Max Planck Institute in Magdeburg on two occasions, for his hospitality during my stays, and for all the insights he has given me during my time as a DPhil student.

There have been countless other people who have assisted and advised me along the way. I would like to thank Tyrone Rees for sharing his research with me during the early months of my DPhil. I am also grateful to Sue Thorne, Jen Pestana, Stefan Takacs, David Silvester, Roland Herzog, Walter Zulehner and many others for conversations we have had about this work and linear algebra more generally, and to Nick Gould and David Kay for their very helpful comments when acting as examiners for my Transfer Thesis. I am thankful to Kathryn Gillow, Sheehan Olver and Mason Porter for introducing me to mathematical research in the first place during my time as an MSc student, to Mick Pont, David Sayers and others for inviting me to work at the Numerical Algorithms Group for a month in 2010 and for making me feel very welcome there, to Nick Trefethen, Endre Süli and others within the Numerical Analysis Group for useful discussions we have had about the subject more generally and for sharing their knowledge, and to Lotti Ekert for her assistance on a number of occasions.

I gratefully acknowledge the Engineering and Physical Sciences Research Council (EPSRC) for funding my DPhil studies, as well as for the award of an EPSRC Doctoral Prize, which I will take up having completed my doctorate. I am also very grateful to the European Science Foundation (ESF) for the award of an Exchange Grant under the OPTPDE scheme, which enabled me to visit the Max Planck Institute in Magdeburg for two months in 2012. I have also been awarded financial support to attend conferences and meetings by the Mathematical Institute, the Oxford Centre for Collaborative and Applied Mathematics (OCCAM), Worcester College, the Society

for Industrial and Applied Mathematics (SIAM), the Institute of Mathematics and its Applications (IMA), the Numerical Algorithms Group (NAG), the University of Manchester, the Max Planck Institute in Magdeburg, the organizers of the 1st Gene Golub Summer School in 2009, the organizers of the Scottish Computational Mathematics Symposium in 2009, the organizers of the workshop Linear Algebra Aspects of Solving PDEs with Random Data in 2012, and the research fund of my supervisor – I am extremely thankful for this funding which has greatly benefited my research.

I am grateful to Ásgeir Birkisson and Mark Richardson, along with whom I have studied for my DPhil, as well as my MSc course before that. I thank them for the memorable conversations, and for helping to make my time as a graduate student a lot of fun. I am also grateful to the rest of the Numerical Analysis Group, within which I have met many great mathematicians and great friends. I thank also those who have departed the group for the memories they have left behind, especially Pedro, Stefan and Ivo. I thank my former housemates Emily, Jess, Kate and Penny, all my friends from Worcester College and Oxford more widely, as well as my friends back home in London, for all the good times.

Last but not least, I am hugely thankful to my family, whose support and love over the years has helped me immeasurably along the way.

Contents

CHAPTER 1

Introduction

For a period of many decades, or in some cases centuries, the analytical and numerical study of partial differential equations (PDEs) has been a field of active interest and considerable research in applied mathematics, and within the scientific community more generally. In scientific computing, a major focus has been the development of potent numerical techniques for solving such problems. Such methods comprise two components. First, a method for representing the continuous solution of a PDE (for instance finite differences, finite elements, spectral methods or radial basis functions) needs to be exploited. Each of these strategies generates a matrix system that has to be solved. The second component of the numerical technique must therefore be the devising of a method for solving the resulting matrix system – either direct or iterative. As the matrix systems arising from the solution of such problems are sparse and frequently of very high dimension, it is often preferable to apply iterative methods to solve these systems. Within the range of iterative methods for such problems, there are two main classes of approach: those of multigrid techniques and those of preconditioned iterative solvers. As preconditioned iterative solvers often have considerable applicability to these problems, it is the exploration of suitable preconditioners, for use within appropriate iterative methods, that we are interested in within this thesis.

More recently, a class of problems that has generated a great deal of attention in the field of applied mathematics and numerical analysis, as well as applied sciences

more widely, is that of PDE-constrained optimization problems [70, 122]. The difference between these problems and PDEs themselves is that within PDE-constrained optimization problems, a certain functional is aimed to be minimized with one or more PDEs serving as constraints within the problem. A large amount of study into this field has been theoretical, but recently there has been much interest in developing efficient numerical techniques for discretized versions of such problems. The classes of numerical methods for solving these problems can be divided up as for the numerical solution of PDEs, and often the methods for PDE-constrained optimization exploit techniques applied successfully to solve the forward PDEs.

Throughout this thesis, we consider the development of preconditioned iterative methods for solving such PDE-constrained optimization problems. For each problem we consider, the matrix system for which we develop preconditioners arises from a finite element discretization of the problem. Moreover, each preconditioner that we construct is justified by the fact that the matrix systems we solve, like matrices arising from more general problems involving optimization with constraints, are of saddle point structure. Therefore, as we will explain later, a large part of the work which is required to develop effective methods for solving these problems involves finding good approximations to the $(1,1)$-block and Schur complement of the matrix systems, and then incorporating these approximations into suitable iterative methods. We find that for a large number of problems, we are able to find approximations that are not only feasible, but result in fast iterative solution schemes for the problem at hand. The ultimate goal when constructing these solvers is to ensure that the iterative method is feasible for a wide range of parameters involved in the problem. This objective may be characterized by two properties. Firstly, if the matrix system doubles in dimension due to decreasing the step-size in space or time, the computation time for solving the problem, in the best case, should roughly double as well – this is one property we desire our solvers to exhibit. Secondly, if a parameter involved in the problem that does not directly affect the size of the matrix system but does affect its numerical features (usually a regularization parameter involved in the PDE-constrained optimization problem) is changed, we hope that the performance of our iterative method will not radically worsen. Our aim is to develop fast iterative solvers that satisfy these two properties – if we are successful we say that we have a robust solver for the problem being considered.

The types of problems that arise in the field of PDE-constrained optimization vary enormously, be it through the functional being minimized, the PDEs acting

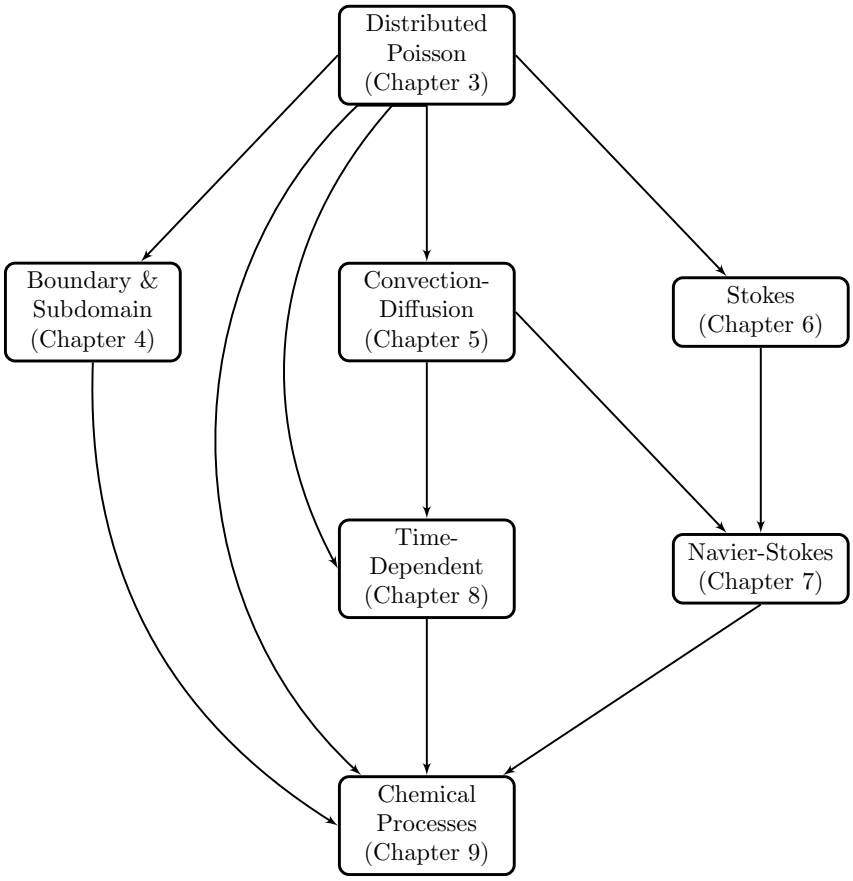


Figure 1.1: Flow chart outlining the problems of which we seek the numerical solution in this thesis, and their links to each other from a preconditioning point-of-view.

as constraints, the imposition of additional inequality constraints on the variables involved in the problem, what types of boundary conditions are specified, or what parts of the domain the variables are measured on. As such, in this thesis, we aim to develop effective preconditioned iterative methods for a number of important PDE-constrained optimization problems. The problems that we tackle in this thesis are summarized in Figure 1.1, which also serves as a guide to the structure of the thesis, as well as the interdependence of methods for different problems we look at. We will return to this figure when we make concluding comments in Chapter 10.

This thesis is structured as follows. In Chapter 2, we detail background theory of PDE-constrained optimization problems, discuss basic linear algebra related to saddle point systems, and introduce iterative methods which we will utilize for the solution of the systems. In Chapter 3, we describe the iterative solution of the distributed Poisson control problem, one of the most fundamental PDE-constrained optimization prob-

lems. In Chapter 4, we will extend this theory to explain how to precondition matrix systems arising from variants of this problem, specifically boundary control problems, subdomain control problems, and problems with additional state constraints. In Chapter 5, we derive and test iterative solvers for the convection-diffusion control problem, which is a more physically realistic problem than the Poisson control problem as it takes account of both convective and diffusive terms within a physical system. In Chapter 6, we explain how our methodology can be extended to solving Stokes control problems, an important type of PDE-constrained optimization problem in fluid dynamics, and in Chapter 7 we extend this to more complex Navier-Stokes control problems. In Chapter 8, we build on our theory to explain how we tackle time-dependent PDE-constrained optimization problems, specifically considering the heat equation. In Chapter 9, we discuss an iterative solution approach for a reaction-diffusion control problem which models chemical processes – this is a problem geared towards a more industrial application, and our method for solving this problem uses and builds on many of the components discussed for other problems. Finally in Chapter 10, we make some concluding remarks and discuss possible avenues for future work.

CHAPTER 2

Background

In this chapter, we wish to provide background information to the problems and methods considered in the remainder of this thesis. This summary is split into three parts. In Section 2.1, we provide a basic overview of PDE-constrained optimization problems. We discuss a variety of such problems which may be considered, ranging from distributed control, boundary control and subdomain control problems, problems with additional state or control constraints, time-dependent problems, and others. Particular attention is paid to the matrix systems obtained using standard conforming finite element methods for solving these problems. In Section 2.2, we introduce the concept of preconditioning matrix systems, and describe the class of systems (namely saddle point systems) of which the iterative solution is considered in this thesis. Finally in Section 2.3, we detail the iterative solvers that we will seek to employ throughout this thesis. In particular, we outline the Chebyshev semi-iteration, multigrid, MINRES, Conjugate Gradient, GMRES and BiCG methods. We aim to provide some background on the motivation of the methods, state in what circumstances they are applied, and detail some results on their convergence. We also state exactly what we will use each of the iterative methods for within the remainder of this thesis.

2.1 Introduction to PDE-Constrained Optimization

One of the most fundamental types of problems within applied mathematics is the solution of partial differential equations. A major class of such problems is that of Dirichlet boundary value problems, which we express in the following general form:

$$\begin{aligned} \mathcal{L}y &= f, \quad \text{in } \Omega, \\ y &= g, \quad \text{on } \partial\Omega, \end{aligned} \tag{2.1}$$

where y is the variable for which the PDE needs to be solved, the function f denotes the *source term* of the PDE, and the function g corresponds to Dirichlet boundary conditions. $\mathcal{L}$ here denotes some differential operator. A problem of this type is often solved on a domain $\Omega \times [0, T]$, where $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$, is a spatial domain with boundary $\partial\Omega$, and $[0, T]$ is some time-interval. Such PDEs can describe a range of real-world processes, and therefore much attention has been paid to the analytical and numerical solution of these problems.

In this thesis, we wish to consider the solution of *PDE-constrained optimization problems*. In this case a PDE is not the problem itself, but merely serves as a constraint subject to which some cost functional is sought to be minimized. In this section, we aim to provide some background as to the types of PDE-constrained optimization problems which are typically solved.

We motivate the class of problems to be examined by stating the following PDE-constrained optimization problem, which relates to the PDE (2.1). This is known as a *distributed control problem* (or *tracking type problem*).

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \|y - \widehat{y}\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & \mathcal{L}y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega. \end{aligned} \tag{2.2}$$

Here, y denotes the *state variable*, with $\widehat{y}$ some *desired state*, and u represents the *control variable*. The value $\beta > 0$ is the *regularization parameter* (frequently referred to as the *penalty parameter* or *Tikhonov parameter*). The L_2-norm is used to measure how "close" the state variable is to the desired state, as well as the "size" of the control

variable, though we note that other norms may be used. We find that it is natural to use a Hilbert space norm, as this choice leads cleanly to the solution of the problem using a finite element method. We typically assume that the desired state $\widehat{y}$ belongs to $L_2(\Omega)$.

One may observe that the motivation of a problem of this type is to attempt to make the state variable "as close as possible" to a desired state, while also penalizing the input of a large control into the system. The constraint subject to which this is carried out arises in the form of a PDE. We may think of this in physical terms as penalizing the input of energy into a physical system.

For instance, let us say a company wishes to store a foodstuff or chemical as close as possible to certain "ideal" atmospheric conditions, and sets up a controlled environment to do this. The reason for doing this could be that it is known the substance is best preserved in certain conditions. However, carrying this out is likely to be very expensive financially, so another consideration would be to minimize this cost. This problem could potentially be solved by a PDE-constrained optimization problem. The state variable in this case may therefore be given by temperature, humidity, light intensity, or a combination of these parameters. The desired state would correspond to an ideal set of atmospheric conditions, and the control variable could be the energy put into the system to achieve these conditions. The PDE in the problem set-up could involve the heat equation, or an equation of similar structure. Solving the PDE-constrained optimization problem would hence involve finding a set of atmospheric conditions close to the 'ideal' conditions, while penalizing to some extent the energy required to create these conditions.

The penalty parameter β therefore plays an important role physically – for small β the control variable is not heavily penalized, and so the state can typically be close to the desired state, but for large β the input of control into the system contributes more heavily to the functional which is sought to be minimized, and so it is harder for the state variable to be near to the desired state in the L_2-norm. For instance, in the above physical example, a large value of β will result in more care being given to minimizing the energy consumption used, whereas a small β corresponds to the conditions of the foodstuff or chemical being prioritised, with less importance attached to the amount of energy expended.

As the essence of PDE-constrained optimization problems is finding the profile of a control variable which results in a state variable being close to some desired profile, these problems are often referred to as *optimal control problems*. Indeed the

optimization problem (2.2) is referred to as a distributed control problem because the control variable is being applied over the entire domain – we note that the control and the source term of the PDE constraint in this case are one and the same. This is not the only type of PDE-constrained optimization problem which we may consider; we will outline a number of alternative formulations in this section.

There is a plethora of industrial and scientific applications for PDE-constrained optimization problems. We refer to [59] for details of a range of such studies. Examples of practical applications of PDE-constrained optimization problems include flow control [55, 82], optimal semiconductor design [69], electromagnetic inverse problems [56, 57], the Monge-Kantorovich mass transfer problem [1, 8, 10], reaction-diffusion problems from chemical processes [5, 49, 50, 51, 52], parameter identification problems within pattern formation from mathematical biology [42, 43], medical imaging and tomography [2, 26, 28, 76, 78], problems in financial markets and option pricing [18, 33, 34], shape optimization problems [84, 97] and other parameter estimation problems [58]. This provides only a handful of useful PDE-constrained optimization problems; there are many more examples in the literature.

2.1.1 Distributed Control of Poisson's Equation

We now wish to describe how to attempt to solve a distributed control problem of the form (2.2). Let us consider for this demonstration the distributed control of Poisson's equation, i.e. the problem

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \left\| y - \widehat{y} \right\|^2_{L_2(\Omega)} + \frac{\beta}{2} \left\| u \right\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega, \end{aligned} \tag{2.3}$$

which is of the general form (2.2), with $\mathcal{L} = -\nabla^2$. Now, as for any PDE-constrained optimization problem, there are two methods available for solving this. Firstly, one can use a *discretize-then-optimize approach*, which involves forming a discrete cost functional and deriving discrete optimality conditions from these. Secondly, one can derive optimality conditions in the continuous setting, and then discretize these – this is known as the *optimize-then-discretize approach*. We now explain these two methods using the problem (2.3) as an example. Our work throughout this thesis is

based on applying a finite element method to these problems.[1] For an overview of the finite element method, we recommend [23, 36, 75].

To provide an introduction, we now consider how the finite element method is applied to the PDE constraint in (2.3). We first define the following spaces for this problem:

$$\begin{aligned}
L_2(\Omega) &= \left\{ v : \int_\Omega |v|^2 \, \mathrm{d}\Omega < \infty \right\}, \\
H^1(\Omega) &= \left\{ v : v \in L_2(\Omega), \frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2}, \frac{\partial v}{\partial x_3} \in L_2(\Omega) \right\}, \\
H^1_E(\Omega) &= \left\{ v : v \in H^1(\Omega), v = g \text{ on } \partial\Omega \right\}, \\
H^1_{E_0}(\Omega) &= \left\{ v : v \in H^1(\Omega), v = 0 \text{ on } \partial\Omega \right\},
\end{aligned}$$

where $\mathbf{x} = [x_1, x_2, x_3]^T$ is the coordinate system in Ω that we use. (Of course if $d = 2$, then this will reduce to $\mathbf{x} = [x_1, x_2]^T$.)

We may now state the weak formulation for the PDE constraint in (2.3). This is typically written [122] as finding $y \in H^1_E(\Omega)$ and $u \in L_2(\Omega)$ such that

$$\int_\Omega \nabla y \cdot \nabla v \, \mathrm{d}\Omega = \int_\Omega uv \, \mathrm{d}\Omega, \quad \forall v \in H^1_{E_0}(\Omega).$$

Let us now select v to be from a set of *Galerkin test functions* $V_{0,h} \subset H^1_{E_0}(\Omega)$. Suppose we take $\{\phi_1, \phi_2, ..., \phi_n\}$ to be a finite element basis for this test space, i.e. we have n basis functions centred within the interior of the domain Ω. Then one common approach, and the one which we will consider for this problem, is to take the finite element approximations y_h, u_h to y, u to be from this same space, and write

$$y_h = \sum_{j=1}^{n} Y_j \phi_j + \sum_{j=n+1}^{n+n_\partial} Y_j \phi_j, \quad u_h = \sum_{j=1}^{n} U_j \phi_j + \sum_{j=n+1}^{n+n_\partial} U_j \phi_j, \tag{2.4}$$

for some coefficients $\{Y_j, U_j, j = 1, ..., n + n_\partial\}$. Here, we have n_∂ nodes on to the boundary $\partial\Omega$, and corresponding finite element basis functions $\{\phi_{n+1}, \phi_{n+2}, ..., \phi_{n+n_\partial}\}$. Note that $Y_{n+1}, ..., Y_{n+n_\partial}$ correspond to the Dirichlet boundary conditions $y = g$, and we later find that $U_{n+1}, ..., U_{n+n_\partial}$ are in fact all equal to zero.

[1] It is possible to solve these problems using methods other than finite element approaches. For instance in [88], radial basis function methods for the solution of PDE-constrained optimization problems are derived and discussed.

We may then write our problem as finding y_h and u_h that belong to $V_{E,h} := \text{span}\{\phi_1, ..., \phi_n, \phi_{n+1}, ..., \phi_{n+n_\partial}\}$, such that

$$\int_\Omega \nabla y_h \cdot \nabla v_h \, \mathrm{d}\Omega = \int_\Omega u_h v_h \, \mathrm{d}\Omega, \quad \forall v_h \in V_{0,h}. \tag{2.5}$$

Finally, substituting (2.4) into (2.5) gives

$$\sum_{j=1}^{n+n_\partial} Y_j \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \, \mathrm{d}\Omega - \sum_{j=1}^{n+n_\partial} U_j \int_\Omega \phi_i\phi_j \, \mathrm{d}\Omega = 0, \quad i = 1, ..., n.$$

This is one of the systems of equations needed to be solved as part of this optimal control problem.

Of course there are many different solutions for y_h and u_h here, but the additional constraints provided by the minimization problem (2.3) generally give rise to a unique solution.

We highlight once more that the assumptions on the finite element solution are typically that $y_h \in H^1(\Omega)$ and $u_h \in L_2(\Omega)$ (in particular y_h and u_h do not have to belong to the same finite element spaces, as we have taken to be the case here). The finite element approximation of the adjoint variable, which we will introduce in the next section, should also belong to $H^1(\Omega)$.

Discretize-then-Optimize. Having motivated the finite element method which we employ for the distributed Poisson control problem, we now consider the two alternative strategies for solving PDE-constrained optimization problems. First, let us consider the discretize-then-optimize strategy. For this, it is important that we study the PDE constraint in the discrete setting. In a finite element formulation, we may use (2.6) to write this constraint as

$$K\mathbf{y} - M\mathbf{u} = \mathbf{g},$$

where $\mathbf{y}$ and $\mathbf{u}$ are vectors corresponding to the coefficients $\{Y_i, i = 1, ..., n\}$ and $\{U_i, i = 1, ..., n\}$. Here, the matrices K and M are the well-known finite element

stiffness and *mass matrices*, defined by

$$K = \{k_{ij}\}_{i,j=1,\dots,n}, \quad k_{ij} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \, \mathrm{d}\Omega, \tag{2.6}$$

$$M = \{m_{ij}\}_{i,j=1,\dots,n}, \quad m_{ij} = \int_\Omega \phi_i\phi_j \, \mathrm{d}\Omega. \tag{2.7}$$

It can easily be shown that the matrices K and M are symmetric positive definite (meaning all their eigenvalues are positive). The vector $\mathbf{g}$ takes account of the Dirichlet boundary condition imposed. We point out that the matrix M as defined in (2.7) is generally referred to as a *consistent mass matrix*. In scientific applications *lumped mass matrices*, defined by

$$\bar{M} = \mathrm{diag}(\bar{m}_{ii}), \quad \bar{m}_{ii} = \sum_j \int_\Omega \phi_i\phi_j \, \mathrm{d}\Omega,$$

are also frequently used. As discussed in [54] for instance, such a lumping strategy is related to approximating a consistent mass matrix using a trapezoidal quadrature rule.

In the discretize-then-optimize formulation, the discrete cost functional that we wish to minimize is given by

$$J_D(\mathbf{y}, \mathbf{u}) = \frac{1}{2} \left\| y_h - \widehat{y} \right\|^2_{L_2(\Omega)} + \frac{\beta}{2} \left\| u_h \right\|^2_{L_2(\Omega)}.$$

Let us now examine the individual terms of J_D in more detail. The first term, upon expanding in terms of finite element basis functions, may be written as follows:

$$\begin{aligned}
\frac{1}{2} \left\| y_h - \widehat{y} \right\|^2_{L_2(\Omega)} &= \frac{1}{2} \int_\Omega \left(\sum_i Y_i\phi_i - \widehat{y} \right) \cdot \left(\sum_j Y_j\phi_j - \widehat{y} \right) \, \mathrm{d}\Omega \\
&= \frac{1}{2} \sum_i \sum_j Y_iY_j \int_\Omega \phi_i\phi_j \, \mathrm{d}\Omega - \sum_i Y_i \int_\Omega \widehat{y}\phi_i \, \mathrm{d}\Omega + \frac{1}{2} \int_\Omega \widehat{y}^2 \, \mathrm{d}\Omega \\
&= \frac{1}{2}\mathbf{y}^T M \mathbf{y} - \mathbf{y}^T \mathbf{z} + \bar{C},
\end{aligned}$$

where the vector $\mathbf{z}$ contains entries of the form $\int_\Omega \widehat{y}\phi_i \, \mathrm{d}\Omega$, and $\bar{C}$ is a constant inde-

pendent of y and u. Similarly, the second term of $J_D(\mathbf{y}, \mathbf{u})$ can be written as

$$\frac{\beta}{2} \|u_h\|^2_{L_2(\Omega)} = \frac{\beta}{2} \int_\Omega \left(\sum_i U_i \phi_i \right) \cdot \left(\sum_j U_j \phi_j \right) \, \mathrm{d}\Omega = \frac{\beta}{2} \mathbf{u}^T M \mathbf{u}.$$

Putting these two terms together gives that

$$J_D(\mathbf{y}, \mathbf{u}) = \frac{1}{2} \mathbf{y}^T M \mathbf{y} - \mathbf{y}^T \mathbf{z} + \bar{C} + \frac{\beta}{2} \mathbf{u}^T M \mathbf{u}.$$

The Lagrangian that we therefore need to work with in the discretize-then-optimize setting will combine the cost functional $J_D(\mathbf{y}, \mathbf{u})$ and a Lagrange multiplier term which enforces the PDE constraint. This Lagrangian is hence given by

$$\mathcal{L}_{DTO}(\mathbf{y}, \mathbf{u}, \mathbf{p}) = \frac{1}{2} \mathbf{y}^T M \mathbf{y} - \mathbf{y}^T \mathbf{z} + \bar{C} + \frac{\beta}{2} \mathbf{u}^T M \mathbf{u} + \mathbf{p}^T (K\mathbf{y} - M\mathbf{u} - \mathbf{g}),$$

where the vector $\mathbf{p}$ corresponds to the coefficients $\{P_i, i = 1, ..., n\}$ for the finite element approximation of the Lagrange multiplier (or adjoint variable) p, that is

$$p_h = \sum_{j=1}^{n} P_j \phi_j + \sum_{j=n+1}^{n+n_\partial} P_j \phi_j.$$

The optimality conditions are obtained by differentiating $\mathcal{L}_{DTO}$ with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$, and setting $\frac{\partial \mathcal{L}}{\partial Y_i} = \frac{\partial \mathcal{L}}{\partial U_i} = \frac{\partial \mathcal{L}}{\partial P_i} = 0$ for $i = 1, ..., n$. Examining the first of these conditions gives that

$$\frac{\partial \mathcal{L}}{\partial Y_i} = 0 \quad \Rightarrow \quad M\mathbf{y} - \mathbf{z} + K\mathbf{p} = \mathbf{0},$$

which is called the *adjoint equation*. The second condition produces

$$\frac{\partial \mathcal{L}}{\partial U_i} = 0 \quad \Rightarrow \quad \beta M\mathbf{u} - M\mathbf{p} = \mathbf{0},$$

which is usually called the *gradient equation*. Finally, we obtain from the third condition

$$\frac{\partial \mathcal{L}}{\partial P_i} = 0 \quad \Rightarrow \quad K\mathbf{y} - M\mathbf{u} - \mathbf{g} = \mathbf{0},$$

which is the *state equation*, i.e. the discretized version of the forward problem.

Combining these three systems of equations[2] gives the following matrix system [100, 122]:

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix}. \tag{2.8}$$

It is this matrix system that we focus on solving in Chapter 3. We note that this system can be of very high dimension for practical problems, so finding effective iterative solvers for this system is an important area of research.

Optimize-then-Discretize. In this formulation, for which we aim to provide the basic motivation rather than the full technical details,[3] we seek to consider the continuous Lagrangian

$$\begin{aligned} \mathcal{L}_{OTD}(y, u, p_\Omega, p_{\partial\Omega}) = \frac{1}{2} \|y - \widehat{y}\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u\|^2_{L_2(\Omega)} \\ + \int_\Omega (-\nabla^2 y - u) p_\Omega \, \mathrm{d}\Omega + \int_{\partial\Omega} (y - g) p_{\partial\Omega} \, \mathrm{d}s, \end{aligned}$$

where here the continuous adjoint variable p has been divided up into interior and boundary components, p_Ω and $p_{\partial\Omega}$, in order to enforce the PDE and boundary condition constraints separately.

We now follow the approach discussed in [99, 122], which is to take the Fréchet derivatives with respect to y, u, p_Ω and $p_{\partial\Omega}$ in some direction $\widetilde{h}$, and use the fact that for the optimal solutions of state, control and adjoint, the derivatives will all be zero.

Using this method, taking the Fréchet derivative with respect to p_Ω and $p_{\partial\Omega}$ in turn recovers the forward problem, or continuous *state equation*

$$\begin{aligned} -\nabla^2 y &= u, \quad \text{in } \Omega, \\ y &= g, \quad \text{on } \partial\Omega. \end{aligned}$$

Furthermore, taking the Fréchet derivative with respect to u in the direction of

[2] We note that the conditions represented within this matrix system are often referred to as the *first-order stationarity conditions* or *Karush-Kuhn-Tucker (KKT) conditions.*

[3] We recommend [122, Chapter 2] for a detailed derivation of the optimality conditions for this approach.

some $\widetilde{h} \in L_2(\Omega)$ gives

$$\int_\Omega (\beta u - p)\widetilde{h} \, \mathrm{d}\Omega = 0, \quad \forall \widetilde{h} \in L_2(\Omega),$$

and hence that $\beta u - p = 0$ almost everywhere. This is the continuous *gradient equation*.

Finally, considering the optimal solution when differentiating $\mathcal{L}_{OTD}$ with respect to y, we obtain that

$$\int_\Omega (y - \widehat{y})\widetilde{h} \, \mathrm{d}\Omega + \int_\Omega -\nabla^2 \widetilde{h} \cdot p_\Omega \, \mathrm{d}\Omega + \int_{\partial\Omega} \widetilde{h} p_{\partial\Omega} \, \mathrm{d}s = 0,$$

which yields, using the Divergence Theorem,

$$\int_\Omega (y - \widehat{y})\widetilde{h} \, \mathrm{d}\Omega - \int_\Omega \widetilde{h}\nabla^2 p_\Omega \, \mathrm{d}\Omega + \int_{\partial\Omega} \left(\frac{\partial \widetilde{h}}{\partial n} p_\Omega - \widetilde{h} \frac{\partial p_\Omega}{\partial n} \right) \mathrm{d}s + \int_{\partial\Omega} \widetilde{h} p_{\partial\Omega} \, \mathrm{d}s = 0. \tag{2.9}$$

From here, as in [99, Chapter 2], we first consider the case $\widetilde{h} \in C_0^\infty(\Omega)$ (which we take to mean that the value and normal derivative of $\widetilde{h}$ vanish at the boundary, i.e. $\widetilde{h} = \frac{\partial \widetilde{h}}{\partial n} = 0$ on $\partial\Omega$). We may write using (2.9) that

$$\int_\Omega (-\nabla^2 p_\Omega + y - \widehat{y})\widetilde{h} \, \mathrm{d}\Omega = 0, \quad \widetilde{h} \in C_0^\infty(\Omega).$$

We then consider the case $\widetilde{h} \in H_0^1(\Omega)$ (and hence that $\widetilde{h} = 0$ on $\partial\Omega$), which gives

$$\int_{\partial\Omega} \frac{\partial \widetilde{h}}{\partial n} p_\Omega \, \mathrm{d}s = 0, \quad \widetilde{h} \in H_0^1(\Omega).$$

These equations give, along with the Fundamental Lemma of the Calculus of Variations, that

$$\begin{aligned} -\nabla^2 p_\Omega &= \widehat{y} - y, \quad \text{in } \Omega, \\ p_\Omega &= 0, \quad \text{on } \partial\Omega. \end{aligned}$$

Furthermore, these equations along with (2.9) give that $p_{\partial\Omega} = \frac{\partial p_\Omega}{\partial n}$ on Ω. Hence, we

conclude that

$$-\nabla^2 p = \widehat{y} - y, \quad \text{in } \Omega,$$
$$p = 0, \quad \text{on } \partial\Omega,$$

which is the continuous *adjoint equation*.

If we now discretize the continuous state, gradient and adjoint equations using the same basis functions as for the discretize-then-optimize approach, we obtain the same matrix system:

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix},$$

as we derived for that approach. We wish to point out that it is possible to discretize the continuous optimality conditions in such a way that this matrix system is not the same as for the discretize-then-optimize method. In this optimize-then-discretize setting, the choice of discretization of the optimality conditions is thus crucial. We observe that the state and adjoint equations both involve the same PDE (Poisson's equation), which needs to be solved for y and p. This motivates the use of the same finite element space for the discrete variables y_h and p_h.

We note that there can be other considerations which result in the matrix systems corresponding to the discretize-then-optimize and optimize-then-discretize methods not being the same. One is presented in Section 2.1.4 and is caused by the choice of time-stepping scheme for time-dependent PDE-constrained optimization problems. Another is discussed in Chapter 5, and results from particular stabilization strategies for convection-dominated problems. In fact if certain methods are chosen for this latter problem, the optimize-then-discretize matrix system is not even symmetric, whereas the discretize-then-optimize matrix system is guaranteed to be symmetric by construction of the method. The symmetry of the discretize-then-optimize approach is in some sense more natural from an optimization point-of-view. Of course it is ideal for the operations of discretization and optimization to commute, and hence that the two resulting matrix systems are the same – we will repeatedly return to this point throughout this thesis.

We note also that it is perfectly feasible to impose Neumann boundary conditions for this distributed control problem, as opposed to the Dirichlet conditions speci-

fied here. The matrix system (for either discretize-then-optimize or optimize-then-discretize) would look the same, except the mass and stiffness matrices would have a marginally different structure. In particular, when Neumann boundary conditions are considered, the matrix M would take account of boundary nodes as well as interior nodes and hence would be enlarged, and K would be a discrete Neumann stiffness matrix rather than a Dirichlet stiffness matrix. In this case the spectral properties of K would be slightly different – in particular K will have one zero eigenvalue, relating to the constant function being a solution of the continuous Neumann problem for Poisson's equation.

2.1.2 Neumann Boundary Control and Subdomain Control of Poisson's Equation

The distributed control problem introduced in the previous section is not the only type of optimal control problem that we may consider. We now wish to examine two problems of a similar flavour to the distributed Poisson control problem: those of Neumann boundary control and subdomain control.

Neumann Boundary Control Problems. In the class of *boundary control problems* the control is applied only at the boundary, as opposed to distributed control problems where it is applied on the entire domain. Such boundary control problems are perhaps more physically realistic, as one can imagine that in real-world applications, in particular applications involving flow, it may only be possible to "access" or "control" the physical features of the boundary of the domain. We therefore wish to investigate such problems. Perhaps the main class of such problems is that of *Neumann boundary control problems* – for Poisson's equation, such a problem is expressed as

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \left\| y - \widehat{y} \right\|_{L_2(\Omega)}^2 + \frac{\beta}{2} \left\| u \right\|_{L_2(\partial\Omega)}^2 \\ \text{s.t.} \ & -\nabla^2 y = f, \quad \text{in } \Omega, \\ & \frac{\partial y}{\partial n} = u, \quad \text{on } \partial\Omega. \end{aligned} \tag{2.10}$$

Here the control is applied in the form of the Neumann boundary condition of the PDE, and the penalization term on the control is now measured as a boundary integral, rather than an integral over the entire domain.

When trying to solve such a problem, it is a natural idea to seek a finite element solution of the form

$$y_h = \sum_{j=1}^{n+n_\partial} Y_j \phi_j, \quad u_h = \sum_{j=1}^{n_\partial} U_j \chi_j, \quad p_h = \sum_{j=1}^{n+n_\partial} P_j \phi_j. \tag{2.11}$$

Clearly, in this case, the state and control must consist of different basis functions $\{\phi_j\}_{j=1,\dots,n+n_\partial}$ and $\{\chi_j\}_{j=1,\dots,n_\partial}$, as one set of functions corresponds to the entire domain, and one solely to its boundary.

The weak formulation of the PDE constraint can be stated as finding some $y \in H^1(\Omega)$ and $u \in L_2(\Omega)$ such that [99, 122]

$$\int_\Omega \nabla y \cdot \nabla v \, \mathrm{d}\Omega - \int_{\partial\Omega} u \cdot \mathrm{tr}(v) \, \mathrm{d}s = \int_\Omega f v \, \mathrm{d}\Omega, \quad \forall v \in H^1(\Omega),$$

where $\mathrm{tr}(\cdot)$ denotes the *trace function*, which restricts v to its values on $\partial\Omega$. Substituting (2.11) into the weak form, and writing the resulting expression in matrix form, gives

$$K\mathbf{y} - N_b\mathbf{u} = \mathbf{f}, \tag{2.12}$$

where K here is the stiffness matrix for the Neumann problem, the matrix N_b contains entries

$$N_b = \{n_{b,ij}\}_{i=1,\dots,n+n_\partial,\ j=1,\dots,n_\partial}, \quad n_{b,ij} = \int_{\partial\Omega} \mathrm{tr}(\phi_i) \cdot \chi_j \, \mathrm{d}\Omega, \tag{2.13}$$

and the vector $\mathbf{f}$ corresponds to the source term of the PDE, that is

$$\mathbf{f} = \{f_i\}_{i=1,\dots,n+n_\partial}, \quad f_i = \int_\Omega f\phi_i \, \mathrm{d}\Omega,$$

where f_i, $i = 1, \dots, n + n_\partial$, would be evaluated by a quadrature rule.

Looking for a discretize-then-optimize solution involves minimizing the following discrete cost functional:

$$J_D(\mathbf{y}, \mathbf{u}) = \frac{1}{2} \|y_h - \widehat{y}\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u_h\|^2_{L_2(\partial\Omega)} = \frac{1}{2}\mathbf{y}^T M \mathbf{y} - \mathbf{y}^T \mathbf{z} + \bar{C} + \frac{\beta}{2}\mathbf{u}^T M_b \mathbf{u},$$

which may be derived in the same way as for the distributed control problem of

the previous section. Here, M is a mass matrix for the Neumann problem, M_b is a boundary mass matrix with entries

$$M_b = \{m_{b,ij}\}_{i,j=1,\dots,n_\partial}, \quad m_{b,ij} = \int_{\partial\Omega} \chi_i \chi_j \, \mathrm{d}s, \tag{2.14}$$

and $\bar{C}$ as before is a constant independent of y, u and p.

Combining this cost functional with the PDE constraints (2.12) leads to the Lagrangian

$$\begin{aligned} \mathcal{L}_{DTO}(\mathbf{y}, \mathbf{u}, \mathbf{p}) &= \frac{1}{2} \|y_h - \widehat{y}\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u_h\|^2_{L_2(\partial\Omega)} + \mathbf{p}^T(K\mathbf{y} - N_b\mathbf{u} - \mathbf{f}) \\ &= \frac{1}{2}\mathbf{y}^T M\mathbf{y} - \mathbf{y}^T\mathbf{z} + \bar{C} + \frac{\beta}{2}\mathbf{u}^T M_b\mathbf{u} + \mathbf{p}^T(K\mathbf{y} - N_b\mathbf{u} - \mathbf{f}), \end{aligned} \tag{2.15}$$

where $\mathbf{p}$ once again denotes the discretized Lagrange multiplier, or adjoint variable.

Differentiating $\mathcal{L}_{DTO}$ from (2.15) with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ gives the following matrix system:

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M_b & -N_b^T \\ K & -N_b & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{f} \end{bmatrix}.$$

It can be shown (as in [99, Chapter 2] for instance) that proceeding using an optimize-then-discretize strategy results in the following (continuous) system of equations:

$$\begin{aligned} -\nabla^2 y &= f, && \text{in } \Omega, \\ \frac{\partial y}{\partial n} &= u, && \text{on } \partial\Omega, \end{aligned}$$

$$\beta u - p = 0, \qquad \text{on } \partial\Omega,$$

$$\begin{aligned} -\nabla^2 p &= \widehat{y} - y, && \text{in } \Omega, \\ \frac{\partial p}{\partial n} &= 0, && \text{on } \partial\Omega. \end{aligned}$$

Galerkin discretization of these equations results in the same matrix system as above, provided the same finite element basis functions are chosen as for the discretize-then-

optimize method.

Subdomain Control Problems. Another problem, which we will see results in a similar matrix structure as that for boundary control problems, involves applying control on an interior subdomain Ω_{sub} of Ω. This is important in cases where it is only possible to "control" some part of the domain – for instance in applications involving flow it is perhaps natural to be only able to control some entry region of the domain. We write the subdomain control problem for Poisson's equation as

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \left\| y - \widehat{y} \right\|_{L_2(\Omega)}^2 + \frac{\beta}{2} \left\| u \right\|_{L_2(\Omega_{\text{sub}})}^2 \\ \text{s.t.} \ & -\nabla^2 y = \begin{cases} u & \text{on } \Omega_{\text{sub}} \subset \Omega, \\ 0 & \text{on } \Omega \backslash \Omega_{\text{sub}}, \end{cases} \\ & y = g, \quad \text{on } \partial\Omega. \end{aligned} \tag{2.16}$$

Here, we still measure the quantity $y - \widehat{y}$ on the entire domain Ω, but the formulation of the problem reflects that the control may only be applied on the subdomain Ω_{sub}.

When trying to solve such a problem, we may discretize state, control and adjoint variables using equal-order finite element basis functions, but allow fewer degrees of freedom for the control variable. That is, we may write

$$y_h = \sum_{j=1}^{n+n_\partial} Y_j \phi_j, \quad u_h = \sum_{j=1}^{n_s} U_j \phi_j, \quad p_h = \sum_{j=1}^{n+n_\partial} P_j \phi_j,$$

where $n_s < n + n_\partial$ corresponds to the number of basis functions within Ω_{sub}.

When applying a discretize-then-optimize strategy to this problem, we proceed very similarly as for the Neumann boundary control problem of this section to derive the discrete Lagrangian

$$\begin{aligned} \mathcal{L}_{DTO}(\mathbf{y}, \mathbf{u}, \mathbf{p}) &= \frac{1}{2} \left\| y_h - \widehat{y} \right\|_{L_2(\Omega)}^2 + \frac{\beta}{2} \left\| u_h \right\|_{L_2(\Omega_{\text{sub}})}^2 + \mathbf{p}^T (K\mathbf{y} - N_s \mathbf{u} - \mathbf{g}) \\ &= \frac{1}{2} \mathbf{y}^T M \mathbf{y} - \mathbf{y}^T \mathbf{z} + \bar{C} + \frac{\beta}{2} \mathbf{u}^T M_{\text{sub}} \mathbf{u} + \mathbf{p}^T (K\mathbf{y} - N_s \mathbf{u} - \mathbf{g}), \end{aligned} \tag{2.17}$$

where $\bar{C}$ is a constant as stated above, $\mathbf{g}$ corresponds to the Dirichlet boundary condition, M_{sub} is a mass matrix on Ω_{sub} of dimension $n_s \times n_s$, and N_s consists of terms of the form $\int_{\Omega_{\text{sub}}} \phi_i \phi_j \, \mathrm{d}\Omega$ for $i = 1, ..., n + n_\partial$, $j = 1, ..., n_s$.

Finding the stationary points of $\mathcal{L}_{DTO}$ in (2.17), by differentiating with respect to

$\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ as before, gives the matrix system

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M_{\text{sub}} & -N_s^T \\ K & -N_s & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix}.$$

We note that the mass and stiffness matrices M and K are of the same form as for the distributed control problem of Section 2.1.1. The reason for this is, as then, Dirichlet boundary conditions need to be imposed – therefore the matrix M is a smaller matrix that excludes the boundary nodes, and K is again the Dirichlet stiffness matrix.

We note that we may alternatively derive continuous optimality conditions and discretize these – this optimize-then-discretize strategy gives the same matrix system to be solved as the discretize-then-optimize approach shown above, assuming a suitable choice of finite element basis functions is made.

We have demonstrated in this section that variants of the distributed control problem of Section 2.1.1 may be tackled using similar strategies. In Chapter 4, we will discuss preconditioned iterative methods for the boundary and subdomain control problems outlined in this section.

2.1.3 Problems with State and Control Constraints

Another possibility for the construction of PDE-constrained optimization problems is to impose additional inequality constraints on either the state or control variable. One may observe how solving such problems could be of great practical use. For instance, in the example described earlier concerning the storage of foodstuffs or chemicals, an additional state constraint would involve rigidly specifying atmospheric conditions which may occur, and an additional control constraint would put fixed limits on the amount of energy allowed to be expended on the physical system.

In this section, we illustrate the theory of such problems by applying the methodology to the distributed control of Poisson's equation.

Control Constraints. Let us first consider a method for solving problems which

involve additional control constraints. We may write such a problem in the form

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega, \\ & u_a \leq u \leq u_b, \quad \text{a.e. in } \Omega. \end{aligned} \tag{2.18}$$

Here, u_a and u_b are either constants or functions of the spatial coordinate $\mathbf{x}$ – these functions provide lower and upper bounds that the control variable u is required to satisfy, in addition to the restrictions caused by the regularization term in the cost functional (2.18). Methods for such problems are described in [11, 117], on which we base our discussion here.

The approach used in [117] involves introducing two new Lagrange multipliers[4] $\mathbf{q}_a := (\beta\bar{\mathbf{u}}-\bar{\mathbf{p}})_+$ and $\mathbf{q}_b := (\beta\bar{\mathbf{u}}-\bar{\mathbf{p}})_-$, that is

$$(q_a)_j = \begin{cases} (\beta\bar{u}-\bar{p})_j, & \text{if } (\beta\bar{u}-\bar{p})_j > 0, \\ 0 & \text{otherwise,} \end{cases}$$
$$(q_b)_j = \begin{cases} |(\beta\bar{u}-\bar{p})_j|, & \text{if } (\beta\bar{u}-\bar{p})_j < 0, \\ 0 & \text{otherwise,} \end{cases}$$

for $j = 1, ..., n+n_\partial$, where $(q_a)_j$, $(q_b)_j$ and $(\beta\bar{u}-\bar{p})_j$ denote the values of $\mathbf{q}_a$, $\mathbf{q}_b$ and $\beta\bar{\mathbf{u}}-\bar{\mathbf{p}}$ at the j-th node, given particular computed iterates $\bar{\mathbf{u}}$ and $\bar{\mathbf{p}}$ of $\mathbf{u}$ and $\mathbf{p}$. At this point, we may define

$$\mathbf{q} := \mathbf{q}_a - \mathbf{q}_b = \beta\bar{\mathbf{u}}-\bar{\mathbf{p}}.$$

Now, let us define the *active sets* for this problem as

$$\begin{aligned} \mathcal{A}_+ &= \{j \in \{1, ..., n+n_\partial\} : (u-q)_j > (u_b)_j\}, \\ \mathcal{A}_- &= \{j \in \{1, ..., n+n_\partial\} : (u-q)_j < (u_a)_j\}, \\ \mathcal{A}_I &= \{1, ..., n+n_\partial\} \backslash (\mathcal{A}_+ \cup \mathcal{A}_-), \end{aligned} \tag{2.19}$$

[4] In [117], the Lagrange multipliers introduced were actually $(\beta\bar{\mathbf{u}}+\bar{\mathbf{p}})_+$ and $(\beta\bar{\mathbf{u}}+\bar{\mathbf{p}})_-$ as the matrix system being solved was of a slightly different form – we modify these to achieve consistency of notation with the remainder of this chapter.

where $(u-q)_j$, $(u_a)_j$ and $(u_b)_j$ denote the values of $u-q$, u_a and u_b, respectively, at the j-th node. These sets correspond to the nodes at which the control constraints are achieved ($\mathcal{A}_I$) or violated ($\mathcal{A}_+$ and $\mathcal{A}_-$). In [117], an Active Set method, presented in Algorithm 1, is used to compute $\mathbf{y}_{(k)}$, $\mathbf{u}_{(k)}$ and $\mathbf{p}_{(k)}$, which are successive iterative approximations to the accurate solutions $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ on the discrete space. It is shown that one may successively solve the optimality conditions at each step of the iterative procedure:

$$\begin{aligned} M\mathbf{y}_{(k)} + K\mathbf{p}_{(k)} &= \mathbf{z}, \\ K\mathbf{y}_{(k)} - M\mathbf{u}_{(k)} &= \mathbf{g}, \\ \beta M\mathbf{u}_{(k)} - M\mathbf{p}_{(k)} - M\mathbf{q}_{(k)} &= \mathbf{0}, \\ \mathbf{u}_{(k)} &= \mathbf{u}_a, \quad \text{on } \mathcal{A}_-^{(k)}, \\ \mathbf{u}_{(k)} &= \mathbf{u}_b, \quad \text{on } \mathcal{A}_+^{(k)}, \\ \mathbf{q}_{(k)} &= \mathbf{0}, \quad \text{on } \mathcal{A}_I^{(k)}. \end{aligned}$$

Choose $\mathbf{y}_{(0)}$, $\mathbf{u}_{(0)}$, $\mathbf{p}_{(0)}$, $\mathbf{q}_{(0)}$
Define active sets $\mathcal{A}_+^{(0)}$, $\mathcal{A}_-^{(0)}$, $\mathcal{A}_I^{(0)}$ using definitions, along with $\mathbf{u}_{(0)}$, $\mathbf{q}_{(0)}$
for $k = 1, 2, \ldots$ **do**
    Solve (2.20) on free variables from previous iteration (on $\mathcal{A}_I^{(k-1)}$)
    Update Lagrange multiplier $\mathbf{q}_{(k)}$
    Define new active sets $\mathcal{A}_+^{(k)}$, $\mathcal{A}_-^{(k)}$, $\mathcal{A}_I^{(k)}$ using (2.19), along with $\mathbf{u}_{(k)}$, $\mathbf{p}_{(k)}$
    if $\mathcal{A}_+^{(k)} = \mathcal{A}_+^{(k-1)}$, $\mathcal{A}_-^{(k)} = \mathcal{A}_-^{(k-1)}$ *and* $\mathcal{A}_I^{(k)} = \mathcal{A}_I^{(k-1)}$ **then**
        Algorithm converged
    end
end

Algorithm 1: Active Set algorithm for control-constrained optimal control problems.

These equations may be written as the matrix system [117]

$$\begin{bmatrix} M & 0 & 0 & 0 & K \\ 0 & \beta M^{\mathcal{A}_I^{(k)},\mathcal{A}_I^{(k)}} & 0 & 0 & -M^{\mathcal{A}_I^{(k)},:} \\ 0 & 0 & \beta M^{\mathcal{A}_+^{(k)},\mathcal{A}_+^{(k)}} & 0 & -M^{\mathcal{A}_+^{(k)},:} \\ 0 & 0 & 0 & \beta M^{\mathcal{A}_-^{(k)},\mathcal{A}_-^{(k)}} & -M^{\mathcal{A}_-^{(k)},:} \\ K & -M^{:,\mathcal{A}_I^{(k)}} & -M^{:,\mathcal{A}_+^{(k)}} & -M^{:,\mathcal{A}_-^{(k)}} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k)} \\ \mathbf{u}_{\mathcal{A}_I^{(k)}} \\ \mathbf{u}_b \\ \mathbf{u}_a \\ \mathbf{p}_{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ (M\mathbf{q})^{\mathcal{A}_+^{(k)}} \\ (M\mathbf{q})^{\mathcal{A}_-^{(k)}} \\ \mathbf{g} \end{bmatrix},$$

and reduced to

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M^{\mathcal{A}_I^{(k)},\mathcal{A}_I^{(k)}} & -M^{\mathcal{A}_I^{(k)},:} \\ K & -M^{:,\mathcal{A}_I^{(k)}} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k)} \\ \mathbf{u}_{\mathcal{A}_I^{(k)}} \\ \mathbf{p}_{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ M^{\mathcal{A}_+^{(k)},:}\mathbf{u}_b + M^{\mathcal{A}_-^{(k)},:}\mathbf{u}_a + \mathbf{g} \end{bmatrix}, \quad (2.20)$$

where $\mathcal{A}_+^{(k)}$, $\mathcal{A}_-^{(k)}$ and $\mathcal{A}_I^{(k)}$ denote the active sets defined as in (2.19), with u and q replaced by the k-th iterates $u_{(k)}$ and $q_{(k)}$. Here, $\mathbf{u}_{\mathcal{A}_I^{(k)}}$ corresponds to the portion of the solution vector of $\mathbf{u}$ at the k-th step of the Active Set method belonging to the active set $\mathcal{A}_I^{(k)}$, and $M^{\mathcal{A}_I^{(k)},\mathcal{A}_I^{(k)}}$, $M^{\mathcal{A}_+^{(k)},\mathcal{A}_+^{(k)}}$, $M^{\mathcal{A}_-^{(k)},\mathcal{A}_-^{(k)}}$ is the block diagonal subdivision of M into nodes within $\mathcal{A}_I^{(k)}$, $\mathcal{A}_+^{(k)}$ and $\mathcal{A}_-^{(k)}$ respectively at each iteration. Correspondingly, the mass matrix at each step may be divided up as $[M^{:,\mathcal{A}_I^{(k)}}, M^{:,\mathcal{A}_+^{(k)}}, M^{:,\mathcal{A}_-^{(k)}}]$ column-wise, and $[M^{\mathcal{A}_I^{(k)},:}, M^{\mathcal{A}_+^{(k)},:}, M^{\mathcal{A}_-^{(k)},:}]^T$ row-wise.

At each step of the Active Set method, one may then recompute the Lagrange multipliers $\mathbf{q}$ associated with $\mathcal{A}_+^{(k)}$ and $\mathcal{A}_-^{(k)}$ via the expressions

$$\begin{aligned} (M\mathbf{q})^{\mathcal{A}_+^{(k)}} &= \beta M^{\mathcal{A}_+^{(k)},\mathcal{A}_+^{(k)}}\mathbf{u}_b - M^{\mathcal{A}_+^{(k)},:}\mathbf{p}_{(k)}, \\ (M\mathbf{q})^{\mathcal{A}_-^{(k)}} &= \beta M^{\mathcal{A}_-^{(k)},\mathcal{A}_-^{(k)}}\mathbf{u}_a - M^{\mathcal{A}_-^{(k)},:}\mathbf{p}_{(k)}. \end{aligned}$$

The procedure is terminated at the iteration where the active sets remain unchanged from the previous iteration. It is shown in [64] that this method is equivalent to a semi-smooth Newton method, which is important as superlinear convergence is therefore guaranteed under suitable conditions [12].

We note that, as in [117], we have decomposed the mass matrices using the assumption that they are lumped – the case for consistent mass matrices is similar and can be found in [103] for instance. Preconditioners for this matrix system have been derived and discussed in [62, 64, 117] for example.

State Constraints. Let us now consider problems with additional inequality constraints on the state variable. Such a problem may be written in the form

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega, \\ & y_a \leq y \leq y_b, \quad \text{a.e. in } \Omega. \end{aligned}$$

Here, y_a and y_b may be constants or functions of the spatial coordinates $\mathbf{x}$. These provide upper and lower bounds for the state variable, which must be satisfied in addition to the PDE constraint. We note that this is in many ways a harder problem than the control constrained problem, as the Lagrange multipliers associated with the state constraints can be non-smooth. We refer the reader to literature such as [27, 62, 73, 90] for further discussion of this problem.

One approach to deal with these problems is to introduce a *Moreau-Yosida regularization* [73, 90]. This involves minimizing instead the cost functional

$$\begin{aligned} J_{MY}(y,u) = \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)} + \frac{1}{2\epsilon}\left\|\min\{0, y-y_a\}\right\|^2_{L_2(\Omega)} \\ + \frac{1}{2\epsilon}\left\|\max\{0, y-y_b\}\right\|^2_{L_2(\Omega)}, \end{aligned}$$

where $\epsilon > 0$ is some small Moreau-Yosida regularization parameter designed to penalize the violation of the state constraints. We can see that if $y < y_a$ or $y > y_b$ anywhere in Ω, the Moreau-Yosida regularization term will increase the value of J_{MY} significantly, and so this condition is constructed to "force" the state variable to attain the lower and upper bounds imposed.

Now, consider the discretized version of this problem. As in [90], we may rewrite the optimization problem in the following (discretize-then-optimize) form:

$$\begin{aligned} \min \ & \frac{1}{2}\mathbf{y}^T M\mathbf{y} - \mathbf{y}^T\mathbf{z} + \frac{\beta}{2}\mathbf{u}^T M\mathbf{u} + \frac{1}{2\epsilon}\min\{\mathbf{0}, \mathbf{y}-\mathbf{y}_a\}^T M \min\{\mathbf{0}, \mathbf{y}-\mathbf{y}_a\} \\ & + \frac{1}{2\epsilon}\max\{\mathbf{0}, \mathbf{y}-\mathbf{y}_b\}^T M \max\{\mathbf{0}, \mathbf{y}-\mathbf{y}_b\} \\ \text{s.t.} \quad & K\mathbf{y} - M\mathbf{u} = \mathbf{g}, \end{aligned}$$

where $\mathbf{y}_a$ and $\mathbf{y}_b$ are vectors which contain the values of y_a and y_b, respectively, at

each node.

Taking $\mathbf{p}$ as a discrete adjoint variable (or Lagrange multiplier) as done previously in this chapter, and differentiating with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$, gives the following optimality conditions:

$$\begin{aligned} M\mathbf{y} + K\mathbf{p} &= \mathbf{z} - \epsilon^{-1}\chi_{\mathcal{A}_+} M \max\{\mathbf{0}, \mathbf{y} - \mathbf{y}_b\} - \epsilon^{-1}\chi_{\mathcal{A}_-} M \min\{\mathbf{0}, \mathbf{y} - \mathbf{y}_a\}, \\ \beta M\mathbf{u} - M\mathbf{p} &= \mathbf{0}, \\ K\mathbf{y} - M\mathbf{u} &= \mathbf{g}, \end{aligned}$$

where $\chi_{\mathcal{A}_+}$ for this problem is the characteristic function for indices of $\mathbf{y}$ where $\mathbf{y} - \mathbf{y}_b > \mathbf{0}$, and $\chi_{\mathcal{A}_-}$ similarly is the characteristic function for indices where $\mathbf{y}_a - \mathbf{y} > \mathbf{0}$. The sets

$$\mathcal{A}_+ = \{j : y_j > (y_b)_j\}, \quad \mathcal{A}_- = \{j : y_j < (y_a)_j\},$$

here are the active sets associated with the bound constraints at each step, where y_j, $(y_a)_j$ and $(y_b)_j$ denote the values of $\mathbf{y}$, $\mathbf{y}_a$ and $\mathbf{y}_b$ at the j-th node.

Applying a semi-smooth Newton method to this problem gives the matrix system [90]

$$\begin{bmatrix} M + \epsilon^{-1} G_{\mathcal{A}} M G_{\mathcal{A}} & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k)} \\ \mathbf{u}_{(k)} \\ \mathbf{p}_{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_{\mathcal{A}} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix}$$

at each Newton step. Here, $\mathbf{y}_{(k)}$, $\mathbf{u}_{(k)}$ and $\mathbf{p}_{(k)}$ denote the k-th iterates for state, control and adjoint, the vector $\mathbf{c}_{\mathcal{A}}$ is given by

$$\mathbf{c}_{\mathcal{A}} = \mathbf{z} + \epsilon^{-1}(G_{\mathcal{A}_+} M G_{\mathcal{A}_+} \mathbf{y}_b + G_{\mathcal{A}_-} M G_{\mathcal{A}_-} \mathbf{y}_a),$$

and the matrices $G_{\mathcal{A}_+}$, $G_{\mathcal{A}_-}$ and $G_{\mathcal{A}}$ are projection matrices onto the active sets given by $\mathcal{A}_+$, $\mathcal{A}_-$ and $\mathcal{A}_\pm := \mathcal{A}_+ \cup \mathcal{A}_-$ respectively. In other words, G is a diagonal matrix, with 1 on the diagonal entries corresponding to indices where the state constraint is violated ($y > y_b$ for the matrix $G_{\mathcal{A}_+}$, $y < y_a$ for $G_{\mathcal{A}_-}$, and either condition for $G_{\mathcal{A}}$), and 0 on the other diagonals.

It is also possible to use this Moreau-Yosida regularization for optimal control problems with additional control constraints, as an alternative to the strategy pro-

posed above – we will consider this method for a particular problem in Chapter 9. Problems that involve a combination of state and control constraints could also be tackled using this technique.

2.1.4 Time-Dependent Problems

It is natural for us to also consider time-dependent variants of the problems discussed in this section. This is important to do, as most real-world problems have a time-dependent element, and therefore time-dependent optimal control problems will have considerable applicability. To provide an outline of such problems, let us consider a distributed control problem involving the heat equation, of the form

$$\begin{aligned} &\min_{y,u} \ J(y,u) \\ &\text{s.t} \quad y_t - \nabla^2 y = u, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T], \\ &\qquad\qquad y = g, \quad \text{on } \partial\Omega, \\ &\qquad\qquad y = y_0, \quad \text{at } t = 0, \end{aligned}$$

where we now work on a space-time domain $\Omega \times [0,T]$, with space-time coordinates $(\mathbf{x},t)$. For this problem, we require Dirichlet boundary conditions on $\partial\Omega$ (though Neumann boundary conditions could also be considered) and initial conditions at $t = 0$. There are many possible choices for the cost functional $J(y,u)$ in this case. In this work, we wish to consider the cases where $J(y,u)$ is equal to either

$$J_1(y,u) = \frac{1}{2}\int_0^T \int_\Omega \left(y(\mathbf{x},t) - \widehat{y}(\mathbf{x},t)\right)^2 \ \mathrm{d}\Omega\mathrm{d}t + \frac{\beta}{2}\int_0^T \int_\Omega \left(u(\mathbf{x},t)\right)^2 \ \mathrm{d}\Omega\mathrm{d}t,$$

or

$$J_2(y,u) = \frac{1}{2}\int_\Omega \left(y(\mathbf{x},T) - \widehat{y}(\mathbf{x})\right)^2 \ \mathrm{d}\Omega + \frac{\beta}{2}\int_0^T \int_\Omega \left(u(\mathbf{x},t)\right)^2 \ \mathrm{d}\Omega\mathrm{d}t.$$

We note that $J_1(y,u)$ is an appropriate choice for the cost functional if we wish to consider the distance between the state and desired state (in the L_2-norm) at all instances in time, and $J_2(y,u)$ is a good choice if we only seek this quantity at the final time $t = T$. We will refer to the optimal control problem with cost functionals $J_1(y,u)$ and $J_2(y,u)$ as the "all-times case" and "final-time case" respectively.

The content of the remainder of this section was discussed in [91, 114].

All-times case with $J_1(y, u)$. Let us first examine the discretize-then-optimize approach for solving this problem. For this case, we need to consider once again the discretized forward problem involving the heat equation. This depends on the time-stepping scheme used: for our investigation we consider a simple backward Euler scheme, in order to demonstrate that iterative solution of such problems is feasible. In this case the forward problem can be written in matrix form as

$$\mathcal{K}\mathbf{y} - \tau\mathcal{M}\mathbf{u} = \mathbf{d}.$$

Here $\mathbf{y}$ and $\mathbf{u}$ correspond to the discretized versions of y and u at each time-step, that is $\mathbf{y} = \left[\left(\mathbf{y}^{(1)}\right)^T, ..., \left(\mathbf{y}^{(N_t)}\right)^T\right]^T$, $\mathbf{u} = \left[\left(\mathbf{u}^{(1)}\right)^T, ..., \left(\mathbf{u}^{(N_t)}\right)^T\right]^T$, with $\mathbf{y}^{(j)}$, $\mathbf{u}^{(j)}$ relating to the state and control at the j-th time-step. The parameter τ denotes the time-step used in the discretization, with N_t being the number of time-steps, and

$$\mathcal{K} = \begin{bmatrix} M+\tau K & & & & & \\ -M & M+\tau K & & & & \\ & \ddots & \ddots & & & \\ & & -M & M+\tau K & & \\ & & & -M & M+\tau K \end{bmatrix}, \tag{2.21}$$

$$\mathcal{M} = \begin{bmatrix} M & & & & \\ & M & & & \\ & & \ddots & & \\ & & & M & \\ & & & & M \end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix} M\mathbf{y}_0 + \mathbf{g} \\ \mathbf{g} \\ \vdots \\ \mathbf{g} \\ \mathbf{g} \end{bmatrix}, \tag{2.22}$$

where K and M are the standard finite element stiffness and mass matrices defined by (2.6) and (2.7). Furthermore, $\mathbf{g}$ takes account of boundary conditions at each time-step, and therefore is the same vector $\mathbf{g}$ which describes the boundary conditions for the distributed Poisson control problem in (2.8). The vector $\mathbf{y}_0$ corresponds to the initial condition y_0.

Let us now use a trapezoidal rule approach to deal with integration in space within the integral terms in the cost functional. Applying similar working as for the time-independent problem leads to the following discrete cost functional that we wish to

minimize:

$$J_D(\mathbf{y},\mathbf{u}) = \frac{\tau}{2}\mathbf{y}^T\mathcal{M}_{1/2}\mathbf{y} - \tau\mathbf{y}^T\mathbf{z}_{1/2} + \bar{C} + \frac{\beta\tau}{2}\mathbf{u}^T\mathcal{M}_{1/2}\mathbf{u},$$

where

$$\mathcal{M}_{1/2} = \begin{bmatrix} \frac{1}{2}M & & & & \\ & M & & & \\ & & \ddots & & \\ & & & M & \\ & & & & \frac{1}{2}M \end{bmatrix}, \tag{2.23}$$

the vector $\mathbf{z}_{1/2} = \left[\frac{1}{2}\left(\mathbf{z}^{(1)}\right)^T, \left(\mathbf{z}^{(2)}\right)^T, ..., \left(\mathbf{z}^{(N_t-1)}\right)^T, \frac{1}{2}\left(\mathbf{z}^{(N_t)}\right)^T\right]^T$, with $\mathbf{z}^{(j)}$ containing entries of the form $\int_\Omega \widehat{y}(\mathbf{x}, j\tau)\phi_i \, \mathrm{d}\Omega$, and $\bar{C}$ once again denotes a constant independent of y and u.

When applying the PDE constraints, this leads to the following Lagrangian:

$$\mathcal{L}_{DTO} = \frac{\tau}{2}\mathbf{y}^T\mathcal{M}_{1/2}\mathbf{y} - \tau\mathbf{y}^T\mathbf{z}_{1/2} + \bar{C} + \frac{\beta\tau}{2}\mathbf{u}^T\mathcal{M}_{1/2}\mathbf{u} + \mathbf{p}^T(\mathcal{K}\mathbf{y} - \tau\mathcal{M}\mathbf{u} - \mathbf{d}),$$

where $\mathbf{p}$ relates to the Lagrange multiplier at each time-step, that is to say $\mathbf{p} = \left[\left(\mathbf{p}^{(1)}\right)^T, ..., \left(\mathbf{p}^{(N_t)}\right)^T\right]^T$, with $\mathbf{p}^{(j)}$ relating to the Lagrange multiplier at the j-th time-step.

Now, differentiating $\mathcal{L}_{DTO}$ with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ gives the following three matrix equations in turn:

$$\begin{aligned} \tau\mathcal{M}_{1/2}\mathbf{y} + \mathcal{K}^T\mathbf{p} &= \tau\mathbf{z}_{1/2}, \\ \beta\tau\mathcal{M}_{1/2}\mathbf{u} - \tau\mathcal{M}\mathbf{p} &= \mathbf{0}, \\ \mathcal{K}\mathbf{y} - \tau\mathcal{M}\mathbf{u} &= \mathbf{d}, \end{aligned}$$

which are again the adjoint, gradient and state equations. Combining these equations gives the matrix system

$$\begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau\mathbf{z}_{1/2} \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}. \tag{2.24}$$

Now, let us consider the alternative optimize-then-discretize approach for solving this problem. In this case, we need to find the stationary points of the continuous Lagrangian

$$\mathcal{L}_{OTD} = \frac{1}{2}\int_0^T \int_\Omega \left(y(\mathbf{x},t) - \widehat{y}(\mathbf{x},t)\right)^2 \, \mathrm{d}\Omega \mathrm{d}t + \frac{\beta}{2}\int_0^T \int_\Omega \left(u(\mathbf{x},t)\right)^2 \, \mathrm{d}\Omega \mathrm{d}t + \int_0^T \int_\Omega (y_t - \nabla^2 y - u) p_\Omega \, \mathrm{d}\Omega \mathrm{d}t + \int_0^T \int_{\partial\Omega} (y - g) p_{\partial\Omega} \, \mathrm{d}s \mathrm{d}t.$$

Now, differentiating $\mathcal{L}_{OTD}$ with respect to p_Ω and $p_{\partial\Omega}$ returns the state equation

$$\begin{aligned} y_t - \nabla^2 y &= u, \quad \text{in } \Omega, \\ y &= g, \quad \text{on } \partial\Omega, \\ y(\mathbf{x},0) &= y_0. \end{aligned}$$

Next, differentiating with respect to y gives, using similar working as for the time-independent problem, the adjoint equation

$$\begin{aligned} -p_t - \nabla^2 p &= \widehat{y} - y, \quad \text{in } \Omega, \\ p &= 0, \quad \text{on } \partial\Omega, \\ p(\mathbf{x},T) &= 0. \end{aligned}$$

As detailed in [114], discretizing the adjoint equation using a backward Euler method gives the matrix system

$$\tau \mathcal{M}_0 \mathbf{y} + \mathcal{K}^T \mathbf{p} = \tau \mathbf{z}_0,$$

where

$$\mathcal{M}_0 = \begin{bmatrix} M & & & & \\ & M & & & \\ & & \ddots & & \\ & & & M & \\ & & & & 0 \end{bmatrix}, \tag{2.25}$$

and $\mathbf{z}_0 = \left[\left(\mathbf{z}^{(1)}\right)^T, ..., \left(\mathbf{z}^{(N_t-1)}\right)^T, \mathbf{0}\right]^T$, with $\mathbf{z}^{(j)}$ again containing entries of the form $\int_\Omega \widehat{y}(\mathbf{x}, j\tau)\phi_i \, \mathrm{d}\Omega$. As in [114], combining this with discretizations of the state and

gradient equations gives the following matrix system:

$$\begin{bmatrix} \tau\mathcal{M}_0 & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau\mathbf{z}_0 \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}. \tag{2.26}$$

We can see that the discretize-then-optimize and optimize-then-discretize approaches do not yield the same matrix system for this problem. To rectify this issue, one would have to choose a time-stepping scheme more carefully.

We consider the solution of the systems (2.24) and (2.26) corresponding to both approaches in Chapter 8.

Final-time case with $J_2(y, u)$. For this case, we will only examine the discretize-then-optimize formulation, although here the matrix system arising from the optimize-then-discretize method does turn out to be the same as the discretize-then-optimize system.

In the discretize-then-optimize approach, due to the modified form of the continuous cost functional, the discrete cost functional would now take the form

$$J_D(\mathbf{y}, \mathbf{u}) = \frac{1}{2}\mathbf{y}^T\mathcal{M}_1\mathbf{y} - \mathbf{y}^T\mathbf{z}_1 + \bar{C} + \frac{\beta\tau}{2}\mathbf{u}^T\mathcal{M}_{1/2}\mathbf{u},$$

where

$$\mathcal{M}_1 = \begin{bmatrix} 0 & & & & \\ & 0 & & & \\ & & \ddots & & \\ & & & 0 & \\ & & & & M \end{bmatrix} \tag{2.27}$$

takes account of the first term in the cost functional $J_2(y, u)$, and the vector $\mathbf{z}_1 = \left[\mathbf{0}^T, ..., \mathbf{0}^T, \left(\mathbf{z}^{(N_t)}\right)^T\right]^T$, with $\mathbf{z}^{(N_t)}$ corresponding to the desired state at $t = T$.

Consequently, the Lagrangian of which we wish to find the stationary points is given by

$$\mathcal{L}_{DTO} = \frac{1}{2}\mathbf{y}^T\mathcal{M}_1\mathbf{y} - \mathbf{y}^T\mathbf{z}_1 + \bar{C} + \frac{\beta\tau}{2}\mathbf{u}^T\mathcal{M}_{1/2}\mathbf{u} + \mathbf{p}^T(\mathcal{K}\mathbf{y} - \tau\mathcal{M}\mathbf{u} - \mathbf{d}).$$

Differentiating this cost functional with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ in turn gives the

following three equations in terms of the discrete variables:

$$\begin{aligned} \mathcal{M}_1\mathbf{y} + \mathcal{K}^T\mathbf{p} &= \mathbf{z}_1, \\ \beta\tau\mathcal{M}_{1/2}\mathbf{u} - \tau\mathcal{M}\mathbf{p} &= \mathbf{0}, \\ \mathcal{K}\mathbf{y} - \tau\mathcal{M}\mathbf{u} &= \mathbf{d}, \end{aligned}$$

which can be written as the matrix system

$$\begin{bmatrix} \mathcal{M}_1 & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}. \tag{2.28}$$

Observe that the matrix systems (2.24), (2.26) and (2.28) are extremely large, even in comparison to the matrix systems corresponding to the time-independent Poisson control problem. This renders a direct method for these problems infeasible, even if such a method could be applied to the time-independent formulation. In addition, even if an iterative method can be devised, it would be likely to be ineffective if matrices of dimension equal to that of the blocks of the matrix systems needed to be stored, as these blocks grow rapidly in dimension if finer discretizations in space or time are used. It is therefore hugely desirable to devise an iterative solution strategy which involves only storing matrices of much lower dimension than the matrix system of which the solution is sought. We will consider whether such a strategy can be developed for solving such time-dependent problems in Chapter 8.

2.1.5 Other Problem Formulations

In this section, we have presented but a few of the possible formulations of PDE-constrained optimization problems. There are many extensions to the examples discussed here. For instance one may consider using different norms within the cost functional we try to minimize – in [102] an H_1-norm regularization of the control variable (rather than L_2-norm) is studied, and in [59, 122], for instance, many other norms applied to the state and control terms within the cost functional are discussed. Furthermore, many different PDEs may be considered within the optimal control problem, which leads to a wealth of application areas as listed earlier in this section. We introduced problems here solely using Poisson's equation and the heat equation, but a variety of other PDEs are studied in later chapters of this thesis.

2.2 Preconditioning Saddle Point Systems

Throughout this thesis, we will encounter (real) matrix systems of the form

$$\mathcal{A}\mathbf{x} = \mathbf{b}, \tag{2.29}$$

which are large and sparse. For a large class of problems, and indeed the problems that we wish to consider, the matrix $\mathcal{A}$ is also symmetric. There are two classes of widely researched strategies for solving such systems: *direct methods*, which use linear algebraic techniques including Gaussian elimination to decompose the matrix such that the solution of the system becomes simpler (see [32] for a summary of such methods), and *iterative methods*. As PDE-constrained optimization problems are frequently of very high dimension, and appropriate iterative methods can work extremely well for such problems, we will apply the latter strategy. In particular, we will aim to solve systems of the form (2.29) using iterative methods, along with *preconditioners*.

The idea of preconditioning is to create a matrix (or operator) $\mathcal{P}$ and look to solve, instead of (2.29), the system

$$\mathcal{P}^{-1}\mathcal{A}\mathbf{x} = \mathcal{P}^{-1}\mathbf{b}, \tag{2.30}$$

which has more "convenient" properties in some sense. Equation (2.30) corresponds to *left preconditioning*. It is also perfectly possible to apply *right preconditioning* – this involves solving, instead of (2.29), the system $\mathcal{A}\mathcal{P}^{-1}\mathcal{P}\mathbf{x} = \mathbf{b}$ by solving in turn

$$\mathcal{A}\mathcal{P}^{-1}\mathbf{y} = \mathbf{b}, \quad \mathcal{P}\mathbf{x} = \mathbf{y}.$$

Whether carrying out left or right preconditioning, the effectiveness of the strategy depends on the choice of preconditioner made. A good choice of preconditioner $\mathcal{P}$ will have two key properties:

1. The matrix $\mathcal{P}$ will capture the behaviour of the matrix $\mathcal{A}$ well. To give an idea of what this means in practice, for a symmetric positive definite preconditioner $\mathcal{P}$ and a symmetric matrix $\mathcal{A}$, the crucial property is that $\kappa(\mathcal{P}^{-1}\mathcal{A}) \ll \kappa(\mathcal{A})$, where κ denotes the *condition number* (in the 2-norm) of a matrix. This will mean that the eigenvalues of the preconditioned system will be more clustered than those of the original system.

2. The matrix $\mathcal{P}$ will be cheap to invert, as at each step of an iterative method, $\mathcal{P}^{-1}$ will need to be applied.

In order to understand these two properties better, we consider two particular choices of preconditioners for the matrix $\mathcal{A}$, neither of which would be used in practice:

- $\mathcal{P} = I$: This choice of preconditioner will certainly satisfy the second stated property of a good preconditioner, as it is trivial to invert. However, it will not satisfy the first property as it will not improve the spectral properties of the matrix system involved.

- $\mathcal{P} = \mathcal{A}$: Naturally, this would capture the behaviour of $\mathcal{A}$ well, as $\mathcal{P}^{-1}\mathcal{A} = I$. However, this would not be cheap to invert as $\mathcal{A}$ is not (if $\mathcal{A}$ were cheap to invert, there would be no use for an iterative method in any case). So this is not a good choice for $\mathcal{P}$ either.

These two artificial choices for $\mathcal{P}$ are at the opposite ends of the scale in terms of the balance between spectral properties and computational cheapness – the preconditioners that we seek result in more of a "happy medium". Our choices of preconditioners will resemble the matrix $\mathcal{A}$ well, but will also be cheap to invert, with the cost of applying $\mathcal{P}^{-1}$ scaling linearly or close to linearly with the dimension of $\mathcal{A}$ (and $\mathcal{P}$).

2.2.1 Saddle Point Systems

When solving PDE-constrained optimization problems, we may in fact say much more about effective preconditioning strategies for the resulting matrix systems than the general observations made above. The reason for this is that the matrix systems that need to be solved have a very specific structure, namely *saddle point* structure.

A saddle point system is defined as a matrix system of the form

$$\underbrace{\begin{bmatrix} \Phi & \Psi^T \\ \Psi & -\Theta \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \tag{2.31}$$

where $\Phi \in \mathbb{R}^{m\times m}$ is invertible, $\Psi \in \mathbb{R}^{p\times m}$, $p \leq m$, has full row rank, and $\Theta \in \mathbb{R}^{p\times p}$. Within all problems that we consider, Φ and Θ are also symmetric. The matrix Θ will also always be positive semi-definite, and will often be equal to the zero matrix.

An excellent review of such matrix systems and numerical methods for their solution is given in [9].

We now introduce the following three general preconditioners for the matrix system (2.31):

$$\mathcal{P}_1 = \begin{bmatrix} \Phi & 0 \\ 0 & \Theta + \Psi\Phi^{-1}\Psi^T \end{bmatrix},$$
$$\mathcal{P}_2 = \begin{bmatrix} \Phi & 0 \\ \Psi & -\Theta - \Psi\Phi^{-1}\Psi^T \end{bmatrix},$$
$$\mathcal{P}_3 = \begin{bmatrix} \Phi & 0 \\ \Psi & \Theta + \Psi\Phi^{-1}\Psi^T \end{bmatrix}.$$

We see that the two common components of each of these three preconditioners are the $(1,1)$-block Φ and the (negative) Schur complement $S := \Theta + \Psi\Phi^{-1}\Psi^T$.

A large part of the explanation of why these matrices are in fact very effective preconditioners arises from the following three theorems. Theorems 1, 2 and 3 were proved for the case $\Theta = 0$ in [79, 83], and Theorems 2 and 3 were extended to the case $\Theta \neq 0$ in [72]. We note that in these theorems, $\lambda(\cdot)$ denotes the eigenvalue spectrum of a particular matrix.

Theorem 1. *Suppose for a matrix system of the form* (2.31) *that* $\mathcal{A}$ *is invertible, and that* $\Theta = 0$. *Then the following preconditioner for* $\mathcal{A}$*:*

$$\mathcal{P}_1 = \begin{bmatrix} \Phi & 0 \\ 0 & \Psi\Phi^{-1}\Psi^T \end{bmatrix}$$

satisfies

$$\lambda(\mathcal{P}_1^{-1}\mathcal{A}) \in \left\{1, \frac{1 \pm \sqrt{5}}{2}\right\}.$$

Theorem 2. *Suppose for a matrix system of the form* (2.31) *that* $\mathcal{A}$ *is invertible. Then the following preconditioner for* $\mathcal{A}$*:*

$$\mathcal{P}_2 = \begin{bmatrix} \Phi & 0 \\ \Psi & -\Theta - \Psi\Phi^{-1}\Psi^T \end{bmatrix}$$

satisfies

$$\lambda(\mathcal{P}_2^{-1}\mathcal{A}) \in \{1\}.$$

Theorem 3. *Suppose for a matrix system of the form* (2.31) *that* $\mathcal{A}$ *is invertible. Then the following preconditioner for* $\mathcal{A}$*:*

$$\mathcal{P}_3 = \begin{bmatrix} \Phi & 0 \\ \Psi & \Theta + \Psi\Phi^{-1}\Psi^T \end{bmatrix}$$

satisfies

$$\lambda(\mathcal{P}_3^{-1}\mathcal{A}) \in \{\pm 1\}.$$

Given that the results for the block triangular preconditioners $\mathcal{P}_2$ and $\mathcal{P}_3$ are much more general than the result for the block diagonal preconditioner $\mathcal{P}_1$ (as there is no assumption that $\Theta = 0$ involved in Theorems 2 and 3, in contrast to Theorem 1), it is natural to ask whether one may predict eigenvalues for block diagonal preconditioners when $\Theta \neq 0$. The following theorem (as stated and proved in [86]) gives a partial answer to this question.[5]

Theorem 4. *Suppose a matrix system of the form* $\mathcal{A}$ *is invertible, and that* Φ *and* Θ *are symmetric positive definite. Then the following preconditioner for* $\mathcal{A}$*:*

$$\mathcal{P}_1 = \begin{bmatrix} \Phi & 0 \\ 0 & \Theta + \Psi\Phi^{-1}\Psi^T \end{bmatrix}$$

satisfies

$$\lambda(\mathcal{P}_1^{-1}\mathcal{A}) \in \left[-1, \frac{1}{2}(1-\sqrt{5})\right] \cup \left[1, \frac{1}{2}(1+\sqrt{5})\right].$$

[5]We note that bounds of this form have been widely discussed in the literature on saddle point systems. For instance, the lower bounds on the negative and positive eigenvalues are shown by Axelsson and Neytcheva in [3, Corollary 1], with the remaining bounds a consequence of the result of Silvester and Wathen in [111, Lemma 2.2].

Proof. Simple calculation gives us that

$$\begin{aligned}\mathcal{P}_1^{-1}\mathcal{A} &= \begin{bmatrix} \Phi^{-1} & 0 \\ 0 & (\Theta + \Psi\Phi^{-1}\Psi^T)^{-1} \end{bmatrix} \begin{bmatrix} \Phi & \Psi^T \\ \Psi & -\Theta \end{bmatrix} \\ &= \begin{bmatrix} I & \Phi^{-1}\Psi^T \\ (\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Psi & -(\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Theta \end{bmatrix}.\end{aligned}$$

We may therefore consider the eigenvalue problem $\mathcal{P}_1^{-1}\mathcal{A}\mathbf{x} = \lambda\mathbf{x}$, and write

$$\mathbf{x}_1 + \Phi^{-1}\Psi^T\mathbf{x}_2 = \lambda\mathbf{x}_1, \tag{2.32}$$

$$(\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Psi\mathbf{x}_1 - (\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Theta\mathbf{x}_2 = \lambda\mathbf{x}_2, \tag{2.33}$$

by expressing $\mathbf{x} = [\mathbf{x}_1^T, \mathbf{x}_2^T]^T$, with $\mathbf{x}_1$ and $\mathbf{x}_2$ vectors of lengths m and p respectively.

Now, (2.32) gives us that

$$\mathbf{x}_1 = \frac{1}{\lambda - 1}\Phi^{-1}\Psi^T\mathbf{x}_2, \tag{2.34}$$

provided that $\lambda \neq 1$. In this case, we may then substitute (2.34) into (2.33) to give

$$\begin{aligned} & \frac{1}{\lambda - 1}(\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Psi\Phi^{-1}\Psi^T\mathbf{x}_2 - (\Theta + \Psi\Phi^{-1}\Psi^T)^{-1}\Theta\mathbf{x}_2 = \lambda\mathbf{x}_2 \\ \Rightarrow \quad & \Psi\Phi^{-1}\Psi^T\mathbf{x}_2 - (\lambda - 1)\Theta\mathbf{x}_2 = \lambda(\lambda - 1)(\Theta + \Psi\Phi^{-1}\Psi^T)\mathbf{x}_2 \\ \Rightarrow \quad & (\lambda^2 - \lambda - 1)(\Theta + \Psi\Phi^{-1}\Psi^T)\mathbf{x}_2 = -\lambda\Theta\mathbf{x}_2. \end{aligned}$$

Given that we have assumed that Θ is positive definite (and hence invertible), we may write this as the following eigenvalue problem:

$$(I + \Theta^{-1}\Psi\Phi^{-1}\Psi^T)\mathbf{x}_2 = \frac{-\lambda}{\lambda^2 - \lambda - 1}\mathbf{x}_2.$$

The pertinent question at this point is what the eigenvalues of the matrix $I + \Theta^{-1}\Psi\Phi^{-1}\Psi^T$ are. As Θ is symmetric positive definite, it has a (unique) matrix square root $\Theta^{1/2}$, and hence the matrix $I+\Theta^{-1}\Psi\Phi^{-1}\Psi^T$ is similar to $I+\Theta^{-1/2}\Psi\Phi^{-1}\Psi^T\Theta^{-1/2}$. We may therefore use the symmetric positive definiteness of Φ and Θ to conclude that the eigenvalues of the symmetric matrix $\Theta^{-1/2}\Psi\Phi^{-1}\Psi^T\Theta^{-1/2}$ are all real and positive, and hence that the eigenvalues of $I + \Theta^{-1/2}\Psi\Phi^{-1}\Psi^T\Theta^{-1/2}$ are all real and at least 1.

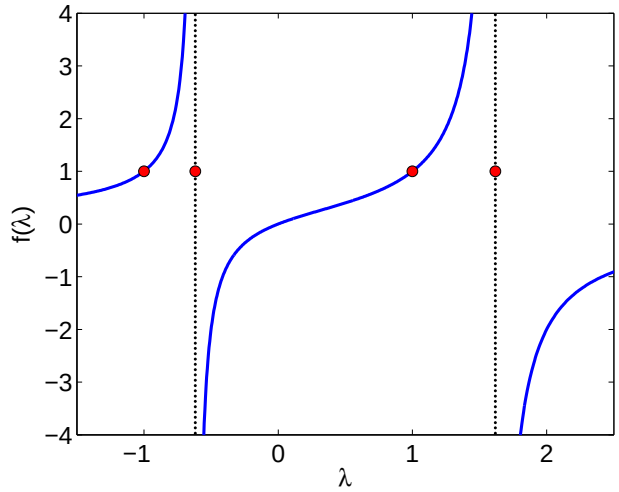


Figure 2.1: Plot of the function $f(\lambda) = \frac{-\lambda}{\lambda^2-\lambda-1}$. Marked are the values $\lambda = -1, 1, \frac{1}{2}(1 \pm \sqrt{5})$.

This makes it clear that the following inequality must hold:

$$\frac{-\lambda}{\lambda^2 - \lambda - 1} \geq 1.$$

In Figure 2.1, we display a plot of the function $f(\lambda) = \frac{-\lambda}{\lambda^2-\lambda-1}$. We can see that $f(\lambda) = 1$ at the points $\lambda = \pm 1$, and that there are asymptotes at $\lambda = \frac{1}{2}(1 \pm \sqrt{5})$. We therefore deduce from the above inequality that

$$\lambda \in \left[-1, \frac{1}{2}(1 - \sqrt{5})\right] \cup \left[1, \frac{1}{2}(1 + \sqrt{5})\right],$$

as required. □

As discussed in [36, 72], if we precondition the matrix $\mathcal{A}$ with $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$, then an appropriate iterative method should converge in 3, 2 and 2 iterations respectively, provided $\mathcal{A}$ is invertible [83], and with the additional assumption that $\Theta = 0$ when $\mathcal{P}_1$ is applied. These important observations result from Theorems 1, 2 and 3 above, along with the fact that $\mathcal{P}_1$ and $\mathcal{P}_3$ are diagonalizable but $\mathcal{P}_2$ is not. The eigenvalues of $\mathcal{P}_1^{-1}\mathcal{A}$ can be shown to be well clustered in certain cases even if $\Theta \neq 0$, as Theorem 4 shows.

The preconditioners $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$ can therefore be regarded as *ideal preconditioners*, given the relevant assumptions detailed above, as they may be used within

iterative methods to solve the matrix system (2.31) in a fixed (small) number of iterations, independently of the dimension of the matrix system.

However, although the preconditioners $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$ are theoretically powerful, they are clearly not practical – in particular the matrix Φ^{-1} may be dense even if Φ is sparse, and so forming the Schur complement exactly would be prohibitively expensive. However, we may be able to establish effective preconditioners for $\mathcal{A}$ by approximating Φ and S with $\widehat{\Phi}$ and $\widehat{S}$, and then applying these matrices within the preconditioners. This would therefore involve considering preconditioners of the form

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \widehat{\Phi} & 0 \\ 0 & \widehat{S} \end{bmatrix}, \quad \widehat{\mathcal{P}}_2 = \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -\widehat{S} \end{bmatrix}, \quad \widehat{\mathcal{P}}_3 = \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & \widehat{S} \end{bmatrix}.$$

The crucial task when constructing preconditioners for matrices of the form (2.31) is therefore developing good approximations to the $(1,1)$-block and Schur complement of $\mathcal{A}$. In practice, we often find the latter to be a harder problem, and we will devote much attention to developing good choices of $\widehat{S}$.

For all matrix systems studied in this thesis, we find that preconditioners of the form $\widehat{\mathcal{P}}_1$, $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_3$ can be considered, and indeed that they can be very well-suited to the problems at hand, if care is taken when choosing such approximations $\widehat{\Phi}$ and $\widehat{S}$.

2.3 Iterative Solution Techniques

In this section, we wish to introduce the iterative methods which are employed to solve the PDE-constrained optimization problems considered in this thesis. In order to solve these problems, it is necessary to consider the precise structures of the resulting saddle point systems. It is therefore very important that the methods we select have particular properties, so that they can be applied in the appropriate way.

We in fact use only six components to generate the iterative solvers that we use to solve PDE-constrained optimization problems. There are four methods which will be used as an outer iteration:

- MINRES, a solver designed for solving symmetric indefinite systems.
- Conjugate Gradients in a non-standard inner product, which can solve matrix systems which are symmetric positive definite in the particular inner product.
- GMRES, a solver which may be applied to non-symmetric matrix systems.
- Biconjugate Gradients, another solver for non-symmetric matrix systems.

Additionally, there are two main techniques which will be exploited within preconditioners for these iterative solvers:

- Chebyshev semi-iteration, which is highly effective for preconditioning symmetric positive definite matrices, such as mass matrices, for which precise spectral information is known.
- Multigrid, which is a particularly effective method for preconditioning symmetric positive definite matrices such as stiffness matrices, and which may also be applied to other matrices.

In this section, we aim to provide a brief overview of these methods, detail convergence bounds of the methods where appropriate, and explain which problems the methods will be applied to in this thesis.

2.3.1 Simple Iteration

In order to provide some necessary background about iterative methods, we first introduce the concept of a *simple iteration*, the idea of which is applied within the Chebyshev semi-iteration and multigrid methods of future sections.

In order to employ a simple iteration when solving a matrix system $\mathcal{A}\mathbf{x} = \mathbf{b}$, one first splits the matrix into the form $\mathcal{A} = \Lambda - \Sigma$, for an appropriate choice of Λ. Then a sequence of approximations $\{\mathbf{x}_j\}_{j=1,2,\ldots}$ to $\mathbf{x}$ is constructed via the iterative sequence

$$\Lambda \mathbf{x}_j = \Sigma \mathbf{x}_{j-1} + \mathbf{b}, \quad j = 1, 2, \ldots ,$$

given a suitable initial guess $\mathbf{x}_0$. We note that this formula may be re-written as follows:

$$\mathbf{x}_j = (I - \Lambda^{-1}\mathcal{A})\mathbf{x}_{j-1} + \Lambda^{-1}\mathbf{b}, \quad j = 1, 2, \ldots .$$

It is known that such a simple iteration scheme will converge to the true solution, for any initial guess, if and only if [48, Lemma 4.1.1]

$$\lim_{j \to \infty} \left\| (I - \Lambda^{-1}\mathcal{A})^j \right\| = 0,$$

and so for an appropriate choice of splitting, this method can be usefully applied.

Some basic examples of simple iterations, stated in terms of the splitting matrix Λ used, are detailed below:

- *Jacobi iteration*: Here, the splitting matrix $\Lambda = D$, where $D = \mathrm{diag}(\mathcal{A})$, is used.
- *Relaxed Jacobi iteration*: In this case, the modified system $\omega\mathcal{A}\mathbf{x} = \omega\mathbf{b}$ is solved, with splitting matrix $\Lambda = D$. Here ω is some *relaxation parameter*.
- *Gauss-Seidel iteration*: Here, the splitting matrix Λ is taken to be the lower triangular part of $\mathcal{A}$.

These methods are frequently very useful for solving matrix systems iteratively; we explain how such splittings can be used within the Chebyshev semi-iteration and multigrid methods in the next sections.

2.3.2 Chebyshev Semi-Iteration

One of the main methods which we make use of within our preconditioners is the *Chebyshev semi-iteration method*, devised by Flanders and Shortley [38], and popularized by Golub and Varga [45, 46]. This method is of great use when one seeks to approximate the inverse of a mass matrix when solving systems of the form $M\mathbf{x} = \mathbf{b}$.

The basic idea is to approximate such an inverse by a Chebyshev series, by making use of known spectral bounds for mass matrices. This method is discussed in detail in literature such as [99, 101, 126].

Chebyshev semi-iteration is based on the relaxed Jacobi method introduced in the previous section, which is a simple iteration applied to the system $\omega M\mathbf{x} = \omega\mathbf{b}$ with splitting matrix $\Lambda = D := \mathrm{diag}(M)$. The relaxation parameter ω is chosen such that the range of eigenvalues of $I - \omega\Lambda^{-1}M$ is symmetric about zero. So suppose $\lambda(D^{-1}M) \in [\lambda_{\min}, \lambda_{\max}]$ – then we wish to choose ω such that

$$1 - \omega\lambda_{\min} = -(1 - \omega\lambda_{\max}) \quad \Leftrightarrow \quad \omega = \frac{2}{\lambda_{\max} + \lambda_{\min}}.$$

Therefore, the eigenvalues of $I - \omega\Lambda^{-1}M$ would be within the interval

$$\left[-\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}, \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}\right] =: [-\rho, \rho]$$

for this choice of ω.

Element Type	Dimension	$\lambda_{\min}$	$\lambda_{\max}$	ω	$\frac{\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}}$
P1	2D	$\frac{1}{2}$	2	$\frac{4}{5}$	$\frac{3}{5}$
arbitrary linear triangles					
Q1	2D	$\frac{1}{4}$	$\frac{9}{4}$	$\frac{4}{5}$	$\frac{4}{5}$
bilinear rectangles					
P2	2D	0.3924	2.0598	0.816	0.680
arbitrary quadratic triangles					
Q2	2D	$\frac{1}{4}$	$\frac{25}{16}$	$\frac{32}{29}$	$\frac{21}{29}$
biquadratic rectangles					
P1	3D	$\frac{1}{2}$	$\frac{5}{2}$	$\frac{2}{3}$	$\frac{2}{3}$
arbitrary linear tetrahedra					
Q1	3D	$\frac{1}{8}$	$\frac{27}{8}$	$\frac{4}{7}$	$\frac{13}{14}$
rectangular triangular 'brick'					

Table 2.1: Bounds on minimum and maximum eigenvalues of $D^{-1}M$ for a number of commonly-used finite element basis functions, as stated in [125], as well as the appropriate choice ω for these functions, and the values of $\frac{\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}}$.

Table 2.1 gives bounds on the minimum and maximum eigenvalues of $D^{-1}M$ for

a number of commonly used finite element basis functions, and also the values of ω which would be used in the Chebyshev semi-iteration method for each set of basis functions. The table illustrates the ease with which we are able to find the parameter ω for these functions.

The aim of the Chebyshev semi-iterative method is to accelerate the relaxed Jacobi method for this problem: that is instead of computing iterates $\mathbf{x}_j$, $j = 1, 2, ...$, for this method using relaxed Jacobi iteration, we take a linear combination $\mathbf{y}_j = \sum_{i=1}^{j} \alpha_i \mathbf{x}_i$ at the j-th step of the iteration, with the goal of constructing a better approximation this way [101]. Here, α_i are the coefficients of some polynomial $p_j(t)$ of degree j. This polynomial needs to be chosen to be small when its argument is given by any eigenvalue of $T = I - \omega\Lambda^{-1}M = I - \omega D^{-1}M$. It is also necessary that the condition $p_j(1) = 1$ holds, as we require that if $\mathbf{x}_0$ is the true solution $\mathbf{x}$, then so are all the subsequent iterates $\mathbf{y}_j$.

With these constraints, the optimal polynomial (that is to say the polynomial p_j such that $\max_{t\in[-\rho,\rho]} |p_j(t)|$ is as small as possible) is the *shifted and scaled Chebyshev polynomial*, given by

$$p_j(t) = \widehat{T}_j(t) := \frac{T_j\left(\frac{t}{\rho}\right)}{T_j\left(\frac{1}{\rho}\right)},$$

where $T_j(t) := \cos\left(j \cos^{-1} t\right)$ denotes the standard *Chebyshev polynomial*.

In [99, Chapter 3], it is demonstrated that one may develop a three-term recurrence relation for this choice of polynomial $p_j(t)$, and use this to construct a fast recursive method for determining successive vectors $\mathbf{y}_j$. We note that this is made possible due to the known values $\lambda_{\min}$ and $\lambda_{\max}$ of $D^{-1}M$, which enable us to compute the parameters ω and ρ in this method.

Pseudocode describing how the Chebyshev semi-iteration method may be implemented, using this approach for constructing successive vectors $\mathbf{y}_j$, is stated in Algorithm 2. We refer to [99, 101, 126] for further explanation as to the derivation of this method.

$\omega = \frac{2}{\lambda_{\max}+\lambda_{\min}}$, $\rho = \frac{\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}}$
$\theta_0 = 1$, $\mathbf{y}_{-1} = \mathbf{0}$
Choose $\mathbf{y}_0$
for $j = 0, 1, 2, ...$ **do**
    $\theta_{j+1} = \left(1 - \frac{\rho\theta_j^2}{4}\right)^{-1}$
    $\mathbf{z}_j = \omega D^{-1}(\mathbf{b} - M\mathbf{y}_j)$
    $\mathbf{y}_{j+1} = \theta_{j+1}(\mathbf{z}_j + \mathbf{y}_j - \mathbf{y}_{j-1}) + \mathbf{y}_{j-1}$
end

Algorithm 2: Chebyshev semi-iteration method for preconditioning the system $M\mathbf{x} = \mathbf{b}$, where M satisfies $\lambda(D^{-1}M) \in [\lambda_{\min}, \lambda_{\max}]$.

It can easily be shown, as in [99], that the error at the j-th step of the Chebyshev semi-iteration method is bounded in the 2-norm as follows:

$$\frac{\|\mathbf{x} - \mathbf{y}_j\|_2}{\|\mathbf{x} - \mathbf{x}_0\|_2} \leq 2\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^j,$$

where κ here denotes the condition number of $D^{-1}M$. Due to the fact that D and M are symmetric, this quantity is given by the ratio of the maximum and minimum eigenvalues. This tells us that we may apply the Chebyshev semi-iteration method as a preconditioner within an iterative method such as MINRES or Conjugate Gradients, and know a priori how many steps of the method need to be taken to achieve a given precision. Table 2.2 shows the eigenvalues of a mass matrix preconditioned using Chebyshev semi-iteration, for a range of iteration numbers. Note for instance that 20 steps of Chebyshev semi-iteration leads to an error of roughly 10^{-6} for the **Q1** mass matrix tested. The rapid convergence of the method is an extremely useful feature of Chebyshev semi-iteration.

Of course Chebyshev semi-iteration is not the only method that guarantees a certain precision when it is applied to a matrix with uniform eigenvalue bounds. What sets this method apart from other strategies is the computational cheapness with which it may be applied. As demonstrated by the pseudocode above, the dominant operation required to apply this method to a sparse matrix system is a single sparse matrix-vector multiply at each iteration. This small amount of computational work required to apply the method makes it very practical to apply. In addition, the application of Chebyshev semi-iteration is a fixed linear operator, as opposed to other

ITERATIONS	$\lambda_{\min}$	$\lambda_{\max}$	$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$
1	0.2	1.8	9
2	0.5294118	1.4698694	2.7764199
3	0.7538462	1.2461538	1.6530612
4	0.8754864	1.1244668	1.2843909
5	0.9375610	1.0624390	1.1331946
10	0.9980469	1.0019516	1.0039124
15	0.9999390	1.0000610	1.0001221
20	0.9999981	1.0000019	1.0000038
25	0.9999999	1.0000001	1.0000001

Table 2.2: Minimum and maximum eigenvalues, as well as the condition number, of $M_{\text{cheb}}^{-1}M$, where M is a 289×289 **Q1** mass matrix in 2D, and M_{cheb} denotes the application of Chebyshev semi-iteration to the matrix. Results are given for a variety of iteration numbers.

methods, such as Conjugate Gradients, which could also be applied to precondition a mass matrix.

Due to the excellent error properties and computational cheapness of the Chebyshev semi-iteration method, we apply it in Chapters 3, 4, 5, 6 and 7 to approximate the inverse of (consistent) mass matrices.

Note. We observe that as $\text{diag}(M)^{-1}M$ is well-conditioned for consistent mass matrices, it would be possible to simply approximate M by its diagonal within an iterative method. However, based on discussion and numerical tests in literature such as [101], as well as our own experiments, we conclude that applying Chebyshev semi-iteration significantly reduces the number of iterations of the iterative methods required, and hence that it is advantageous to approximate M this way within such a method.

2.3.3 Multigrid Methods

Another algorithm which will be exploited often during the course of solving problems detailed within this thesis is the *multigrid method*. We detail this method briefly in this section in the context of solving a matrix system $\mathcal{A}\mathbf{x} = \mathbf{b}$. For comprehensive introductions to this method, we recommend [24, 60, 127]. The class of such methods

we spend most of this section introducing is referred to as the *geometric multigrid* (GMG) method.

The idea of the multigrid method is to approximate the solution of a PDE on a fine grid by approximating the differential operator and residuals on a coarser grid, solve the problem set-up on this coarser grid, and then map the solution back to the finer grid. For particular problems, this method is known to be highly effective.

The main ingredients which are required for such a method are:

- **Prolongation operator:** This is an operator, which we denote by a matrix P, used to transfer a computed correction term of the solution from a coarse grid to a fine grid. We will generally construct the matrix P by approximating each value on the coarse grid as a weighted average of those on the fine grid, as discussed in [36, Section 2.1].

- **Restriction operator:** This is an operator, which we denote R, used to restrict the residual from a fine grid to a coarse grid. In this thesis, the choice $R = P^T$ is taken.

- **Smoothing operator:** We employ pre- and post-smoothing, using matrices Λ and Λ^T respectively, to mitigate high frequency errors within the solution. Smoothing methods relate to a simple iteration with splitting $\mathcal{A} = \Lambda - \Sigma$: common examples are (relaxed) Jacobi smoothing and Gauss-Seidel smoothing, as described in Section 2.3.1.

We note that the multigrid method carried out on two grids may be shown to be exactly a simple iteration with the following iteration matrix [36, Section 2.1]:

$$\mathbf{e}_{j+1} = \left(I - \Lambda^{-T}\mathcal{A}\right)^s \left(\mathcal{A}^{-1} - P\bar{\mathcal{A}}^{-1}P^T\right)\mathcal{A}\left(I - \Lambda^{-1}\mathcal{A}\right)^s \mathbf{e}_j, \quad j = 0, 1, 2, \dots ,$$

where $\mathbf{e}_j = \mathcal{A}\mathbf{r}_j$ denotes the error at the j-th iterate, and s denotes the number of pre- and post-smoothing steps used (we only consider the case where the number of pre- and post-smoothing steps is the same).

Algorithm 3 details a multigrid V-cycle, forms of which will be utilized often within this thesis, on $l_c \geq 2$ grids. We note that other strategies, such as a multigrid W-cycle, have been studied, but we do not focus on these in this work.

function $\mathbf{x} = \texttt{GMG}(\mathcal{A}, \mathbf{x}, \mathbf{u}, \texttt{level})$
for $j = 1, ..., s$ **do**
  $\mathbf{x} = (I - \Lambda^{-1}\mathcal{A})\mathbf{x} - \Lambda^{-1}\mathbf{b}$
end
if $level = l_c$ **then**
  Solve $\mathcal{A}\mathbf{x} = \mathbf{b}$
else
  $\bar{\mathbf{r}} = P^T(\mathbf{b} - \mathcal{A}\mathbf{x})$
  $\bar{\mathcal{A}} = P^T\mathcal{A}P$
  $\bar{\mathbf{e}} = \texttt{GMG}(\bar{\mathcal{A}}, \bar{\mathbf{r}}, \bar{\mathbf{e}}, \texttt{level} + 1)$
  $\mathbf{x} = \mathbf{x} + P\bar{\mathbf{e}}$
end
for $j = 1, ..., s$ **do**
  $\mathbf{x} = (I - \Lambda^{-T}\mathcal{A})\mathbf{x} - \Lambda^{-T}\mathbf{b}$
end

Algorithm 3: Geometric multigrid V-cycle for preconditioning the system $\mathcal{A}\mathbf{x} = \mathbf{b}$. The algorithm is applied on l_c grids, with Jacobi smoothing.

An important result is proved in [36, Lemma 6.2] concerning a multigrid process applied to the linear system $K\mathbf{x} = \mathbf{b}$, involving the stiffness matrix. This result states that, provided there exists a contraction factor $\rho_c < 1$ such that the iteration error for a simple iteration with splitting matrix K_s satisfies

$$\|\mathbf{u} - \mathbf{u}_{j+1}\|_K \leq \rho_c \|\mathbf{u} - \mathbf{u}_j\|_K , \quad j = 0, 1, 2, ... , \tag{2.35}$$

with $\mathbf{u}_j$ denoting the j-th iterative approximation, then

$$1 - \rho_c \leq \frac{\mathbf{v}^T K \mathbf{v}}{\mathbf{v}^T K_s \mathbf{v}} \leq 1 + \rho_c,$$

where ρ_c is a constant independent of the size of the matrix, and $\mathbf{v} \neq \mathbf{0}$ is a vector of appropriate dimension. As shown in [23, Chapter 6], a multigrid method with Jacobi smoother achieves the condition (2.35), provided an appropriate grid is used to solve the problem. We may therefore conclude that, for this problem, a multigrid routine will guarantee a fixed rate of contraction in the solution error, and is therefore an excellent preconditioner for the stiffness matrix. This is a crucial property which

motivates the use of such a method. Moreover, arguments of this form may be carried over to solving matrix systems involving the sum of stiffness and mass matrices, as such matrices are positive definite as well as the stiffness matrix itself. In the majority of cases where we use a multigrid process in this thesis the method is applied to matrices which are a linear combination of stiffness and mass matrices, so the results described above are very important for our purposes.

We observe that if such a rate of contraction may be guaranteed, then the computational cost of solving such a sparse matrix system to a given tolerance should scale linearly with the size of the matrix system. This is an important property that we desire the applied methods to have.

Although multigrid is an effective method in its own right, in this thesis we wish to make use of the method as a preconditioner within an iterative method. We apply various forms of the multigrid method in this way in each of Chapters 3–9 to solve the PDE-constrained optimization problems considered. For the most part, we will consider an *algebraic multigrid* (AMG) method as opposed to the geometric multigrid approach defined above (see [22, 106] for an introduction to AMG). The basis of AMG is that the operators on each level are constructed purely using the algebraic structure of the matrix, as opposed to GMG which exploits known geometric information concerning the domain on which the physical problem is based. We utilize the Harwell Subroutine Library (HSL) code `HSL_MI20` [19] or the smoothed aggregation AMG code within the Trilinos ML package [44] for the algebraic multigrid procedure. However, we will make use of a specialized GMG routine for the convection-diffusion control problem in Chapter 5. Here, the operator to which multigrid is applied is neither symmetric nor positive definite, meaning that the strategy described above needs to be modified – specifically, the construction of operators on coarse grids, as well as the type of smoothing used, will be tailored to the problem at hand. We note at this point that all the multigrid approaches we make use of appear to scale linearly with the size of the matrix system being solved, and so multigrid is a very appropriate method to employ.

2.3.4 MINRES

We now wish to introduce the iterative methods used as outer iterations in our solvers. The method that we make the most use of is the Minimum Residual method (MINRES). This was developed by Paige and Saunders in [85] for the purpose of solving

symmetric matrix systems, which may be definite or indefinite. It is therefore a highly useful method for the solution of saddle point systems arising in PDE-constrained optimization problems, as these systems are frequently symmetric indefinite themselves.

The MINRES method is an example of a *Krylov subspace method*. These are iterative methods which result in the j-th iterative approximation $\mathbf{x}_j$ to the solution $\mathbf{x}$ of (2.29) satisfying

$$\mathbf{x}_j \in \mathbf{x}_0 + \mathcal{K}_j(\mathcal{A}, \mathbf{r}_0), \quad j = 1, 2, \dots , \tag{2.36}$$

where $\mathbf{x}_0$ is the initial guess of the solution, $\mathbf{r}_0 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$ is the initial *residual*, and $\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$ is the *Krylov subspace* defined by

$$\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0) = \text{span}\left\{\mathbf{r}_0, \mathcal{A}\mathbf{r}_0, \mathcal{A}^2\mathbf{r}_0, ..., \mathcal{A}^{j-1}\mathbf{r}_0\right\}.$$

As in much of the literature about such methods, for instance [36], the expression (2.36) is stated rather loosely – to be more precise, we say that $\mathbf{x}_j$ is equal to the sum of $\mathbf{x}_0$ and some vector that is a member of $\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$.

The MINRES method is based on the *Lanczos algorithm* [80] for generating an orthogonal basis $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_j\}$ of $\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$ when $\mathcal{A}$ is symmetric. This method involves setting $\mathbf{v}_0$, choosing $\mathbf{v}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$, and applying the recurrence

$$\gamma_{i+1}\mathbf{v}_{i+1} = \mathcal{A}\mathbf{v}_i - \delta_i\mathbf{v}_i - \gamma_i\mathbf{v}_{i-1}, \quad i = 1, 2, ..., j-1, \tag{2.37}$$

with $\delta_i = \langle \mathcal{A}\mathbf{v}_i, \mathbf{v}_i \rangle$, and γ_{i+1} selected to enforce $\|\mathbf{v}_{i+1}\|_2 = 1$ for $i = 1, 2, ..., j-1$.

As described in [36, Section 2.4], generating an approximation $\mathbf{x}_j \in \mathbf{x}_0 + \mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$ to $\mathbf{x}$ at the j-th MINRES iteration, and choosing $\mathbf{v}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$, involves writing

$$\mathbf{x}_j = \mathbf{x}_0 + V_j\mathbf{y}_j,$$

for some vector $\mathbf{y}_j \in \mathbb{R}^j$, with $V_j = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_j]$.

The fundamental property of the MINRES method is that, at each iteration, it minimizes the vector 2-norm of the residual $\mathbf{r}_j$ over the vector subspace $\mathbf{r}_0 + \text{span}\{\mathcal{A}\mathbf{r}_0, \mathcal{A}^2\mathbf{r}_0, ..., \mathcal{A}^j\mathbf{r}_0\}$. At each iteration, this involves solving

$$\min \|\mathbf{r}_j\|_2 = \min_{\mathbf{y}_j} \left\| \|\mathbf{r}_0\|_2 \mathbf{e}_1 - \widehat{W}_j\mathbf{y}_j \right\|_2,$$

where $\widehat{W}_j$ is a tridiagonal matrix consisting of entries obtained from the repeated application of the expression (2.37), and $\mathbf{e}_1 = [1, 0, ..., 0]^T$. The minimum is calculated within the method using Givens rotations for a QR factorization.

We recommend [36, 108] for a more detailed derivation of the MINRES algorithm.

We note that one crucial property of the MINRES algorithm is that we may also consider a preconditioned MINRES algorithm, provided we select a preconditioner $\mathcal{P}$ that is symmetric positive definite. In this case, the preconditioner has a Cholesky decomposition $\mathcal{P} = \bar{H}\bar{H}^T$, and so we may in effect consider the solution of [36, 99]

$$\begin{aligned} \mathcal{P}^{-1}\mathcal{A}\mathbf{x} = \mathcal{P}^{-1}\mathbf{b} \quad &\Leftrightarrow \quad \bar{H}^{-1}\mathcal{A}\bar{H}^{-T}\left(\bar{H}^T\mathbf{x}\right) = \bar{H}^{-1}\mathbf{b} \\ &\Leftrightarrow \quad \bar{H}^{-1}\mathcal{A}\bar{H}^{-T}\mathbf{y} = \bar{H}^{-1}\mathbf{b}, \quad \mathbf{y} = \bar{H}^T\mathbf{x}. \end{aligned}$$

So in this reformulation, the matrix being solved for is $\bar{H}^{-1}\mathcal{A}\bar{H}^{-T}$, which is symmetric since $\mathcal{A}$ is. Therefore, the MINRES algorithm may be applied to this preconditioned system, though of course the only matrices worked with in practice are $\mathcal{A}$ and $\mathcal{P}$ themselves, and not $\bar{H}$ or $\bar{H}^T$. The preconditioned MINRES method, which is used extensively in this thesis, is stated in Algorithm 4.

We may observe the ease with which preconditioners may be applied within this algorithm. For instance, suppose we wished to solve the matrix system

$$K\mathbf{x} = \mathbf{b},$$

which arises from the finite element solution of Poisson's equation. Then we may apply the preconditioned MINRES algorithm as the matrix system is symmetric, provided that the preconditioner is symmetric positive definite. A good choice of such a preconditioner [36, Chapter 2] is a multigrid process of the form described in Section 2.3.3, which may be applied within the algorithm to accelerate the rate of convergence.

The preconditioned MINRES algorithm minimizes the residual in the $\mathcal{P}^{-1}$ norm, which is defined by $\|\mathbf{r}\|_{\mathcal{P}^{-1}} = \sqrt{\mathbf{r}^T\mathcal{P}^{-1}\mathbf{r}}$. We can bound the residual at the j-th iteration as follows [36, Section 2.4]:

$$\frac{\|\mathbf{r}_j\|_{\mathcal{P}^{-1}}}{\|\mathbf{r}_0\|_{\mathcal{P}^{-1}}} \leq \min_{p_j \in \Pi_j, p_j(0)=1} \max_{\lambda_i} |p_j(\lambda_i)|, \tag{2.38}$$

where λ_i denote the eigenvalues of $\mathcal{P}^{-1}\mathcal{A}$ and Π_j defines the set of polynomials of degree at most j. One may therefore describe the convergence of the preconditioned

MINRES method in terms of the eigenvalues of the preconditioned matrix system considered, and so may select preconditioners with this objective in mind.

$\mathbf{v}_0 = \mathbf{0}$, $\mathbf{w}_0 = \mathbf{0}$, $\mathbf{w}_1 = \mathbf{0}$
Choose $\mathbf{x}_0$
$\mathbf{v}_1 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$
Solve $\mathcal{P}\mathbf{z}_1 = \mathbf{v}_1$
$\gamma_1 = \sqrt{\langle \mathbf{z}_1, \mathbf{v}_1 \rangle}$
$\eta = \gamma_1$, $s_0 = s_1 = 0$, $c_0 = c_1 = 1$
for $j = 1, 2, \ldots$ **do**
    $\mathbf{z}_j = \frac{\mathbf{z}_j}{\gamma_j}$
    $\delta_j = \langle \mathcal{A}\mathbf{z}_j, \mathbf{z}_j \rangle$
    $\mathbf{v}_{j+1} = \mathcal{A}\mathbf{v}_j - \frac{\delta_j}{\gamma_j}\mathbf{v}_j - \frac{\gamma_j}{\gamma_{j-1}}\mathbf{v}_{j-1}$
    Solve $\mathcal{P}\mathbf{z}_{j+1} = \mathbf{v}_{j+1}$
    $\gamma_{j+1} = \sqrt{\langle \mathbf{z}_{j+1}, \mathbf{v}_{j+1} \rangle}$
    $\alpha_0 = c_j\delta_j - c_{j-1}s_j\gamma_j$
    $\alpha_1 = \sqrt{\alpha_0^2 + \gamma_{j+1}^2}$
    $\alpha_2 = s_j\delta_j + c_{j-1}c_j\gamma_j$
    $\alpha_3 = s_{j-1}\gamma_j$
    $c_{j+1} = \frac{\alpha_0}{\alpha_1}$, $s_{j+1} = \frac{\gamma_{j+1}}{\alpha_1}$
    $\mathbf{w}_{j+1} = \frac{1}{\alpha_1}(\mathbf{z}_j - \alpha_3\mathbf{w}_{j-1} - \alpha_2\mathbf{w}_j)$
    $\mathbf{x}_j = \mathbf{x}_{j-1} + c_{j+1}\eta\mathbf{w}_{j+1}$
    $\eta = -s_{j+1}\eta$
    <Convergence Test>
end

Algorithm 4: MINRES algorithm for solving $\mathcal{A}\mathbf{x} = \mathbf{b}$ with preconditioner $\mathcal{P}$.

We emphasize that the preconditioned MINRES algorithm, when applied to a sparse matrix system, can be applied in a number of computational operations that scales linearly with the size of the system being solved, provided the preconditioner can itself be applied in a number of operations that also scales as such. We may therefore obtain useful solvers based on the MINRES algorithm, provided suitable preconditioners are found which may be cheaply applied.

We use the MINRES algorithm for solving a range of problems, including Poisson control problems in Chapters 3 and 4, convection-diffusion control problems in

Chapter 5, Stokes control problems in Chapter 6 and time-dependent optimal control problems in Chapter 8.

2.3.5 Conjugate Gradients

Another key method for solving symmetric linear systems is the Conjugate Gradient (CG) method. The method, in its standard form, was devised by Hestenes and Stiefel [63] for solving matrix systems of the form $\mathcal{A}\mathbf{x} = \mathbf{b}$, where $\mathcal{A}$ is symmetric positive definite.

The Conjugate Gradient method is another example of a Krylov subspace method. In CG successive iterates, of the form

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j, \quad j = 0, 1, 2, \dots ,$$

are generated. Here $\mathbf{p}_j$ denote *search directions* – these are $\mathcal{A}$-conjugate vectors, meaning that $\mathbf{p}_j^T \mathcal{A} \mathbf{p}_i = 0$ for $i \neq j$, which are found using the Gram-Schmidt method, and are related to the residual vectors $\mathbf{r}_j$. The values of α_j are chosen at each iteration to achieve stationary points of the quadratic form

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T \mathcal{A}\mathbf{x} - \mathbf{x}^T \mathbf{b},$$

the minimization of which is an equivalent problem to solving the matrix system $\mathcal{A}\mathbf{x} = \mathbf{b}$.

We do not give a detailed description of the Conjugate Gradient method in this thesis, and instead refer to [36, 63] for more thorough discussion.

An important property of CG is that it is a method which, at the j-th iterate, minimizes the $\mathcal{A}$-norm of the error, $\|\mathbf{x} - \mathbf{x}_j\|_{\mathcal{A}}$, over $\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$. Further, it may be shown [121, Lecture 38] that this error can be bounded by

$$\frac{\|\mathbf{x} - \mathbf{x}_j\|_{\mathcal{A}}}{\|\mathbf{x} - \mathbf{x}_0\|_{\mathcal{A}}} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^j ,$$

where κ again denotes the condition number (in the 2-norm) of $\mathcal{A}$, which for a symmetric matrix $\mathcal{A}$ is given by the quotient of the maximum and minimum eigenvalues of $\mathcal{A}$.

As for the MINRES method of the previous section, we may also consider a preconditioned CG method, provided we use a preconditioner $\mathcal{P}$ that is symmetric positive

definite. In this case, we may write $\mathcal{P} = \bar{H}\bar{H}^T$ for some matrix $\bar{H}$, and hence consider instead [36, 99]

$$\bar{H}^{-1}\mathcal{A}\bar{H}^{-T}\mathbf{y} = \bar{H}^{-1}\mathbf{b}, \quad \mathbf{y} = \bar{H}^T\mathbf{x}.$$

So, as for the MINRES algorithm discussed in Section 2.3.4, the matrix being solved for is $\bar{H}^{-1}\mathcal{A}\bar{H}^{-T}$, which is guaranteed to be symmetric and positive definite as $\mathcal{A}$ is. Again, the matrices $\bar{H}$ and $\bar{H}^T$ do not need to be applied in practice to enact the preconditioner – only $\mathcal{P}$ itself needs to be used. A preconditioned CG method can therefore be used, and is stated in Algorithm 5. We note that, as for the MINRES algorithm of the previous section, the computational cost of applying this algorithm to a sparse matrix system scales linearly with the size of the system being solved, provided the cost of applying the preconditioner $\mathcal{P}$ scales in this way.

Choose $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$
Solve $\mathcal{P}\mathbf{z}_0 = \mathbf{r}_0$
$\mathbf{p}_0 = \mathbf{z}_0$
for $j = 0, 1, 2, \ldots$ **do**
  $\alpha_j = \frac{\langle \mathbf{z}_j, \mathbf{r}_j \rangle}{\langle \mathcal{A}\mathbf{p}_j, \mathbf{p}_j \rangle}$
  $\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j$
  $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \mathcal{A}\mathbf{p}_j$
  <Convergence Test>
  Solve $\mathcal{P}\mathbf{z}_{j+1} = \mathbf{r}_{j+1}$
  $\beta_j = \frac{\langle \mathbf{z}_{j+1}, \mathbf{r}_{j+1} \rangle}{\langle \mathbf{z}_j, \mathbf{r}_j \rangle}$
  $\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$
end

Algorithm 5: Conjugate Gradient algorithm for solving $\mathcal{A}\mathbf{x} = \mathbf{b}$ with preconditioner $\mathcal{P}$.

Now, for the CG algorithm as described, we rely heavily on the matrix $\mathcal{A}$ being positive definite. However, for the problems we will seek to solve, this is not the case – in fact we will consider problems for which $\mathcal{A}$ is symmetric but indefinite – so the standard algorithm cannot be used. However, as discussed in [21, 99, 113] for instance, it is possible to write a CG algorithm in a non-standard inner product

$\langle \cdot, \cdot \rangle_{\mathcal{H}}$, in which the preconditioned matrix system is symmetric positive definite. In more detail, we select a symmetric positive definite inner product matrix $\mathcal{H}$ in which the preconditioned matrix is self-adjoint, i.e. for all vectors $\mathbf{v}$, $\mathbf{w}$ of appropriate dimension,

$$\begin{aligned} \langle \mathcal{P}^{-1}\mathcal{A}\mathbf{v}, \mathbf{w} \rangle_{\mathcal{H}} = \langle \mathbf{v}, \mathcal{P}^{-1}\mathcal{A}\mathbf{w} \rangle_{\mathcal{H}} \quad &\Leftrightarrow \quad \mathbf{w}^T \mathcal{H}\mathcal{P}^{-1}\mathcal{A}\mathbf{v} = \mathbf{w}^T \left(\mathcal{P}^{-1}\mathcal{A}\right)^T \mathcal{H}\mathbf{v} \\ &\Leftrightarrow \quad \mathbf{w}^T \mathcal{H}\mathcal{P}^{-1}\mathcal{A}\mathbf{v} = \mathbf{w}^T \mathcal{A}\mathcal{P}^{-1}\mathcal{H}\mathbf{v}, \end{aligned}$$

or equivalently,

$$\mathcal{H}\mathcal{P}^{-1}\mathcal{A} = \mathcal{A}\mathcal{P}^{-1}\mathcal{H}.$$

So provided we can find a symmetric and positive definite inner product $\mathcal{H}$, and preconditioner $\mathcal{P}$ such that the preconditioned system is symmetric positive definite in $\mathcal{H}$, we may apply a non-standard CG method in this inner product. This procedure is stated in Algorithm 6.

Choose $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$
Solve $\mathcal{P}\mathbf{z}_0 = \mathbf{r}_0$
$\mathbf{p}_0 = \mathbf{z}_0$
for $j = 0, 1, 2, \ldots$ **do**
  $\alpha_j = \frac{\langle \mathbf{z}_j, \mathbf{r}_j \rangle_{\mathcal{H}}}{\langle \mathcal{A}\mathbf{p}_j, \mathbf{p}_j \rangle_{\mathcal{H}}}$
  $\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j$
  $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \mathcal{A}\mathbf{p}_j$
  <**Convergence Test**>
  Solve $\mathcal{P}\mathbf{z}_{j+1} = \mathbf{r}_{j+1}$
  $\beta_j = \frac{\langle \mathbf{z}_{j+1}, \mathbf{r}_{j+1} \rangle_{\mathcal{H}}}{\langle \mathbf{z}_j, \mathbf{r}_j \rangle_{\mathcal{H}}}$
  $\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$
end

Algorithm 6: Conjugate Gradient algorithm for solving $\mathcal{A}\mathbf{x} = \mathbf{b}$ with preconditioner $\mathcal{P}$ in a non-standard inner product $\mathcal{H}$.

For a number of problems considered in this thesis, it is in fact possible to construct suitable and practical inner products such that a non-standard CG method may be

applied. We will employ the CG method in non-standard inner products in Chapter 3 for the distributed Poisson control problem, and in Chapter 5 for the convection-diffusion control problem – in each case we will explicitly specify the inner product needed to apply the algorithm.

2.3.6 GMRES

The preconditioned MINRES and Conjugate Gradient methods outlined above rely heavily on the symmetric positive definiteness of the preconditioner used. However, for a number of problems we consider in the remainder of this thesis, in particular more complex optimal control problems, the preconditioners we develop will not be positive definite, and in some cases will not be symmetric either. It is therefore very important that there are iterative methods available which can be used with such preconditioners also.

One such method which will be made use of is the Generalized Minimum Residual method (GMRES). This is a generalized form of MINRES developed by Saad and Schultz in [108] for the purpose of solving non-symmetric matrix systems $\mathcal{A}\mathbf{x} = \mathbf{b}$, as opposed to symmetric systems. GMRES is possibly the most widely used iterative solver for non-symmetric matrix systems.

This algorithm is based on the *Arnoldi method*,[6] which at the j-th step constructs an orthogonal basis $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_j\}$ of the Krylov subspace $\mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$, with $\mathbf{r}_0$ the initial residual, by using the modified Gram-Schmidt process. The method can be written as follows [36, Section 4.1.1]:

$$\mathcal{A}V_i = V_i H_i + h_{i+1,i}\left[\mathbf{0}, ..., \mathbf{0}, \mathbf{v}_{i+1}\right], \quad i = 1, 2, ..., j-1, \tag{2.39}$$

where $V_i = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_i]$, and H_i is an *upper Hessenberg matrix*, that is, a matrix with zeros everywhere except the diagonal and super-diagonal.

Finding an approximate solution $\mathbf{x}_j \in \mathbf{x}_0 + \mathcal{K}_j(\mathcal{A}, \mathbf{r}_0)$ at the j-th GMRES iteration, and choosing $\mathbf{v}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$, gives [36, Section 4.1.1]

$$\mathbf{x}_j = \mathbf{x}_0 + V_j \mathbf{y}_j,$$

for some $\mathbf{y}_j \in \mathbb{R}^j$.

[6] This generalizes the Lanczos method, as previously defined, to non-symmetric matrix systems.

The aim of the GMRES method is to minimize the vector 2-norm of the residual $\mathbf{r}_j$ at each iteration, which can be shown to be found by solving [36, Section 4.1.1]

$$\min \|\mathbf{r}_j\|_2 = \min_{\mathbf{y}_j} \left\| \|\mathbf{r}_0\|_2 \mathbf{e}_1 - \widehat{H}_j \mathbf{y}_j \right\|_2,$$

using a QR factorization applied to the matrix $\widehat{H}_j$, which is equal to the matrix H_j but with a final row $[0, ..., 0, h_{j+1,j}]$ added. Again, the vector $\mathbf{e}_1 = [1, 0, ..., 0]^T$.

We omit a thorough derivation of the GMRES method here, and refer instead to [36, 108] for detailed discussions. The preconditioned GMRES method we will use is discussed in [107] and stated in Algorithm 7.

Choose $\mathbf{x}_0$
$\mathbf{r}_1 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$
$\beta_0 = \|\mathbf{r}_0\|_2$
$\mathbf{v}_1 = \frac{\mathbf{r}_0}{\beta_0}$
for $j = 1, 2, ...$ **do**
  Solve $\mathcal{P}\mathbf{z}_j = \mathbf{v}_j$
  $\mathbf{w}_{j+1}^{(1)} = \mathcal{A}\mathbf{z}_j$
  for $k = 1, ..., j$ **do**
    $h_{k,j} = \langle \mathbf{w}_{j+1}^{(k)}, \mathbf{v}_k \rangle$
    $\mathbf{w}_{j+1}^{(k+1)} = \mathbf{w}_{j+1}^{(k)} - h_{k,j}\mathbf{v}_k$
  end
  $h_{j+1,j} = \left\| \mathbf{w}_{j+1}^{(j+1)} \right\|_2$
  $\mathbf{v}_{j+1} = \frac{\mathbf{w}_{j+1}^{(j+1)}}{h_{j+1,j}}$
  Find $\mathbf{y}_j$ such that $\mathbf{y}_j$ minimizes $\beta_j = \left\| \beta_0 \mathbf{e}_1 - \widehat{H}_j \mathbf{y}_j \right\|_2$
  <**Convergence Test**>
end
$V_j = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_j]$
Solve $\mathcal{P}\mathbf{z}_j = V_j \mathbf{y}_j$
$\mathbf{x} = \mathbf{x}_0 + \mathbf{z}_j$

Algorithm 7: GMRES algorithm for solving $\mathcal{A}\mathbf{x} = \mathbf{b}$ with (right) preconditioner $\mathcal{P}$.

The preconditioned GMRES algorithm has been used to good effect to solve a

number of non-symmetric matrix systems arising from practical problems. In the context of the numerical solution of PDEs, the applicability of the method to solving the convection-diffusion equation and Navier-Stokes equations is discussed in [36, Chapters 4 and 8]. As PDE-constrained optimization problems include many of the matrices and structures involved in such simpler PDEs, it seems a natural choice to apply the GMRES method to problems of this type as well, in the cases where methods such as MINRES and non-standard Conjugate Gradients cannot be used.

However, there are some drawbacks of this method when compared to, say, the MINRES method, of which it is a generalization. The major additional computational expenses of the GMRES method as opposed to the MINRES method are the computation and storage of Hessenberg matrices $\widehat{H}_j$ and matrices V_j containing Arnoldi vectors, as well as operations resulting from the presence of these matrices. Of course one only has to store two such matrices at a time, as $\widehat{H}_j$ and V_j may be created by modifying $\widehat{H}_{j-1}$ and V_{j-1}; however both matrices grow at each iteration. Storing and solving the least squares problems involving the Hessenberg matrices would become very expensive if many iterations were required to solve the matrix system.

Another disadvantageous feature of applying GMRES instead of MINRES is that there is not as full a picture of the rate of convergence of the GMRES algorithm as for MINRES. In particular, whereas the convergence bound (2.38) can be estimated purely using information about the eigenvalues of the (preconditioned) matrix, there is no bound on the convergence of GMRES that requires only this information. One can make some comments on GMRES convergence – for instance provided the matrix $\mathcal{A}$ is diagonalizable, one known bound on the j-th residual obtained by applying the (unpreconditioned) method is given by [36, Theorem 4.1]

$$\frac{\|\mathbf{r}_j\|_2}{\|\mathbf{r}_0\|_2} \leq \kappa(Y) \min_{p_j \in \Pi_j, p_j(0)=1} \max_{\lambda_i} |p_j(\lambda_i)|,$$

where the eigendecomposition of $\mathcal{A}$ is $Y\widehat{\Lambda}Y^{-1}$, λ_i denote the eigenvalues of $\mathcal{A}$ (i.e. the diagonal entries of $\widehat{\Lambda}$), and Π_j again defines the set of polynomials of degree at most j. However, this bound can clearly not be predicted solely using information about eigenvalues of the matrix system – this is a significant drawback when attempting to ascertain the effectiveness of our preconditioners theoretically.

We do however find the GMRES method to be an effective one for a number of problems, and we apply this method when solving Stokes control problems in Chapter

6 and Navier-Stokes control problems in Chapter 7. In each case, the preconditioners which we motivate for the problem at hand are not symmetric positive definite, so it is crucial to have such a method which may be applied to these problems.

2.3.7 BiCG

The GMRES method introduced in the previous section is by no means the only useful iterative solver for non-symmetric matrix systems. Another powerful example of such a solver is the Biconjugate Gradient method (or BiCG method). This method was designed as an analogue of the Conjugate Gradient method, for the purpose of solving non-symmetric systems (see [39, 81]). In fact, the matrices that we will use this method for solving are symmetric, but the preconditioners we apply are neither symmetric nor positive definite – we find that this method works well for the problems we consider.

Choose $\mathbf{x}_0$

$\mathbf{r}_0 = \mathbf{b} - \mathcal{A}\mathbf{x}_0$

Solve $\mathcal{P}\mathbf{p}_0 = \mathbf{r}_0$

Choose $\widehat{\mathbf{r}}_0$ such that $\langle \mathbf{r}_0, \widehat{\mathbf{r}}_0 \rangle \neq 0$

Solve $\mathcal{P}^T\widehat{\mathbf{p}}_0 = \widehat{\mathbf{r}}_0$

for $j = 0, 1, 2, \ldots$ **do**

$\alpha_j = \frac{\langle \mathbf{z}_j, \widehat{\mathbf{r}}_j \rangle}{\langle \mathcal{A}\mathbf{p}_j, \widehat{\mathbf{p}}_j \rangle}$

$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j$

$\widehat{\mathbf{x}}_{j+1} = \widehat{\mathbf{x}}_j + \alpha_j \widehat{\mathbf{p}}_j$

$\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \mathcal{A}\mathbf{p}_j$

$\widehat{\mathbf{r}}_{j+1} = \widehat{\mathbf{r}}_j - \alpha_j \mathcal{A}^T\widehat{\mathbf{p}}_j$

<**Convergence Test**>

Solve $\mathcal{P}\mathbf{z}_{j+1} = \mathbf{r}_{j+1}$

Solve $\mathcal{P}^T\widehat{\mathbf{z}}_{j+1} = \widehat{\mathbf{r}}_{j+1}$

$\beta_j = \frac{\langle \mathbf{z}_{j+1}, \widehat{\mathbf{r}}_{j+1} \rangle}{\langle \mathbf{z}_j, \widehat{\mathbf{r}}_j \rangle}$

$\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$

$\widehat{\mathbf{p}}_{j+1} = \widehat{\mathbf{z}}_{j+1} + \beta_j \widehat{\mathbf{p}}_j$

end

Algorithm 8: Biconjugate Gradient algorithm for solving $\mathcal{A}\mathbf{x} = \mathbf{b}$ (where $\mathcal{A}$ is a real matrix) with preconditioner $\mathcal{P}$.

The method is expressed as pseudocode in Algorithm 8. The main differences from CG involve computing two residuals $\mathbf{r}_j$ and $\widehat{\mathbf{r}}_j$ instead of a single one for CG, as well as applying operations resulting from the preconditioner $\mathcal{P}$ twice per BiCG iteration rather than once, and using this to calculate quantities $\mathbf{p}_j$ and $\widehat{\mathbf{p}}_j$ at each step.

For symmetric matrix systems being solved with symmetric positive definite preconditioners, this algorithm reduces to the preconditioned Conjugate Gradient algorithm introduced earlier. The convergence of BiCG for this case is therefore the same as for Conjugate Gradients, but there are few other results on the convergence of this method. This is a similar drawback as for the Gmres method, as discussed in the previous section. Of course there are advantages of Gmres and BiCG over each other. For instance, the Gmres method requires the construction of a Hessenberg matrix and a matrix containing Arnoldi vectors which grow with each iteration; BiCG does not require this. Conversely, each iteration of preconditioned BiCG requires two applications of the selected preconditioner, versus the single application required at each Gmres iteration.

We note that, like the Conjugate Gradient method, it is possible to apply this algorithm in a non-standard inner product (see [105]). We also note that a stabilized version of this algorithm, Biconjugate Gradients Stabilized (BiCGStab), was created by Van der Vorst in [124], however we find that the simpler BiCG algorithm is sufficient for our purposes. We apply this method to the problem of reaction-diffusion control for chemical processes in Chapter 9.

There is a variety of solvers for non-symmetric matrix systems other than the Gmres and BiCG algorithms discussed in this chapter, for example the Quasi-Minimal Residual (QMR) algorithm [41], the Ideal Transpose-Free QMR (ITFQMR) algorithm [40], and the Conjugate Gradients Squared (CGS) algorithm [112]. However, we focus our attention on Gmres and BiCG as our non-symmetric solvers in this thesis.

CHAPTER 3

Distributed Poisson Control

As discussed in the previous chapter, one of the most fundamental problems in the field of PDE-constrained optimization is that of the distributed Poisson control problem:

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2}\left\|y-\widehat{y}\right\|_{L_2(\Omega)}^2+\frac{\beta}{2}\left\|u\right\|_{L_2(\Omega)}^2 \\ \text{s.t.} \quad & -\nabla^2 y=u, \quad \text{in } \Omega, \\ & y=g, \quad \text{on } \partial\Omega. \end{aligned}$$

In this chapter, we consider the iterative solution of this problem.[7] Much work has gone into developing effective iterative methods for such Poisson control problems (see [100, 101, 109] for instance for preconditioned iterative methods, and [15, 16, 17, 118] for multigrid methods). We will focus on three preconditioned iterative methods in particular that have been previously developed for this problem, all motivated using the saddle point theory of Section 2.2. Firstly, in [100], Rees, Dollar and Wathen developed a block diagonal preconditioner for use with MINRES. Secondly, a block

[7] This chapter is based on the following paper, which is Ref. [94]:

J. W. Pearson, and A. J. Wathen, *A New Approximation of the Schur Complement in Preconditioners for PDE-Constrained Optimization*, Numerical Linear Algebra with Applications, 19(5), pp.816–829, 2012.

The author notes that the content of Section 3.3 did not appear in this article.

triangular preconditioner for use with a non-standard (Bramble-Pasciak) Conjugate Gradient algorithm was introduced by Rees and Stoll in [101]. Thirdly, a symmetric indefinite preconditioner was derived by Schöberl and Zulehner in [109] to be used with a different non-standard Conjugate Gradient method. In their construction of a block preconditioner for use with a non-standard inner product, Schöberl and Zulehner pay particular attention to achieving robust behaviour with respect to small values of β, however the other two methods are independent with respect to h only. In fact, as β is decreased, the performance of the solvers discussed in [100] and [101] deteriorates quite rapidly. We wish to see if we may improve the effectiveness of these preconditioners.

Our specific goal when developing solvers for this problem, as in the remainder of this thesis, is to construct preconditioners that are robust with respect to problem size, as well as the regularization parameter β. Motivated by this objective, we describe a new Schur complement approximation for PDE-constrained optimization problems which can be employed in the three approaches mentioned (as well as others), and which yields convergence of the appropriate iterative method in a number of steps which is independent of the value of the regularization parameter β, as well as the mesh-size h. We prove the relevant eigenvalue bounds which guarantee this property.

This chapter is structured as follows. In Section 3.1, we review the three preconditioners we have mentioned that have previously been proposed for solving the matrix system (2.8). In Section 3.2, we propose a new approximation to the Schur complement which, along with a good approximation to the mass matrix M, can be incorporated into any of the three preconditioners of Section 3.1. We prove a result on the eigenvalues of the preconditioned Schur complement with this new approximation, to demonstrate analytically why our approximation to the Schur complement is effective. In Section 3.3, we consider the eigenvalues of the entire preconditioned matrix system with this approximation. In Section 3.4, we present numerical results to show how well our approximation works within the three preconditioners in practice, and in Section 3.5, we make some concluding comments.

3.1 Previously Developed Preconditioners for Distributed Poisson Control

The problem we will aim to solve efficiently is of the saddle point form described in Section 2.2: we write the matrix system in the form $\mathcal{A}\mathbf{x} = \mathbf{b}$, with

$$\mathcal{A} = \begin{bmatrix} M & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix}.$$

We may write, in the notation of (2.31), that for this problem

$$\Phi = \begin{bmatrix} M & 0 \\ 0 & \beta M \end{bmatrix}, \quad \Psi = \begin{bmatrix} K & -M \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

Two natural options for approximating $\mathcal{A}$ would be to devise preconditioners of the form

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \widehat{\Phi} & 0 \\ 0 & \widehat{S} \end{bmatrix}, \quad \widehat{\mathcal{P}}_2 = \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -\widehat{S} \end{bmatrix}, \tag{3.1}$$

where $\widehat{S} \approx S = \Psi\Phi^{-1}\Psi^T$, as discussed in the context of general saddle point systems in Section 2.2. The formulations of the preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ motivate our discussion in Sections 3.1.1 and 3.1.2 respectively.

A third option is to construct a preconditioner for the matrix system (2.31) of the form

$$\widehat{\mathcal{P}}_4 = \begin{bmatrix} \widehat{\Phi} & \Psi^T \\ \Psi & \Psi\widehat{\Phi}^{-1}\Psi^T - \widehat{S} \end{bmatrix} = \begin{bmatrix} I & 0 \\ \Psi\widehat{\Phi}^{-1} & I \end{bmatrix} \begin{bmatrix} \widehat{\Phi} & \Psi^T \\ 0 & -\widehat{S} \end{bmatrix},$$

meaning that one application of the approximation of $\widehat{S}$ and two of $\widehat{\Phi}$ are required to effect this preconditioner. We will discuss this preconditioner further in Section 3.1.3.

In Sections 3.1.1–3.1.3, we consider three preconditioners, one each of the form $\widehat{\mathcal{P}}_1$, $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_4$. These are "optimal" provided that the inverses of $\widehat{\Phi}$ and $\widehat{S}$ can themselves be applied with optimal complexity. That is to say, an appropriate Krylov subspace method combined with each preconditioner has linear complexity in matrix size, or

alternatively that the number of iterations required for convergence of the solver is bounded independently of the mesh, with linear work required for each iteration, as long as the inverses of $\widehat{\Phi}$ and $\widehat{S}$ can be applied with optimal complexity.

We will note in particular the Schur complement approximations used in these preconditioners. This work will later be incorporated when testing the Schur complement approximation that we propose in Section 3.2.

3.1.1 Block Diagonal Preconditioners

As motivated by the discussion above, one option for constructing a solver for the Poisson control problem is to apply the following block diagonal preconditioner:

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \widehat{M} & 0 & 0 \\ 0 & \beta\widehat{M} & 0 \\ 0 & 0 & \widehat{S} \end{bmatrix} \tag{3.2}$$

to the matrix system in (2.8), where $\widehat{M}$ and $\widehat{S}$ are approximations of the mass matrix M and the Schur complement S respectively. Here, the approximation of the $(1,1)$-block of the saddle point system is given by $\widehat{\Phi} = \text{blkdiag}(\widehat{M}, \beta\widehat{M})$. The approximations $\widehat{M}$ and $\widehat{S}$ must be symmetric and positive definite; thus $\widehat{\mathcal{P}}_1$ is symmetric and positive definite, and so it is possible to build this into a Krylov subspace algorithm for symmetric matrices, such as the MINRES algorithm introduced in Section 2.3.4.

Rees, Dollar and Wathen [100] introduced the block diagonal preconditioner (3.2) for PDE-constrained optimization; they took the mass matrix approximation $\widehat{M}$ to be a fixed number of steps of the Chebyshev semi-iteration method described in Section 2.3.2 – this strategy is a well-founded one, and we make use of it frequently for other problems also.

The authors then considered the Schur complement of the matrix system

$$S = KM^{-1}K + \frac{1}{\beta}M,$$

which they approximated by dropping the second term, that is

$$S \approx KM^{-1}K =: \widehat{S}_1. \tag{3.3}$$

This choice of approximation is motivated by the fact that the $\frac{1}{\beta}M$ term is a higher

order term in the mesh-size h than $KM^{-1}K$. (Theorem 7 of Section 3.2 gives upper and lower bounds for $\widehat{S}_1^{-1}S$.) To approximate K when required, a limited number of multigrid cycles, described in Section 2.3.3, are applied. That is to say the multigrid algorithm is not run to convergence, but one or two cycles are used to accelerate the convergence of the outer MINRES iteration. Two such multigrid processes to approximate the matrix K, as well as one matrix multiplication to represent M, are required per application of $\widehat{S}_1$. This approximation leads to mesh-independent convergence, but convergence is not β-independent. The preconditioner we recommend in Section 3.2 enables us to overcome this issue, so that our method is insensitive to the value of β, without resulting in significant additional work.

3.1.2 Block Triangular Preconditioners

When employing a block triangular preconditioner of the form $\widehat{\mathcal{P}}_2$, there is no simple symmetric formulation, as for the block diagonal preconditioner of the previous section, since the matrix $\widehat{\mathcal{P}}_2$ is non-symmetric. This means that a method such as MINRES or Conjugate Gradients with a standard inner product cannot be used with this preconditioner. However, it is possible to apply a non-standard Conjugate Gradient method of the form described in Section 2.3.5. As outlined in [21, 116, 128] and discussed in the context of Poisson control by Rees and Stoll in [101], $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$ is self-adjoint and positive definite in the inner product defined by $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{H}} = \mathbf{u}^T \mathcal{H} \mathbf{v}$, where

$$\mathcal{H} = \begin{bmatrix} \Phi - \widehat{\Phi} & 0 \\ 0 & \widehat{S} \end{bmatrix}.$$

One is therefore able to use the theory of Section 2.3.5, and apply the Conjugate Gradient method with this non-standard inner product. This is a method known as the *Bramble-Pasciak Conjugate Gradient method*. Note that $\Phi - \widehat{\Phi}$ (as well as $\widehat{S}$) must be positive definite for this to define an inner product, and so $\widehat{\Phi}$ must be carefully constructed with this in mind. For the Poisson control problem, this can readily be achieved.

In [101], the application of this method to Poisson control is discussed. In this

context, we may write the preconditioner as

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \gamma\widehat{M} & 0 & 0 \\ 0 & \beta\gamma\widehat{M} & 0 \\ K & -M & -\widehat{S} \end{bmatrix}, \tag{3.4}$$

and the inner product as

$$\mathcal{H} = \begin{bmatrix} M - \gamma\widehat{M} & 0 & 0 \\ 0 & \beta(M - \gamma\widehat{M}) & 0 \\ 0 & 0 & \widehat{S} \end{bmatrix},$$

for a parameter γ which ensures that $M - \gamma\widehat{M}$ is positive definite.

As discussed in [101], it is effective to again take $\widehat{M}$ as a fixed number of Chebyshev semi-iterations, and $\widehat{S}$ as two multigrid approximations and a matrix multiplication. In [101] $\widehat{S}$ was again taken to be $\widehat{S}_1$ as in (3.3). Particular emphasis is made in [101] of the ease of choosing γ, which is a scaling constant introduced to ensure that $M - \gamma\widehat{M}$ is positive definite for the PDE-constrained optimization problem, and hence that the inner product matrix $\mathcal{H}$ is positive definite. As explained by the authors, known eigenvalue bounds of $\widehat{M}^{-1}M$ for a fixed number of Chebyshev semi-iterations, as highlighted in Table 2.2 for **Q1** basis functions for instance, mean that choosing a constant γ less than but close to 1 will ensure positive definiteness of this inner product matrix.

3.1.3 Symmetric Indefinite Preconditioners

In [109], a preconditioner of the form $\widehat{\mathcal{P}}_4$ is proposed for use with the Conjugate Gradient method in a non-standard inner product (as outlined in Section 2.3.5). Here, $\widehat{\Phi}$ and $\widehat{S}$ are chosen such that the preconditioned system is positive definite with respect to the inner product $\langle\cdot,\cdot\rangle_{\bar{\mathcal{H}}}$, where

$$\langle \mathbf{u}, \mathbf{v} \rangle_{\bar{\mathcal{H}}} = \mathbf{u}^T \underbrace{\begin{bmatrix} \widehat{\Phi} - \Phi & 0 \\ 0 & \Psi\widehat{\Phi}^{-1}\Psi^T - \widehat{S} \end{bmatrix}}_{\bar{\mathcal{H}}} \mathbf{v}.$$

We note that $\widehat{\Phi}$ here must be carefully constructed such that $\widehat{\Phi} - \Phi$ is positive definite, and $\widehat{S}$ constructed such that $\Psi\widehat{\Phi}^{-1}\Psi^T - \widehat{S}$ is positive definite. For the Poisson control

problem, this can feasibly be done using the strategy presented in this section.

When considering this problem, Schöberl and Zulehner [109] took approximations

$$\widehat{\Phi} = \frac{1}{\sigma}\widehat{\Phi}_0, \quad \widehat{S} = \frac{1}{\bar{\tau}}\widehat{S}_0, \tag{3.5}$$

with

$$\widehat{\Phi}_0 = \begin{bmatrix} \widehat{Y} & 0 \\ 0 & \beta\widehat{M} \end{bmatrix}, \quad \widehat{S}_0 = \frac{\sigma}{\beta}\widehat{Y}, \tag{3.6}$$

for positive constants σ and $\bar{\tau}$. Here, $\widehat{Y}$ denotes a multigrid solver applied to the matrix $\sqrt{\beta}K + M$, and $\widehat{M}$ denotes application of the symmetric Gauss-Seidel method to the appropriate mass matrix. (Throughout the remainder of this chapter, we will replace this mass matrix approximation with the Chebyshev semi-iteration method described in Section 2.3.2.) The formulation of (3.5) and (3.6) guarantees that

$$\frac{1}{\sigma}\widehat{\Phi}_0 > \Phi, \quad \frac{1}{\bar{\tau}}\widehat{S}_0 < \Psi\widehat{\Phi}^{-1}\Psi^T,$$

and hence that the preconditioned system is positive definite in the inner product defined by $\bar{\mathcal{H}}$. The overall preconditioner for the Poisson control problem therefore looks as follows:

$$\begin{aligned} \widehat{\mathcal{P}}_4 &= \begin{bmatrix} \frac{1}{\sigma}\widehat{Y} & 0 & K \\ 0 & \frac{\beta}{\sigma}\widehat{M} & -M \\ K & -M & \sigma K\widehat{Y}^{-1}K + \frac{\sigma}{\beta}M\widehat{M}^{-1}M - \frac{\sigma}{\bar{\tau}\beta}\widehat{Y} \end{bmatrix} \\ &= \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ \sigma K\widehat{Y}^{-1} & -\frac{\sigma}{\beta}M\widehat{M}^{-1} & I \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma}\widehat{Y} & 0 & K \\ 0 & \frac{\beta}{\sigma}\widehat{M} & -M \\ 0 & 0 & -\frac{\sigma}{\bar{\tau}\beta}\widehat{Y} \end{bmatrix}. \end{aligned} \tag{3.7}$$

In [109], it is recommended that $\sigma \approx 1$ and $\bar{\tau} \approx \frac{4}{3}$, and these are the values we use in the computations of Section 3.4. We note that this solver was designed by the authors to exhibit independence with respect to mesh-size and regularization parameter, in contrast to the solvers discussed in Sections 3.1.1 and 3.1.2.

3.2 A New Schur Complement Approximation

All three methods detailed in Section 3.1 rely heavily on an accurate approximation of the Schur complement. When the block diagonal preconditioner for MINRES and block triangular preconditioner for Bramble-Pasciak Conjugate Gradients are applied, the iteration count becomes prohibitively large for small values of β. This is due to the neglected $\frac{1}{\beta}M$ term in the Schur complement approximation (3.3). A detailed computational analysis of the asymptotic behaviour of eigenvalue properties of the matrix $\mathcal{A}$ for small values of β is given in [119, 120]. In this section, we introduce an alternative Schur complement approximation that is robust with respect to all values of h and β.

Instead of (3.3), it is easily checked that we can write

$$S = \left(K + \frac{1}{\sqrt{\beta}}M\right) M^{-1} \left(K + \frac{1}{\sqrt{\beta}}M\right) - \frac{2}{\sqrt{\beta}}K,$$

which enables us to take the approximation

$$S \approx \widehat{S}_2 := \left(K + \frac{1}{\sqrt{\beta}}M\right) M^{-1} \left(K + \frac{1}{\sqrt{\beta}}M\right) \tag{3.8}$$

by dropping the $-\frac{2}{\sqrt{\beta}}K$ term. We highlight that the term discarded here is $\mathcal{O}(\beta^{-1/2})$ rather than $\mathcal{O}(\beta^{-1})$, as was the case in the Schur complement approximation $\widehat{S}_1$. When we wish to use preconditioners involving the approximate Schur complement, the factorization (3.8) enables us to apply multigrid on two occasions to the matrix $K + \frac{1}{\sqrt{\beta}}M$ rather than the matrix K, together, as before, with a mass matrix multiplication. We note the similarity of this to applying the multigrid process $\widehat{Y}$ to the matrix $\sqrt{\beta}K + M$ as discussed in Section 3.1.3, but note also that the multigrid cycles are applied at different points in the preconditioners.

We demonstrate theoretically why this approximation is more potent than the approximation $\widehat{S}_1$ defined in (3.3). Note that Schöberl and Zulehner derive the different Schur complement approximation (3.5), (3.6) in [109]; we will demonstrate in Section 3.4 that our approximation (3.5), (3.8) also fits nicely into the preconditioning framework developed in [109].

In order to obtain convergence bounds or estimates, we wish to obtain eigenvalue bounds for $\widehat{S}_1^{-1}S$ and $\widehat{S}_2^{-1}S$. To calculate the eigenvalues for the former, as done in

[100], Theorems 5 and 6, which are stated in [36, pp.57–60], need to be utilized.[8]

Theorem 5. *For the problem* (2.8) *for* $\Omega \subset \mathbb{R}^2$*, with a degree of approximation* $\mathbf{P}_m$ *or* $\mathbf{Q}_m$ *with* $m \geq 1$*, the following bound holds:*

$$c_1 h^2 \leq \frac{\mathbf{v}^T M \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \leq C_1 h^2, \quad \forall \mathbf{v} \in \mathbb{R}^{\bar{N}},$$

where the positive (real) constants c_1 *and* C_1 *are independent of the mesh-size* h *but dependent on* m*.*
For $\Omega \subset \mathbb{R}^3$*, the equivalent result is*

$$c_1 h^3 \leq \frac{\mathbf{v}^T M \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \leq C_1 h^3, \quad \forall \mathbf{v} \in \mathbb{R}^{\bar{N}}.$$

Theorem 6. *For the problem* (2.8) *for* $\Omega \subset \mathbb{R}^2$*, with a degree of approximation* $\mathbf{P}_m$ *or* $\mathbf{Q}_m$ *with* $m \geq 1$*, the following bound holds:*

$$c_2 h^2 \leq \frac{\mathbf{v}^T K \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \leq C_2, \quad \forall \mathbf{v} \in \mathbb{R}^{\bar{N}},$$

where the positive (real) constants c_2 *and* C_2 *are independent of the mesh-size* h *but dependent on* m*.*
For $\Omega \subset \mathbb{R}^3$*, the equivalent result is*

$$c_2 h^3 \leq \frac{\mathbf{v}^T K \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \leq C_2 h, \quad \forall \mathbf{v} \in \mathbb{R}^{\bar{N}}.$$

Theorems 5 and 6 give us that in 2D or 3D, for any $\mathbf{v} \in \mathbb{R}^{\bar{N}}$,

$$ch^2 \leq \frac{\mathbf{v}^T M \mathbf{v}}{\mathbf{v}^T K \mathbf{v}} \leq C, \tag{3.9}$$

for positive (real) constants c and C independent of the mesh-size h, and therefore that the eigenvalues of $K^{-1}M$ are contained in an interval of the form $[ch^2, C]$.

Theorem 7 as stated below is proved in [100] using the result (3.9).

Theorem 7. *The eigenvalues of* $\widehat{S}_1^{-1} S$ *are bounded as follows:*

$$\lambda(\widehat{S}_1^{-1} S) \in \left[\frac{1}{\beta} \widetilde{c} h^4 + 1, \frac{1}{\beta} \widetilde{C} + 1 \right],$$

[8] In Theorems 5 and 6, the parameter $\bar{N}$ refers to the size of the matrices K and M being considered.

for positive (real) constants $\widetilde{c}$ and $\widetilde{C}$ independent of h and β.

The Schur complement approximation (3.8) is an improved one, as Theorem 8 demonstrates.

Theorem 8. *The eigenvalues of $\widehat{S}_2^{-1}S$ satisfy the following bound:*

$$\lambda(\widehat{S}_2^{-1}S) \in \left[\frac{1}{2}, 1\right],$$

independently of the values of h and β.

Proof. First note that $\widehat{S}_2$ is always non-singular since M is positive definite, and that $K^{-1}M + \sqrt{\beta}I$ is positive real (as it is similar to the symmetric positive definite matrix $M^{1/2}K^{-1}M^{1/2} + \sqrt{\beta}I$) and hence invertible. If we denote the eigenvalues of $\widehat{S}_2^{-1}S$ by $\bar{\mu}$, and the corresponding eigenvectors by $\mathbf{x}$, then

$$\begin{aligned}
\widehat{S}_2^{-1}S\mathbf{x} = \bar{\mu}\mathbf{x} \quad &\Rightarrow \quad \left(KM^{-1}K + \frac{1}{\beta}M\right)\mathbf{x} = \bar{\mu}\left(K + \frac{1}{\sqrt{\beta}}M\right)M^{-1}\left(K + \frac{1}{\sqrt{\beta}}M\right)\mathbf{x} \\
&\Rightarrow \quad \left(KM^{-1}K + \frac{1}{\beta}M\right)\mathbf{x} = \bar{\mu}\left[KM^{-1}K + \frac{2}{\sqrt{\beta}}K + \frac{1}{\beta}M\right]\mathbf{x} \\
&\Rightarrow \quad \left(I + \frac{1}{\beta}K^{-1}MK^{-1}M\right)\mathbf{x} = \bar{\mu}\left[I + \frac{2}{\sqrt{\beta}}K^{-1}M + \frac{1}{\beta}K^{-1}MK^{-1}M\right]\mathbf{x} \\
&\Rightarrow \quad \left(\beta I + K^{-1}MK^{-1}M\right)\mathbf{x} = \bar{\mu}\left[\beta I + 2\sqrt{\beta}K^{-1}M + K^{-1}MK^{-1}M\right]\mathbf{x} \\
&\Rightarrow \quad \left(\beta I + (K^{-1}M)^2\right)\mathbf{x} = \bar{\mu}\left(K^{-1}M + \sqrt{\beta}I\right)^2\mathbf{x} \\
&\Rightarrow \quad \left(K^{-1}M + \sqrt{\beta}I\right)^{-2}\left(\beta I + (K^{-1}M)^2\right)\mathbf{x} = \bar{\mu}\mathbf{x}.
\end{aligned}$$

So we deduce for each eigenvalue χ of $K^{-1}M$, that $\frac{\chi^2+\beta}{(\chi+\sqrt{\beta})^2}$ is an eigenvalue of $\widehat{S}_2^{-1}S$. Now since $K^{-1}M$ is similar to a real symmetric matrix ($M^{1/2}K^{-1}M^{1/2}$), it is diagonalizable, and hence this describes all eigenvalues of $\widehat{S}_2^{-1}S$. But $\frac{\chi^2+\beta}{(\chi+\sqrt{\beta})^2}$ is simply a function of the form $\frac{a^2+b^2}{(a+b)^2}$ with a and b real and positive. It is a simple algebraic task to show that $\frac{1}{2} \leq \frac{a^2+b^2}{(a+b)^2} \leq 1$, and hence that

$$\lambda(\widehat{S}_2^{-1}S) \in \left[\frac{1}{2}, 1\right]. \tag{3.10}$$

□

We thus have a very simple bound for $\widehat{S}_2^{-1}S$. Note that to demonstrate that this

bound holds, we did not need to use any spectral properties of K or M besides the fact that all eigenvalues of $K^{-1}M$ are real. Hence the bound (3.10) must also hold for any positive definite self-adjoint operator appropriately approximated by a symmetric matrix, regardless of the order of the elliptic operator.

In Figure 3.1, we compare the eigenvalue spectra of $\widehat{S}_1^{-1}S$ and $\widehat{S}_2^{-1}S$ for the Poisson control problem for a range of values of β. We can see that for large β the approximation $\widehat{S}_1$ of S is likely to be very effective, as all eigenvalues of $\widehat{S}_1^{-1}S$ are clustered close to 1; but, as β becomes smaller the eigenvalues become increasingly spread out, and the convergence of an iterative method will generally suffer as a result. However, for the new approximation $\widehat{S}_2$, the eigenvalues of $\widehat{S}_2^{-1}S$ are pinned down into a fixed interval as predicted by (3.10), so an appropriate iterative method should perform well regardless of how small β is – note carefully the vertical scales of the individual plots in Figure 3.1 to see this. To illustrate the spreading of the eigenvalues of $\widehat{S}_1^{-1}S$, we include more detailed plots of the intermediate eigenvalues for $\beta = 10^{-4}$ and $\beta = 10^{-7}$ in Figure 3.2.

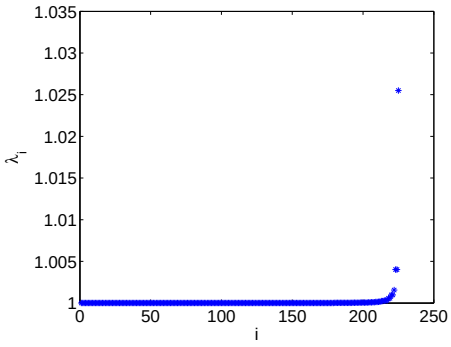
Note. As the state, control and adjoint are here all discretized using the same piecewise polynomial approximation spaces (as in [95] for example), it would be possible to use the discretized gradient equation to eliminate the second block of the matrix system (2.8), and then solve the remaining 2×2 block system

$$\begin{bmatrix} M & K \\ K & -\frac{1}{\beta}M \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{g} \end{bmatrix}, \tag{3.11}$$

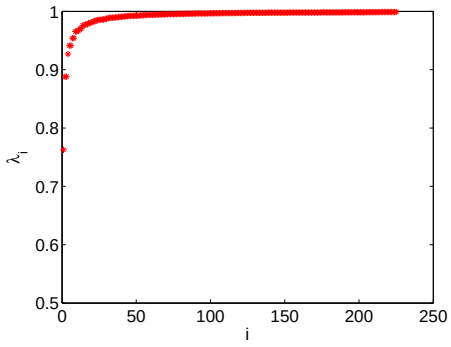
which is still a saddle point system of the form (2.31), but now with $\Theta \neq 0$. The Schur complement of the resulting system would then be exactly the same as that of the original 3×3 block system, so the approximation $\widehat{S}_2$ to S detailed in this section is equally useful in this case. Due to Theorem 4 of Section 2.2, this is a viable solution strategy when block diagonal preconditioners are used; Theorem 2 also guarantees the effectiveness of a block triangular preconditioner for this matrix system.

Note. We highlight at this point the presence of another block diagonal preconditioner for the Poisson control problem in the literature: in [129], Zulehner uses a non-standard norm argument to derive the following preconditioner for (3.11):

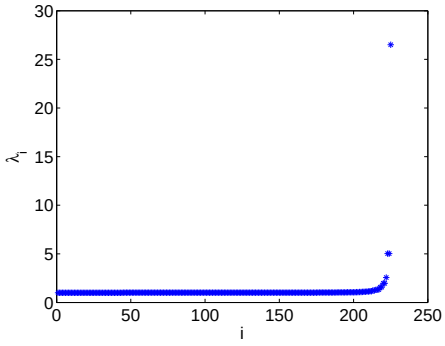
$$\begin{bmatrix} M + \sqrt{\beta}K & 0 \\ 0 & \frac{1}{\beta}(M + \sqrt{\beta}K) \end{bmatrix},$$



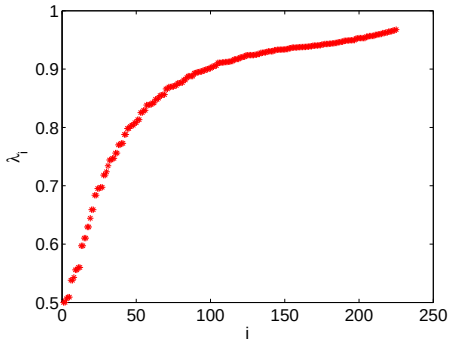
(a) $\lambda_i(\widehat{S}_1^{-1}S)$, $\beta = 10^{-1}$, $i = 1, ..., 225$



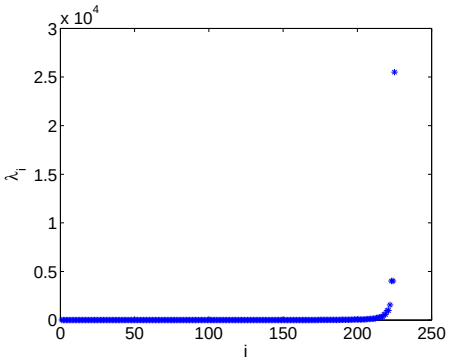
(b) $\lambda_i(\widehat{S}_2^{-1}S)$, $\beta = 10^{-1}$, $i = 1, ..., 225$



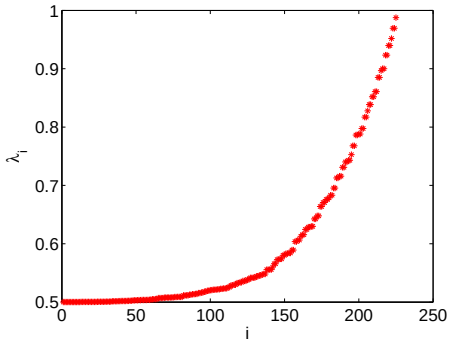
(c) $\lambda_i(\widehat{S}_1^{-1}S)$, $\beta = 10^{-4}$, $i = 1, ..., 225$



(d) $\lambda_i(\widehat{S}_2^{-1}S)$, $\beta = 10^{-4}$, $i = 1, ..., 225$

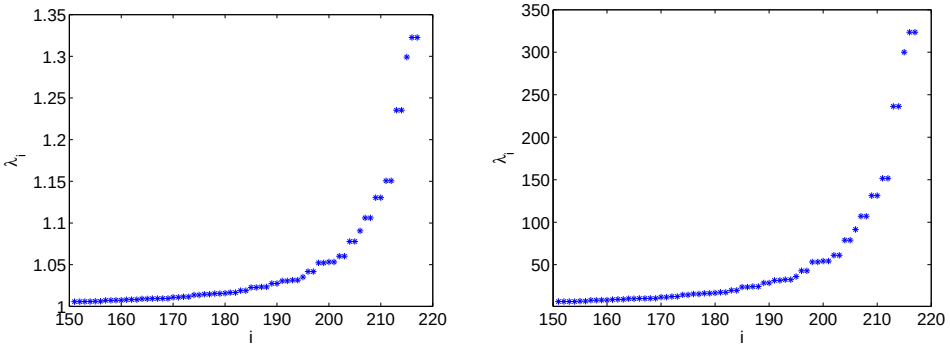


(e) $\lambda_i(\widehat{S}_1^{-1}S)$, $\beta = 10^{-7}$, $i = 1, ..., 225$



(f) $\lambda_i(\widehat{S}_2^{-1}S)$, $\beta = 10^{-7}$, $i = 1, ..., 225$

Figure 3.1: Spectra of $\widehat{S}_1^{-1}S$ [(a), (c), (e)] and $\widehat{S}_2^{-1}S$ [(b), (d), (f)] for $\beta = 10^{-1}$, $\beta = 10^{-4}$ and $\beta = 10^{-7}$, for an evenly spaced grid on $\Omega = [0, 1]^2$ with $h = 2^{-4}$.



(a) $\lambda_i(\widehat{S}_1^{-1}S)$, $\beta = 10^{-4}$ (b) $\lambda_i(\widehat{S}_1^{-1}S)$, $\beta = 10^{-7}$

Figure 3.2: Illustrations of the intermediate eigenvalues of $\widehat{S}_1^{-1}S$ for $\beta = 10^{-4}$ and $\beta = 10^{-7}$, for an evenly spaced grid on $\Omega = [0,1]^2$ with $h = 2^{-4}$.

for the reduced 2×2 matrix system. It is demonstrated that this yields an effective solver when used with the MINRES algorithm. This preconditioner is closely related to the one introduced in Section 3.1.3.

The approximation $\widehat{S}_2$ to the Schur complement we have stated here can be used in effective preconditioners for the three solution approaches we have discussed in Section 3.1. The use of a fixed number of Chebyshev semi-iteration cycles to approximate mass matrices, along with our approximation $\widehat{S}_2$ to S, can be used as part of a block diagonal, block triangular or symmetric indefinite preconditioner, as in Sections 3.1.1, 3.1.2 and 3.1.3 respectively. Analysis of how well the approximations detailed perform in practice is given in Section 3.4. In all three cases, the bound (3.10) ensures that the convergence rate of the iteration will be independent of β; independence with respect to h is ensured if a spectrally equivalent[9] approximation of $K + \frac{1}{\sqrt{\beta}}M$ such as a multigrid process is used. The availability of such a spectrally equivalent approximation which can be applied with $\mathcal{O}(\bar{N})$ work (where $\bar{N}$ here denotes the dimension of the matrices K and M) is crucial here: our preconditioned iterative solvers will only have optimal complexity with such an approximation. In our computations we use the HSL algebraic multigrid code `HSL_MI20` [19] via a MATLAB interface, which as expected is seen to satisfy these conditions. We highlight once more that only a fixed number of multigrid cycles are applied, as opposed to running multigrid until convergence.

[9]We say that two matrices $A_1(h)$ and $A_2(h)$ are *spectrally equivalent* if the eigenvalues of $A_2^{-1}A_1$ are bounded within a constant (positive) interval, whatever the value of h is.

This new Schur complement approximation thus gives rise to three new methods for solving the Poisson control problem. Firstly, we can apply MINRES with the preconditioner (3.2), but now using $\widehat{S}_2$ to approximate the Schur complement instead of $\widehat{S}_1$ as in (3.3). Secondly, we may apply Bramble-Pasciak CG preconditioned by (3.4) (using an appropriate choice of γ, which is typically very close to 1), but again replacing the Schur complement approximation $\widehat{S}_1$ with $\widehat{S}_2$. Finally, we may apply a non-standard CG method with a symmetric indefinite preconditioner of the form

$$\widehat{\mathcal{P}}_4 = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ \sigma K\widehat{M}^{-1} & -\frac{\sigma}{\beta}M\widehat{M}^{-1} & I \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma}\widehat{M} & 0 & K \\ 0 & \frac{\beta}{\sigma}\widehat{M} & -M \\ 0 & 0 & -\frac{1}{\bar{\tau}}\widehat{S}_2 \end{bmatrix}, \tag{3.12}$$

with $\widehat{M}$ again an application of Chebyshev semi-iteration to the relevant mass matrix. The bound (3.10) guarantees that we may choose appropriate values σ and $\bar{\tau}$ using the same reasoning with which we choose γ for the Bramble-Pasciak CG method, taking into account Chebyshev semi-iteration results described in Section 2.3.2, as well as [101]. We test our three new methods in Section 3.4.

3.3 Eigenvalues of the Preconditioned Matrix System

We now present results concerning the eigenvalues of $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$ and $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$ when our new Schur complement approximation $\widehat{S}_2$ is used. These results demonstrate that, in an idealized setting (i.e. if the mass matrix M and the matrix $K + \frac{1}{\sqrt{\beta}}M$ are inverted exactly), then the bounds for the eigenvalues of the preconditioned Schur complement also guarantee that robust bounds for the entire matrix system can be obtained.

In order to consider the eigenvalues of $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$, relating to our block diagonal preconditioner, we make use of the following result, which is shown in [100, Proposition 3.2] when considering eigenvalues for the preconditioned system with Schur complement approximation $\widehat{S}_1$.

Theorem 9. *Any eigenvalue λ of the matrix*

$$\begin{bmatrix} I & \widetilde{B}^T \\ \widetilde{B} & 0 \end{bmatrix},$$

where $\widetilde{B}$ has full row-rank, satisfies one of the following:

$$\begin{gathered}\lambda = 1, \\ \frac{1+\sqrt{1+4\sigma_{\min}}}{2} \leq \lambda \leq \frac{1+\sqrt{1+4\sigma_{\max}}}{2}, \\ \frac{1-\sqrt{1+4\sigma_{\max}}}{2} \leq \lambda \leq \frac{1-\sqrt{1+4\sigma_{\min}}}{2}.\end{gathered}$$

Here $\sigma_{\min}$ and $\sigma_{\max}$ denote the minimum and maximum eigenvalues of $\widetilde{B}\widetilde{B}^T$ respectively.

We may use Theorem 9 to prove the following result on the eigenvalues of $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$.

Theorem 10. *The eigenvalues of $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$, where*

$$\mathcal{A} = \begin{bmatrix} M & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix}, \quad \widehat{\mathcal{P}}_1 = \begin{bmatrix} M & 0 & 0 \\ 0 & \beta M & 0 \\ 0 & 0 & \widehat{S}_2 \end{bmatrix},$$

are all real, and are contained within the following intervals:

$$\begin{aligned}\lambda(\widehat{\mathcal{P}}_1^{-1}\mathcal{A}) \in & \left[\frac{1-\sqrt{5}}{2}, \frac{1-\sqrt{3}}{2}\right] \cup \{1\} \cup \left[\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{5}}{2}\right] \\ & \approx [-0.618, -0.366] \cup \{1\} \cup [1.366, 1.618].\end{aligned}$$

Proof. We follow the approach used in [100] to prove an analogous result when the Schur complement is approximated by $\widehat{S}_1$ instead.[10] There, it is noted that the eigenvalues of $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$ are the same as those of $\widehat{\mathcal{P}}_1^{-1/2}\mathcal{A}\widehat{\mathcal{P}}_1^{-1/2}$ by similarity transformation.

[10] The equivalent result when the Schur complement is approximated by $\widehat{S}_1$ is given by [100, Corollary 3.3]

$$\begin{aligned}\lambda(\widehat{\mathcal{P}}_1^{-1}\mathcal{A}) \in & \left[\frac{1}{2}\left(1-\sqrt{5+\frac{4\widetilde{C}}{\beta}}\right), \frac{1}{2}\left(1-\sqrt{5+\frac{4\widetilde{c}h^4}{\beta}}\right)\right] \cup \{1\} \\ & \cup \left[\frac{1}{2}\left(1+\sqrt{5+\frac{4\widetilde{c}h^4}{\beta}}\right), \frac{1}{2}\left(1+\sqrt{5+\frac{4\widetilde{C}}{\beta}}\right)\right].\end{aligned}$$

Expanding out the matrix $\widehat{\mathcal{P}}_1^{-1/2}\mathcal{A}\widehat{\mathcal{P}}_1^{-1/2}$ gives

$$\begin{aligned}
\widehat{\mathcal{P}}_1^{-1/2}\mathcal{A}\widehat{\mathcal{P}}_1^{-1/2} &= \begin{bmatrix} M^{-1/2} & 0 & 0 \\ 0 & \frac{1}{\sqrt{\beta}}M^{-1/2} & 0 \\ 0 & 0 & \widehat{S}_2^{-1/2} \end{bmatrix} \mathcal{A} \begin{bmatrix} M^{-1/2} & 0 & 0 \\ 0 & \frac{1}{\sqrt{\beta}}M^{-1/2} & 0 \\ 0 & 0 & \widehat{S}_2^{-1/2} \end{bmatrix} \\
&= \begin{bmatrix} M^{-1/2} & 0 & 0 \\ 0 & \frac{1}{\sqrt{\beta}}M^{-1/2} & 0 \\ 0 & 0 & \widehat{S}_2^{-1/2} \end{bmatrix} \begin{bmatrix} M^{1/2} & 0 & K\widehat{S}_2^{-1/2} \\ 0 & \sqrt{\beta}M^{1/2} & -M\widehat{S}_2^{-1/2} \\ KM^{-1/2} & -\frac{1}{\sqrt{\beta}}M^{1/2} & 0 \end{bmatrix} \\
&= \begin{bmatrix} I & 0 & M^{-1/2}K\widehat{S}_2^{-1/2} \\ 0 & I & -\frac{1}{\sqrt{\beta}}M^{1/2}\widehat{S}_2^{-1/2} \\ \widehat{S}_2^{-1/2}KM^{-1/2} & -\frac{1}{\sqrt{\beta}}\widehat{S}_2^{-1/2}M^{1/2} & 0 \end{bmatrix} =: \begin{bmatrix} I & \widetilde{B}^T \\ \widetilde{B} & 0 \end{bmatrix}.
\end{aligned}$$

We are interested in the eigenvalues of this matrix. Theorem 9 tells us that we need to consider the minimum and maximum eigenvalues $\sigma_{\min}$ and $\sigma_{\max}$ of $\widetilde{B}\widetilde{B}^T$. Now,

$$\begin{aligned}
\widetilde{B}\widetilde{B}^T &= \begin{bmatrix} \widehat{S}_2^{-1/2}KM^{-1/2} & -\frac{1}{\sqrt{\beta}}\widehat{S}_2^{-1/2}M^{1/2} \end{bmatrix} \begin{bmatrix} M^{-1/2}K\widehat{S}_2^{-1/2} \\ -\frac{1}{\sqrt{\beta}}M^{1/2}\widehat{S}_2^{-1/2} \end{bmatrix} \\
&= \widehat{S}_2^{-1/2}KM^{-1}K\widehat{S}_2^{-1/2} + \frac{1}{\beta}\widehat{S}_2^{-1/2}M\widehat{S}_2^{-1/2} \\
&= \widehat{S}_2^{-1/2}\left(KM^{-1}K + \frac{1}{\beta}M\right)\widehat{S}_2^{-1/2},
\end{aligned}$$

so the eigenvalues of $\widetilde{B}\widetilde{B}^T$ are the same as those of

$$\widehat{S}_2^{-1}\left(KM^{-1}K + \frac{1}{\beta}M\right) = \widehat{S}_2^{-1}S,$$

again using a similarity transformation. Hence $\sigma_{\min} \geq \frac{1}{2}$ and $\sigma_{\max} \leq 1$ in the notation of Theorem 9, and therefore, using this theorem, we have that

$$\lambda(\widehat{\mathcal{P}}_1^{-1}\mathcal{A}) \in \left[\frac{1-\sqrt{5}}{2}, \frac{1-\sqrt{3}}{2}\right] \cup \{1\} \cup \left[\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{5}}{2}\right],$$

as claimed. □

Let us now consider the derivation of eigenvalue bounds for $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$. In order to derive such bounds, we exploit the following result, which was proved in [101, Theorem 3.1] where the authors considered eigenvalues of the preconditioned system when the

Schur complement approximation $\widehat{S}_1$ was used.

Theorem 11. *Consider a general saddle point system of the form* (2.31)*, with* $\Theta = 0$*. Then if*

$$1 < \widetilde{\theta} \leq \frac{\mathbf{v}^T \Phi \mathbf{v}}{\mathbf{v}^T \widehat{\Phi} \mathbf{v}} \leq \widetilde{\Theta}, \qquad 0 < \widetilde{\omega} \leq \frac{\mathbf{w}^T \Psi \Phi^{-1} \Psi^T \mathbf{w}}{\mathbf{w}^T \widehat{S} \mathbf{w}} \leq \widetilde{\Omega},$$

for positive definite approximations $\widehat{\Phi}$ *and* $\widehat{S}$*, and for all non-zero vectors* $\mathbf{v}$ *and* $\mathbf{w}$ *of the appropriate dimension, then the eigenvalues* λ *of the matrix*

$$\begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -\widehat{S} \end{bmatrix}^{-1} \begin{bmatrix} \Phi & \Psi^T \\ \Psi & 0 \end{bmatrix}$$

are contained within one of the following intervals:

$$\begin{aligned} \widetilde{\theta} \leq \lambda \leq \widetilde{\Theta}, \\ \frac{(1+\widetilde{\omega})\widetilde{\Theta} - \sqrt{(1+\widetilde{\omega})^2\widetilde{\Theta}^2 - 4\widetilde{\omega}\widetilde{\Theta}}}{2} \leq \lambda \leq \frac{(1+\widetilde{\Omega})\widetilde{\theta} - \sqrt{(1+\widetilde{\Omega})^2\widetilde{\theta}^2 - 4\widetilde{\Omega}\widetilde{\theta}}}{2}, \\ \frac{(1+\widetilde{\omega})\widetilde{\theta} + \sqrt{(1+\widetilde{\omega})^2\widetilde{\theta}^2 - 4\widetilde{\omega}\widetilde{\theta}}}{2} \leq \lambda \leq \frac{(1+\widetilde{\Omega})\widetilde{\Theta} + \sqrt{(1+\widetilde{\Omega})^2\widetilde{\Theta}^2 - 4\widetilde{\Omega}\widetilde{\Theta}}}{2}. \end{aligned}$$

We note that the condition $1 < \widetilde{\theta}$ in Theorem 11 is applied above to ensure positivity in the non-standard inner product being used for the Conjugate Gradient algorithm.

Theorem 11 leads to the following result on the eigenvalues of $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$.

Theorem 12. *The eigenvalues of* $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$*, where*

$$\mathcal{A} = \begin{bmatrix} M & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix}, \quad \widehat{\mathcal{P}}_2 = \begin{bmatrix} \gamma M & 0 & 0 \\ 0 & \beta\gamma M & 0 \\ K & -M & -\widehat{S}_2 \end{bmatrix},$$

with $\gamma < 1$*, are all real, and are contained within the following intervals:*

$$\lambda(\widehat{\mathcal{P}}_2^{-1}\mathcal{A}) \in \left[\frac{3-\sqrt{9-8\gamma}}{4\gamma}, \frac{1-\sqrt{1-\gamma}}{\gamma}\right] \cup \left\{\frac{1}{\gamma}\right\} \cup \left[\frac{3+\sqrt{9-8\gamma}}{4\gamma}, \frac{1+\sqrt{1-\gamma}}{\gamma}\right].$$

For instance, if $\gamma = 0.95$*, the intervals of the eigenvalues of* $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$ *are given approx-*

imately by

$$\lambda(\widehat{\mathcal{P}}_2^{-1}\mathcal{A}) \in [0.478, 0.817] \cup \{1.053\} \cup [1.101, 1.288].$$

Proof. We follow the approach of Rees and Stoll in [101], where an analogous result is sought when the Schur complement is approximated by $\widehat{S}_1$ instead,[11] and make use of Theorem 11. It is therefore necessary to ascertain values of $\widetilde{\theta}$, $\widetilde{\Theta}$, $\widetilde{\omega}$ and $\widetilde{\Omega}$, in the notation of the theorem. By construction of the approximation $\widehat{\Phi}$ to the $(1,1)$-block Φ, it is clear that we may write $\widetilde{\theta} = \widetilde{\Theta} = \frac{1}{\gamma}$. We also write $\widetilde{\omega} = \frac{1}{2}$ and $\widetilde{\Omega} = 1$, using the eigenvalue bounds of $\widehat{S}_2^{-1}S$ proved in Theorem 8. Therefore, inserting these values into the intervals of Theorem 11, we obtain that

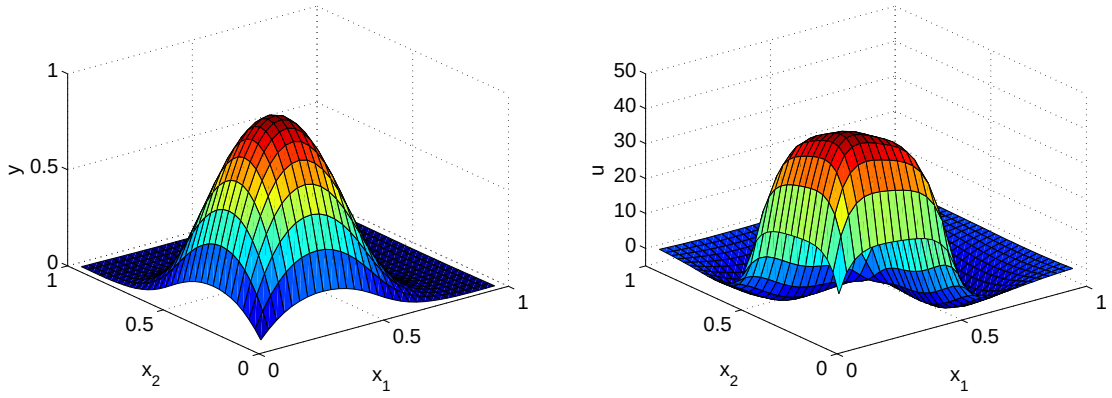
$$\lambda(\widehat{\mathcal{P}}_2^{-1}\mathcal{A}) \in \left[\frac{3-\sqrt{9-8\gamma}}{4\gamma}, \frac{1-\sqrt{1-\gamma}}{\gamma}\right] \cup \left\{\frac{1}{\gamma}\right\} \cup \left[\frac{3+\sqrt{9-8\gamma}}{4\gamma}, \frac{1+\sqrt{1-\gamma}}{\gamma}\right],$$

as claimed. □

These above results indicate that our Schur complement approximation, along with effective procedures to apply this approximation as well as the approximate inverses of mass matrices, will ensure that the condition numbers of the preconditioned matrix systems $\widehat{\mathcal{P}}_1^{-1}\mathcal{A}$ and $\widehat{\mathcal{P}}_2^{-1}\mathcal{A}$ are small. When such preconditioners are used within iterative methods such as MINRES and Conjugate Gradients, this property guarantees fast convergence of the methods [36, Chapters 2 and 6]. This therefore provides a further argument to advocate our proposed preconditioners.

[11] The equivalent result when the Schur complement is approximated by $\widehat{S}_1$ is given by (using results from [99, 101])

$$\begin{aligned}
\lambda(\widehat{\mathcal{P}}_2^{-1}\mathcal{A}) &\in \left\{\frac{1}{\gamma}\right\} \\
&\cup \left[\frac{1}{2\gamma}\left(2+\frac{\widetilde{c}h^4}{\beta}-\sqrt{\left(2+\frac{\widetilde{c}h^4}{\beta}\right)^2-4\gamma\left(1+\frac{\widetilde{c}h^4}{\beta}\right)}\right), \frac{1}{2\gamma}\left(2+\frac{\widetilde{C}}{\beta}-\sqrt{\left(2+\frac{\widetilde{C}}{\beta}\right)^2-4\gamma\left(1+\frac{\widetilde{C}}{\beta}\right)}\right)\right] \\
&\cup \left[\frac{1}{2\gamma}\left(2+\frac{\widetilde{c}h^4}{\beta}+\sqrt{\left(2+\frac{\widetilde{c}h^4}{\beta}\right)^2-4\gamma\left(1+\frac{\widetilde{c}h^4}{\beta}\right)}\right), \frac{1}{2\gamma}\left(2+\frac{\widetilde{C}}{\beta}+\sqrt{\left(2+\frac{\widetilde{C}}{\beta}\right)^2-4\gamma\left(1+\frac{\widetilde{C}}{\beta}\right)}\right)\right].
\end{aligned}$$



(a) State y (b) Control u

Figure 3.3: State and control solution to the numerical example in 2D, within $\Omega \backslash \partial\Omega$ with axes x_1 and x_2, with $h = 2^{-5}$ and $\beta = 10^{-4}$.

3.4 Numerical Results

We have demonstrated the theoretical capability of our new Schur complement approximation by the eigenvalue results of Section 3.2 and 3.3. To illustrate the practical effectiveness of the preconditioners we have derived, we now test them with the block diagonal, block triangular and symmetric indefinite preconditioners proposed in [100], [101] and [109] respectively, and described in Section 3.1. The test problem we use is given by

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2} \left\| y - \widehat{y} \right\|^2_{L_2(\Omega)} + \frac{\beta}{2} \left\| u \right\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = u \quad \text{in } \Omega, \\ & y = 0 \quad \text{on } \partial\Omega, \end{aligned}$$

where $\Omega = [0,1]^2$, $\partial\Omega$ denotes its boundary, and $\widehat{y}$ is given by

$$\widehat{y} = \begin{cases} 1 & \text{in } [0, \frac{1}{2}]^2 =: \Omega_1, \\ 0 & \text{in } \Omega \backslash \Omega_1. \end{cases}$$

The solution for the state and control of this problem when $\beta = 10^{-4}$ is shown in Figure 3.3.

In Table 3.1, we compare the number of MINRES iterations required and CPU time (in seconds) taken to solve this problem to a tolerance of 10^{-6} using the block

ITERATIONS		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	9	21	73	248	13	16	15	(12)*
	2^{-5}	9	20	82	484	13	17	16	15
	2^{-6}	9	22	85	592	13	17	16	16
	2^{-7}	11	22	84	619	13	17	16	16
	2^{-8}	11	21	85	646	15	17	17	16

TIME (s)		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	0.0209	0.0451	0.151	0.508	0.0290	0.0354	0.0398	(0.0359)*
	2^{-5}	0.0660	0.105	0.409	2.41	0.0693	0.0902	0.0864	0.121
	2^{-6}	0.198	0.461	1.67	11.5	0.272	0.351	0.342	0.554
	2^{-7}	0.901	1.89	6.53	47.1	1.05	1.35	1.27	1.33
	2^{-8}	4.56	8.40	33.5	241	6.29	7.01	6.81	6.61

Table 3.1: Number of MINRES iterations and time taken with block diagonal preconditioner (3.2) to solve the test problem in 2D, using **Q1** basis functions for state and control, for a variety of h and β. Results are given when the Schur complement is approximated by $\widehat{S}_1$ and $\widehat{S}_2$.

diagonal preconditioner (3.2) for a range of h and β, when the Schur complement is approximated by $\widehat{S}_1$ and $\widehat{S}_2$.[12]

To demonstrate that our method is equally applicable in three dimensions, in Table 3.2 we present results for solving the problem above, except with $\Omega = [0,1]^3$ and $\Omega_1 = [0, \frac{1}{2}]^3$, using MINRES with block diagonal preconditioner (3.2). In Table 3.3, we show the number of Bramble-Pasciak Conjugate Gradient iterations required and CPU time taken to solve the test problem using the block triangular preconditioner (3.4), approximating S by $\widehat{S}_1$ and $\widehat{S}_2$, with $\gamma = 0.95$.[13] Finally in Table 3.4, we compare the number of Conjugate Gradient iterations and CPU time taken with $\widehat{\Phi}$ and $\widehat{S}$ defined by (3.5), (3.6) as described in [109], with the number of iterations required

[12] All results in Tables 3.1–3.4 were generated on a tri-core 2.5 GHz workstation.

[13] We wish to choose γ reasonably close to 1 in order for the preconditioner for the $(1,1)$-block to be effective, but also far enough away from 1 to ensure that the inner product we work with is clearly positive definite. We found that the value $\gamma = 0.95$ met these criteria in practice. Similar issues are discussed in [101] for instance.

ITERATIONS		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-1}	10^{-3}	10^{-5}	10^{-7}	10^{-1}	10^{-3}	10^{-5}	10^{-7}
h	2^{-2}	8	12	26	28	10	14	(12)*	(8)*
	2^{-3}	8	12	42	130	10	16	14	(12)*
	2^{-4}	8	12	48	272	12	17	15	13
	2^{-5}	10	14	49	341	12	18	16	16

TIME (s)		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-1}	10^{-3}	10^{-5}	10^{-7}	10^{-1}	10^{-3}	10^{-5}	10^{-7}
h	2^{-2}	0.0119	0.0169	0.0344	0.0370	0.0147	0.0200	(0.0178)*	(0.0951)*
	2^{-3}	0.0417	0.0598	0.198	0.601	0.0536	0.0826	0.0748	(0.0883)*
	2^{-4}	0.568	0.735	2.76	15.4	0.793	1.10	1.10	0.937
	2^{-5}	9.21	12.6	41.4	266	11.6	18.0	14.2	17.7

Table 3.2: Number of MINRES iterations and time taken with block diagonal preconditioner (3.2) to solve the test problem in 3D, using **Q1** basis functions for state and control, for a variety of h and β. Results are given when the Schur complement is approximated by $\widehat{S}_1$ and $\widehat{S}_2$.

and CPU time taken when using the preconditioner (3.12) with Schur complement approximation $\widehat{S}_2$ (except that we use Chebyshev semi-iterations to approximate a mass matrix rather than Gauss-Seidel iteration as in [109]; this equally improves both approaches).

On each occasion, we use 20 Chebyshev semi-iterations to approximate a mass matrix, and 2 algebraic multigrid V-cycles with 2 pre- and post- (relaxed Jacobi) smoothing steps whenever a multigrid solve is required. We employ the HSL code `HSL_MI20` [19] via a MATLAB interface for the (algebraic) multigrid cycles. In Tables 3.1–3.4, * denotes that the coarsening failed when the code `HSL_MI20` was applied to $K + \frac{1}{\sqrt{\beta}}M$, which occurs only when h is large and β is very small. This clearly occurs in relatively few cases, and is caused by the presence of positive off-diagonal entries. The use of large h and small β is not an interesting practical parameter regime, but for completeness we apply a sparse direct solve for this regime and include the results in brackets in Tables 3.1–3.4, next to the * symbol. We note that an application of the Chebyshev semi-iteration method would also be an effective approach for this regime.

Iterations		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	8	15	74	480	11	14	15	(15)*
	2^{-5}	8	16	84	864	11	15	16	17
	2^{-6}	9	18	84	975	11	15	17	18
	2^{-7}	9	19	96	1061	13	15	17	18
	2^{-8}	10	19	104	1257	13	17	17	19

Time (s)		Approximation $\widehat{S}_1$				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	0.0200	0.0355	0.165	1.06	0.0267	0.0336	0.0418	(0.0405)*
	2^{-5}	0.0479	0.0913	1.45	4.58	0.0640	0.0852	0.0938	0.142
	2^{-6}	0.209	0.392	1.82	20.9	0.250	0.332	0.376	0.645
	2^{-7}	0.828	1.66	8.16	85.5	1.14	1.30	1.45	1.59
	2^{-8}	4.58	8.21	43.4	523	5.82	7.50	7.54	8.35

Table 3.3: Number of Bramble-Pasciak CG iterations and time taken with block triangular preconditioner (3.4) to solve the test problem in 2D, using **Q1** basis functions for state and control, for a variety of h and β. Results are given when the Schur complement is approximated by $\widehat{S}_1$ and $\widehat{S}_2$.

Tables 3.1–3.4 clearly illustrate the potency of our new Schur complement approximation; h and β-independent convergence is exhibited in all cases. In Tables 3.1–3.3, we can see that for larger values of β, the approximation of the Schur complement $\widehat{S}_1$ is marginally more effective than the approximation $\widehat{S}_2$, as the eigenvalues of $\widehat{S}_1^{-1}S$ are more clustered than those of $\widehat{S}_2^{-1}S$ (see Figure 3.1). However, as β gets smaller, it is easily observable that, as the eigenvalues of $\widehat{S}_1^{-1}S$ become less clustered but those of $\widehat{S}_2^{-1}S$ remain in the interval $\left[\frac{1}{2}, 1\right]$, our new Schur complement approximation performs far better. Table 3.4 also illustrates that our approximations of mass matrices and Schur complement leave us with a preconditioner that is at least competitive with the symmetric indefinite preconditioner proposed in [109].

ITERATIONS		Approximation as in [109]				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	11	14	13	(13)*	7	11	13	(13)*
	2^{-5}	11	14	16	14	8	11	13	14
	2^{-6}	11	15	17	16	8	12	13	15
	2^{-7}	12	16	18	19	8	12	14	15
	2^{-8}	12	16	18	20	9	13	14	16

TIME (S)		Approximation as in [109]				Approximation $\widehat{S}_2$			
		β				β			
		10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
h	2^{-4}	0.0566	0.0721	0.0802	(0.0991)*	0.0479	0.0714	0.0886	(0.103)*
	2^{-5}	0.138	0.173	0.199	0.264	0.120	0.161	0.188	0.236
	2^{-6}	0.535	0.718	0.820	1.33	0.471	0.678	0.734	1.05
	2^{-7}	2.32	3.00	3.32	3.64	1.73	2.51	2.90	3.13
	2^{-8}	12.0	16.0	17.8	19.8	10.1	13.8	14.7	16.8

Table 3.4: Number of CG iterations and time taken with symmetric indefinite preconditioner (3.7) to solve the test problem in 2D, using **Q1** basis functions for state and control, for a variety of h and β. Results are given with $\widehat{\Phi}$ and $\widehat{S}$ as stated in (3.5), (3.6) and detailed in [109], and using the preconditioner (3.12) with $\widehat{S}$ approximated by $\widehat{S}_2$.

3.5 Summary

In this chapter, we wished to consider how we might apply the saddle point theory introduced in Section 2.2 to one of the most fundamental optimal control problems: a distributed control problem with Poisson's equation as a constraint. We first reviewed three methods previously introduced for this problem: a MINRES approach with a block diagonal preconditioner, a Bramble-Pasciak CG solver with a block triangular preconditioner, and a CG method with a symmetric indefinite preconditioner. All of these solvers exhibited independence with respect to the mesh-size h taken, but only the last one was also independent with respect to the regularization parameter β.

We then introduced a new approximation of the Schur complement of the relevant matrix system, and proved analytically that it is spectrally equivalent to the exact Schur complement. We demonstrated how we may build this new approximation into

the three solvers previously discussed, with the goal that all three methods exhibit mesh and β-independence. We were able to show analytically and numerically that our suggested preconditioners did indeed give rapid parameter-independent convergence when applied within each of the three methods described, used in conjunction with a Chebyshev semi-iteration method to approximate the relevant mass matrices and a multigrid process to apply the Schur complement approximation.

We will find that the new Schur complement approximation introduced in this chapter is an important one. This is not only because of the improved results generated for the distributed Poisson control problem, but also because we will be able to use similar strategies to solve many other optimal control problems in future chapters.

CHAPTER 4

Boundary and Subdomain Control

In this chapter, we wish to discuss a number of problems related to the distributed Poisson control problem detailed in the previous chapter.[14] Firstly, we consider the Neumann boundary control problem governed by Poisson's equation, as stated in (2.10), where the control variable is applied in the form of a Neumann boundary condition. Secondly, we tackle the subdomain control problem (2.16), which is a distributed control problem, but with the control only applied in some subdomain rather than the entire domain. We also briefly detail how we may approach distributed control problems with additional inequality constraints imposed on the state variable.

These problems are to some extent more physically relevant than the distributed Poisson control problem discussed in the previous section. It seems natural to restrict the portion of a domain on which control may be applied, be that the boundary of the domain or some interior subdomain. It also seems a pertinent question to examine the effect of placing bound constraints on the solution of the state variable. This motivates

[14] Section 4.1 is partially based on Sections 3.4–3.5 of Ref. [91], which is the following published paper:

J. W. Pearson, M. Stoll and A. J. Wathen, *Regularization-Robust Preconditioners for Time-Dependent PDE-Constrained Optimization Problems*, SIAM Journal on Matrix Analysis and Applications, 33(4), pp.1126–1152, 2012.

Additionally, Section 4.3 is a short summary of Ref. [90], which is

J. W. Pearson, M. Stoll and A. J. Wathen, *Preconditioners for State Constrained Optimal Control Problems with Moreau-Yosida Penalty Function*, to appear in Numerical Linear Algebra with Applications, available online, DOI: 10.1002/nla.1863, 2012.

our investigations of such problems. Our goal when carrying out these investigations is to discover whether robust solvers may be developed for these problems, as they were for the distributed Poisson control problem in the previous chapter.

We wish to follow similar strategies as for the distributed Poisson control problem, and investigate to what extent our methods are effective for the problems examined in this chapter. For the most part we will seek block diagonal preconditioners, though we are not limited to considering them. As for the work of the previous chapter, one may consider either block diagonal or block triangular preconditioners for each problem detailed in this chapter.

4.1 Neumann Boundary Control Problem

Let us first consider the solution of Neumann boundary control problems. We examine the following problem formulation:

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\partial\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = f, \quad \text{in } \Omega, \\ & \frac{\partial y}{\partial n} = u, \quad \text{on } \partial\Omega, \end{aligned}$$

discussed in Section 2.1.2. As detailed there, applying a finite element method with equal-order basis functions to this problem leads to the matrix system

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M_b & -N_b^T \\ K & -N_b & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{f} \end{bmatrix},$$

with M and K being mass and stiffness matrices for the entire domain, and M_b, N_b representing the boundary matrices defined by (2.14), (2.13). It is possible to eliminate the second block of this matrix system (corresponding to the gradient equation) leaving the system

$$\begin{bmatrix} M & K \\ K & -\frac{1}{\beta}M_\Gamma \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{f} \end{bmatrix}. \tag{4.1}$$

Here, $M_\Gamma := N_b M_b^{-1} N_b^T$ contains entries of M_b at indices corresponding to nodes on $\partial\Omega$, and zeros elsewhere. To put it another way, if the nodes are ordered such that all the interior nodes are listed first followed by the boundary nodes, then $M_\Gamma = \text{blkdiag}\,(0, M_b)$. We will seek to solve matrix systems of the form (4.1) in this section.

We may write this system in the saddle point form of (2.31), with

$$\Phi = M, \quad \Psi = K, \quad \Theta = \frac{1}{\beta} M_\Gamma.$$

Therefore, we are faced with two challenges when constructing a block diagonal preconditioner of the form $\text{blkdiag}\left(\widehat{\Phi}, \widehat{S}\right)$. The first is to accurately approximate the $(1,1)$-block Φ – we are able to do this, as in the previous chapter, by applying the Chebyshev semi-iteration method of Section 2.3.2. The second task, on which we focus our attention now, is to approximate the Schur complement of the matrix system

$$S = K M^{-1} K + \frac{1}{\beta} M_\Gamma.$$

4.1.1 Approximating the Schur Complement

The Schur complement for this problem has a different structure to that of the distributed Poisson control problem. In particular, the rank-deficiency of the second term of the Schur complement, as well as the non-invertibility of the matrix K (as it corresponds to a Neumann problem), create complications when we seek to develop an effective Schur complement approximation.

Another aspect of this problem that makes its iterative solution more difficult in comparison to the distributed control problem is the presence of two different mass matrices – M and M_Γ. When analysing these matrices, we find it is therefore often useful to consider them in lumped form (though we implement mass matrices in consistent form for all numerical results in this chapter). In this case, M is a diagonal matrix that is spectrally equivalent to $h^d I$ (where d is the dimension of the domain in which the problem is being solved), and the non-zero diagonal part of M_Γ (corresponding to boundary terms) is spectrally equivalent to $h^{d-1} I_b$, where I_b denotes the identity matrix of dimension equal to the number of boundary nodes. These observations enable us to make the following approximations to aid us when carrying out analysis for this problem:

$$M_\Gamma M^{-1} \approx \text{blkdiag}\left(0, h^{-1} I_b\right), \quad M_\Gamma \approx h M_\Gamma M^{-1} M_\Gamma. \tag{4.2}$$

We now propose two options for approximating S, both of which are designed to be robust with respect to both mesh-size and regularization parameter.[15] Firstly, we may approximate S by

$$\widehat{S}_1 = \left(K + \frac{1}{\sqrt{\beta}}\widehat{M}\right) M^{-1} \left(K + \frac{1}{\sqrt{\beta}}\widehat{M}\right),$$

where $\widehat{M}$ is a matrix chosen such that we maintain the structure of the exact Schur complement in our approximation. Examining in more detail the contributions to $\widehat{S}_1$, we obtain

$$\widehat{S}_1 = KM^{-1}K + \frac{1}{\beta}\widehat{M}M^{-1}\widehat{M} + \frac{1}{\sqrt{\beta}}\left(KM^{-1}\widehat{M} + \widehat{M}M^{-1}K\right).$$

We now wish to choose $\widehat{M}$ so that the second term in the above expansion of $\widehat{S}_1$ approximates the second term of S, i.e.

$$\frac{1}{\beta}\widehat{M}M^{-1}\widehat{M} \approx \frac{1}{\beta}M_\Gamma. \tag{4.3}$$

This can be described as a "matching" strategy, in which we seek to capture the nature of both terms of the exact Schur complement in our approximation. This is based on the discussion above concerning the spectral equivalence of various mass matrices. Note once more that the eigenvalues of M are all of $\mathcal{O}(h^d)$, and, since it corresponds to boundary terms, M_Γ contains eigenvalues of at most $\mathcal{O}(h^{d-1})$ – the matrix M_Γ also has many zero eigenvalues due to its rank-deficiency. Therefore, if we make the choice

$$\widehat{M} = \sqrt{h}M_\Gamma,$$

then we find that the largest eigenvalues of both sides of (4.3) should balance, i.e.

$$\beta^{-1} \cdot h^{1/2}h^{d-1} \cdot h^{-d} \cdot h^{1/2}h^{d-1} = \beta^{-1} \cdot h^{d-1}.$$

[15] We note the change in notation from Chapter 3 – before, $\widehat{S}_1$ denoted an approximation that was not robust with respect to β, whereas in this section, both $\widehat{S}_1$ and $\widehat{S}_2$ represent preconditioners which are aimed to be parameter-robust.

This motivates our choice of Schur complement approximation

$$\widehat{S}_1 = \left(K + \sqrt{\frac{h}{\beta}}M_\Gamma\right) M^{-1} \left(K + \sqrt{\frac{h}{\beta}}M_\Gamma\right).$$

We may motivate very similarly a second choice of approximation

$$\widehat{S}_2 = \left(K + \sqrt{\frac{h}{\beta}}M_\Gamma\right) \left(h\widehat{M}_\Gamma\right)^{-1} \left(K + \sqrt{\frac{h}{\beta}}M_\Gamma\right),$$

where $\widehat{M}_\Gamma$ denotes a matrix which contains exactly the entries of M_Γ at boundary nodes, and entries of $\mathcal{O}(h^{d-1})$ on the diagonals corresponding to interior nodes.[16]

Eigenvalues of $\widehat{S}_1^{-1}S$. We now wish to establish bounds for the preconditioned Schur complement when our first approximation is used. We will carry out analysis for the 2D case, though the case for 3D problems is very similar and the same results can be obtained. We find that, whereas we cannot pin the eigenvalues into a clear interval as in the previous chapter, we may argue that they should be contained within bounds of $\mathcal{O}(1)$.

To do this, we consider the Rayleigh quotient

$$\begin{aligned}\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_1\mathbf{v}} &= \frac{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_\Gamma\mathbf{v}}{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{h}{\beta}\mathbf{v}^T M_\Gamma M^{-1}M_\Gamma\mathbf{v} + \sqrt{\frac{h}{\beta}}\mathbf{v}^T\left[KM^{-1}M_\Gamma + M_\Gamma M^{-1}K\right]\mathbf{v}} \\ &= \frac{\mathbf{v}^T KM^{-1}K\mathbf{v} + \mathbf{v}^T\left(\frac{1}{\beta}M_\Gamma\right)\mathbf{v}}{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{h}{\beta}\mathbf{v}^T M_\Gamma M^{-1}M_\Gamma\mathbf{v} + 2\sqrt{\frac{h}{\beta}}\mathbf{v}^T M_\Gamma M^{-1}K\mathbf{v}},\end{aligned}$$

which will give us a range for the eigenvalues of $\widehat{S}_1^{-1}S$.

If $\mathbf{v} \in \text{null}(M_\Gamma)$, then $\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_1\mathbf{v}} = 1$. If this is not the case, we may write the above expression as

$$\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_1\mathbf{v}} = \frac{1}{\frac{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{h}{\beta}\mathbf{v}^T M_\Gamma M^{-1}M_\Gamma\mathbf{v}}{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_\Gamma\mathbf{v}} + \frac{2\sqrt{\frac{h}{\beta}}\mathbf{v}^T M_\Gamma M^{-1}K\mathbf{v}}{\mathbf{v}^T KM^{-1}K\mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_\Gamma\mathbf{v}}}. \tag{4.4}$$

[16] In fact in our implementation, we set the diagonal entries corresponding to the interior nodes to be the same as one of the diagonal entries of the boundary nodes – this way the relevant entries are of the same order.

We now use the fact that the non-zero parts of $M_\Gamma \left(\frac{1}{h}M\right)^{-1} M_\Gamma = hM_\Gamma M^{-1} M_\Gamma$ and M_Γ are spectrally equivalent (which we justified in (4.2), and used to motivate our choice for $\widehat{M}$ above),[17] to observe that

$$0 < \frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{h}{\beta}\mathbf{v}^T M_\Gamma M^{-1} M_\Gamma \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_\Gamma \mathbf{v}} =: \widetilde{D}_1 = \mathcal{O}(1),$$

where $\widetilde{D}_1$ is a mesh and β-independent constant. Specifically, this is a constant that measures the spectral equivalence of the non-zero parts of $M_\Gamma \left(\frac{1}{h}M\right)^{-1} M_\Gamma = hM_\Gamma M^{-1} M_\Gamma$ and M_Γ.

We now examine in more detail the term

$$\frac{2\sqrt{\frac{h}{\beta}}\mathbf{v}^T M_\Gamma M^{-1} K \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_\Gamma \mathbf{v}} =: \frac{T_1}{T_2},$$

in particular its maximum and minimum values. For the purposes of this heuristic, it is useful to consider M and M_Γ as lumped mass matrices, as explained earlier in this section. We then have $M \approx h^2 I$ and $M_\Gamma \approx \text{blkdiag}\,(0, hI_b)$, ignoring multiplicative constants. The eigenvalues of K are within the interval $[c_K h^2, C_K]$, where c_K and C_K are constants independent of h and β (apart from one zero eigenvalue with a corresponding eigenvector of ones – as discussed in Section 2.1.2 this corresponds to an arbitrary constant being a solution of the continuous Neumann problem for Poisson's equation).

Let us consider a possible lower bound for the quantity $\frac{T_1}{T_2}$, paying attention first to T_1. This is a quantity which corresponds to the product of the matrices $M_\Gamma M^{-1}$ and K (which is symmetric positive semi-definite). Observe once more that when lumped mass matrices are considered, the matrix $M_\Gamma M^{-1}$ may be reasonably approximated by $\text{blkdiag}\,(0, h^{-1}I_b)$. In this case the quantity T_1 reduces to a scaled quadratic form with the matrix K, which we may expect to be non-negative in most cases. We also note that T_2 must always be strictly positive.[18] If T_1 and T_2 are both non-negative,

[17] The matrices M_Γ and $hM_\Gamma M^{-1} M_\Gamma$ themselves are not spectrally equivalent using our previous definition, as neither matrix is invertible. However, the non-zero sub-matrices of M_Γ and $hM_\Gamma M^{-1} M_\Gamma$ corresponding to boundary nodes are spectrally equivalent.

[18] This may be argued as follows. Both $\mathbf{v}^T K M^{-1} K \mathbf{v}$ and $\frac{1}{\beta}\mathbf{v}^T M_\Gamma \mathbf{v}$ are non-negative terms. The first term is strictly positive unless $\mathbf{v}$ is the vector of ones, corresponding to the zero eigenvalue of K. If this is the case, the $\mathbf{v}^T M_\Gamma \mathbf{v}$ term is strictly positive. So for each $\mathbf{v}$, at least one of the terms is strictly positive.

we may write that $\frac{T_1}{T_2} \geq 0$, and hence that $\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_1 \mathbf{v}} \leq \frac{1}{\widetilde{D}_1}$, giving us an upper bound for the largest eigenvalue, $\lambda_{\max}(\widehat{S}_1^{-1} S)$.

We now consider the maximum and minimum values of $\frac{T_1}{T_2}$. We consider the maximum such value by writing

$$\frac{T_1}{T_2} = \frac{2\beta^{-1/2} h^{1/2} h h^{-2} \xi}{h^{-2}\xi^2 + \beta^{-1} h} = \frac{2\beta^{-1/2} h^{-1/2} \xi}{h^{-2}(\xi^2 + \beta^{-1} h^3)} = \frac{2\upsilon\xi}{\xi^2 + \upsilon^2},$$

with $\upsilon = \beta^{-1/2} h^{3/2}$, and ξ corresponding to the relevant eigenvalue of K. Here, both υ and ξ are positive, and so in this case $\frac{2\upsilon\xi}{\xi^2+\upsilon^2} \leq 1$ by straightforward algebraic manipulation. This means that the denominator in (4.4) will be bounded above by a constant independent of h and β, as both terms are of $\mathcal{O}(1)$. This gives us a lower bound for the smallest eigenvalue, $\lambda_{\min}(\widehat{S}_1^{-1} S)$.

Putting our analysis together, and reinstating multiplicative constants, we conclude that

$$\lambda_{\min}(\widehat{S}_1^{-1} S) \geq \widetilde{c}_1, \quad \lambda_{\max}(\widehat{S}_1^{-1} S) \leq \widetilde{C}_1,$$

where $\widetilde{c}_1$ and $\widetilde{C}_1$ are positive constants independent of h and β.

Eigenvalues of $\widehat{S}_2^{-1} S$. It is possible to carry out a similar analysis for the approximation $\widehat{S}_2$ of S, by considering the Rayleigh quotient

$$\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_2 \mathbf{v}} = \frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \mathbf{v}^T \left(\frac{1}{\beta} M_\Gamma\right) \mathbf{v}}{\mathbf{v}^T K (h\widehat{M}_\Gamma)^{-1} K \mathbf{v} + \frac{h}{\beta} \mathbf{v}^T M_\Gamma (h\widehat{M}_\Gamma)^{-1} M_\Gamma \mathbf{v} + 2\sqrt{\frac{h}{\beta}} \mathbf{v}^T M_\Gamma (h\widehat{M}_\Gamma)^{-1} K \mathbf{v}},$$

and writing that $M \approx h^2 I$, $\widehat{M}_\Gamma \approx hI$, and $M_\Gamma \approx \text{blkdiag}\,(0, hI_b)$.

Proceeding as above, we obtain using our heuristic approach that

$$\lambda_{\min}(\widehat{S}_2^{-1} S) \geq \widetilde{c}_2, \qquad \lambda_{\max}(\widehat{S}_2^{-1} S) \leq \widetilde{C}_2,$$

where $\widetilde{c}_2$ and $\widetilde{C}_2$ are positive constants independent of h and β.

Although we are not able to bound the eigenvalues of the preconditioned Schur complements as precisely as for the distributed Poisson control problem, it is helpful that we may at least bound the eigenvalues heuristically by constants of $\mathcal{O}(1)$.

We therefore conclude that the two preconditioners

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} M & 0 \\ 0 & \widehat{S}_1 \end{bmatrix}, \qquad \widehat{\mathcal{P}}_2 = \begin{bmatrix} M & 0 \\ 0 & \widehat{S}_2 \end{bmatrix},$$

are potentially effective and robust ones for the Neumann boundary control problem. We may also consider block triangular preconditioners of the form

$$\begin{bmatrix} M & 0 \\ K & -\widehat{S}_1 \end{bmatrix}, \qquad \begin{bmatrix} M & 0 \\ K & -\widehat{S}_2 \end{bmatrix},$$

as described in the previous chapter.

4.1.2 Numerical Tests

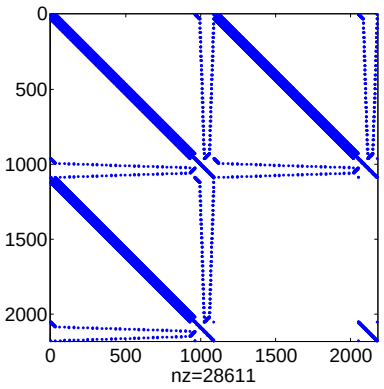
To demonstrate numerically the performance of our block diagonal preconditioners, we now wish to test them on the following problem:

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \left\| y - \widehat{y} \right\|^2_{L_2(\Omega)} + \frac{\beta}{2} \left\| u \right\|^2_{L_2(\partial\Omega)} \\ \text{s.t.} \ & -\nabla^2 y = 0, \quad \text{in } \Omega := [0,1]^2, \\ & \frac{\partial y}{\partial n} = u, \quad \text{on } \partial\Omega, \\ & \widehat{y} = \begin{cases} 1 & \text{on } \Omega_1 := [0, \frac{1}{2}]^2, \\ 0 & \text{on } \Omega \backslash \Omega_1. \end{cases} \end{aligned}$$

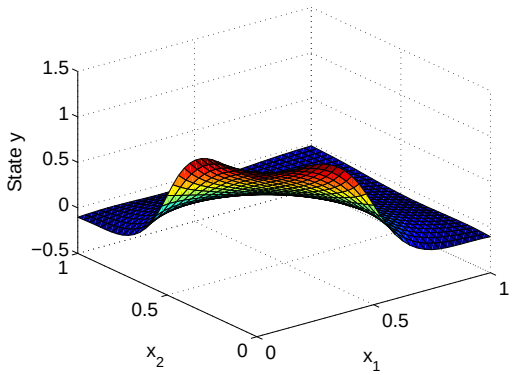
We note that the target function within the test problem is the same as that used to test our solvers for the distributed Poisson control problem in the previous chapter. However the nature of the problem is very different. Figure 4.1 is a plot of solutions obtained for a particular choice of h and β, along with a MATLAB '`spy`' plot of the matrix within the system we solve iteratively.

To solve this problem, we use **Q1** basis functions for state and adjoint variables. We use the MINRES algorithm, and observe the number of iterations required to solve the matrix system (4.1) to a tolerance of 10^{-6}, when preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ are used. We again approximate the inverse of a mass matrix by 20 Chebyshev semi-iterations, and approximate the inverse of the matrix $K + \sqrt{\frac{h}{\beta}} M_\Gamma$ using 2 AMG cycles with 2 pre- and post- (relaxed Jacobi) smoothing steps.

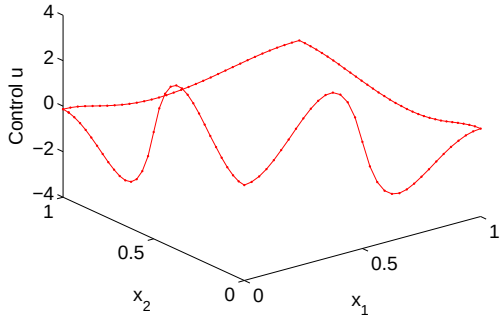
In Tables 4.1 and 4.2, we present iteration numbers required for solving the prob-



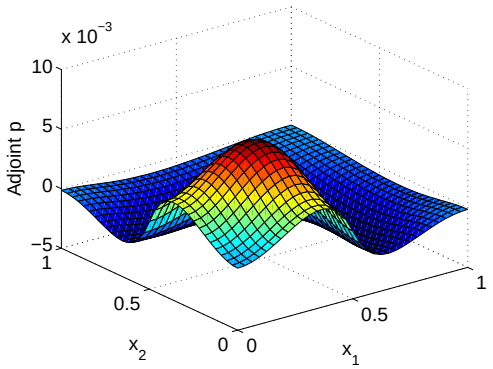
(a) '`spy`' plot



(b) State y



(c) Control u



(d) Adjoint p

Figure 4.1: Plots of state, control and adjoint variables for the Neumann boundary control test problem, with $h = 2^{-5}$ and $\beta = 10^{-3}$, as well as a MATLAB '`spy`' plot of the matrix within the system being solved.

$\widehat{\mathcal{P}}_1$ Iterations		β								
		1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
h	2^{-2}	18	19	21	21	18	17	15	14	14
	2^{-3}	17	21	31	34	35	30	31	27	26
	2^{-4}	18	26	34	44	50	48	40	43	42
	2^{-5}	18	26	39	52	63	65	61	57	61
	2^{-6}	23	29	44	59	75	87	88	79	75
	2^{-7}	22	30	46	67	90	113	121	120	106

Table 4.1: Number of MINRES iterations with preconditioner $\widehat{\mathcal{P}}_1$ required to solve the test problem, using **Q1** basis functions for state and adjoint, for a variety of h and β.

$\widehat{\mathcal{P}}_2$ Iterations		β								
		1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
h	2^{-2}	21	21	21	20	16	13	11	11	11
	2^{-3}	35	35	34	34	32	32	31	29	29
	2^{-4}	40	39	41	43	43	44	42	40	40
	2^{-5}	43	43	46	48	50	54	53	52	50
	2^{-6}	41	44	48	55	56	63	67	67	66
	2^{-7}	44	46	51	58	67	78	87	90	89

Table 4.2: Number of MINRES iterations with preconditioner $\widehat{\mathcal{P}}_2$ required to solve the test problem, using **Q1** basis functions for state and adjoint, for a variety of h and β.

lem with preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ respectively. We note that, whereas the iteration numbers do not remain as small or uniform as for the results presented in the previous chapter on distributed control, they do remain bounded as well as moderate in size, for all values of h and β tested (with $\widehat{\mathcal{P}}_2$ performing marginally better than $\widehat{\mathcal{P}}_1$ in the majority, though not all, of the cases). We also observe that the iterative methods seem to perform better when h^3 and β are far apart – we observe this in Tables 4.1 and 4.2, as, for a given h and decreasing β, the iteration numbers first seem to increase and then decrease again. This feature may be explained as follows: when $h^3 \ll \beta$, the first term of the Schur complement ($KM^{-1}K$) is very dominant and is captured well by the Schur complement approximations; similarly if $\beta \ll h^3$, the second term of the Schur complement ($\frac{1}{\beta}M_\Gamma$) dominates, and this too is captured well by our approximations.

As a result of the iteration counts we observe when employing our solvers, as well as CPU timings not presented here, we conclude that our solvers exhibit robustness when applied to this problem for a wide range of values of h and β.

4.2 Subdomain Control Problem

We now turn our attention to the numerical solution of subdomain control problems. We work with the following formulation:

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2} \left\| y - \widehat{y} \right\|^2_{L_2(\Omega)} + \frac{\beta}{2} \left\| u \right\|^2_{L_2(\Omega_{\text{sub}})} \\ \text{s.t.} \ & -\nabla^2 y = \begin{cases} u & \text{in } \Omega_{\text{sub}} \subset \Omega, \\ 0 & \text{in } \Omega \backslash \Omega_{\text{sub}}, \end{cases} \\ & y = g, \quad \text{on } \partial\Omega, \end{aligned}$$

which we discussed in Section 2.1.2. As explained there, an equal-order finite element method yields the matrix system

$$\begin{bmatrix} M & 0 & K \\ 0 & \beta M_{\text{sub}} & -N_s^T \\ K & -N_s & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix},$$

which we may reduce to

$$\begin{bmatrix} M & K \\ K & -\frac{1}{\beta} M_s \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{g} \end{bmatrix}, \tag{4.5}$$

by eliminating the gradient equation. The matrix $M_s := N_s M_{\text{sub}}^{-1} N_s^T$ contains entries of M_{sub} at indices corresponding to Ω_{sub}, and zeros elsewhere. We will consider the iterative solution of the matrix system (4.5) in this section.

We write this system in the form of the saddle point system (2.31), with

$$\Phi = M, \quad \Psi = K, \quad \Theta = \frac{1}{\beta} M_s.$$

We may hence consider block diagonal preconditioners of the form blkdiag $\left(\widehat{\Phi}, \widehat{S}\right)$ for this system (or indeed block triangular or other preconditioners). We may approxi-

mate the $(1,1)$-block using Chebyshev semi-iteration as before, leaving the main task as approximating the Schur complement

$$S = KM^{-1}K + \frac{1}{\beta}M_s.$$

4.2.1 Approximating the Schur Complement

As for the Neumann boundary control problem, we wish to propose options for approximating the Schur complement S which are robust with respect to mesh-size and regularization parameter.

As in the previous section, when carrying out our analysis we believe it to be helpful to consider the mass matrices in lumped form. Once again, M is then a diagonal matrix that is spectrally equivalent to $h^d I$, and the non-zero diagonal part of M_{sub} (corresponding to nodes in Ω_{sub}) is spectrally equivalent to $h^d I_s$, with I_s denoting the identity matrix of dimension equal to the number of nodes within Ω_{sub}. These observations lead to the following approximations, which will be helpful when analysing our Schur complement approximation for this problem:[19]

$$M_s M^{-1} \approx \text{blkdiag}\left(I_s, 0\right), \quad M_s \approx M_s M^{-1} M_s. \tag{4.6}$$

Using the above set-up, let us consider first an approximation of the form

$$\begin{aligned}\widehat{S}_1 &= \left(K + \frac{1}{\sqrt{\beta}}\widehat{M}\right) M^{-1} \left(K + \frac{1}{\sqrt{\beta}}\widehat{M}\right) \\ &= KM^{-1}K + \frac{1}{\beta}\widehat{M}M^{-1}\widehat{M} + \frac{1}{\sqrt{\beta}}\left(KM^{-1}\widehat{M} + \widehat{M}M^{-1}K\right),\end{aligned}$$

where here $\widehat{M}$ is chosen so that

$$\frac{1}{\beta}\widehat{M}M^{-1}\widehat{M} \approx \frac{1}{\beta}M_s. \tag{4.7}$$

Now, as for the boundary control case, the eigenvalues of M are all of $\mathcal{O}(h^d)$, and M_s contains eigenvalues of at most $\mathcal{O}(h^d)$. It is therefore natural to consider the choice

$$\widehat{M} = M_s,$$

[19] We have assumed from the point of view of this notation that the nodes are ordered with the nodes in Ω_{sub} first, followed by the remaining nodes.

as this will balance the largest eigenvalues of both sides of the expression (4.7), i.e.

$$\beta^{-1} \cdot h^d \cdot h^{-d} \cdot h^d = \beta^{-1} \cdot h^d.$$

We therefore propose the Schur complement approximation

$$\widehat{S}_1 = \left(K + \frac{1}{\sqrt{\beta}} M_s\right) M^{-1} \left(K + \frac{1}{\sqrt{\beta}} M_s\right),$$

as well as the alternative approximation

$$\widehat{S}_2 = \left(K + \frac{1}{\sqrt{\beta}} M_s\right) \widehat{M}_s^{-1} \left(K + \frac{1}{\sqrt{\beta}} M_s\right),$$

using similar motivation. The matrix $\widehat{M}_s$ contains the entries of M_s at nodes within Ω_{sub}, and entries of $\mathcal{O}(h^d)$ on the diagonals corresponding to nodes outside Ω_{sub}.[20]

Eigenvalues of $\widehat{S}_1^{-1}S$. A pertinent question at this point is whether we may apply a similar heuristic as for the Neumann boundary control problem to bound the largest and smallest eigenvalues of $\widehat{S}_1^{-1}S$ by values of $\mathcal{O}(1)$. We again analyse the eigenvalues in the 2D case, with the 3D case being similar.

We examine the Rayleigh quotient

$$\begin{aligned}\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_1 \mathbf{v}} &= \frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s M^{-1} M_s \mathbf{v} + \frac{1}{\sqrt{\beta}}\mathbf{v}^T \left[K M^{-1} M_s + M_s M^{-1} K\right] \mathbf{v}} \\ &= \frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s M^{-1} M_s \mathbf{v} + \frac{2}{\sqrt{\beta}}\mathbf{v}^T M_s M^{-1} K \mathbf{v}},\end{aligned}$$

which will give us a range for the eigenvalues of $\widehat{S}_1^{-1}S$ as before.

If $\mathbf{v} \in \text{null}(M_s)$, then $\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_1 \mathbf{v}} = 1$. If not, we may write

$$\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_1 \mathbf{v}} = \frac{1}{\frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s M^{-1} M_s \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}} + \frac{\frac{2}{\sqrt{\beta}}\mathbf{v}^T M_s M^{-1} K \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}}}. \tag{4.8}$$

Motivated by the above expression, we may utilize that the non-zero sub-matrices of

[20] In our implementation, we set the diagonal entries corresponding to nodes outside Ω_{sub} to be the same as one of the diagonal entries corresponding to a node within Ω_{sub}, to ensure the entries are of the same order.

$M_s M^{-1} M_s$ and M_s are spectrally equivalent,[21] as explained above in (4.6), and hence write

$$0 < \frac{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s M^{-1} M_s \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}} =: \widetilde{D}_2 = \mathcal{O}(1),$$

with $\widetilde{D}_2$ an h and β-independent constant corresponding to the spectral equivalence of the sub-matrices of $M_s M^{-1} M_s$ and M_s.

We now look more carefully at the maximum and minimum values of

$$\frac{\frac{2}{\sqrt{\beta}}\mathbf{v}^T M_s M^{-1} K \mathbf{v}}{\mathbf{v}^T K M^{-1} K \mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s \mathbf{v}} =: \frac{T_3}{T_4}.$$

We again consider, for the purposes of our heuristic, lumped versions of the mass matrices M and M_s, and assume that $M \approx h^2 I$ and $M_s \approx \text{blkdiag}\,(h^2 I_s, 0)$, ignoring all multiplicative constants. We note that the eigenvalues of K are within $[c_K h^2, C_K]$, for constants c_K and C_K which are independent of h and β.

Using our approximations for M and M_s, we may write that the matrix $M_s M^{-1} \approx \text{blkdiag}\,(I_s, 0)$. This puts us in a position to argue, as in the previous section for the quantity T_1, that T_3 is essentially a scaled quadratic form for K, and is therefore typically non-negative. Further, T_4 must be strictly positive, using a similar argument as for the quantity T_2 in the previous section. If T_3 and T_4 are indeed non-negative, we write that $\frac{T_3}{T_4} \geq 0$, and therefore that $\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T \widehat{S}_1 \mathbf{v}} \leq \frac{1}{\widetilde{D}_2}$, giving us an upper bound for $\lambda_{\max}(\widehat{S}_2^{-1} S)$.

We may then consider the maximum value of $\frac{T_3}{T_4}$, which we do by writing

$$\frac{T_3}{T_4} = \frac{2\beta^{-1/2} h^2 h^{-2} \xi}{h^{-2}\xi^2 + \beta^{-1} h^2} = \frac{2\beta^{-1/2}\xi}{h^{-2}(\xi^2 + \beta^{-1} h^4)} = \frac{2\varpi\xi}{\xi^2 + \varpi^2},$$

with $\varpi = \beta^{-1/2} h^2$, and ξ corresponding to the relevant eigenvalue of K as before. Both ϖ and ξ are positive, and so $\frac{2\varpi\xi}{\xi^2+\varpi^2} \leq 1$. This means that the denominator in (4.8) is bounded above by a constant independent of h and β. This gives us the lower bound for $\lambda_{\min}(\widehat{S}_2^{-1} S)$ that we seek.

Putting all the pieces together (with multiplicative constants reintroduced) we

[21] The non-zero sub-matrices of $M_s M^{-1} M_s$ and M_s corresponding to nodes in Ω_{sub} are spectrally equivalent using our definition.

may conclude that

$$\lambda_{\min}(\widehat{S}_1^{-1}S) \geq \widetilde{c}_3, \quad \lambda_{\max}(\widehat{S}_1^{-1}S) \leq \widetilde{C}_3,$$

with $\widetilde{c}_3$ and $\widetilde{C}_3$ positive constants independent of h and β.

Eigenvalues of $\widehat{S}_2^{-1}S$. We may perform a similar analysis for $\widehat{S}_2^{-1}S$, by examining the Rayleigh quotient

$$\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_2\mathbf{v}} = \frac{\mathbf{v}^T K M^{-1}K\mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s\mathbf{v}}{\mathbf{v}^T K\widehat{M}_s^{-1}K\mathbf{v} + \frac{1}{\beta}\mathbf{v}^T M_s\widehat{M}_s^{-1}M_s\mathbf{v} + \frac{2}{\sqrt{\beta}}\mathbf{v}^T M_s\widehat{M}_s^{-1}K\mathbf{v}},$$

and writing that $M \approx h^2 I$, $\widehat{M}_s \approx h^2 I$, and $M_s \approx \text{blkdiag}\,(h^2 I_s, 0)$.

Proceeding once more as we did for $\widehat{S}_1$, we obtain that

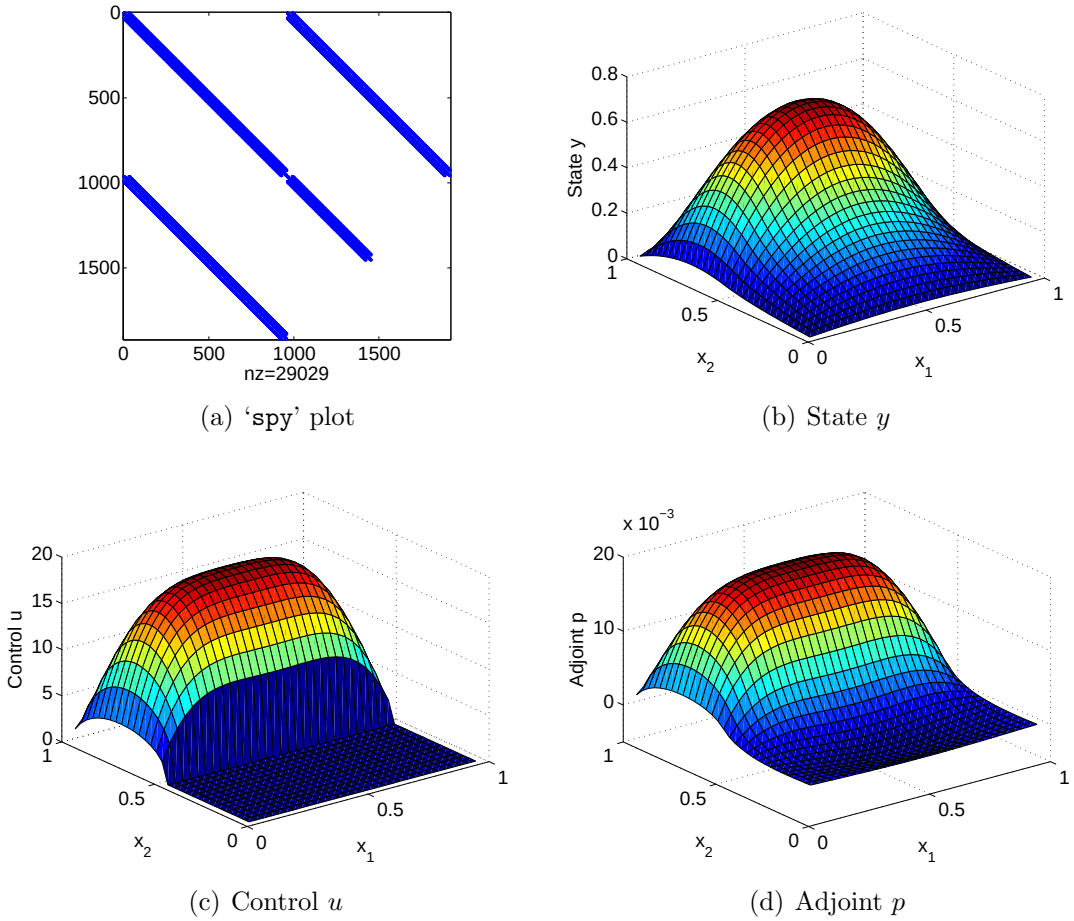
$$\lambda_{\min}(\widehat{S}_2^{-1}S) \geq \widetilde{c}_4, \qquad \lambda_{\max}(\widehat{S}_2^{-1}S) \leq \widetilde{C}_4,$$

where $\widetilde{c}_4$ and $\widetilde{C}_4$ are positive constants independent of h and β.

We therefore postulate that the two preconditioners

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} M & 0 \\ 0 & \widehat{S}_1 \end{bmatrix}, \qquad \widehat{\mathcal{P}}_2 = \begin{bmatrix} M & 0 \\ 0 & \widehat{S}_2 \end{bmatrix},$$

are potentially effective and robust ones for the subdomain control problem, as may be related block triangular preconditioners.



(a) '`spy`' plot

(b) State y

(c) Control u

(d) Adjoint p

Figure 4.2: Plots of state, control and adjoint variables for the subdomain control test problem, with $h = 2^{-5}$ and $\beta = 10^{-3}$, as well as a MATLAB '`spy`' plot of the matrix within the system being solved.

4.2.2 Numerical Tests

In order to demonstrate how the above preconditioners perform in practice, we wish to test our methods on the following test problem:

$$\begin{aligned}
&\min_{y,u} \ \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega_{\text{sub}})} \\
&\text{s.t.} \quad -\nabla^2 y = \begin{cases} u & \text{in } \Omega_{\text{sub}} := [0,1]\times[\frac{1}{2},1], \\ 0 & \text{in } \Omega\backslash\Omega_{\text{sub}}, \end{cases} \\
&\qquad\qquad y = 0, \quad \text{on } \partial\Omega, \\
&\qquad\qquad \widehat{y} = \begin{cases} 1 & \text{on } \Omega_{\text{sub}}, \\ 0 & \text{on } \Omega\backslash\Omega_{\text{sub}}. \end{cases}
\end{aligned}$$

Figure 4.2 contains solution plots for this problem, along with a MATLAB '`spy`' plot of the matrix we consider solving for.

$\widehat{\mathcal{P}}_1$ ITERATIONS		β								
		1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
h	2^{-2}	10	11	11	12	12	(12)*	(12)*	(12)*	(12)*
	2^{-3}	17	17	19	21	23	24	(30)*	(30)*	(33)*
	2^{-4}	25	27	28	30	31	36	41	(50)*	(51)*
	2^{-5}	38	40	41	44	48	57	66	76	87
	2^{-6}	44	44	47	52	62	77	84	97	114
	2^{-7}	44	46	50	56	69	84	90	103	118

Table 4.3: Number of MINRES iterations with preconditioner $\widehat{\mathcal{P}}_1$ required to solve the test problem, using **Q1** basis functions for state and adjoint, for a variety of h and β.

We again use **Q1** basis functions for state and adjoint variables, and apply the MINRES algorithm to solve this problem. We solve to a tolerance of 10^{-6} with preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$, approximating the inverse of a mass matrix by 20 Chebyshev semi-iterations and the inverse of the matrix $K + \frac{1}{\sqrt{\beta}}M_s$ using 2 AMG cycles with 2 pre- and post- (relaxed Jacobi) smoothing steps.[22]

[22] In Tables 4.3 and 4.4, the symbol * denotes that the AMG routine did not solve for the matrix $K + \frac{1}{\sqrt{\beta}}M_s$ in the parameter regime being tested, due to the presence of positive off-diagonal entries. In these cases Gaussian elimination solves were used instead. We note that this problem only occurs when h is large and β is small, which is not an interesting practical case.

$\widehat{\mathcal{P}}_2$ ITERATIONS		β								
		1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
h	2^{-2}	12	12	12	12	12	(12)*	(12)*	(12)*	(11)*
	2^{-3}	27	28	28	30	31	32	(34)*	(33)*	(33)*
	2^{-4}	45	45	45	45	48	52	53	(55)*	(58)*
	2^{-5}	67	69	70	71	76	87	92	90	97
	2^{-6}	77	77	80	85	96	115	121	123	131
	2^{-7}	78	80	83	88	102	127	128	129	136

Table 4.4: Number of MINRES iterations with preconditioner $\widehat{\mathcal{P}}_2$ required to solve the test problem, using **Q1** basis functions for state and adjoint, for a variety of h and β.

Tables 4.3 and 4.4 show iteration numbers for solving the problem with preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ respectively. Again, the iteration numbers remain bounded, and are very reasonable in size for all parameter regimes of h and β tested (with the preconditioner $\widehat{\mathcal{P}}_1$ marginally outperforming $\widehat{\mathcal{P}}_2$). We find that the CPU times demonstrate parameter robustness also. We observe for this problem that the iterative methods seem to perform better when h^4 and β are far apart, as when this is the case one of the terms of the Schur complement ($KM^{-1}K$ or $\frac{1}{\beta}M_s$) is dominant and is captured well by our Schur complement approximations. However, we can again conclude that our solvers perform well for the subdomain control problem considered, for all values of h and β tested.

4.3 Comment on Distributed Poisson Control with State Constraints

Finally, in this section we briefly discuss how we may approach solving the matrix system

$$\begin{bmatrix} M + \epsilon^{-1} G_{\mathcal{A}} M G_{\mathcal{A}} & 0 & K \\ 0 & \beta M & -M \\ K & -M & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k)} \\ \mathbf{u}_{(k)} \\ \mathbf{p}_{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_{\mathcal{A}} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix}, \tag{4.9}$$

which arises at every semi-smooth Newton step when seeking the solution of the following problem:

$$\begin{aligned} \min_{y,u} \ & \frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nabla^2 y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega, \\ & y_a \leq y \leq y_b, \quad \text{a.e. in } \Omega. \end{aligned}$$

We discussed optimal control problems with such additional state constraints previously in Section 2.1.3, and we wish to consider the iterative solution of this problem, as it is a further example of a Poisson control problem with additional complexities arising in the structure of the matrix system. The notation with which this system is written is as in Section 2.1.3.

In [90], preconditioners for the matrix system (4.9) are sought. The $(1,1)$-block

$$\Phi = \begin{bmatrix} M+\epsilon^{-1}G_{\mathcal{A}}MG_{\mathcal{A}} & 0 \\ 0 & \beta M \end{bmatrix}$$

is easily obtained using exact diagonal solves, as lumped mass matrices are used. If this were not the case, then the matrix $M+\epsilon^{-1}G_{\mathcal{A}}MG_{\mathcal{A}}$ could be split into two parts (one corresponding to the active set and one to the other nodes) – the inverse of the matrix could then be approximated by applying Chebyshev semi-iteration to the separated matrices. The matrix βM could be dealt with by Chebyshev semi-iteration as usual in the case of consistent mass matrices being used.

The most difficult task, once again, is approximating the Schur complement

$$S = K\left(M+\epsilon^{-1}G_{\mathcal{A}}MG_{\mathcal{A}}\right)^{-1}K+\frac{1}{\beta}M.$$

This time, we seek robust approximations of S with respect to three variables – h, β and ϵ. The presence of an additional parameter creates extra complications when it comes to finding robust solvers.

In [90], a proposed approximation of S is given by

$$\widehat{S}_1 = \left(K+\frac{1}{\sqrt{\beta}}\widehat{M}\right)\left(M+\epsilon^{-1}G_{\mathcal{A}}MG_{\mathcal{A}}\right)^{-1}\left(K+\frac{1}{\sqrt{\beta}}\widehat{M}\right),$$

where here $\widehat{M} = \text{blkdiag}\left(M_{\mathcal{I}}, \sqrt{1+\epsilon^{-1}}M_{\mathcal{A}}\right)$, and $M_{\mathcal{I}}$, $M_{\mathcal{A}}$ denote the parts of the mass matrix corresponding to the inactive and active sets respectively. The approximation $\widehat{S}_1$ is justified by exactly the same "matching" strategy as for the other problems discussed in this section.

Using similar motivation, the second Schur complement approximation

$$\widehat{S}_2 = \left[K + \frac{1}{\sqrt{\beta}} M I_{G,\epsilon}\right] I_{G,\epsilon}^{-1} M^{-1} I_{G,\epsilon}^{-1} \left[K + \frac{1}{\sqrt{\beta}} I_{G,\epsilon} M\right],$$

where $I_{G,\epsilon} = I + \frac{1}{\sqrt{\epsilon}} G_{\mathcal{A}}$, is also proposed.

Although it is again not possible to prove rigorous bounds for $\widehat{S}_1^{-1}S$ or $\widehat{S}_2^{-1}S$ as for the distributed control problem without state constraints, numerical experiments indicate that the (block triangular) preconditioners,

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \gamma\left(M + \epsilon^{-1} G_{\mathcal{A}} M G_{\mathcal{A}}\right) & 0 & 0 \\ 0 & \beta\gamma M & 0 \\ K & -M & -\widehat{S}_1 \end{bmatrix},$$

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \gamma\left(M + \epsilon^{-1} G_{\mathcal{A}} M G_{\mathcal{A}}\right) & 0 & 0 \\ 0 & \beta\gamma M & 0 \\ K & -M & -\widehat{S}_2 \end{bmatrix},$$

are effective ones for this problem for a wide range of parameter regimes for h, β and ϵ. As for the block triangular preconditioners discussed in the previous chapter, γ denotes a scaling parameter to ensure positive definiteness within the inner product being used. These preconditioners were used with the Bramble-Pasciak CG method.

Of course related block diagonal preconditioners could also be used along with the MINRES algorithm – these may be derived in the same way as for the boundary and subdomain control problems of this chapter.

4.4 Summary

In this chapter, we have described the derivation of solvers for Poisson control problems of different structure to the distributed control problem discussed in Chapter 3. The classes of problems detailed were the Neumann boundary control problem, the subdomain control problem, and, briefly, the state-constrained Poisson control problem.

We pursued a similar strategy as for the distributed Poisson control problem. The main added difficulty for these problems was the more complex structure of the Schur complement. However, we were able to use similar ideas as in Chapter 3 to derive effective approximations. We motivated heuristic eigenvalue bounds (in contrast to the more rigorous bounds for the distributed control problem), and numerical results indicate that our preconditioners are robust and efficient ones for these harder problems.

The main conclusion of this chapter is that the methodology detailed in Chapter 3, involving saddle point theory, effective mass matrix approximation, and approximating the Schur complement using a "matching" strategy as we have described, can be effective for tackling a range of optimal control problems, even if the problems are not of the exact nature for which the strategy was originally derived. This indicates that the family of preconditioners we have developed so far for optimal control problems may be applicable for a yet wider range of problems – we will examine this conjecture in future chapters.

CHAPTER 5

Convection-Diffusion Control

In this chapter, we examine the iterative solution of convection-diffusion control problems, a more sophisticated class of PDE-constrained optimization problems than that of Poisson control problems, and also a more challenging one from a numerical point-of-view.[23] Equations of convection-diffusion type are an important class of problems motivated by applications such as contaminant transport. When solving forward problems of this type using a finite element method, they result in non-symmetric matrix systems, which needs to be considered when solving optimal control problems of this form, as well as the forward problem in isolation.

In addition, convection-diffusion type problems often require stabilization techniques to be applied, in order that the solution obtained is reliable. For convection-diffusion control problems, we thus wish to select a stabilization scheme which results in the matrix systems obtained using the discretize-then-optimize and optimize-then-discretize approaches being the same. Commonly-used stabilization strategies, such as the Streamline Upwind Petrov-Galerkin stabilization [71], do not satisfy this property. However, the stabilization method we select, that of the Local Projection Stabilization approach [6, 7, 20], does meet this requirement. With this stabilization, along with appropriate approximations of the $(1,1)$-block and Schur complement of

[23] This chapter is based on the following paper, which is Ref. [93]:

J. W. Pearson, and A. J. Wathen, *Fast Iterative Solvers for Convection-Diffusion Control Problems*, Electronic Transactions on Numerical Analysis, 40, pp.294–310, 2013.

the matrix system involved, and a carefully constructed multigrid routine to enforce the Schur complement approximation, we will develop effective block diagonal and block triangular preconditioners for this problem.

This chapter is structured as follows. In Section 5.1, we discuss the solution of the convection-diffusion equation – we will incorporate elements of this solver into our solution strategies for the control problem. In Section 5.2, we introduce the convection-diffusion control problem, discuss stabilization strategies for the problem, and motivate our choice of stabilization by considering the matrix systems obtained using the discretize-then-optimize and optimize-then-discretize strategies. In Section 5.3, we derive block diagonal and block triangular preconditioners for the control problem, to be used with MINRES and Bramble-Pasciak CG respectively. In Section 5.4, we provide results obtained from numerical experiments, and in Section 5.5, we make some concluding comments.

5.1 Iterative Solution of the Convection-Diffusion Equation

In order to motivate the work we will carry out on the convection-diffusion control problem, we first consider the finite element solution of the convection-diffusion equation with Dirichlet boundary conditions

$$-\nu\nabla^2 y + \mathbf{w}\cdot\nabla y = f, \quad \text{in } \Omega, \tag{5.1}$$

$$y = g, \quad \text{on } \partial\Omega, \tag{5.2}$$

where the domain $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$, has boundary $\partial\Omega$, $\nu > 0$, and $\mathbf{w}$ is a divergence-free wind vector (i.e. $\nabla \cdot \mathbf{w} = 0$).

The $-\nu\nabla^2 y$ term in the above equation denotes the diffusive element, and the $\mathbf{w}\cdot\nabla y$ term represents convection. As pointed out, for example in [36, Chapter 3], convection typically plays a more significant physical role than diffusion, so $\nu \ll \|\mathbf{w}\|$ for many practical problems. However, this in turn makes the problem more difficult to solve [36, 98], as the solution procedure will need to be robust with respect to the direction of the wind $\mathbf{w}$ and any boundary or internal layers that form.

The finite element representation of the equations (5.1)–(5.2) is given by

$$\bar{K}\mathbf{y} = \mathbf{f}, \tag{5.3}$$

where $\mathbf{y} = \{Y_i\}_{i=1,...,n}$, with Y_i denoting the coefficients of the finite element solution $y_h = \sum_{i=1}^{n+n_\partial} Y_i\phi_i$ with interior finite element basis functions $\phi_1, ..., \phi_n$ and boundary basis functions $\phi_{n+1}, ..., \phi_{n+n_\partial}$. The matrix $\bar{K}$ is defined by

$$\bar{K} = \nu K + N + \bar{T},$$

where K is a stiffness matrix as defined by (2.6),

$$N = \{\widetilde{n}_{ij}\}_{i,j=1,...,n}, \qquad \widetilde{n}_{ij} = \int_\Omega (\mathbf{w}\cdot\nabla\phi_j)\phi_i \, \mathrm{d}\Omega,$$

$\bar{T}$ is a matrix corresponding to the stabilization strategy used (which depends on the mesh-size h, a stabilization parameter δ and an orthogonal projection operator π_h), and $\mathbf{f}$ is a vector corresponding to the functions f and g (and sometimes the stabilization as well). We discuss the definitions of $\bar{T}$ and $\mathbf{f}$ for two different stabilization methods in Section 5.2.1, and note that $\bar{T} = 0$ if no stabilization is used.

A method discussed in [36, Chapter 4] for solving (5.1)–(5.2) is a GMRES method preconditioned with a geometric multigrid process described by Ramage in [98]. The multigrid process contains standard prolongation and restriction operators, but there are two major differences between it and a more typical multigrid routine:

- **Construction of the coarse grid operator** – In most geometric multigrid algorithms, the construction of a coarse grid operator is carried out using the scaled Galerkin coarse grid operator (that is $\bar{K}_{\text{coarse}} = R\bar{K}_{\text{fine}}P$, where P is the projection operator and R the restriction operator). However, in Ramage's method, the coarse grid operator is *explicitly constructed* on all grids on which it is required. This involves constructing the matrices K, N and $\bar{T}$ on each sub-grid, and incorporates different stabilization parameters δ for each grid.

- **Pre- and post- smoothing** – The smoothing strategy we employ is *block Gauss-Seidel smoothing* (which involves taking as a splitting matrix the block lower triangular part of the matrix $\bar{K}$, ordered in different ways), applied in each direction to take account of all possible wind directions, that is to say we employ 4 (2 pre- and 2 post-) smoothing steps for a two dimensional problem, and 6

smoothing steps for a three dimensional problem. This strategy is shown to be effective for a wide range of problems with our formulation, as demonstrated in [36, Chapter 4] and [98].

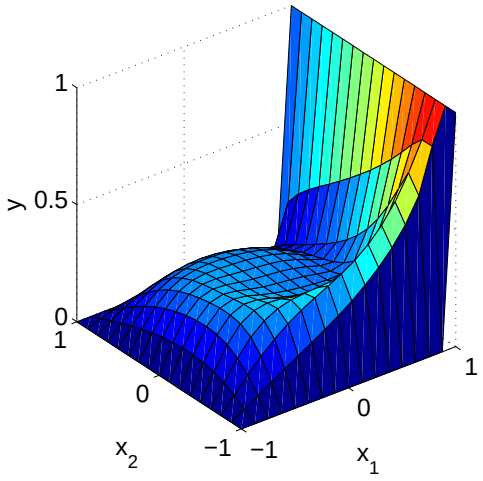


Figure 5.1: Finite element solution plot of (5.1)–(5.2), with $\Omega = [-1, 1]^2$, $h = 2^{-3}$, $\nu = \frac{1}{200}$, $f = 0$, $g = 1$ on $x_1 = 1$ and 0 otherwise, and $\mathbf{w} = [\frac{1}{2}x_2(1-x_1^2), -\frac{1}{2}x_1(1-x_2^2)]^T$, where $\mathbf{x} = [x_1, x_2]^T$ denotes the spatial coordinates.

In Figure 5.1, we display the solution of the convection-diffusion equation for a particular problem set-up.

5.2 Convection-Diffusion Control Problems

For the remainder of this chapter, we will consider the distributed convection-diffusion control problem

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2}\|y - \widehat{y}\|^2_{L_2(\Omega)} + \frac{\beta}{2}\|u\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nu\nabla^2 y + \mathbf{w}\cdot\nabla y = u, \quad \text{in } \Omega, \\ & y = g, \quad \text{on } \partial\Omega, \end{aligned} \tag{5.4}$$

where y again denotes the state variable with $\widehat{y}$ some desired state, u denotes the control, and $\beta > 0$ is a regularization parameter.

We again employ a finite element method to solve the problem, that is we write

$$y_h = \sum_{i=1}^{n+n_\partial} Y_i \phi_i, \quad u_h = \sum_{i=1}^{n+n_\partial} U_i \phi_i, \quad p_h = \sum_{i=1}^{n+n_\partial} P_i \phi_i,$$

where p denotes the Lagrange multiplier (or adjoint variable) we use. Note that we discretize the state y, the control u and the Lagrange multiplier p using the same basis functions here. Note also that the coefficients $Y_{n+1}, ..., Y_{n+n_\partial}$ are trivially obtained by considering the specified Dirichlet boundary condition $y = g$.

For the rest of this section, we define $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ as follows:

$$\mathbf{y} = \{Y_i\}_{i=1,...,n}, \quad \mathbf{u} = \{U_i\}_{i=1,...,n}, \quad \mathbf{p} = \{P_i\}_{i=1,...,n},$$

as for the Poisson control problem of Chapter 3.

5.2.1 Stabilization of the Control Problem

One important consideration when solving the convection-diffusion control problem (or indeed the convection-diffusion equation itself) is that of stabilizing the problem. It is well known that, without any form of stabilization, accurate solution of the convection-diffusion equation [36, 98] and the convection-diffusion control problem [7, 61] is compromised due to the formation of layers in the approximate solution, potentially leading to large errors for small ν.

One popular method for avoiding this problem is by using the *Streamline Upwind Petrov-Galerkin* (SUPG) stabilization, which was introduced in [71] and discussed further in literature such as [36, 61, 99]. For the forward problem, using this stabilization would result in a system of the form (5.3), with K and N as above, and

$$\begin{aligned}
\bar{T} = \{\tau^\delta_{h,ij}\}_{i,j=1,...,n}, \quad \tau^\delta_{h,ij} &= \delta \int_\Omega (\mathbf{w} \cdot \nabla \phi_i)(\mathbf{w} \cdot \nabla \phi_j) \, \mathrm{d}\Omega \\
&\quad - \nu\delta \sum_k \int_{\bar{\Delta}_k} (\nabla^2 \phi_i)(\mathbf{w} \cdot \nabla \phi_j) \, \mathrm{d}\Omega, \\
\mathbf{f} = \{f_i\}_{i=1,...,n}, \quad f_i &= \int_\Omega f \phi_i \, \mathrm{d}\Omega + \delta \int_\Omega f \mathbf{w} \cdot \nabla \phi_i \, \mathrm{d}\Omega,
\end{aligned}$$

with stabilization parameter δ, and $\bar{\Delta}_k$ denoting the k-th element in our finite element discretization. Here we have taken zero Dirichlet conditions for illustrative purposes.

It is well recognised that this method is effective for solving the forward problem (see [36, Chapters 3 and 4] for instance). However, for the convection-diffusion control problem, difficulties arise – the matrix systems that we obtain when we use the discretize-then-optimize and optimize-then-discretize formulations of Sections 5.2.2 and 5.2.3 do not commute [99, Chapter 6]. This is problematic as we would then have to choose between solving the discretize-then-optimize matrix system, which would not be strongly consistent (meaning the solutions to the optimization problem would not satisfy all the optimality conditions), or the optimize-then-discretize system, which is non-symmetric and so is not the optimality system for any finite dimensional problem. Further, the non-symmetry of the matrix system that arises when using the optimize-then-discretize approach means that we cannot apply the methodology detailed in Section 5.3 to solve it, as our methods depend on the matrix being symmetric. It is also believed that applying SUPG to the optimal control problem will guarantee at most first-order accuracy in the solution [61].

To deal with these two problems, we now introduce the *Local Projection Stabilization* (LPS) method, which is discussed in [7, 53] for example. Applying this stabilization to the forward problem again yields a matrix system of the form (5.3), with K and N as above and

$$\begin{aligned} \bar{T} = \{\tau^{\delta}_{h,ij}\}_{i,j=1,\ldots,n}, \quad \tau^{\delta}_{h,ij} &= \delta \int_{\Omega} \big(\mathbf{w}\cdot\nabla\phi_i - \pi_h(\mathbf{w}\cdot\nabla\phi_i)\big) \\ &\qquad \times \big(\mathbf{w}\cdot\nabla\phi_j - \pi_h(\mathbf{w}\cdot\nabla\phi_j)\big)\,\mathrm{d}\Omega, \\ \mathbf{f} = \{f_i\}_{i=1,\ldots,n}, \quad f_i &= \int_{\Omega} f\phi_i \,\mathrm{d}\Omega, \end{aligned} \tag{5.5}$$

with δ again a stabilization parameter and π_h an orthogonal projection operator from $L_2(\Omega)$ to the space of piecewise constant functions on patches of the domain. We have again taken zero Dirichlet conditions for this working. Furthermore, as we will demonstrate in Sections 5.2.2 and 5.2.3, when this stabilization is applied in the optimal control setting, the discretize-then-optimize and optimize-then-discretize systems are consistent and self-adjoint; that is the discretization and optimization steps commute.

There are a number of considerations detailed below which need to be taken into account when applying this method in the control setting with a uniform grid and bilinear basis functions, as we will do in Section 5.4.

- **Stabilization parameter** δ – We take δ to be the following, as in [7]:

$$\delta = \begin{cases} 0 & \text{if Pe} < 1, \\ \frac{h}{\|\mathbf{w}\|_2} & \text{if Pe} \geq 1, \end{cases}$$

  where the mesh Péclet number Pe is defined on each element as

$$\text{Pe} = \frac{h \|\mathbf{w}\|_2}{\nu}.$$

  Clearly this means that the stabilization depends on the mesh-size, and if the mesh-size h is less than $\frac{\nu}{\|\mathbf{w}\|_2}$, then no stabilization procedure will be applied.

- **Orthogonal projection operator** π_h – We require an L_2-orthogonal projection operator defined on patches of the domain, that satisfies L_2-norm properties specified on [7, p.4]. We will proceed by working with **Q1** elements with equally spaced nodes, and divide the domain into patches consisting of 2 elements in each dimension. From this, we will take $\pi_h(v)$ (where v has support solely on that patch) to be equal to the integral of v over the patch divided by the area of the patch (in 2D this will be $4h^2$). This definition will satisfy the required properties in our formulation.

- **Error of LPS method** – In [7], it is shown that the LPS stabilization gives $\mathcal{O}(h^{3/2})$ convergence for problems of the form (5.4) for bilinear finite elements. This further motivates the use of the LPS stabilization method for the remainder of this chapter.

5.2.2 Discretize-then-Optimize Strategy

We now demonstrate that, when using the LPS method described in Section 5.2.1, the matrix systems obtained with the discretize-then-optimize and optimize-then-discretize approaches are the same. The derivation of the matrix system when using the former approach is straightforward. We first note that the discretized version of the PDE constraint is given by

$$\bar{K}\mathbf{y} - M\mathbf{u} = \mathbf{g},$$

where $\mathbf{g}$ is as defined for the Poisson control problem.

We also note that we may write the functional that we are trying to minimize, $\frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)}$, as

$$\frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)}=\frac{1}{2}\mathbf{y}^TM\mathbf{y}-\mathbf{y}^T\mathbf{z}+\bar{C}+\frac{\beta}{2}\mathbf{u}^TM\mathbf{u},$$

where $\bar{C}$ is a constant independent of $\mathbf{y}$, M denotes a mass matrix as defined previously, and $\mathbf{z}$ contains entries of the form $\int_\Omega \widehat{y}\phi_i \,\mathrm{d}\Omega$.

We therefore deduce that the Lagrangian, the stationary point of which we wish to find, is given by

$$\mathcal{L}_{DTO}(\mathbf{y},\mathbf{u},\mathbf{p})=\frac{1}{2}\mathbf{y}^TM\mathbf{y}-\mathbf{y}^T\mathbf{z}+\bar{C}+\frac{\beta}{2}\mathbf{u}^TM\mathbf{u}+\mathbf{p}^T(\bar{K}\mathbf{y}-M\mathbf{u}-\mathbf{g}). \tag{5.6}$$

Differentiating (5.6) with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ yields the following system of equations:

$$\begin{bmatrix} M & 0 & \bar{K}^T \\ 0 & \beta M & -M \\ \bar{K} & -M & 0 \end{bmatrix}\begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix}=\begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix}. \tag{5.7}$$

This is of the saddle point form discussed in Section 2.2. To provide an illustration of the appearance of this matrix system, we display the sparsity pattern for a particular problem in Figure 5.2.

5.2.3 Optimize-then-Discretize Strategy

To derive the optimize-then-discretize formulation, as in [7], we need to consider a Lagrangian of the form

$$\begin{aligned}\mathcal{L}_{OTD}(y,u,p_\Omega,p_{\partial\Omega})=&\frac{1}{2}\left\|y-\widehat{y}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|u\right\|^2_{L_2(\Omega)}\\&+\int_\Omega(-\nu\nabla^2y+\mathbf{w}\cdot\nabla y-u)p_\Omega\,\mathrm{d}\Omega+\int_{\partial\Omega}(y-g)p_{\partial\Omega}\,\mathrm{d}s,\end{aligned}$$

where y and u relate to the weak solutions of the forward problem, and p_Ω, $p_{\partial\Omega}$ (which correspond to the adjoint variable p on the interior of Ω and $\partial\Omega$ respectively) are assumed to be sufficiently smooth.

As in [99, Chapter 6] for example, we differentiate $\mathcal{L}_{OTD}$ with respect to the state y, the control u and the Lagrange multipliers p_Ω and $p_{\partial\Omega}$, and study the resulting

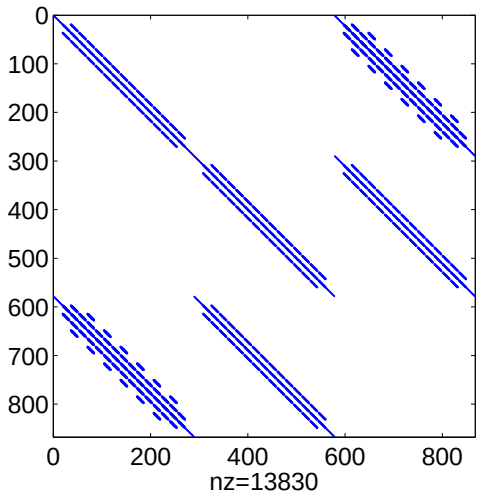


Figure 5.2: Sparsity pattern of the system (5.7) for Problem 1 as stated in Section 5.4, with $h = 2^{-3}$, $\beta = 10^{-2}$ and $\nu = \frac{1}{200}$.

equations. Calculating the Fréchet derivative with respect to y, and applying the Divergence Theorem and the Fundamental Lemma of Calculus of Variations, as in [99, Chapter 6], yields

$$\begin{aligned} -\nu\nabla^2 p - \mathbf{w}\cdot\nabla p - (\nabla\cdot\mathbf{w})p &= \widehat{y} - y, \quad \text{in } \Omega, \\ p &= 0, \qquad \text{on } \partial\Omega, \end{aligned}$$

from which we use the assumption $\nabla\cdot\mathbf{w} = 0$ to obtain the *adjoint equation*

$$\begin{aligned} -\nu\nabla^2 p - \mathbf{w}\cdot\nabla p &= \widehat{y} - y, \quad \text{in } \Omega, \\ p &= 0, \qquad \text{on } \partial\Omega. \end{aligned}$$

Further, differentiating with respect to u generates the *gradient equation*

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

and differentiating with respect to the Lagrange multipliers p_Ω and $p_{\partial\Omega}$ yields the *state equation*

$$\begin{aligned} -\nu\nabla^2 y + \mathbf{w}\cdot\nabla y &= u, \quad \text{in } \Omega, \\ y &= f, \quad \text{on } \partial\Omega. \end{aligned}$$

Discretizing these three equations using the stabilization (5.5) yields the matrix system

$$\begin{bmatrix} M & 0 & \bar{K}^T \\ 0 & \beta M & -M \\ \bar{K} & -M & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \\ \mathbf{g} \end{bmatrix},$$

which is the same saddle point system as that derived using the discretize-then-optimize approach. We therefore consider the solution of this system for the remainder of this chapter.

5.3 Preconditioning the Matrix System

We now consider how one might precondition the matrix system (5.7) for solving the convection-diffusion control problem with Local Projection Stabilization. We will use the saddle point theory of Section 2.2 in this section.

We first note that we may write (5.7) as a sparse saddle point system of the form (2.31), with

$$\Phi = \begin{bmatrix} M & 0 \\ 0 & \beta M \end{bmatrix}, \quad \Psi = \begin{bmatrix} \bar{K} & -M \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

By the theory of Section 2.2, we see that we may obtain an effective solver if we have a good approximation to Φ, as well as the Schur complement of the matrix system which is given by

$$S = \bar{K} M^{-1} \bar{K}^T + \frac{1}{\beta} M.$$

We therefore start by considering an accurate approximation of these two matrices. As previously in this thesis, the Chebyshev semi-iterative method is effective for approximating mass matrices, so in our preconditioners we approximate Φ by $\text{blkdiag}\left(\widehat{M}, \beta\widehat{M}\right)$, where $\widehat{M}$ denotes the application of Chebyshev semi-iteration to M.

To find an accurate approximation of the Schur complement, we apply the result of Theorem 13 below. This theorem gives us a Schur complement approximation for which the eigenvalues of the Schur complement preconditioned with this approx-

imation are bounded robustly, given positive semi-definiteness of the symmetric matrix $\nu K + T$ and skew-symmetry of the matrix N (see [36, Chapters 3 and 5] for more details), and therefore positive semi-definiteness of the symmetric part of $\bar{K}$, $H := \frac{1}{2}(\bar{K} + \bar{K}^T)$. We note that Theorem 13 is an extension of Theorem 8 proved in Chapter 3, which applied to symmetric operators rather than the non-symmetric operator $\bar{K}$ we are considering in this chapter. The stronger result presented below is highly useful, as it may be applied to a vastly wider class of problems.

Theorem 13. *Suppose that the symmetric part of $\bar{K}$, $H := \frac{1}{2}(\bar{K} + \bar{K}^T)$, is positive semi-definite. Then, if we approximate the Schur complement S by*

$$\widehat{S} = \left(\bar{K} + \frac{1}{\sqrt{\beta}}M\right) M^{-1} \left(\bar{K} + \frac{1}{\sqrt{\beta}}M\right)^T,$$

we can bound the eigenvalues of $\widehat{S}^{-1}S$ as follows:

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

Proof. We have that the eigenvalues $\bar{\mu}$ and eigenvectors $\mathbf{x}$ of $\widehat{S}^{-1}S$ satisfy:

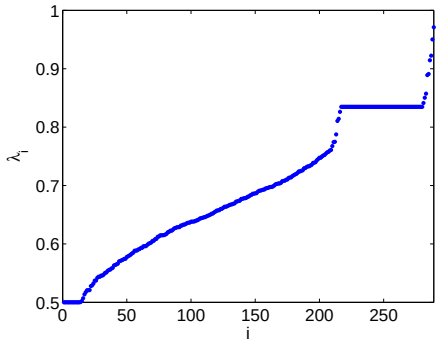
$$\widehat{S}^{-1}S\mathbf{x} = \bar{\mu}\mathbf{x} \quad \Leftrightarrow \quad \left(\beta\bar{K}M^{-1}\bar{K}^T + M\right)\mathbf{x} = \bar{\mu}\left[\beta\bar{K}M^{-1}\bar{K}^T + M + \sqrt{\beta}(\bar{K} + \bar{K}^T)\right]\mathbf{x}.$$

Note that it is sufficient to show that the Rayleigh quotient $R := \frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}\mathbf{v}} \in \left[\frac{1}{2}, 1\right]$. To show this, we write

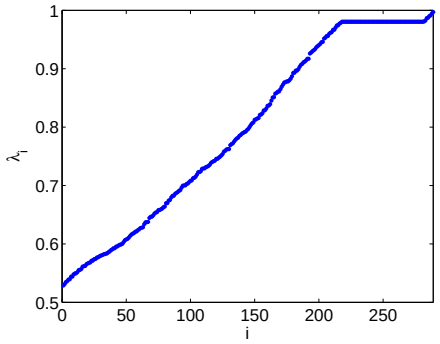
$$R = \frac{\mathbf{v}^T\left[\beta\bar{K}M^{-1}\bar{K}^T + M\right]\mathbf{v}}{\mathbf{v}^T\left[\beta\bar{K}M^{-1}\bar{K}^T + M + \sqrt{\beta}(\bar{K} + \bar{K}^T)\right]\mathbf{v}} = \frac{\mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b}}{(\mathbf{a}+\mathbf{b})^T(\mathbf{a}+\mathbf{b})},$$

where $\mathbf{a} = (\sqrt{\beta}\bar{K}M^{-1/2})^T\mathbf{v}$, $\mathbf{b} = (M^{1/2})^T\mathbf{v}$.

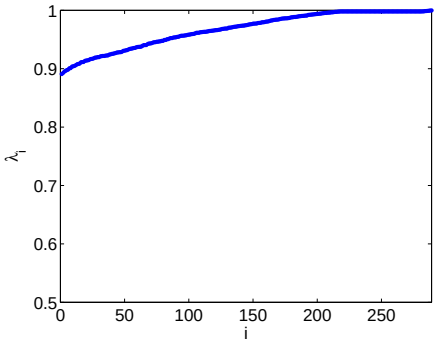
The upper bound follows immediately from the fact that $\sqrt{\beta}\mathbf{v}^T(\bar{K} + \bar{K}^T)\mathbf{v} = 2\sqrt{\beta}\mathbf{v}^T H\mathbf{v} \geq 0$ by the assumption of positive semi-definiteness of H, as well as the positivity of $\mathbf{b}^T\mathbf{b} = \mathbf{v}^T M\mathbf{v}$.



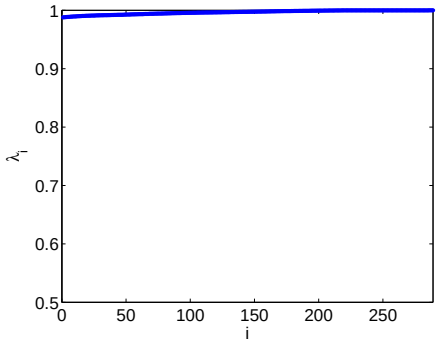
(a) $\lambda_i(\widehat{S}^{-1}S)$, $\beta = 10^{-2}$, $i = 1, ..., 289$



(b) $\lambda_i(\widehat{S}^{-1}S)$, $\beta = 10^{-4}$, $i = 1, ..., 289$



(c) $\lambda_i(\widehat{S}^{-1}S)$, $\beta = 10^{-6}$, $i = 1, ..., 289$



(d) $\lambda_i(\widehat{S}^{-1}S)$, $\beta = 10^{-8}$, $i = 1, ..., 289$

Figure 5.3: Spectra of $\widehat{S}^{-1}S$ for $\beta = 10^{-2}$, $\beta = 10^{-4}$, $\beta = 10^{-6}$ and $\beta = 10^{-8}$, for an evenly spaced grid on $\Omega = [-1, 1]^2$ with $h = 2^{-3}$, $\nu = \frac{1}{100}$ and $\mathbf{w} = \left[\sin\frac{\pi}{6}, \cos\frac{\pi}{6}\right]^T$.

To show that $R \geq \frac{1}{2}$, we proceed as follows, noting again that $\mathbf{b}^T\mathbf{b} > 0$:

$$\begin{aligned} R \geq \frac{1}{2} \quad &\Leftrightarrow \quad \mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b} \geq \frac{1}{2}\left[\mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b} + \mathbf{a}^T\mathbf{b} + \mathbf{b}^T\mathbf{a}\right] \\ &\Leftrightarrow \quad \frac{1}{2}\left[\mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b} - \mathbf{a}^T\mathbf{b} - \mathbf{b}^T\mathbf{a}\right] \geq 0 \\ &\Leftrightarrow \quad (\mathbf{a} - \mathbf{b})^T(\mathbf{a} - \mathbf{b}) \geq 0. \end{aligned}$$

As $(\mathbf{a} - \mathbf{b})^T(\mathbf{a} - \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|_2^2 \geq 0$ is clearly satisfied, the result is proved. □

Illustrations of the eigenvalue distribution of $\widehat{S}^{-1}S$ for a variety of values of β in a particular practical case are shown in Figure 5.3 – we note that the eigenvalues are particularly clustered for small values of β.

Therefore, using Theorem 13, we may obtain an effective Schur complement approximation if we can find a good way of approximating the matrices $\bar{K} + \frac{1}{\sqrt{\beta}}M$ and

$\left(\bar{K} + \frac{1}{\sqrt{\beta}}M\right)^T$. The method we use for approximating these matrices is the geometric multigrid process described for the forward problem in Section 5.1, with the coarse grid matrices formed explicitly rather than by the use of prolongation and restriction operators, and with block Gauss-Seidel smoothing.

So, as we now have good approximations of the matrices $\widehat{\Phi}$ and $\widehat{S}$, we can propose two effective preconditioners of the form

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \widehat{\Phi} & 0 \\ 0 & \widehat{S} \end{bmatrix}, \quad \widehat{\mathcal{P}}_2 = \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -\widehat{S} \end{bmatrix},$$

as described in Section 2.2.

Unlike the forward problem, the convection-diffusion control problem is symmetric with our (symmetric) stabilization, and so $\widehat{\mathcal{P}}_1$ is symmetric positive definite. Therefore, our first method for solving the matrix system (5.7) would be to apply a MINRES method with preconditioner

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \widehat{M} & 0 & 0 \\ 0 & \beta\widehat{M} & 0 \\ 0 & 0 & \widehat{S} \end{bmatrix}. \tag{5.8}$$

In our preconditioner, $\widehat{M}$ denotes 20 steps of Chebyshev semi-iteration to approximate the mass matrix M, and $\widehat{S}$ denotes the approximation to the Schur complement discussed above.

Our second method involves applying the Bramble-Pasciak Conjugate Gradient method as described in Section 2.3.5, with preconditioner

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \gamma\widehat{M} & 0 & 0 \\ 0 & \beta\gamma\widehat{M} & 0 \\ \bar{K} & -M & -\widehat{S} \end{bmatrix}, \tag{5.9}$$

and inner product given by

$$\mathcal{H} = \begin{bmatrix} M - \gamma\widehat{M} & 0 & 0 \\ 0 & \beta(M - \gamma\widehat{M}) & 0 \\ 0 & 0 & \widehat{S} \end{bmatrix},$$

where γ is a constant which can be chosen a priori to ensure that $M - \gamma\widehat{M}$ is positive

definite, as in previous chapters.

At this juncture, we make two points about our preconditioning strategy and its applicability:

1. The matrix system (5.7) for the distributed convection-diffusion control problem could potentially be reduced to the following system of equations by elimination of the discretized gradient equation:

$$\begin{bmatrix} M & \bar{K}^T \\ \bar{K} & -\frac{1}{\beta}M \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{g} \end{bmatrix}.$$

   We note that our preconditioning strategies are also valid for this problem (due to Theorems 2 and 4 of Section 2.2), as we still obtain a saddle point system of the structure discussed in Section 2.2 (except now with $\Theta \neq 0$). We will again need to implement a Chebyshev semi-iteration process to approximate M and enact the approximation of the Schur complement S, which remains the same as for the system (5.7).

2. We believe that other similar methods could be devised to solve the convection-diffusion control problem based on the framework discussed in this section. For instance, we see no reason why a preconditioner of the form

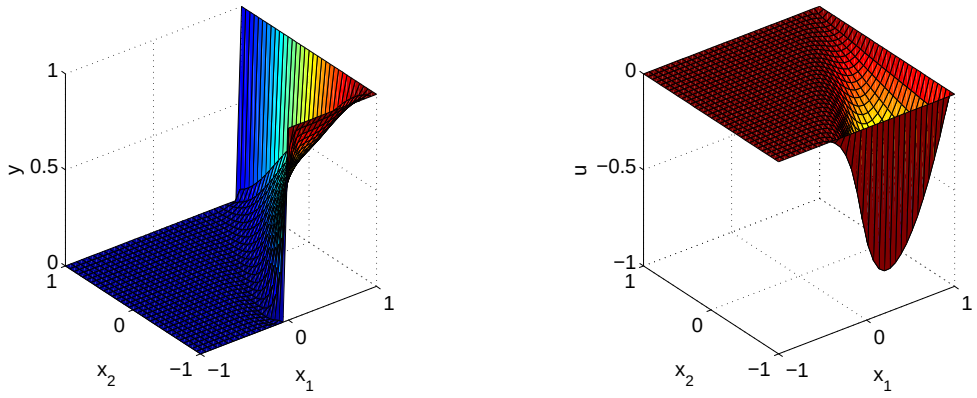
$$\widehat{\mathcal{P}}_4 = \begin{bmatrix} \widehat{\Phi} & \Psi^T \\ \Psi & \Psi\widehat{\Phi}^{-1}\Psi^T - \widehat{S} \end{bmatrix} = \begin{bmatrix} I & 0 \\ \Psi\widehat{\Phi}^{-1} & I \end{bmatrix} \begin{bmatrix} \widehat{\Phi} & \Psi^T \\ 0 & -\widehat{S} \end{bmatrix},$$

   which we discussed in the context of the Poisson control problem in Section 3.1.3, could not be applied to this problem using our approximations $\widehat{\Phi}$ and $\widehat{S}$.

5.4 Numerical Results

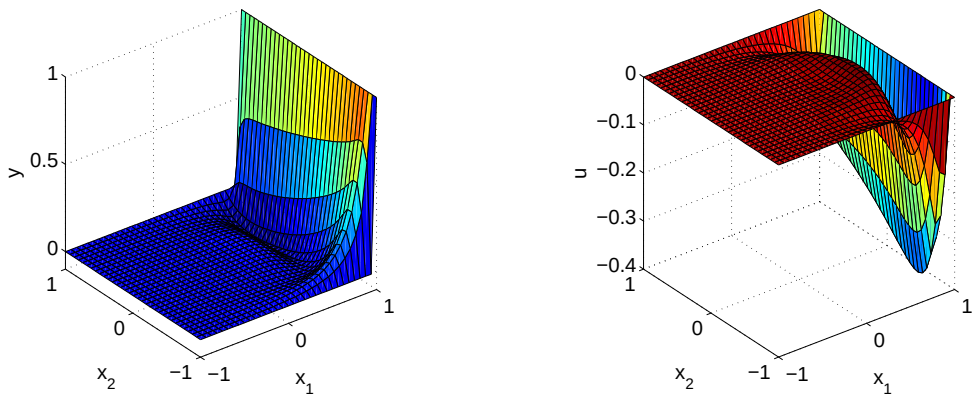
In this section, we provide numerical results to demonstrate the effectiveness of our suggested methods. In our numerical tests, we discretize the state y, control u and adjoint p using **Q1** finite element basis functions. We construct the relevant matrices for our two test problems in the same way as is done in the Incompressible Flow & Iterative Solver Software (IFISS) package [35, 110].

The two problems that we consider are stated below, with plots of their solutions shown in Figures 5.4 and 5.5 respectively.



(a) State y (b) Control u

Figure 5.4: Solutions of state and control for Problem 1 using **Q1** basis functions with $\nu = \frac{1}{100}$ and $\beta = 1$.



(a) State y (b) Control u

Figure 5.5: Solutions of state and control for Problem 2 using **Q1** basis functions with $\nu = \frac{1}{100}$ and $\beta = 1$.

$\nu = \frac{1}{250}$		MINRES				BPCG			
		β				β			
$\frac{h}{2}$	SIZE	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
2^{-2}	75	13	7	5	3	11	9	6	6
2^{-3}	243	13	9	5	3	12	10	7	6
2^{-4}	867	13	11	5	3	12	13	9	7
2^{-5}	3,267	13	12	7	3	13	14	10	7
2^{-6}	12,675	13	12	7	4	13	14	12	8
2^{-7}	49,923	12	11	9	5	13	15	15	10

Table 5.1: Number of MINRES iterations with 'ideal' block diagonal preconditioner (5.8), and Bramble-Pasciak CG iterations with 'ideal' block triangular preconditioner (5.9), needed to solve Problem 1. Results are given for a range of values of $\frac{h}{2}$ (which is equal to the inverse of the number of steps in space in each coordinate) and β, where $\nu = \frac{1}{250}$, and **Q1** basis functions are used to approximate the state, control and adjoint.

- **Problem 1:** We wish to solve the following distributed convection-diffusion control problem on $\Omega = [-1, 1]^2$:

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2} \|y\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nu \nabla^2 y + \mathbf{w} \cdot \nabla y = u, \quad \text{in } \Omega, \\ & y = \begin{cases} 1 & \text{on } \partial\Omega_1 := ([0,1] \times \{-1\}) \cup (\{1\} \times [-1,1]), \\ 0 & \text{on } \partial\Omega_2 := \partial\Omega \backslash \partial\Omega_1, \end{cases} \end{aligned}$$

  where $\mathbf{w} = \left[\sin\frac{\pi}{6}, \cos\frac{\pi}{6}\right]^T$. This is an optimal control problem involving a constant wind $\mathbf{w}$; forward problems of this form have previously been considered in literature such as [36, 99].

- **Problem 2:** We wish to solve the following distributed convection-diffusion

MINRES $\nu = \frac{1}{100}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	SIZE	ITER.	TIME	ITER.	TIME	ITER.	TIME	ITER.	TIME
2^{-2}	75	13	0.070	7	0.051	5	0.040	3	0.038
2^{-3}	243	13	0.11	9	0.092	5	0.072	3	0.063
2^{-4}	867	13	0.20	11	0.17	5	0.078	3	0.064
2^{-5}	3,267	13	0.54	12	0.50	7	0.29	3	0.23
2^{-6}	12,675	13	2.36	13	2.24	7	1.52	5	1.53
2^{-7}	49,923	13	14.1	11	12.9	9	11.1	5	8.10

MINRES $\nu = \frac{1}{500}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	SIZE	ITER.	TIME	ITER.	TIME	ITER.	TIME	ITER.	TIME
2^{-2}	75	13	0.072	7	0.054	5	0.044	3	0.038
2^{-3}	243	13	0.13	9	0.098	4	0.066	3	0.060
2^{-4}	867	13	0.27	11	0.15	5	0.084	3	0.062
2^{-5}	3,267	13	0.58	12	0.52	7	0.42	3	0.27
2^{-6}	12,675	13	2.93	12	2.73	7	1.76	4	1.21
2^{-7}	49,923	12	15.2	11	15.1	9	10.2	5	9.51

Table 5.2: Number of MINRES iterations with block diagonal preconditioner (5.8) needed to solve Problem 1, and computation times taken to do so (in seconds). Results are given for a range of values of $\frac{h}{2}$ (and hence problem size) and β, with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$, where **Q1** basis functions are used to approximate the state, control and adjoint.

control problem on $\Omega = [-1, 1]^2$:

$$\begin{aligned} \min_{y,u} \quad & \frac{1}{2} \|y\|^2_{L_2(\Omega)} + \frac{\beta}{2} \|u\|^2_{L_2(\Omega)} \\ \text{s.t.} \quad & -\nu \nabla^2 y + \mathbf{w} \cdot \nabla y = u, \quad \text{in } \Omega, \\ & y = \begin{cases} 1 & \text{on } \partial\Omega_1 := \{1\} \times [-1, 1], \\ 0 & \text{on } \partial\Omega_2 := \partial\Omega \backslash \partial\Omega_1, \end{cases} \end{aligned}$$

where $\mathbf{w} = \left[\frac{1}{2}x_2(1 - x_1^2), -\frac{1}{2}x_1(1 - x_2^2)\right]^T$, and $\mathbf{x} = [x_1, x_2]^T$ denotes the spatial coordinates. This is an optimal control formulation of the *double-glazing problem* discussed in [36, p.119.]: a model of the temperature in a cavity with

BPCG $\nu = \frac{1}{100}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	Size	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
2^{-2}	75	10	0.056	9	0.050	6	0.040	6	0.044
2^{-3}	243	12	0.11	10	0.11	7	0.084	6	0.075
2^{-4}	867	12	0.20	13	0.22	9	0.17	7	0.13
2^{-5}	3,267	13	0.60	14	0.62	10	0.46	7	0.38
2^{-6}	12,675	13	2.89	15	2.99	12	2.60	9	2.31
2^{-7}	49,923	13	14.5	15	16.0	15	15.8	11	11.6

BPCG $\nu = \frac{1}{500}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	Size	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
2^{-2}	75	11	0.057	8	0.048	6	0.047	6	0.043
2^{-3}	243	12	0.11	10	0.10	7	0.080	6	0.079
2^{-4}	867	12	0.22	13	0.22	9	0.16	7	0.14
2^{-5}	3,267	13	0.52	14	0.55	10	0.45	7	0.36
2^{-6}	12,675	13	2.91	14	2.96	12	2.68	8	2.01
2^{-7}	49,923	13	13.7	15	14.8	14	14.2	9	10.5

Table 5.3: Number of Bramble-Pasciak CG iterations with block triangular preconditioner (5.9) needed to solve Problem 1, and computation times taken to do so (in seconds). Results are given for a range of values of $\frac{h}{2}$ (and hence problem size) and β, with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$, where **Q1** basis functions are used to approximate the state, control and adjoint.

recirculating wind $\mathbf{w}$. We note that we have chosen the wind so that the maximum value of $\|\mathbf{w}\|_2$ on Ω is equal to 1.

We first provide a proof-of-concept that our proposed preconditioners are effective ones. In Table 5.1, we present iteration numbers for solving Problem 1 with $\nu = \frac{1}{250}$ and a range of h and β, using 'ideal' versions of our two preconditioners (specifically, where we invert $\bar{K} + \frac{1}{\sqrt{\beta}}M$ and its transpose using direct solves in the preconditioners, rather than using multigrid). The results shown demonstrate that in theory our preconditioners are highly potent for a range of parameters. All other results presented are thus generated using the geometric multigrid procedure previously described.

In Table 5.2, we present the number of Minres iterations and computation times

MINRES $\nu = \frac{1}{100}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	SIZE	ITER.	TIME	ITER.	TIME	ITER.	TIME	ITER.	TIME
2^{-2}	75	13	0.071	7	0.050	4	0.044	3	0.039
2^{-3}	243	15	0.13	7	0.063	4	0.061	3	0.059
2^{-4}	867	13	0.19	7	0.13	5	0.076	3	0.065
2^{-5}	3,267	13	0.52	9	0.42	5	0.32	3	0.25
2^{-6}	12,675	13	2.39	11	2.14	7	1.49	3	1.06
2^{-7}	49,923	13	13.9	11	13.2	9	10.8	5	8.32

MINRES $\nu = \frac{1}{500}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	SIZE	ITER.	TIME	ITER.	TIME	ITER.	TIME	ITER.	TIME
2^{-2}	75	15	0.074	7	0.053	5	0.041	3	0.040
2^{-3}	243	21	0.20	7	0.085	4	0.071	3	0.060
2^{-4}	867	19	0.35	9	0.17	5	0.085	3	0.064
2^{-5}	3,267	12	0.55	9	0.47	5	0.33	3	0.28
2^{-6}	12,675	12	2.81	9	2.34	5	2.10	3	1.17
2^{-7}	49,923	12	15.4	11	14.7	5	8.92	3	7.71

Table 5.4: Number of MINRES iterations with block diagonal preconditioner (5.8) needed to solve Problem 2, and computation times taken to do so (in seconds). Results are given for a range of values of $\frac{h}{2}$ (and hence problem size) and β, with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$, where **Q1** basis functions are used to approximate the state, control and adjoint.

(including the time taken to construct the relevant matrices on sub-grids) required to solve Problem 1 with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$, using preconditioner $\widehat{\mathcal{P}}_1$, to a tolerance of 10^{-6}. In Table 5.3 we show how many Bramble-Pasciak CG iterations are required to solve the same problem to the same tolerance with preconditioner $\widehat{\mathcal{P}}_2$ and with $\gamma = 0.95$. We observe that both our solvers generate convergence in a small number of iterations for both values of viscosity. The convergence rate actually improves as β decreases, probably because our Schur complement approximation becomes better for smaller β, as illustrated by Figure 5.3. Although we take the wind $\mathbf{w} = \left[\sin\frac{\pi}{6}, \cos\frac{\pi}{6}\right]^T$ and specific values of ν, we find, in other computations not presented here, that the results were similar for any constant wind with vector 2-norm equal to 1, for a wide

BPCG $\nu = \frac{1}{100}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	Size	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
2^{-2}	75	10	0.056	7	0.050	6	0.040	6	0.044
2^{-3}	243	12	0.10	8	0.097	6	0.078	6	0.077
2^{-4}	867	12	0.19	10	0.18	7	0.14	6	0.12
2^{-5}	3,267	13	0.58	12	0.52	9	0.44	7	0.38
2^{-6}	12,675	13	2.93	15	3.02	11	2.38	8	2.10
2^{-7}	49,923	13	14.2	15	15.6	15	15.5	10	10.4

BPCG $\nu = \frac{1}{500}$		β							
		10^{-2}		10^{-4}		10^{-6}		10^{-8}	
$\frac{h}{2}$	Size	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
2^{-2}	75	12	0.061	7	0.046	6	0.045	6	0.043
2^{-3}	243	16	0.13	8	0.091	6	0.071	6	0.075
2^{-4}	867	17	0.25	9	0.16	7	0.13	6	0.13
2^{-5}	3,267	13	0.54	11	0.45	7	0.38	6	0.34
2^{-6}	12,675	13	2.86	13	2.88	9	2.28	7	1.85
2^{-7}	49,923	13	13.6	15	15.4	11	12.7	7	9.14

Table 5.5: Number of Bramble-Pasciak CG iterations with block triangular preconditioner (5.9) needed to solve Problem 2, and computation times taken to do so (in seconds). Results are given for a range of values of $\frac{h}{2}$ (and hence problem size) and β, with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$, where **Q1** basis functions are used to approximate the state, control and adjoint.

range of ν. We note that altering the boundary conditions or target function $\widehat{y}$ would not change the matrix within the system being solved, so our solvers seem to be very robust for problems involving constant winds and values of β which are of computational interest.

In Table 5.4, we present the number of preconditioned Minres iterations and CPU times required to solve Problem 2, a harder problem, to the same tolerance, with $\nu = \frac{1}{100}$ and $\nu = \frac{1}{500}$; the number of preconditioned Bramble-Pasciak CG iterations required to solve this problem is shown in Table 5.5. Once more, for this problem and a wide range of values of β, our solvers are effective, with convergence achieved in a very small number of iterations.

We can see that the MINRES and Bramble-Pasciak CG methods are very competitive, and results for both methods are similar. Whereas MINRES tends to converge in fewer iterations, the Bramble-Pasciak CG method is computationally cheaper for a fixed number of iterations. We note that the computation times for Bramble-Pasciak CG seem to be better for larger β (in particular for smaller h), and that the MINRES solver works better for smaller β due to the lower iteration numbers. However the analysis of Section 5.3 and these results demonstrate that the iteration count should be bounded by a low number for these problems when either method is used, for any choice of h and β.

The results in this section have demonstrated that the solvers we have proposed are potent ones for a number of convection-diffusion control problems, a class of problems which, as for the convection-diffusion equation itself, is fraught with numerical difficulties. The number of iterations required to solve these problems is small, and the convergence of the solvers improves rather than degrades as β is decreased. As is observable in the computation times shown in Tables 5.2–5.5, the convergence is close to linear with respect to the size of the matrix system – we find that the only component of our solvers that does not scale linearly in time is the construction of matrices on the sub-grids.

5.5 Summary

In this chapter, we have explored preconditioned iterative methods for convection-diffusion control problems, a more complex and general class of problems than the Poisson control problems considered up to this point. In order to tackle these problems, we first needed to introduce an appropriate stabilization technique to ensure that the discretize-then-optimize and optimize-then-discretize approaches resulted in the same matrix system.

We then found that extensions of the preconditioning strategies introduced for Poisson control problems could in fact be applied to these harder problems – the major additional step was proving a stronger result on the approximation of the Schur complement of the matrix system. Having done this, as well as implementing an appropriate multigrid routine to enact our Schur complement approximation, we were able to motivate two solvers for convection-diffusion control problems – one block diagonal and one block triangular. We presented numerical results which validated our strategy. We note that our solution methods worked well whether SUPG or LPS

stabilization was used, and were also effective when no stabilization was applied at all.

We conclude that our approach for approximating Schur complements of PDE-constrained optimization problems is in fact valid for a wider class of problems than we originally envisaged. In future chapters, we will build on the solvers derived so far, in order to motivate iterative methods for harder optimal control problems arising in fluid dynamics.

CHAPTER 6

Stokes Control

In this chapter, we consider the iterative solution of the (distributed) Stokes control problem, an important optimal control problem in the field of fluid dynamics.[24] There has been much previous work on iterative methods for this problem. For instance, in [104] solvers for the time-independent problem were derived that were independent of the mesh-size used, and in [115] this methodology was extended to the time-dependent problem. In [66], a multigrid solver was developed for the time-dependent problem that also exhibited mesh-independence. In addition, in [129], Zulehner derived a solver for the time-independent problem that exhibited independence with respect to both the mesh-size and the regularization parameter used. We wish to see whether we may use the theory introduced so far in this thesis to derive methods for this important problem.

In this chapter, we consider the development of (block diagonal and block triangular) preconditioners for a time-independent Stokes control problem, by using the saddle point theory of Section 2.2, as well as applying commutator arguments to the matrix system, the concept of which we will describe. We find this work to be useful for a number of reasons. For one, it enables us to re-derive the block diago-

[24] This chapter is based on the following paper, which is Ref. [86]:

J. W. Pearson, *On the Role of Commutator Arguments in the Development of Parameter-Robust Preconditioners for Stokes Control Problems*, submitted to Electronic Transactions on Numerical Analysis, 2013.

nal preconditioner of Zulehner [129]; we may also use this theory to motivate a new block diagonal preconditioner, as well as two block triangular preconditioners, for this problem. Furthermore, as discussed in Chapter 7, we are able to extend some of the methodology of this chapter to the harder Navier-Stokes control problem – this is perhaps the most interesting aspect of the work introduced here.

This chapter is structured as follows. We introduce the Stokes control problem that we seek to solve in Section 6.1. In Section 6.1.1, we derive 2 block diagonal preconditioners for this problem, and in Section 6.1.2, we use our methodology to motivate 2 block triangular preconditioners. In Section 6.1.3, we discuss the effectiveness of the new commutator arguments we introduce, and outline the main operations required to apply our preconditioners. In Section 6.2 we carry out numerical tests on our preconditioners, and finally in Section 6.3 we make some concluding comments.

6.1 Preconditioners for the Stokes Control Problem

The problem that we wish to consider in this chapter is the following distributed Stokes control problem:

$$\begin{aligned} \min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}-\underline{\widehat{\mathbf{v}}}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \ & -\nabla^2\underline{\mathbf{v}}+\nabla p=\underline{\mathbf{u}}, \quad \text{in } \Omega, \\ & -\nabla\cdot\underline{\mathbf{v}}=0, \quad \text{in } \Omega, \\ & \underline{\mathbf{v}}=\underline{\mathbf{g}}, \quad \text{on } \partial\Omega. \end{aligned}$$

Again we work on a domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$, with boundary $\partial\Omega$, and with regularization parameter β. Here, $\underline{\mathbf{v}}$ denotes the velocity in d dimensions and p the pressure field, both of which are state variables in this problem. $\underline{\mathbf{u}}$ here is the control variable in d dimensions. We also introduce at this point the adjoint variables $\underline{\boldsymbol{\lambda}}$ (in d dimensions) and μ. The Stokes equations, on which this optimal control problem is based, are a fundamental system of equations in the field of fluid dynamics: in particular they describe slow incompressible flow.

We note that, in the construction of the functional being minimized in this optimal control problem, we have not regularized the pressure term – the problem where

pressure is regularized was considered in [104, 115] for instance, and solvers for problems of this form undoubtedly have a big role to play when considering the iterative solution of Stokes control problems in general.

In order to arrive at a matrix system for the problem stated above, we may either employ a discretize-then-optimize or optimize-then-discretize approach. The discretize-then-optimize strategy in this case involves differentiating a discrete cost functional of the form

$$\mathcal{L}_{DTO} = \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} - \mathbf{v}^T\mathbf{z} + \frac{\beta}{2}\mathbf{u}^T\mathbf{M}\mathbf{u} + \boldsymbol{\lambda}^T(\mathbf{K}\mathbf{v} + B^T\mathbf{p} - \mathbf{M}\mathbf{u} - \mathbf{g}) + \boldsymbol{\mu}^T(B\mathbf{v}),$$

with respect to the discrete variables $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{p}$. Here, $\mathbf{M}$ and $\mathbf{K}$ denote $d \times d$ block matrices with mass and stiffness matrices on the velocity space on the block diagonals, and B represents the negative of the divergence operator on the finite element space in matrix form. The vector $\mathbf{z}$ corresponds to the target function $\widehat{\underline{\mathbf{v}}}$, and $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ relate to the adjoint variables $\underline{\boldsymbol{\lambda}}$, μ.

Alternatively, an optimize-then-discretize method involves differentiating a continuous cost functional of the form

$$\begin{aligned}\mathcal{L}_{OTD} = \frac{1}{2}\left\|\underline{\mathbf{v}} - \widehat{\underline{\mathbf{v}}}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} + \int_\Omega (-\nabla^2\underline{\mathbf{v}} + \nabla p - \underline{\mathbf{u}})\underline{\boldsymbol{\lambda}}_\Omega \,\mathrm{d}\Omega \\ + \int_{\partial\Omega} (\underline{\mathbf{v}} - \underline{\mathbf{g}})\underline{\boldsymbol{\lambda}}_{\partial\Omega} \,\mathrm{d}s + \int_\Omega (-\nabla\cdot\underline{\mathbf{u}})\mu \,\mathrm{d}\Omega,\end{aligned}$$

with respect to the continuous variables (where $\underline{\boldsymbol{\lambda}}$ is split into interior and boundary components $\underline{\boldsymbol{\lambda}}_\Omega$ and $\underline{\boldsymbol{\lambda}}_{\partial\Omega}$), and then discretizing the equations.

Applying either method (and eliminating the gradient equation $\beta\mathbf{u} - \boldsymbol{\lambda} = \mathbf{0}$) results in the following matrix system [129]:

$$\begin{bmatrix} \mathbf{M} & 0 & \mathbf{K} & B^T \\ 0 & 0 & B & 0 \\ \mathbf{K} & B^T & -\frac{1}{\beta}\mathbf{M} & 0 \\ B & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{p} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu} \end{bmatrix} = \begin{bmatrix} \mathbf{z} + \mathbf{g}_1 \\ \mathbf{0} \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix}, \tag{6.1}$$

where the vectors $\mathbf{g}_1$, $\mathbf{g}_2$ and $\mathbf{g}_3$ take account of boundary conditions. We note at this point that this matrix is in general singular, as it is well known that the vector of ones is a member of the nullspace of B^T (see [36, Chapter 5] for instance) – the matrix in (6.1) therefore has 2 zero eigenvalues (one corresponding to each appearance of B^T).

On the continuous level, the zero eigenvalues arise from the fact that an arbitrary constant may be added to the solution of the pressure p or the adjoint variable μ and yield another solution. However, the presence of these eigenvalues may be avoided by restricting the pressure space to the orthogonal complement of the nullspace, as in this case the matrix B^T will clearly no longer have a nullspace.

We note that the discrete representation of the forward problem occurs in the bottom left of the matrix system (6.1), with the adjoint problem in the top right and the terms arising from the cost functional the minimum of which we seek on the block diagonal.

We consider discretizing this problem using the well-studied (inf-sup stable) *Taylor-Hood* finite element basis functions, that is discretizing the velocity $\underline{\mathbf{v}}$ using **Q2** basis functions, and the pressure p using **Q1** basis functions. We discretize the control $\underline{\mathbf{u}}$ and adjoint variable $\underline{\boldsymbol{\lambda}}$ using **Q2** functions, and the adjoint variable μ using **Q1** functions.

It is not immediately obvious how the preconditioners derived for the Poisson control problem in Chapter 3 can be applied to the more difficult Stokes control problem. Specifically, the appearance of a saddle point system within the PDE constraint (as opposed to a single matrix which is symmetric positive definite) greatly increases the complexity of the problem. In this section, we discover that we may in fact apply preconditioning strategies for the Poisson control problem to the harder Stokes control problem, and we explain how this can be done.

6.1.1 Motivation of Block Diagonal Preconditioners

To commence our derivation, we reorder the matrix system (6.1) so that we are dealing with the system

$$\underbrace{\begin{bmatrix} \mathbf{M} & \mathbf{K} & B^T & 0 \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} & 0 & B^T \\ B & 0 & 0 & 0 \\ 0 & B & 0 & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} + \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \\ \mathbf{0} \end{bmatrix}.$$

This is a saddle point system of the form (2.31), with

$$\Phi = \begin{bmatrix} \mathbf{M} & \mathbf{K} \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}, \quad \Psi = \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Note that the $(1,1)$-block Φ is of the form of the matrix system (2.8) relating to the Poisson control problem. We will use this to motivate two block diagonal preconditioners, related to two preconditioners for Poisson control detailed in Chapter 3. These preconditioners will be of the form

$$\begin{bmatrix} \widehat{\Phi} & 0 \\ 0 & (\Psi\Phi^{-1}\Psi^T)_{\text{approx}} \end{bmatrix}. \tag{6.2}$$

Such a strategy also leads to block triangular preconditioners of the form

$$\begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & (\Psi\Phi^{-1}\Psi^T)_{\text{approx}} \end{bmatrix} \text{ or } \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -(\Psi\Phi^{-1}\Psi^T)_{\text{approx}} \end{bmatrix}. \tag{6.3}$$

We will derive two such block triangular preconditioners in Section 6.1.2.

First Block Diagonal Preconditioner

We motivate our first preconditioner for the Stokes control system (6.1) using a preconditioner derived by Zulehner [129] for the Poisson control problem, as discussed in Chapter 3. We first note that the $(1,1)$-block of the Stokes control problem (6.1) is of the form of the matrix involved in the Poisson control problem, so we write, in the notation of (6.2),

$$\Phi = \begin{bmatrix} \mathbf{M} & \mathbf{K} \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix} \approx \begin{bmatrix} \mathbf{M} + \sqrt{\beta}\mathbf{K} & 0 \\ 0 & \frac{1}{\beta}(\mathbf{M} + \sqrt{\beta}\mathbf{K}) \end{bmatrix} =: \widehat{\Phi}.$$

Here, the statement $\Phi \approx \widehat{\Phi}$ means that $\widehat{\Phi}$ has been selected with the aim that the singular values of $\widehat{\Phi}^{-1}\Phi$ are all contained within a fixed (small) interval that is robust with respect to mesh-size and regularization parameter.

The next step is to find a potentially good approximation to the Schur complement

$\Psi\Phi^{-1}\Psi^T$ of $\mathcal{A}$; we proceed by writing

$$\begin{aligned}\Psi\Phi^{-1}\Psi^T &= \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & \mathbf{K} \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\ &\approx \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M}+\sqrt{\beta}\mathbf{K} & 0 \\ 0 & \frac{1}{\beta}(\mathbf{M}+\sqrt{\beta}\mathbf{K}) \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} =: \Psi\widehat{\Phi}^{-1}\Psi^T \\ &= \begin{bmatrix} B(\mathbf{M}+\sqrt{\beta}\mathbf{K})^{-1}B^T & 0 \\ 0 & \beta B(\mathbf{M}+\sqrt{\beta}\mathbf{K})^{-1}B^T \end{bmatrix}.\end{aligned}$$

We therefore now seek an idea for approximating $\widetilde{\Sigma} := B(\mathbf{M}+\sqrt{\beta}\mathbf{K})^{-1}B^T$, so that we obtain a cheap and invertible approximation to the Schur complement. We do this using a commutator argument, a type of which is described in [36, Chapter 8] for the Navier-Stokes equations for instance. We examine the commutator

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

where $\widetilde{\mathcal{L}} = -\sqrt{\beta}\Delta + I := -\sqrt{\beta}\nabla^2 + I$; this is an operator carefully chosen to give us a matrix that we can use to approximate $\widetilde{\Sigma}$. In the continuous setting, the commutator $\mathcal{E}$ should be small – the foundation of the commutator argument we employ is the assumption that the discretized version of this commutator is also small.

Discretizing the commutator $\mathcal{E}$ using finite elements gives

$$\mathcal{E}_h = (\mathbf{M}^{-1}\mathbf{L})\mathbf{M}^{-1}B^T - \mathbf{M}^{-1}B^T(M_p^{-1}L_p),$$

where $\mathbf{L} = \mathbf{M} + \sqrt{\beta}\mathbf{K}$ and $L_p = M_p + \sqrt{\beta}K_p$. Pre-multiplying by $B\mathbf{L}^{-1}\mathbf{M}$ and post-multiplying by $L_p^{-1}M_p$ then gives

$$B\mathbf{M}^{-1}B^TL_p^{-1}M_p \approx B\mathbf{L}^{-1}B^T,$$

using the assumption that $\mathcal{E}_h$ is small.

We are faced in the above expression with the non-trivial matrix term $B\mathbf{M}^{-1}B^T$ – this term appears frequently in our working. However, it is known that $B\mathbf{M}^{-1}B^T$ may be well approximated by K_p (see [36, Chapter 8]). This may be motivated as follows: the matrices B and B^T represent the negative of the divergence operator $-\nabla\cdot$ and the gradient operator ∇ respectively, on the continuous space. The mass matrix

corresponds to the identity operator, so $B\mathbf{M}^{-1}B^T$ corresponds to $-\nabla\cdot\nabla = -\nabla^2$, which in turn relates to the stiffness matrix K_p.

We may therefore use the approximation $B\mathbf{M}^{-1}B^T \approx K_p$ and substitute in the expression for $\mathbf{L}$ to give that[25]

$$\widetilde{\Sigma} = B(\mathbf{M}+\sqrt{\beta}\mathbf{K})^{-1}B^T \approx K_p L_p^{-1} M_p,$$

and therefore that

$$\widetilde{\Sigma}^{-1} \approx M_p^{-1}L_pK_p^{-1} = M_p^{-1}(M_p+\sqrt{\beta}K_p)K_p^{-1} = \sqrt{\beta}M_p^{-1}+K_p^{-1}.$$

We note that such a motivational argument has been used before, in order to derive preconditioners for a range of fluid dynamics problems [25, 36].

We may use the result of the above argument to write that

$$\Psi\Phi^{-1}\Psi^T \approx \begin{bmatrix} (\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} & 0 \\ 0 & \beta(\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} \end{bmatrix} =: (\Psi\Phi^{-1}\Psi^T)_{\text{approx}}.$$

Therefore, putting all of the above working together, we postulate that

$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \mathbf{M}+\sqrt{\beta}\mathbf{K} & 0 & 0 & 0 \\ 0 & \frac{1}{\beta}(\mathbf{M}+\sqrt{\beta}\mathbf{K}) & 0 & 0 \\ 0 & 0 & (\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} & 0 \\ 0 & 0 & 0 & \beta(\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} \end{bmatrix}$$

might be an effective preconditioner for $\mathcal{A}$. We note that it is of course the action of $\widehat{\mathcal{P}}_1^{-1}$ that is needed within our solver.

This is exactly the preconditioner proposed by Zulehner in [129] using a non-standard norm argument. We will demonstrate the effectiveness of this preconditioner by displaying numerical results in Section 6.2.

Second Block Diagonal Preconditioner

We are also able to derive a new block diagonal preconditioner for the Stokes control system (6.1) using the new block diagonal preconditioner for the Poisson control

[25] An approximation of the form $B\mathbf{L}^{-1}B^T \approx K_pL_p^{-1}M_p$ was first introduced by Cahouet and Chabard in [25] for the forward Stokes problem. Such arguments have since been used to develop iterative solvers for a variety of fluid dynamics problems.

problem derived in Chapter 3. We treat the $(1,1)$-block of the Stokes control system by using the preconditioner for Poisson control, writing (in the notation of (6.2))

$$\Phi = \begin{bmatrix} \mathbf{M} & \mathbf{K} \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix} \approx \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M} \end{bmatrix}$$
$$\approx \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right) \end{bmatrix} =: \widehat{\Phi}.$$

We now once again search for a potentially good approximation to the Schur complement – we proceed as follows:

$$\Psi\Phi^{-1}\Psi^T \approx \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix}$$
$$= \begin{bmatrix} B\mathbf{M}^{-1}B^T & 0 \\ 0 & B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T \end{bmatrix}.$$

We do not as yet have a feasible preconditioner, as the matrices $B\mathbf{M}^{-1}B^T$ and $B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T$ cannot be inverted without computing the inverses of $\mathbf{M}$ or $\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}$. However, we may use once again that $B\mathbf{M}^{-1}B^T$ may be well approximated by K_p.

The crucial task is now to approximate the matrix $B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T$, so that this approximation may be fed into our preconditioner. We again apply a commutator argument, examining the term

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

and assume that its discretized version is small. For this approximation, we take $\widetilde{\mathcal{L}} = \Delta^2 + \frac{1}{\beta}I = \nabla^4 + \frac{1}{\beta}I$.

Applying the same working as above gives once again that

$$B\mathbf{L}^{-1}B^T \approx K_p L_p^{-1} M_p,$$

where $\mathbf{L} = \mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}$ and $L_p = K_p M_p^{-1} K_p + \frac{1}{\beta} M_p$. This tells us that

$$\begin{aligned}\left[B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M}\right)^{-1} B^T\right]^{-1} &\approx M_p^{-1}\left(K_p M_p^{-1} K_p + \frac{1}{\beta} M_p\right) K_p^{-1} \\ &= M_p^{-1} K_p M_p^{-1} + \frac{1}{\beta} K_p^{-1}.\end{aligned}$$

So, combining our working, we have that a second possible preconditioner for $\mathcal{A}$ is

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \mathbf{M} & 0 & 0 & 0 \\ 0 & \left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right) & 0 & 0 \\ 0 & 0 & K_p & 0 \\ 0 & 0 & 0 & \left(M_p^{-1} K_p M_p^{-1} + \frac{1}{\beta} K_p^{-1}\right)^{-1} \end{bmatrix},$$

which we postulate may be an effective preconditioner. Again, the action of the inverse of this preconditioner is required within an iterative method. We verify the potency of this preconditioner with numerical experiments presented in Section 6.2.

We note at this point that the preconditioner $\widehat{\mathcal{P}}_2$ is more "flexible". We will find that a preconditioner of this form may be applied to the more difficult and general Navier-Stokes control problem – this will be demonstrated in the next chapter.

6.1.2 Block Triangular Preconditioners

A further useful aspect of our approach is that we may consider developing robust preconditioners for the Stokes control problem that are not of the block diagonal form of $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$. We do this by considering various block triangular preconditioners of the Poisson control matrix system.

Firstly, we may consider a preconditioner of the form $\begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & (\Psi\Phi^{-1}\Psi^T)_{\text{approx}} \end{bmatrix}$ stated in (6.3) that is in some sense analogous to $\widehat{\mathcal{P}}_1$ as derived in Section 6.1.1. We could in fact consider the same approximations $\widehat{\Phi}$ and $(\Psi\Phi^{-1}\Psi^T)_{\text{approx}}$ as we did to construct $\widehat{\mathcal{P}}_1$,

$$\widehat{\Phi} = \begin{bmatrix} \mathbf{M}+\sqrt{\beta}\mathbf{K} & 0 \\ 0 & \frac{1}{\beta}(\mathbf{M}+\sqrt{\beta}\mathbf{K}) \end{bmatrix},$$
$$(\Psi\Phi^{-1}\Psi^T)_{\text{approx}} = \begin{bmatrix} (\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} & 0 \\ 0 & \beta(\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} \end{bmatrix},$$

to develop the following block triangular preconditioner for $\mathcal{A}$:

$$\widehat{\mathcal{P}}_3 = \begin{bmatrix} \mathbf{M}+\sqrt{\beta}\mathbf{K} & 0 & 0 & 0 \\ 0 & \frac{1}{\beta}(\mathbf{M}+\sqrt{\beta}\mathbf{K}) & 0 & 0 \\ B & 0 & (\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} & 0 \\ 0 & B & 0 & \beta(\sqrt{\beta}M_p^{-1}+K_p^{-1})^{-1} \end{bmatrix},$$

which may be applied within the GMRES algorithm.

In addition to this preconditioner, we may form a block lower triangular preconditioner for the Stokes control problem that is based on the following block triangular preconditioner for the Poisson control problem:

$$\begin{bmatrix} M & 0 \\ K & -\left(K+\frac{1}{\sqrt{\beta}}M\right)M^{-1}\left(K+\frac{1}{\sqrt{\beta}}M\right) \end{bmatrix},$$

which we showed to be effective for that problem in Chapter 3. We may, once again, use this as an approximation to the $(1,1)$-block of the Stokes control matrix $\mathcal{A}$.

Let us consider how we may precondition the Schur complement of $\mathcal{A}$ while using

this approximation of the $(1,1)$-block. We write, in the notation of (6.3),

$$\begin{aligned}
\Psi\Phi^{-1}\Psi^T &= \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & \mathbf{K} \\ \mathbf{K} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\
&\approx \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & 0 \\ \mathbf{K} & -\widehat{S}_P \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\
&= \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M}^{-1} & 0 \\ \widehat{S}_P^{-1}\mathbf{K}\mathbf{M}^{-1} & -\widehat{S}_P^{-1} \end{bmatrix} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} =: \Psi\widehat{\Phi}^{-1}\Psi^T \\
&= \begin{bmatrix} B\mathbf{M}^{-1}B^T & 0 \\ B\widehat{S}_P^{-1}\mathbf{K}\mathbf{M}^{-1}B^T & -B\widehat{S}_P^{-1}B^T \end{bmatrix} \\
&\approx \begin{bmatrix} K_p & 0 \\ B\widehat{S}_P^{-1}\mathbf{K}\mathbf{M}^{-1}B^T & -\left(M_p^{-1}K_pM_p^{-1} + \frac{1}{\beta}K_p^{-1} + \frac{2}{\sqrt{\beta}}M_p^{-1}\right)^{-1} \end{bmatrix} \\
&=: (\Psi\Phi^{-1}\Psi^T)_{\text{approx}},
\end{aligned}$$

where

$$\widehat{S}_P = \left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right).$$

In the working above, we have again used the approximation $B\mathbf{M}^{-1}B^T \approx K_p$. To approximate the matrix $B\widehat{S}_P^{-1}B^T$, we have used the same commutator argument as in Section 6.1.1 except with $\mathbf{L} = \widehat{S}_P = \mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \frac{1}{\beta}\mathbf{M} + \frac{2}{\sqrt{\beta}}\mathbf{K}$ and $L_p = K_pM_p^{-1}K_p + \frac{1}{\beta}M_p + \frac{2}{\sqrt{\beta}}K_p$.

Therefore, applying the (block triangular) saddle point theory of Section 2.2, we have arrived at a block triangular preconditioner for $\mathcal{A}$, namely:

$$\widehat{\mathcal{P}}_4 = \begin{bmatrix} \mathbf{M} & 0 & 0 & 0 \\ \mathbf{K} & -\widehat{S}_P & 0 & 0 \\ B & 0 & K_p & 0 \\ 0 & B & B\widehat{S}_P^{-1}\mathbf{K}\mathbf{M}^{-1}B^T & -\left(M_p^{-1}K_pM_p^{-1} + \frac{1}{\beta}K_p^{-1} + \frac{2}{\sqrt{\beta}}M_p^{-1}\right)^{-1} \end{bmatrix}.$$

Of course, we would not be able to apply the MINRES algorithm with the preconditioners $\widehat{\mathcal{P}}_3$ or $\widehat{\mathcal{P}}_4$; instead we would use the GMRES algorithm of [108]. Numerical tests indicate that $\widehat{\mathcal{P}}_3$ and $\widehat{\mathcal{P}}_4$ are also effective preconditioners for $\mathcal{A}$ when applied within this algorithm – we refer to Section 6.2 for a demonstration of this assertion.

6.1.3 Further Comments

We now wish to make some further observations about the preconditioners which we have proposed. Firstly, it is natural to consider the effectiveness of the new commutator arguments we have introduced, as such arguments are heuristic in nature. We therefore carry out numerical tests on our approximations; in particular we look for the maximum and minimum (non-zero) eigenvalues of

$$\left[M_p^{-1}K_pM_p^{-1}+\frac{1}{\beta}K_p^{-1}\right]B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K}+\frac{1}{\beta}\mathbf{M}\right)^{-1}B^T, \tag{6.4}$$

$$\left[M_p^{-1}K_pM_p^{-1}+\frac{1}{\beta}K_p^{-1}+\frac{2}{\sqrt{\beta}}M_p^{-1}\right]B\left(\mathbf{K}\mathbf{M}^{-1}\mathbf{K}+\frac{1}{\beta}\mathbf{M}+\frac{2}{\sqrt{\beta}}\mathbf{K}\right)^{-1}B^T, \tag{6.5}$$

which relate to the two new commutator arguments introduced in this chapter, and which are utilized in the preconditioners $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_4$ respectively. In Table 6.1, we provide eigenvalues of the matrix (6.4) for a range of mesh-sizes and values of β, and in Table 6.2, we present the same results for (6.5). For the results in both tables, an evenly spaced grid with Taylor-Hood elements was used, with the values of h stated corresponding to the distance between **Q2** nodes. We can see that the approximations are effective ones for a range of parameter values, especially for smaller values of β. We note a benign dependence of the effectiveness of the approximations on h, but the eigenvalue ranges obtained are all relatively tight, apart from when β is very large (in which case the minimum eigenvalue is small). In this case, numerical tests indicate that the rate at which λ_2 approaches zero slows down as h decreases, and the iteration numbers obtained in Section 6.2, using the approximations (6.4) and (6.5), are very

		β							
		10		10^{-2}		10^{-5}		10^{-8}	
		λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$
h	2^{-2}	0.0584	1.3315	0.1271	1.2617	0.4537	0.9776	0.4975	1.0096
	2^{-3}	0.0400	1.3495	0.0843	1.3245	0.2988	0.9591	0.5000	1.0090
	2^{-4}	0.0295	1.3730	0.0560	1.3560	0.1721	1.1442	0.4876	0.9994
	2^{-5}	0.0227	1.3645	0.0396	1.3624	0.1065	1.2964	0.3872	0.9968

Table 6.1: Maximum and minimum (non-zero) eigenvalues for commutator approximation (6.4) used in block diagonal preconditioner, for different values of h and β.

		β							
		10		10^{-2}		10^{-5}		10^{-8}	
		λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$	λ_2	$\lambda_{\max}$
h	2^{-2}	0.0653	1.3211	0.1541	1.1475	0.3922	0.9171	0.4924	1.0026
	2^{-3}	0.0446	1.3443	0.1048	1.2563	0.2881	0.9550	0.4812	0.9839
	2^{-4}	0.0326	1.3694	0.0699	1.3167	0.1951	1.0707	0.4355	0.9876
	2^{-5}	0.0249	1.3633	0.0487	1.3466	0.1294	1.2051	0.3418	0.9968

Table 6.2: Maximum and minimum (non-zero) eigenvalues for commutator approximation (6.5) used in block triangular preconditioner, for different values of h and β.

reasonable for all values of h and β tested.

Another pertinent question is how cheap it is to apply our proposed preconditioners. We therefore detail the main computational operations required to approximate $\widehat{\mathcal{P}}_1^{-1}$, $\widehat{\mathcal{P}}_2^{-1}$, $\widehat{\mathcal{P}}_3^{-1}$ and $\widehat{\mathcal{P}}_4^{-1}$ (excluding matrix multiplications, which are comparatively cheap). For the purposes of these descriptions, we view the preconditioners as 4×4 block matrices, and refer to each block as such.

- **Operations needed to apply $\widehat{\mathcal{P}}_1^{-1}$:**
  - $(\mathbf{1},\mathbf{1})$: 1 multigrid operation for $\mathbf{M} + \sqrt{\beta}\mathbf{K}$
  - $(\mathbf{2},\mathbf{2})$: 1 multigrid operation for $\mathbf{M} + \sqrt{\beta}\mathbf{K}$
  - $(\mathbf{3},\mathbf{3})$: 1 Chebyshev semi-iteration for M_p, and 1 multigrid operation for K_p
  - $(\mathbf{4},\mathbf{4})$: 1 Chebyshev semi-iteration for M_p, and 1 multigrid operation for K_p
  - **Total**: 2 Chebyshev semi-iterations and 4 multigrids.

- **Operations needed to apply $\widehat{\mathcal{P}}_2^{-1}$:**
  - $(\mathbf{1},\mathbf{1})$: 1 Chebyshev semi-iteration for $\mathbf{M}$
  - $(\mathbf{2},\mathbf{2})$: 2 multigrid operations for $\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}$
  - $(\mathbf{3},\mathbf{3})$: 1 multigrid operation for K_p

  - $(\mathbf{4}, \mathbf{4})$: 2 Chebyshev semi-iterations for M_p, and 1 multigrid operation for K_p
  - **Total**: 3 Chebyshev semi-iterations and 4 multigrids.

- **Operations needed to apply $\widehat{\mathcal{P}}_{\mathbf{3}}^{-\mathbf{1}}$:** These are the same as for $\widehat{\mathcal{P}}_1^{-1}$, and hence total:
  - **Total**: 2 Chebyshev semi-iterations and 4 multigrids.

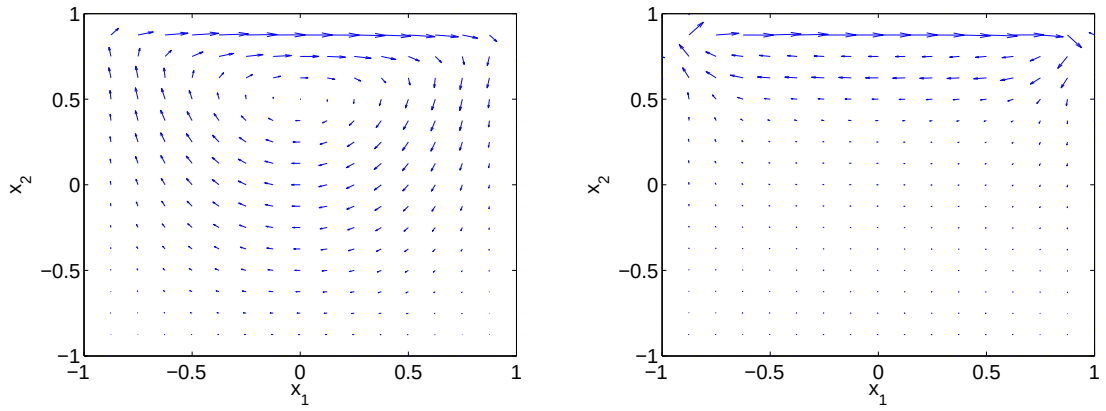
- **Operations needed to apply $\widehat{\mathcal{P}}_{\mathbf{4}}^{-\mathbf{1}}$:**
  - $(\mathbf{1}, \mathbf{1})$: 1 Chebyshev semi-iteration for $\mathbf{M}$
  - $(\mathbf{2}, \mathbf{2})$: 2 multigrid operations for $\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}$
  - $(\mathbf{3}, \mathbf{3})$: 1 multigrid operation for K_p
  - $(\mathbf{4}, \mathbf{3})$: 1 Chebyshev semi-iteration for $\mathbf{M}$, and 2 multigrid operations for $\mathbf{K} + \frac{1}{\sqrt{\beta}}\mathbf{M}$
  - $(\mathbf{4}, \mathbf{4})$: 2 Chebyshev semi-iterations for M_p, and 1 multigrid operation for K_p
  - **Total**: 4 Chebyshev semi-iterations and 6 multigrids.

We can see from this list of operations that the application of each preconditioner (especially $\widehat{\mathcal{P}}_1$, $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_3$) is fairly cheap, and therefore that our methods should be computationally effective ones.

6.2 Numerical Results

Having motivated the theoretical potential of our approach, we now seek to demonstrate how our preconditioners perform in practice. To do this, we consider an optimal control analogue of the *lid-driven cavity* problem on the domain $\Omega = [-1, 1]^2$:

$$\begin{aligned} \min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \ & -\nabla^2\underline{\mathbf{v}} + \nabla p = \underline{\mathbf{u}}, \quad \text{in } \Omega, \\ & -\nabla\cdot\underline{\mathbf{v}} = 0, \quad \text{in } \Omega, \\ & \underline{\mathbf{v}} = \begin{cases} [1,0]^T & \text{on } [-1,1]\times\{1\}, \\ [0,0]^T & \text{on } \partial\Omega\backslash\left([-1,1]\times\{1\}\right). \end{cases} \end{aligned}$$

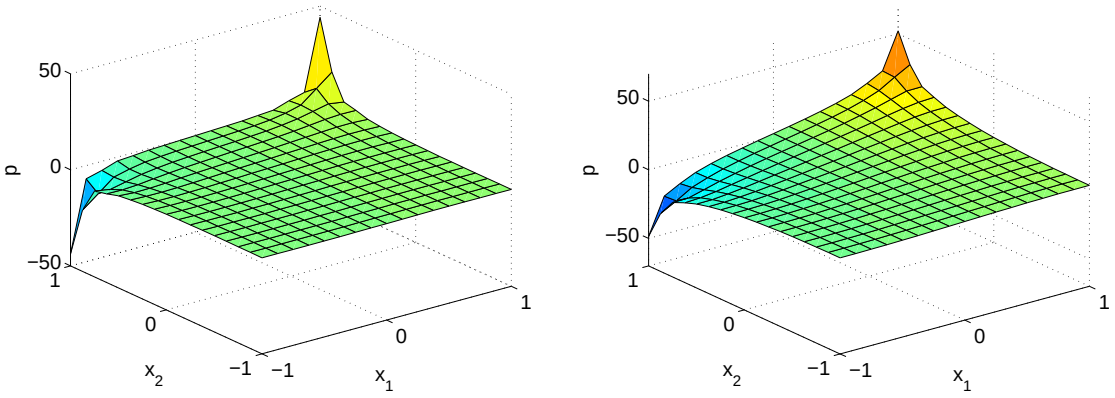


(a) $\beta = 1$ (b) $\beta = 10^{-4}$

Figure 6.1: Plots of computed velocity solution to the test problem for different β. The value of h used here is 2^{-3}.

This is a somewhat academic problem, but it is useful for testing the effectiveness of iterative solvers for Stokes-type problems. We therefore wish to observe how well the 4 preconditioners presented in this chapter perform when solving the matrix system relating to this problem. In Table 6.3, we show the number of MINRES iterations and CPU times[26] for solving this problem with preconditioner $\widehat{\mathcal{P}}_1$ to a tolerance of

[26]The CPU times include the time taken to construct the matrices M_p and K_p involved in the preconditioner. We construct these matrices in the same way as in the Incompressible Flow & Iterative Solver Software (IFISS) package [35, 110]. Where appropriate we follow the recipe detailed in [36, Chapter 8] of imposing a Dirichlet boundary condition in the matrix K_p at the node on the velocity space corresponding to the *inflow boundary condition*.



(a) $\beta = 10^{-2}$ (b) $\beta = 10^{-6}$

Figure 6.2: Plots of computed pressure solution to the test problem for different β. The value of h used here is 2^{-4}.

10^{-6}, for a variety of h and β. In Table 6.4, we show the number of iterations and CPU times for solving the same problem, using MINRES with preconditioner $\widehat{\mathcal{P}}_2$, to the same tolerance. Finally in Tables 6.5 and 6.6, we show the iteration count and CPU times for solving the problem to the same tolerance, with the GMRES algorithm used in the Incompressible Flow & Iterative Solver Software (IFISS) package [35, 110], preconditioned with the matrices $\widehat{\mathcal{P}}_3$ and $\widehat{\mathcal{P}}_4$.[27] In Figures 6.1 and 6.2, we display solutions to the test problem for velocity and pressure, for different values of β. In each of the tables and figures, the value of h indicated corresponds to the spacing between **Q2** nodes.

When generating these results, we once more use 20 steps of Chebyshev semi-iteration to approximate the inverse of mass matrices. To approximate the inverses of K_p, $\mathbf{M}+\sqrt{\beta}\mathbf{K}$ and $\mathbf{K}+\frac{1}{\sqrt{\beta}}\mathbf{M}$ in our preconditioners (note that the last two matrices are the same up to a multiplicative factor), we employ the algebraic multigrid routine `HSL_MI20` from the Harwell Subroutine Library (HSL) [19], using 2 V-cycles with 2 pre- and post- (relaxed Jacobi) smoothing steps to approximate each matrix inverse.[28]

The results shown in Tables 6.3–6.6 indicate that all 4 preconditioners discussed in this chapter are robust with respect to mesh-size and regularization parameter.

[27] All results in Tables 6.3–6.6 were obtained using a tri-core 2.5 GHz workstation.

[28] In Tables 6.3–6.6, the symbol * denotes that the coarsening of the AMG routine failed when applied to $\mathbf{M}+\sqrt{\beta}\mathbf{K}$ or $\mathbf{K}+\frac{1}{\sqrt{\beta}}\mathbf{M}$ – as in previous chapters this occurs in the specific and impractical parameter regime where h is large and β is small, and is caused by the presence of positive off-diagonal entries. In these cases, we present results obtained using direct solves rather than AMG.

$\widehat{\mathcal{P}}_1$		Size	β						
			10^2	1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
h	2^{-3}	1,318	80 0.281	80 0.283	60 0.216	44 0.189	36 0.156	(32)* (0.290)	(26)* (0.232)
	2^{-4}	4,934	84 0.755	85 0.766	66 0.601	52 0.488	37 0.475	(32)* (1.59)	(26)* (1.38)
	2^{-5}	19,078	88 3.03	90 3.08	70 2.41	58 2.04	44 2.05	32 1.54	(28)* (7.42)
	2^{-6}	75,014	86 12.6	90 13.6	74 11.0	62 10.3	50 7.96	33 8.39	(28)* (40.1)
	2^{-7}	297,478	86 62.0	88 58.8	76 53.5	66 46.2	54 39.3	40 29.2	26 28.1

Table 6.3: Number of iterations and CPU times (in seconds) when applying MINRES to the test problem with preconditioner $\widehat{\mathcal{P}}_1$, for a variety of h and β.

$\widehat{\mathcal{P}}_2$		Size	β						
			10^2	1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
h	2^{-3}	1,318	112 0.502	107 0.480	85 0.388	59 0.317	42 0.222	(30)* (0.316)	(25)* (0.249)
	2^{-4}	4,934	125 1.51	123 1.49	97 1.18	68 0.847	48 0.768	(33)* (1.87)	(25)* (1.52)
	2^{-5}	19,078	142 6.68	137 6.42	102 4.79	75 3.59	60 3.54	39 2.37	(27)* (7.33)
	2^{-6}	75,014	156 32.5	148 30.9	106 22.3	80 17.2	67 14.1	48 17.0	(29)* (46.5)
	2^{-7}	297,478	165 141	160 138	106 91.2	84 80.5	72 61.9	54 49.4	34 89.6

Table 6.4: Number of iterations and CPU times (in seconds) when applying MINRES to the test problem with preconditioner $\widehat{\mathcal{P}}_2$, for a variety of h and β.

In fact the only parameter regime where we do not observe complete robustness is that of very small β, when we observe some degradation in the performance of the AMG routine used. The iteration count is low, considering the complexity of the problem, when each of the 4 preconditioners is used. In many practical problems, the

$\widehat{\mathcal{P}}_3$		Size	β						
			10^2	1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
h	2^{-3}	1,318	64 0.238	62 0.247	53 0.195	44 0.188	39 0.184	(33)* (0.287)	(28)* (0.263)
	2^{-4}	4,934	65 0.671	63 0.674	56 0.573	50 0.516	41 0.565	(38)* (1.98)	(31)* (1.65)
	2^{-5}	19,078	63 2.54	63 2.53	56 2.26	53 2.11	48 2.81	38 2.27	(35)* (9.42)
	2^{-6}	75,014	63 13.7	61 12.5	57 13.1	54 11.5	51 10.9	41 13.8	(37)* (58.1)
	2^{-7}	297,478	63 60.8	62 62.8	56 55.3	52 45.1	52 51.8	48 43.8	39 45.5

Table 6.5: Number of iterations and CPU times (in seconds) when applying Gmres to the test problem with preconditioner $\widehat{\mathcal{P}}_3$, for a variety of h and β.

$\widehat{\mathcal{P}}_4$		Size	β						
			10^2	1	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
h	2^{-3}	1,318	91 0.755	85 0.675	67 0.529	46 0.415	26 0.238	(19)* (0.376)	(14)* (0.261)
	2^{-4}	4,934	107 2.55	101 2.38	79 1.84	59 1.38	34 1.02	(24)* (2.47)	(15)* (1.63)
	2^{-5}	19,078	123 11.7	114 10.8	88 8.20	73 6.75	47 5.42	29 3.42	(21)* (11.7)
	2^{-6}	75,014	138 63.5	131 58.5	99 43.7	81 37.0	62 27.6	37 24.1	(25)* (74.8)
	2^{-7}	297,478	156 327	150 287	109 224	89 161	73 130	48 92.3	30 148

Table 6.6: Number of iterations and CPU times (in seconds) when applying Gmres to the test problem with preconditioner $\widehat{\mathcal{P}}_4$, for a variety of h and β.

value of β is within the range $[10^{-6}, 10^{-1}]$; all solvers perform well in this regime. We note that the block diagonal preconditioner $\widehat{\mathcal{P}}_1$ (introduced in [129]), and the block triangular preconditioner $\widehat{\mathcal{P}}_3$ based on it, solve the problem in the shortest time in all cases considered, and the lowest iteration count in most cases. However, the strategy

involved in constructing these preconditioners is highly specific to this problem. As shown in the next chapter, the flexibility in the methodology used to construct $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_4$ enables us to consider the more general and much harder Navier-Stokes control problem, and therefore it is important to note that these preconditioners also seem to achieve robustness, albeit with larger iteration counts and CPU times than $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_3$.

Of the two preconditioners $\widehat{\mathcal{P}}_2$ and $\widehat{\mathcal{P}}_4$, we note that the preconditioner $\widehat{\mathcal{P}}_4$ solves the problem in fewer iterations than $\widehat{\mathcal{P}}_2$, but greater CPU time due to the added complexity of the GMRES algorithm (though this could partially be offset by using restarts within the GMRES method). We also note that in the parameter regime of small β, the iteration count when the preconditioner $\widehat{\mathcal{P}}_4$ is used is even smaller than that when $\widehat{\mathcal{P}}_1$ (or indeed $\widehat{\mathcal{P}}_2$) is applied.

6.3 Summary

In this chapter, we have been able to utilize our methodology for preconditioning optimal control problems to develop efficient solvers for the distributed Stokes control problem. Our methods for this problem have made use of effective preconditioners for the Poisson control problem which we discussed in Chapter 3. In addition, to solve the Stokes control problem we required a specialized commutator argument – we found that applying such an argument, alongside the preconditioning strategies for Poisson control, resulted in fast iterative methods for this problem.

We have discussed 2 block diagonal (one of which was previously derived in [129]) and 2 block triangular preconditioners for this problem, to be used with the MINRES or GMRES algorithms. We analysed the major operations required to apply each preconditioner, showing that the computational work required is very reasonable, and we carried out numerical tests to demonstrate the effectiveness of our new commutator arguments for a wide range of parameter values.

Putting all the pieces together, we were able to create new preconditioners which exhibited robustness with respect to mesh-size and regularization parameter. Perhaps the most intriguing aspect of the work presented in this chapter is that two of the preconditioners introduced may be adapted to tackle the harder and more general Navier-Stokes control problem. We detail how this may be done in the next chapter.

CHAPTER 7

Navier-Stokes Control

In this chapter, we aim to extend the methodology presented in the previous chapter to the more difficult and general Navier-Stokes control problem.[29] This problem has aroused much interest within the applied mathematics community, and we recommend literature such as [13, 14, 29, 30, 67, 68, 74, 96, 123] for discussions on research into this field. Developing preconditioners for this problem is therefore an interesting and important research topic. Indeed the construction of preconditioned iterative methods for the Navier-Stokes equations themselves is a fairly recent development; in [77] Kay, Loghin and Wathen utilized saddle point theory, along with a commutator argument to approximate the Schur complement of the matrix system, and were able to develop good solvers for the relevant matrix systems in this way. We will aim to use similar ideas to build efficient solvers for the control problem; we will incorporate an Oseen-type iteration to deal with the nonlinear terms in the forward and adjoint problems, apply saddle point theory to motivate good preconditioners, and utilize the preconditioning strategy of Chapter 5 for the convection-diffusion control problem, as well as develop new commutator arguments to approximate the Schur complement at each outer iteration.

This chapter is structured as follows. In Section 7.1, we outline the problem that

[29] This chapter is based on the following paper, which is Ref. [87]:

J. W. Pearson, *Preconditioned Iterative Methods for Navier-Stokes Control Problems*, submitted to SIAM Journal on Scientific Computing, 2013.

we wish to solve, and detail the Oseen-type iteration that we apply. In Section 7.2, we explain our strategies for preconditioning the matrix system that arises at each step of this outer iteration. In Section 7.3 we present numerical results for our solvers, and in Section 7.4 we summarize the work undertaken.

7.1 Outline of the Navier-Stokes Control Problem

The problem we will consider in this chapter is the time-independent (distributed) Navier-Stokes control problem, given by

$$\begin{aligned} \min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}-\widehat{\underline{\mathbf{v}}}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\ \text{s.t.} \ & -\nu\nabla^2\underline{\mathbf{v}}+(\underline{\mathbf{v}}\cdot\nabla)\underline{\mathbf{v}}+\nabla p=\underline{\mathbf{u}}, \quad \text{in } \Omega, \\ & -\nabla\cdot\underline{\mathbf{v}}=0, \quad \text{in } \Omega, \\ & \underline{\mathbf{v}}=\underline{\mathbf{g}}, \quad \text{on } \partial\Omega. \end{aligned} \tag{7.1}$$

We again work on the domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$, with boundary $\partial\Omega$. As in the previous chapter, the variables $\underline{\mathbf{v}}$ and p are the state variables, denoting the velocity (in d-dimensions) and pressure respectively, the variable $\underline{\mathbf{u}}$ denotes the control variable, the parameter β is a regularization parameter, and $\widehat{\underline{\mathbf{v}}}$ is some desired state. Once more, $\underline{\boldsymbol{\lambda}}$ and μ are the adjoint variables to $\underline{\mathbf{v}}$ and p respectively. The parameter ν as written above denotes viscosity.

The Navier-Stokes equations, on which this problem is based, are a fundamental system of equations describing viscous flow. They are an extension of the Stokes equations described in the previous chapter. Similarly as for the convection-diffusion equation introduced in Chapter 5, the $-\nu\nabla^2\underline{\mathbf{v}}$ term describes diffusive processes, and the $(\underline{\mathbf{v}}\cdot\nabla)\underline{\mathbf{v}}$ term represents convection. Additionally, an incompressibility condition $-\nabla\cdot\underline{\mathbf{v}}=0$ is stated.

In order to linearize this problem, we have a number of possibilities. The approach we follow is an Oseen-type (or Picard-type) iteration, as examined in [96]. This involves choosing an initial guess for states and control, and then solving in turn the

problems

$$\begin{aligned}\min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}-\widehat{\underline{\mathbf{v}}}\right\|^2_{L_2(\Omega)}+\frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)}-\int_\Omega(\underline{\mathbf{v}}\cdot\nabla\bar{\underline{\mathbf{v}}})\bar{\underline{\boldsymbol{\lambda}}}\,\mathrm{d}\Omega \\ \text{s.t.} \ & -\nu\nabla^2\underline{\mathbf{v}}+(\bar{\underline{\mathbf{v}}}\cdot\nabla)\underline{\mathbf{v}}+\nabla p=\underline{\mathbf{u}}, \quad \text{in } \Omega, \\ & -\nabla\cdot\underline{\mathbf{v}}=0, \quad \text{in } \Omega, \\ & \underline{\mathbf{v}}=\underline{\mathbf{g}}, \quad \text{on } \partial\Omega,\end{aligned}$$

where $\bar{\underline{\mathbf{v}}}$ denotes the most recent iterate of $\underline{\mathbf{v}}$ and $\bar{\underline{\boldsymbol{\lambda}}}$ the most recent iterate of $\underline{\boldsymbol{\lambda}}$, until convergence of the solution is achieved. As shown in [96], $-\int_\Omega(\underline{\mathbf{v}}\cdot\nabla\bar{\underline{\mathbf{v}}})\bar{\underline{\boldsymbol{\lambda}}}\,\mathrm{d}\Omega$ is the appropriate correction term required to prove convergence. Further, this choice of correction term also makes the cost functional strictly convex on a linear manifold, giving a unique solution for any given $\bar{\underline{\mathbf{v}}}$ and $\bar{\underline{\boldsymbol{\lambda}}}$ [96]. We will again use a **Q2-Q1** Taylor-Hood element to solve this problem.

We next apply a discretize-then-optimize approach, considering a discrete cost functional of the form

$$\begin{aligned}\mathcal{L}_{DTO}=\frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v}-\mathbf{v}^T\mathbf{z}+\frac{\beta}{2}\mathbf{u}^T\mathbf{M}\mathbf{u}-\mathbf{v}^T\bar{\mathbf{w}} \\ +\boldsymbol{\lambda}^T(\mathbf{F}\mathbf{v}+B^T\mathbf{p}-\mathbf{M}\mathbf{u}-\mathbf{g})+\boldsymbol{\mu}^T(B\mathbf{v}),\end{aligned}$$

at each step of the outer iteration. Here $\mathbf{F}=\nu\mathbf{K}+\mathbf{N}$, with $\mathbf{M}$ and $\mathbf{K}$ again denoting $d\times d$ block matrices consisting of standard finite element mass and stiffness matrices on their block diagonals. Furthermore, $\mathbf{N}$ is a matrix denoting convective terms of the type $\int_\Omega(\bar{\underline{\mathbf{v}}}\cdot\nabla\phi_j)\phi_i\,\mathrm{d}\Omega$, and $\bar{\mathbf{w}}$ contains terms of the form $\int_\Omega(\phi_i\cdot\nabla\bar{\underline{\mathbf{v}}})\bar{\underline{\boldsymbol{\lambda}}}\,\mathrm{d}\Omega$.

Differentiating $\mathcal{L}_{DTO}$ with respect to the discrete variables $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{p}$, and eliminating the gradient equation $\beta\mathbf{u}-\boldsymbol{\lambda}=\mathbf{0}$, gives the following matrix system at each Oseen step:

$$\begin{bmatrix}\mathbf{M} & 0 & \mathbf{F}^T & B^T\\ 0 & 0 & B & 0\\ \mathbf{F} & B^T & -\frac{1}{\beta}\mathbf{M} & 0\\ B & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{v}\\ \mathbf{p}\\ \boldsymbol{\lambda}\\ \boldsymbol{\mu}\end{bmatrix}=\begin{bmatrix}\mathbf{z}+\mathbf{g}_1\\ \mathbf{0}\\ \mathbf{g}_2\\ \mathbf{g}_3\end{bmatrix}+\begin{bmatrix}\bar{\mathbf{w}}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}. \tag{7.2}$$

We note that the discretize-then-optimize approach results in a symmetric matrix system at each Oseen iteration. The alternative optimize-then-discretize approach, as highlighted previously, will not necessarily result in symmetric matrix systems

for this problem, and the systems obtained will depend on the precise discretization strategy used.

We note also that, as for the Stokes control problem in the previous chapter, the bottom-left of the matrix system corresponds to the forward problem, the top-right relates to the adjoint problem, and the terms corresponding to the cost functional occur on the block diagonal. Equation (7.2) states the form of matrix system for which we will consider preconditioned iterative methods in this chapter.

We note that the Oseen-type iteration we have selected to deal with the non-linearity of the optimal control problem is not the only option we have to do this. Another possibility is to apply a Newton-type method (see [30, 67, 68, 123] for instance); however we discover that applying such a method causes the $(1,1)$-block of the resulting matrix systems to be dominated by convective terms. This is highly problematic when constructing an iterative solver for such a system using our approach, and we therefore conclude that the Oseen iteration strategy presented is the one best suited to our preconditioners. Numerical tests in [96] also indicate that this method is competitive overall with other approaches for solving these problems.

7.2 Preconditioners for the Navier-Stokes Control Problem

We now seek to develop good preconditioners for the matrix system (7.2). Motivated by the saddle point theory of Section 2.2 and the strategies employed for the Stokes control problem in the previous chapter, we rearrange the matrix system (7.2) obtained at each step of the Oseen iteration for the Navier-Stokes control problem. Specifically, we write it as

$$\underbrace{\begin{bmatrix} \mathbf{M} & \mathbf{F}^T & B^T & 0 \\ \mathbf{F} & -\frac{1}{\beta}\mathbf{M} & 0 & B^T \\ B & 0 & 0 & 0 \\ 0 & B & 0 & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z} + \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \bar{\mathbf{w}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \tag{7.3}$$

which is a saddle point system with

$$\Phi = \begin{bmatrix} \mathbf{M} & \mathbf{F}^T \\ \mathbf{F} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}, \quad \Psi = \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

We once more wish to derive effective preconditioners by developing good approximations to Φ and $S = \Psi\Phi^{-1}\Psi^T$.

We spend the majority of this section motivating a block diagonal preconditioner for the matrix system (7.3). We first note that the $(1,1)$-block Φ of $\mathcal{A}$ is itself a saddle point system – specifically, it relates to the matrix system corresponding to the (distributed) convection-diffusion control problem discussed in Chapter 5. We recall that the matrix

$$\widehat{\Phi} = \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)^T \end{bmatrix} =: \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \widehat{S}_{CD} \end{bmatrix} \tag{7.4}$$

is an effective preconditioner for the matrix Φ, and so we wish to employ the matrix $\widehat{\Phi}$ within our preconditioners for Navier-Stokes control.

In order to develop a good approximation of the Schur complement

$$S = \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & \mathbf{F}^T \\ \mathbf{F} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix}$$

of $\mathcal{A}$, we approximate the matrix $\begin{bmatrix} \mathbf{M} & \mathbf{F}^T \\ \mathbf{F} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}$ in terms of its $(1,1)$-block and exact Schur complement, and write (similarly as for the Stokes control problem in the previous chapter)

$$\begin{aligned} S &\approx \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T + \frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\ &= \begin{bmatrix} B\mathbf{M}^{-1}B^T & 0 \\ 0 & B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T \end{bmatrix}. \end{aligned}$$

The matrices $B\mathbf{M}^{-1}B^T$ and $B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T$ cannot be used directly within the approximation of the Schur complement – we must therefore introduce approximations of these matrices. However we once again note that $B\mathbf{M}^{-1}B^T \approx K_p$, so we

utilize this within our block diagonal preconditioner.

We now consider how best to approximate the matrix $B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T+\frac{1}{\beta}\mathbf{M}\right)^{-1}B^T$. We do this by applying a similar commutator argument to that introduced in the previous chapter.

We again start by examining the commutator

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

where now $\widetilde{\mathcal{L}}=(-\nu\nabla^2+\underline{\bar{\mathbf{v}}}\cdot\nabla)\cdot(-\nu\nabla^2+\underline{\bar{\mathbf{v}}}\cdot\nabla)^T+\frac{1}{\beta}I$, and make the assumption that $\mathcal{E}$ is small. The operator $\widetilde{\mathcal{L}}$ is chosen to represent $\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T+\frac{1}{\beta}\mathbf{M}$ on the continuous space, so that an approximation of $\widetilde{\Sigma}:=B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T+\frac{1}{\beta}\mathbf{M}\right)^{-1}B^T$ may be developed.

We next discretize this commutator using finite elements to obtain

$$\mathcal{E}_h=(\mathbf{M}^{-1}\mathbf{L})\mathbf{M}^{-1}B^T-\mathbf{M}^{-1}B^T(M_p^{-1}L_p)\approx 0, \tag{7.5}$$

where $\mathbf{L}=\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T+\frac{1}{\beta}\mathbf{M}$ and $L_p=F_pM_p^{-1}F_p^T+\frac{1}{\beta}M_p$. Note that we have again carried over to the discrete space our assumption that the commutator is small. Pre-multiplying (7.5) by $B\mathbf{L}^{-1}\mathbf{M}$ and post-multiplying by $L_p^{-1}M_p$ then gives that $B\mathbf{M}^{-1}B^TL_p^{-1}M_p\approx B\mathbf{L}^{-1}B^T$. We then use that $B\mathbf{M}^{-1}B^T\approx K_p$ and substitute in the expressions for $\mathbf{L}$ and L_p to give

$$\begin{aligned}\widetilde{\Sigma}=B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T+\frac{1}{\beta}\mathbf{M}\right)^{-1}B^T&\approx K_p\left(F_pM_p^{-1}F_p^T+\frac{1}{\beta}M_p\right)^{-1}M_p\\ \Rightarrow\widetilde{\Sigma}^{-1}\approx M_p^{-1}\left(F_pM_p^{-1}F_p^T+\frac{1}{\beta}M_p\right)K_p^{-1}&=M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1}+\frac{1}{\beta}K_p^{-1}.\end{aligned}$$

This commutator argument therefore generates an approximation of $\widetilde{\Sigma}$ to be used in our preconditioner for $\mathcal{A}$.

We use the result of this argument to approximate the Schur complement of $\mathcal{A}$ by

$$\widehat{S}:=\begin{bmatrix}K_p & 0\\ 0 & \left(M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1}+\frac{1}{\beta}K_p^{-1}\right)^{-1}\end{bmatrix}=:\begin{bmatrix}K_p & 0\\ 0 & \widehat{S}_{NS,1}^{-1}\end{bmatrix}.$$

Putting all the pieces together, we may write a proposed preconditioner $\widehat{\mathcal{P}}_1$ for $\mathcal{A}$

as blkdiag $\left(\widehat{\Phi}, \widehat{S}\right)$, i.e.

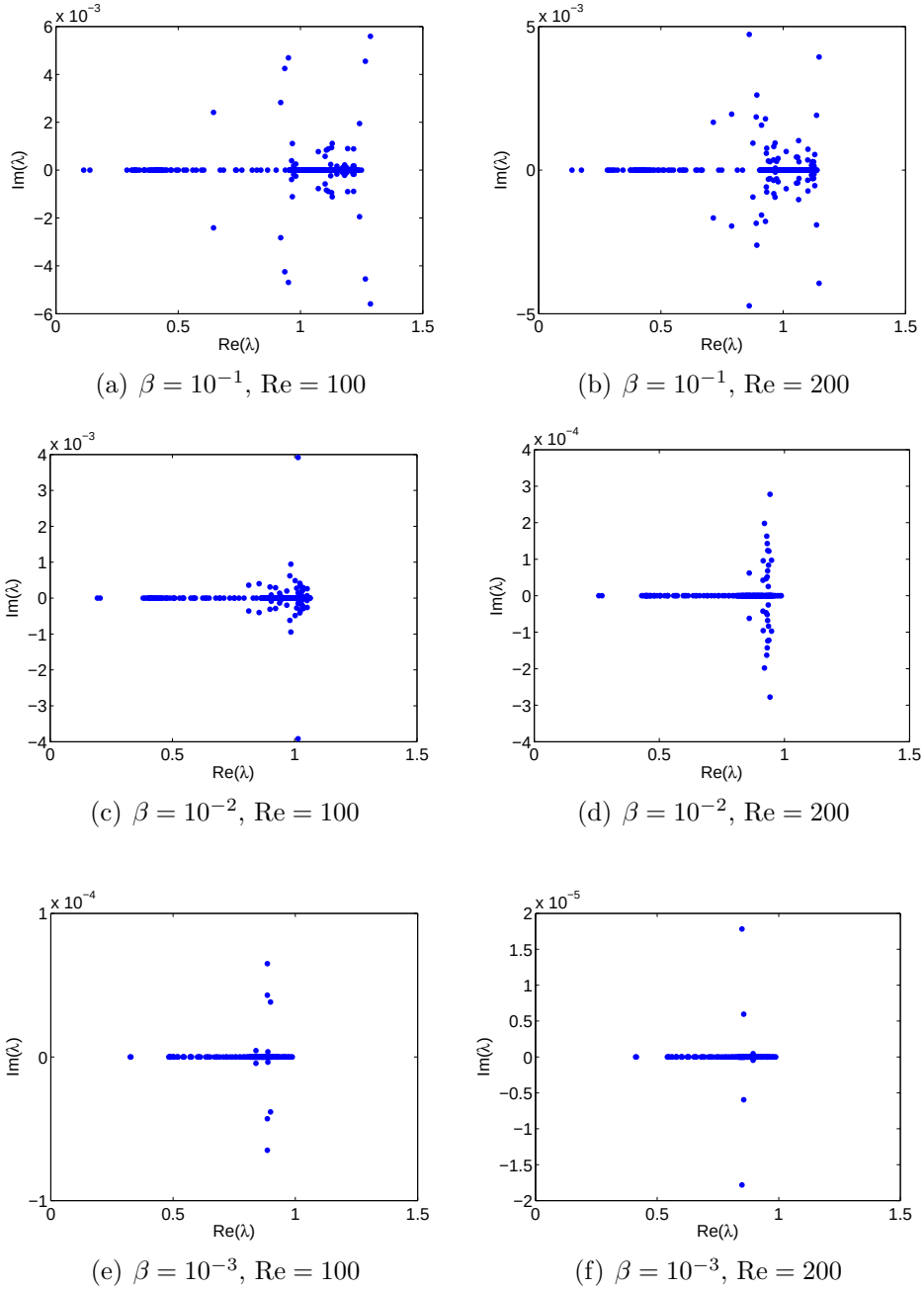
$$\widehat{\mathcal{P}}_1 = \begin{bmatrix} \mathbf{M} & 0 & 0 & 0 \\ 0 & \left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)^T & 0 & 0 \\ 0 & 0 & K_p & 0 \\ 0 & 0 & 0 & \widehat{S}_{NS,1}^{-1} \end{bmatrix}.$$

We highlight once again that this is not a symmetric preconditioner, due to the non-symmetry of the matrix $M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1}$, despite the fact that $\mathcal{A}$ is itself symmetric. We are therefore not able to use a symmetric solver with this preconditioner, and would instead need to use a solver such as GMRES with preconditioner $\widehat{\mathcal{P}}_1$.

We note that, as before, the above commutator argument is a heuristic approach, and due to the non-symmetry of the matrix approximation generated it would be very difficult to analyze in great detail. In Figure 7.1 we provide eigenvalue plots for the matrix $\left(M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1} + \frac{1}{\beta}K_p^{-1}\right)\left[B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T\right]$, for small matrix systems arising from the final Oseen iteration applied to Problem 1 (as stated in Section 7.3). Plots are given for a range of β and Reynolds number Re. The matrix we consider for our plots is equal to $\widetilde{\Sigma}$ preconditioned by our approximation of $\widetilde{\Sigma}$. In the plots, we omit the zero eigenvalue resulting from the vector of ones belonging to the nullspace of B^T, as well as the largest eigenvalue. The reason for this latter omission is that we find there is a single eigenvalue of this matrix of much larger magnitude than the others – identifying a way of isolating and removing this very large eigenvalue would improve our approximation. We find however that the remainder of the eigenvalues are well clustered for a range of parameters, and that an individual eigenvalue is unlikely to greatly delay the convergence of a Krylov subspace method.

The non-symmetry of the preconditioner $\widehat{\mathcal{P}}_1$ means that we could also create a block triangular preconditioner for the matrix $\mathcal{A}$, without imposing further restrictions on the solvers we could use it with. We therefore now derive such a block triangular preconditioner. We start by approximating Φ with the non-symmetric matrix

$$\begin{bmatrix} \mathbf{M} & 0 \\ \mathbf{F} & -\left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)\mathbf{M}^{-1}\left(\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}\right)^T \end{bmatrix},$$



(a) $\beta = 10^{-1}$, Re $= 100$ (b) $\beta = 10^{-1}$, Re $= 200$

(c) $\beta = 10^{-2}$, Re $= 100$ (d) $\beta = 10^{-2}$, Re $= 200$

(e) $\beta = 10^{-3}$, Re $= 100$ (f) $\beta = 10^{-3}$, Re $= 200$

Figure 7.1: Spectra of $\left(M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1} + \frac{1}{\beta}K_p^{-1}\right)\left[B\left(\mathbf{F}\mathbf{M}^{-1}\mathbf{F}^T + \frac{1}{\beta}\mathbf{M}\right)^{-1}B^T\right]$ for $\beta = 10^{-1}$, 10^{-2}, 10^{-3} and Re $= \frac{2}{\nu} = 100$ and 200, for the final Oseen step in the solution of Problem 1, on an evenly spaced grid on $\Omega = [-1, 1]^2$ with $h = 2^{-4}$.

as motivated in Chapter 5.

When approximating the Schur complement of $\mathcal{A}$ for this block triangular preconditioner, we may write

$$\begin{aligned}
&\begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & \mathbf{F}^T \\ \mathbf{F} & -\frac{1}{\beta}\mathbf{M} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\
&\qquad \approx \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M} & 0 \\ \mathbf{F} & -\widehat{S}_{CD} \end{bmatrix}^{-1} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\
&\qquad = \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \mathbf{M}^{-1} & 0 \\ \widehat{S}_{CD}^{-1}\mathbf{F}\mathbf{M}^{-1} & -\widehat{S}_{CD}^{-1} \end{bmatrix} \begin{bmatrix} B^T & 0 \\ 0 & B^T \end{bmatrix} \\
&\qquad = \begin{bmatrix} B\mathbf{M}^{-1}B^T & 0 \\ B\widehat{S}_{CD}^{-1}\mathbf{F}\mathbf{M}^{-1}B^T & -B\widehat{S}_{CD}^{-1}B^T \end{bmatrix} \\
&\qquad \approx \begin{bmatrix} K_p & 0 \\ B\widehat{S}_{CD}^{-1}\mathbf{F}\mathbf{M}^{-1}B^T & -\widehat{S}_{NS,2}^{-1} \end{bmatrix},
\end{aligned}$$

where $\widehat{S}_{CD}$ is as defined in (7.4), and

$$\widehat{S}_{NS,2} := M_p^{-1}F_pM_p^{-1}F_p^TK_p^{-1} + \frac{1}{\beta}K_p^{-1} + \frac{1}{\sqrt{\beta}}M_p^{-1}(F_p + F_p^T)K_p^{-1}.$$

As for the block diagonal preconditioner, we use the fact that $B\mathbf{M}^{-1}B^T \approx K_p$. We then take $\widetilde{\mathcal{L}} = \left(-\nu\nabla^2 + \bar{\underline{\mathbf{v}}}\cdot\nabla + \frac{1}{\sqrt{\beta}}I\right)\cdot\left(-\nu\nabla^2 + \bar{\underline{\mathbf{v}}}\cdot\nabla + \frac{1}{\sqrt{\beta}}I\right)^T$ in the above commutator argument to obtain that $B\widehat{S}_{CD}^{-1}B^T \approx \widehat{S}_{NS,2}^{-1}$.

Putting all the pieces together, we may postulate that

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \mathbf{M} & 0 & 0 & 0 \\ \mathbf{F} & -\widehat{S}_{CD} & 0 & 0 \\ B & 0 & K_p & 0 \\ 0 & B & B\widehat{S}_{CD}^{-1}\mathbf{F}\mathbf{M}^{-1}B^T & -\widehat{S}_{NS,2}^{-1} \end{bmatrix}$$

is an appropriate choice of a block triangular preconditioner for $\mathcal{A}$. We may use this with an iterative method such as GMRES.

Having derived our two proposed preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$, we now examine the dominant processes required to apply the inverses of these preconditioners – we of course do not invert any of the matrices exactly, but instead approximate them. We

approximate the inverse of a mass matrix by Chebyshev semi-iteration as discussed in Section 2.3.2 and in [126], and we deal with the matrix K_p by using the algebraic multigrid routine `HSL_MI20` from the Harwell Subroutine Library (HSL) [19]. Whenever the matrix $\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}$ or its transpose appears, we also use the same AMG routine, but we note that, for flows with a very large Reynolds number (which is a harder problem numerically due to dominant convective terms within the matrix), we would need to apply a more specialized multigrid routine, such as that described in [98] for the forward convection-diffusion problem.[30]

Below we detail the dominant operations required to approximate $\widehat{\mathcal{P}}_1^{-1}$ (for this we view $\widehat{\mathcal{P}}_1$ as a 4×4 block matrix, and refer to the blocks as such):

- $(\mathbf{1}, \mathbf{1})$: 1 Chebyshev semi-iteration for $\mathbf{M}$
- $(\mathbf{2}, \mathbf{2})$: 2 multigrid operations: 1 for $\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}$ and 1 for its transpose
- $(\mathbf{3}, \mathbf{3})$: 1 multigrid operation for K_p
- $(\mathbf{4}, \mathbf{4})$: 1 multigrid operation for K_p, and 2 Chebyshev semi-iterations for M_p
- **Total**: 3 Chebyshev semi-iterations and 4 multigrids (2 dealing with terms involving convection).

We similarly detail the dominant operations required to approximate $\widehat{\mathcal{P}}_2^{-1}$:

- $(\mathbf{1}, \mathbf{1})$: 1 Chebyshev semi-iteration for $\mathbf{M}$
- $(\mathbf{2}, \mathbf{2})$: 2 multigrid operations: 1 for $\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}$ and 1 for its transpose
- $(\mathbf{3}, \mathbf{3})$: 1 multigrid operation for K_p
- $(\mathbf{4}, \mathbf{3})$: 1 Chebyshev semi-iteration for M_p, and 2 multigrid operations: 1 for $\mathbf{F} + \frac{1}{\sqrt{\beta}}\mathbf{M}$ and 1 for its transpose
- $(\mathbf{4}, \mathbf{4})$: 1 multigrid operation for K_p, and 3 Chebyshev semi-iterations for M_p
- **Total**: 5 Chebyshev semi-iterations and 6 multigrids (4 dealing with terms involving convection).

[30] In the case of high Reynolds number flow, the problem may require strategies such as stabilization as well, which would be taken account of within the multigrid routine. As this is a specialized subject area, with the appropriate stabilization technique highly dependent on the type of finite elements used to solve the problem, we do not investigate this in this chapter, but instead provide a more general picture of the strategies required to solve this problem.

We note that a single application of the inverse of $\widehat{\mathcal{P}}_2$ is therefore more expensive than an application of the inverse of $\widehat{\mathcal{P}}_1$, and hence a fixed number of GMRES iterations will be cheaper when used with the preconditioner $\widehat{\mathcal{P}}_1$. We also comment that one convenient feature of both preconditioners is that one never has to invert the matrices F_p or F_p^T exactly. When they appear in the preconditioners, a matrix multiply is involved rather than an inversion – this is a positive aspect of our preconditioners as these matrices may contain large convective terms, and hence applying a multigrid routine to them may be troublesome numerically.

We will demonstrate the potential effectiveness of both proposed preconditioners in the next section. We will use the GMRES method to show this, but we note that other methods, such as BICG or BICGSTAB, could also be applied.

7.3 Numerical Results

In this section, we test our proposed solvers on the following two examples:

- **Problem 1:** We consider the following optimal control variant of the lid-driven cavity problem:

$$\begin{aligned}
\min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\
\text{s.t.} \ & -\nu\nabla^2\underline{\mathbf{v}} + (\underline{\mathbf{v}}\cdot\nabla)\underline{\mathbf{v}} + \nabla p = \underline{\mathbf{u}}, \quad \text{in } \Omega := [-1,1]^2, \\
& -\nabla\cdot\underline{\mathbf{v}} = 0, \quad \text{in } \Omega, \\
& \underline{\mathbf{v}} = \begin{cases} [1,0]^T & \text{on } [-1,1]\times\{1\}, \\ [0,0]^T & \text{on } \partial\Omega\backslash\left([-1,1]\times\{1\}\right). \end{cases}
\end{aligned}$$

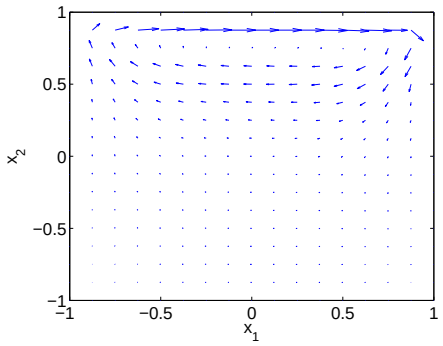
- **Problem 2:** We consider a target state which involves a recirculating wind near to the boundary and zero velocity near the centre of the domain, with the problem statement as follows:

$$\begin{aligned}
\min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ & \frac{1}{2}\left\|\underline{\mathbf{v}}-\widehat{\underline{\mathbf{v}}}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\
\text{s.t.} \ & -\nu\nabla^2\underline{\mathbf{v}} + (\underline{\mathbf{v}}\cdot\nabla)\underline{\mathbf{v}} + \nabla p = \underline{\mathbf{u}}, \quad \text{in } \Omega := [-1,1]^2, \\
& -\nabla\cdot\underline{\mathbf{v}} = 0, \quad \text{in } \Omega, \\
& \underline{\mathbf{v}} = \widehat{\underline{\mathbf{v}}}, \quad \text{on } \partial\Omega,
\end{aligned}$$

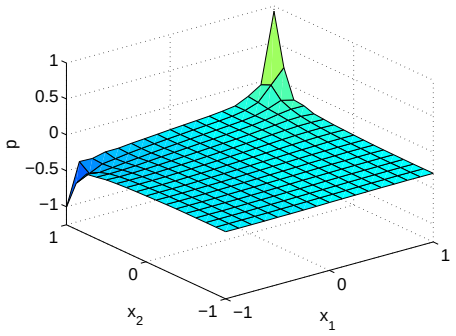
where

$$\widehat{\underline{\mathbf{v}}} = \begin{cases} \left[\frac{1}{2}x_2(1-x_1^2), -\frac{1}{2}x_1(1-x_2^2)\right]^T & \text{if } x_1^2 + x_2^2 \geq \frac{1}{2}, \\ [0,0]^T & \text{otherwise,} \end{cases}$$

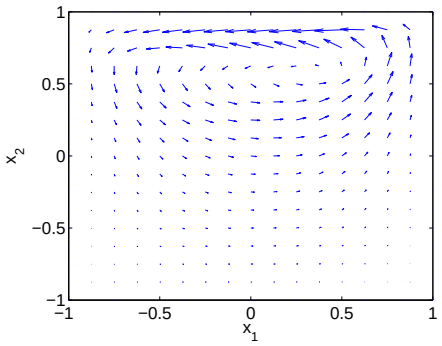
and $\mathbf{x} = [x_1, x_2]^T$ denotes the spatial coordinates.



(a) Velocity $\underline{\mathbf{v}}$



(b) Pressure p



(c) Adjoint $\underline{\boldsymbol{\lambda}}$

Figure 7.2: Solution plots for velocity $\underline{\mathbf{v}}$, pressure p and adjoint $\underline{\boldsymbol{\lambda}}$ for Problem 1, with $\beta = 1$ and $\mathrm{Re} = 100$.

Solution plots are given for these problems in Figures 7.2 and 7.3. We solve these problems iteratively with an outer iteration tolerance of 10^{-5} (we measure convergence using the ratio of the vector 2-norm of the difference between the current and previous iterates for $\mathbf{v}$ divided by the vector 2-norm of the previous iterate for $\mathbf{v}$) and with a GMRES tolerance of 10^{-6}. When applying our preconditioners, we use 20 steps of Chebyshev semi-iteration whenever we need to approximate the inverse of a mass matrix, and 2 V-cycles of the AMG routine from the Harwell Subroutine Library [19] when approximating the inverses of all other matrices. In order to construct the

relevant finite element matrices, we use and modify code from the IFISS software system [35, 110], from which we also modify the version of the GMRES code for solving our problems.

Re = 50, $\widehat{\mathcal{P}}_1$		β = 10^{-1}		10^{-2}		10^{-3}		10^{-4}		10^{-5}		10^{-6}	
h	2^{-3}	5	67	5	55	4	44	4	(34)*	3	(23)*	2	(18)*
	2^{-4}	5	83	4	70	4	58	4	48	3	(34)*	3	(24)*
	2^{-5}	5	104	4	85	3	71	3	60	3	48	3	(36)*
	2^{-6}	5	124	4	97	3	80	3	71	3	62	3	50
	2^{-7}	5	158	4	118	3	95	3	80	3	71	3	62

Re = 50, $\widehat{\mathcal{P}}_1$		β = 10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
h	2^{-3}	0.389	0.331	0.252	(0.374)*	(0.237)*	(0.181)*
	2^{-4}	1.23	1.20	1.00	0.83	(1.87)*	(1.28)*
	2^{-5}	5.93	4.90	4.80	4.50	5.88	(9.56)*
	2^{-6}	33.6	25.4	20.3	19.8	21.3	16.9
	2^{-7}	191	140	111	90.5	82.1	99.5

Table 7.1: Top: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 1 with preconditioner $\widehat{\mathcal{P}}_1$, for a variety of h and β, and with Re = 50. Bottom: Average CPU times (in seconds) for the same values.

$h = 2^{-5}$, $\widehat{\mathcal{P}}_1$		β = 1		10^{-1}		10^{-2}		10^{-3}		10^{-4}	
Re	1	3	182	3	180	3	157	2	135	2	119
	10	4	183	3	140	3	111	3	92	3	80
	50	7	144	5	104	4	85	3	71	3	60
	100	8	122	5	93	4	74	4	61	4	50
	200	12	118	6	87	5	65	4	52	4	41

Table 7.2: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 1 with preconditioner $\widehat{\mathcal{P}}_1$, for a variety of β and Re, and with $h = 2^{-5}$.

We wish to test the performance of our methods for different values of h (which we define to be the mesh-size between **Q2** nodes), regularization parameter β and viscosity ν.[31] The Reynolds number of the flow we consider is again $\text{Re} = \frac{2}{\nu}$, as we are working on a domain of length scale 2. In Table 7.1, we fix the value of Re to be 50, and test our preconditioner $\widehat{\mathcal{P}}_1$ on Problem 1 for a variety of h and β. We display the number of Oseen iterations (with our initial guess the solution of the equivalent Stokes control problem – that is the same problem but without the convective terms), the average number of GMRES iterations per Oseen iteration, and the average CPU time for each Oseen iteration. Table 7.3 displays the same results with preconditioner $\widehat{\mathcal{P}}_2$. We also wish to examine the effect of altering the Reynolds number. Therefore, in Tables 7.2 and 7.4, we present results from tests of the preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ on Problem 1 for a variety of values of β and Re, with a fixed mesh-size $h = 2^{-5}$.[32]

Re = 50, $\widehat{\mathcal{P}}_2$		$\beta = 10^{-1}$		$\beta = 10^{-2}$		$\beta = 10^{-3}$		$\beta = 10^{-4}$		$\beta = 10^{-5}$		$\beta = 10^{-6}$	
h	2^{-3}	5	45	5	30	4	23	4	(18)*	3	(14)*	2	(12)*
	2^{-4}	5	61	4	41	4	31	4	24	3	(19)*	3	(15)*
	2^{-5}	5	72	4	59	3	42	3	32	3	24	3	(20)*
	2^{-6}	5	86	4	68	3	52	3	43	3	34	3	24
	2^{-7}	5	113	4	82	3	64	3	52	3	45	3	37

Re = 50, $\widehat{\mathcal{P}}_2$		$\beta = 10^{-1}$	$\beta = 10^{-2}$	$\beta = 10^{-3}$	$\beta = 10^{-4}$	$\beta = 10^{-5}$	$\beta = 10^{-6}$
h	2^{-3}	0.421	0.288	0.213	(0.348)*	(0.268)*	(0.234)*
	2^{-4}	1.46	1.17	0.905	0.710	(2.07)*	(1.58)*
	2^{-5}	6.80	5.57	4.96	4.31	5.56	(10.8)*
	2^{-6}	37.8	29.6	22.1	20.4	22.1	14.9
	2^{-7}	227	160	125	101	86.1	107

Table 7.3: Top: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 1 with preconditioner $\widehat{\mathcal{P}}_2$, for a variety of h and β, and with Re = 50. Bottom: Average CPU times (in seconds) for the same values.

[31] All results were obtained using a tri-core 2.5 GHz workstation.

[32] In Tables 7.1, 7.3, 7.5 and 7.6, the numbers in brackets and labelled * correspond to values where the AMG routine used did not work, due to the presence of positive off-diagonal entries. We again use direct solves for such matrices in place of a multigrid routine.

$h = 2^{-5}$		β									
$\widehat{\mathcal{P}}_2$		1		10^{-1}		10^{-2}		10^{-3}		10^{-4}	
Re	1	3	143	3	136	3	115	2	99	2	84
	10	4	130	3	102	3	77	3	67	3	50
	50	7	105	5	72	4	59	3	42	3	32
	100	8	88	5	65	4	46	4	35	4	25
	200	12	84	6	60	5	39	4	28	4	21

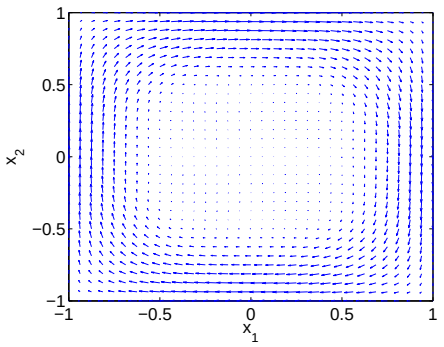
Table 7.4: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 1 with preconditioner $\widehat{\mathcal{P}}_2$, for a variety of β and Re, and with $h = 2^{-5}$.

From the tables, we first note that the number of outer (Oseen/Picard) iterations is reasonable for all parameter values tested (though the number rises as the Reynolds number is increased), so we believe our choice of this outer iteration is an appropriate one for these problems. Looking at the average number of GMRES iterations and CPU times in Tables 7.1 and 7.3, we note a benign dependence on h when using our solvers, though we believe that the increase in iteration numbers as h decreases is reasonable, as the size of the matrix system increases by roughly a factor of 4 as h is halved. Our methods also perform better as β and ν are decreased, as shown in Tables 7.2 and 7.4. The decrease in iteration numbers as ν is decreased (i.e. for higher Reynolds numbers) is in some sense surprising, though we point out that the accuracy of the finite element solution is likely to be worse in these cases for a fixed h. We also note that beyond values of Re ≈ 200, the AMG routine used begins to struggle due to the dominant convective terms within the relevant matrices, so a more sophisticated multigrid routine would need to be employed.

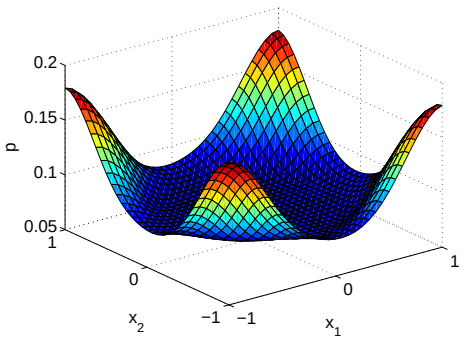
In Tables 7.5 and 7.6, we display the iteration numbers taken to solve Problem 2 using preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$, for a range of h, β and Re. We see that for this harder problem, the iteration numbers are slightly larger, but all are still reasonable given the complexity of the problem.

We also briefly examine the following test problem:

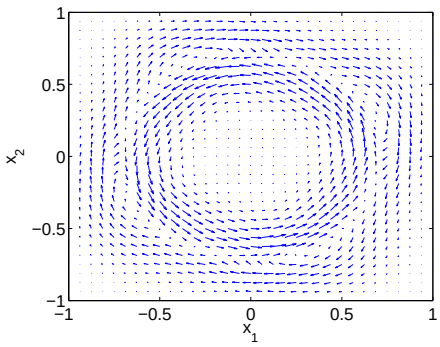
- **Problem 3:** We look at an optimal control analogue of a *channel flow problem*



(a) Velocity $\underline{\mathbf{v}}$



(b) Pressure p



(c) Adjoint $\underline{\boldsymbol{\lambda}}$

Figure 7.3: Solution plots for velocity $\underline{\mathbf{v}}$, pressure p and adjoint $\underline{\boldsymbol{\lambda}}$ for Problem 2, with $\beta = 1$ and $\mathrm{Re} = 200$.

introduced in [36, Chapter 7]:

$$
\begin{aligned}
&\min_{\underline{\mathbf{v}},\underline{\mathbf{u}}} \ \frac{1}{2}\left\|\underline{\mathbf{v}}\right\|^2_{L_2(\Omega)} + \frac{\beta}{2}\left\|\underline{\mathbf{u}}\right\|^2_{L_2(\Omega)} \\
&\text{s.t.} \ -\nu\nabla^2\underline{\mathbf{v}} + (\underline{\mathbf{v}}\cdot\nabla)\underline{\mathbf{v}} + \nabla p = \underline{\mathbf{u}}, \quad \text{in } \Omega := [-1,1]^2, \\
&\qquad\qquad -\nabla\cdot\underline{\mathbf{v}} = 0, \quad \text{in } \Omega, \\
&\qquad\qquad \underline{\mathbf{v}} = \begin{cases} [1-x_2^2, 0]^T & \text{on } \{-1\}\times[-1,1], \\ [0,0]^T & \text{on } [-1,1]\times\{1\}, \\ [0,0]^T & \text{on } [-1,1]\times\{-1\}, \end{cases} \\
&\qquad\qquad \left[\frac{\partial v_{x_1}}{\partial x_1}, \frac{\partial v_{x_2}}{\partial x_1}\right]^T = [p, 0]^T, \quad \text{on } \{1\}\times[-1,1],
\end{aligned}
$$

where $\underline{\mathbf{v}} = [v_{x_1}, v_{x_2}]^T$, and $\mathbf{x} = [x_1, x_2]^T$ as before. We present solution plots for this problem in Figure 7.4.

Re = 50, $\widehat{\mathcal{P}}_1$		β									
		10^{-1}		10^{-2}		10^{-3}		10^{-4}		10^{-5}	
h	2^{-3}	4	88	4	61	4	44	3	(35)*	2	(30)*
	2^{-4}	4	116	3	85	3	60	3	44	3	(33)*
	2^{-5}	4	155	3	108	3	75	3	54	3	41
	2^{-6}	4	218	3	143	3	94	3	66	2	50

Re = 100, $\widehat{\mathcal{P}}_1$		β									
		10^{-1}		10^{-2}		10^{-3}		10^{-4}		10^{-5}	
h	2^{-3}	6	84	5	52	4	(36)*	3	(30)*	2	(27)*
	2^{-4}	5	120	4	78	4	51	3	38	3	(29)*
	2^{-5}	5	167	3	102	3	68	3	45	3	35
	2^{-6}	5	244	3	135	3	84	3	59	3	44

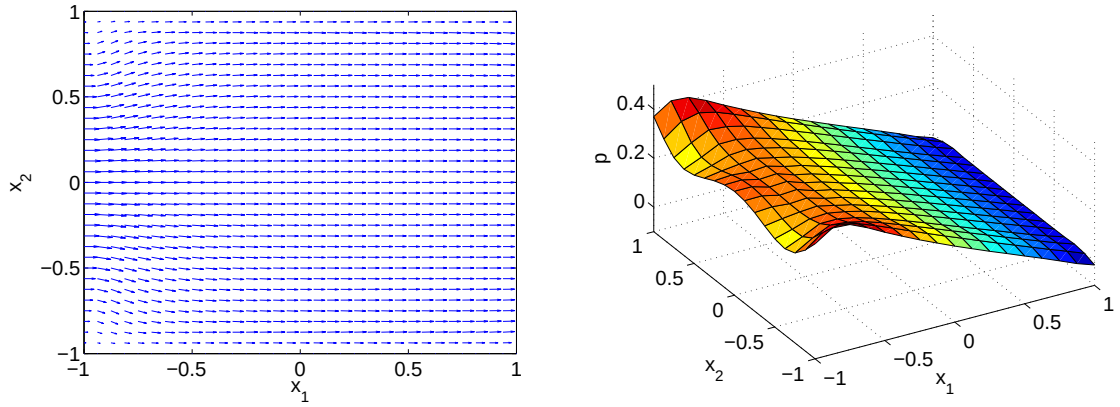
Table 7.5: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 2 with preconditioner $\widehat{\mathcal{P}}_1$, for a variety of h and β, with Re = 50 and Re = 100.

Re = 50, $\widehat{\mathcal{P}}_2$		β									
		10^{-1}		10^{-2}		10^{-3}		10^{-4}		10^{-5}	
h	2^{-3}	4	56	4	35	4	24	3	(20)*	2	(16)*
	2^{-4}	4	81	3	54	3	35	3	24	3	(20)*
	2^{-5}	4	111	3	74	3	48	3	31	3	22
	2^{-6}	4	151	3	93	3	63	3	45	2	30

Re = 100, $\widehat{\mathcal{P}}_2$		β									
		10^{-1}		10^{-2}		10^{-3}		10^{-4}		10^{-5}	
h	2^{-3}	6	54	5	31	4	(22)*	3	(17)*	2	(14)*
	2^{-4}	5	79	4	46	4	28	3	21	3	(17)*
	2^{-5}	5	115	3	70	3	42	3	26	3	19
	2^{-6}	5	169	3	94	3	58	3	38	3	25

Table 7.6: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 2 with preconditioner $\widehat{\mathcal{P}}_2$, for a variety of h and β, with Re = 50 and Re = 100.

We note that this problem set-up is not of the precise form of (7.1), as there are Neumann boundary conditions present. However we find that our approach can be



(a) Velocity $\underline{\mathbf{v}}$

(b) Pressure p

Figure 7.4: Solution plots for velocity $\underline{\mathbf{v}}$ and pressure p for Problem 3, with $\beta = 10^{-2}$ and $\mathrm{Re} = 100$.

used for this problem as well, and we present results for solving this problem with preconditioner $\widehat{\mathcal{P}}_1$, for a range of h, β and Re, in Table 7.7. The set-up of the problem, including the presence of Neumann boundary conditions, makes it a much harder one from a numerical point-of-view (where the value of Re can play a large role). Therefore when solving this problem we run the Oseen-type iteration to a higher tolerance of 10^{-3} for our numerical tests, with GMRES run to a tolerance of 10^{-4}. We also note that the AMG routine used for other problems fails in a large number of cases, so we use exact solves instead where appropriate. We therefore regard our solver for this problem as a "proof-of-concept" implementation – a more complete solver would require a suitable multigrid routine, such as that described by Ramage in [98]. Overall however, the iteration numbers shown suggest that the framework for solving difficult problems such as this one may be in place; the only parameter regime where our solver struggles occurs when β is very large. Given the incorporation of a more robust multigrid routine however, our solver should once more be a feasible and effective one for this problem, for a range of parameter values.

From the results obtained for each problem, we observe that our two solvers involving the preconditioners $\widehat{\mathcal{P}}_1$ and $\widehat{\mathcal{P}}_2$ perform quite similarly. However, it appears that, although the block triangular preconditioner $\widehat{\mathcal{P}}_2$ consistently solves the problem in fewer iterations, the block diagonal preconditioner $\widehat{\mathcal{P}}_1$ does so in lower CPU time for the majority of parameter values studied. This is due to the larger number of operations required to apply the preconditioner $\widehat{\mathcal{P}}_2$, as detailed in the previous section.

$\widehat{\mathcal{P}}_1$		$h = 2^{-4}$								$h = 2^{-5}$							
		β								β							
		10^{-2}		10^{-3}		10^{-4}		10^{-5}		10^{-2}		10^{-3}		10^{-4}		10^{-5}	
Re	50	3	107	3	80	3	69	2	63	3	196	2	117	2	95	2	86
	100	4	100	3	73	3	63	2	57	4	175	2	106	3	85	2	79
	200	4	102	4	68	3	56	2	55	4	156	3	97	3	75	2	71
	500	7	114	4	63	3	53	2	52	5	191	4	101	3	68	2	65

Table 7.7: Number of outer iterations (in blue) and average number of GMRES iterations per outer iteration (rounded to the nearest integer) when solving Problem 3 with preconditioner $\widehat{\mathcal{P}}_1$, for a variety of β and Re, with $h = 2^{-4}$ and $h = 2^{-5}$, and with exact solves used instead of algebraic multigrid for the relevant matrices.

h	SIZE	DIRECT	ITERATIVE
2^{-3}	1,318	0.050	0.272
2^{-4}	4,934	0.321	1.11
2^{-5}	19,078	1.76	4.12
2^{-6}	75,014	16.1	23.5
2^{-7}	297,478	—	121
2^{-8}	1,184,774	—	1240

Table 7.8: CPU times (in seconds) for solving Problem 1, with sparse direct solves in MATLAB, and our iterative approach with preconditioner $\widehat{\mathcal{P}}_1$, for a range of h and with $\beta = 10^{-2}$, Re = 100. Where '—' is denoted, the sparse direct method failed to give a solution.

Importantly however, we have demonstrated that solving a number of complicated Navier-Stokes control problems is feasible for a range of parameter values using either of our methods, and we believe that the iteration numbers obtained are satisfactory considering the complexity of the problem. To illustrate the importance of developing such feasible iterative methods for Navier-Stokes control, we compare direct and iterative solution strategies in Table 7.8 – here it is shown that our method gives solutions in reasonable times for matrix systems which are sufficiently large that direct methods fail when attempting to solve them.

7.4 Summary

In this chapter, we have been able to exploit saddle point theory, solvers for convection-diffusion control problems and commutator arguments, in order to create preconditioned iterative methods for the difficult and practically useful incompressible Navier-Stokes control problem. We have demonstrated numerically that our approach yields feasible solution strategies for a wide range of mesh-sizes, regularization parameters and Reynolds numbers, and for a number of test problems.

There are many pieces of future research that could be spawned by the work presented in this chapter. One major challenge would be to create preconditioners for Navier-Stokes control problems solved by Newton iteration. This would be likely to require a markedly different approach in order to deal with convection-dominated terms which would appear in the $(1,1)$-block of the matrix system. However if this could be tackled it would be a major breakthrough, as the convergence of the outer iteration would be more reliable. One could also modify this approach to deal with boundary control problems, and problems which include additional state or control constraints. Finally one could seek to solve time-dependent variants of the problem introduced in this section – such an extension could be of substantial practical use, and would be a natural problem to consider. In the next chapter, we will introduce solvers for a class of time-dependent optimal control problems, which could form the foundation of preconditioners for the time-dependent analogue of this optimal control problem.

CHAPTER 8

Time-Dependent Optimal Control Problems

In this chapter, we wish to extend the methodology we have developed so far to time-dependent PDE-constrained optimization problems.[33] As the majority of real-world mathematical models involve a time-dependent element, it is natural and important to consider how to develop effective solvers for such problems. We base our studies in this chapter on the optimal control of the heat equation, for the most part the following problem that we introduced in Section 2.1.4:

$$\begin{aligned} &\min_{y,u} \ J(y,u) \qquad (8.1)\\ &\text{s.t} \quad y_t - \nabla^2 y = u, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T],\\ &\qquad\qquad y = g, \quad \text{on } \partial\Omega,\\ &\qquad\qquad y = y_0, \quad \text{at } t = 0, \end{aligned}$$

[33] This chapter is based on the following two papers, which are Refs. [91] and [92] respectively:

J. W. Pearson, M. Stoll and A. J. Wathen, *Regularization-Robust Preconditioners for Time-Dependent PDE-Constrained Optimization Problems*, SIAM Journal on Matrix Analysis and Applications, 33(4), pp.1126–1152, 2012.

J. W. Pearson, M. Stoll and A. J. Wathen, *Robust Iterative Solution of a Class of Time-Dependent Optimal Control Problems*, Proceedings in Applied Mathematics and Mechanics (PAMM), 12(1), pp.3–6, 2012.

Specifically, the "final-time case" portion of Section 8.1.1, as well as some numerical results, are based on the latter article, with the remainder based on the former.

where

$$J(y,u) = J_1(y,u) = \frac{1}{2}\int_0^T \int_\Omega (y(\mathbf{x},t) - \widehat{y}(\mathbf{x},t))^2 \ \mathrm{d}\Omega\mathrm{d}t + \frac{\beta}{2}\int_0^T \int_\Omega (u(\mathbf{x},t))^2 \ \mathrm{d}\Omega\mathrm{d}t,$$

or

$$J(y,u) = J_2(y,u) = \frac{1}{2}\int_\Omega (y(\mathbf{x},T) - \widehat{y}(\mathbf{x}))^2 \ \mathrm{d}\Omega + \frac{\beta}{2}\int_0^T \int_\Omega (u(\mathbf{x},t))^2 \ \mathrm{d}\Omega\mathrm{d}t.$$

Recall that we referred to the problems involving the minimization of $J_1(y,u)$ and $J_2(y,u)$ as the "all-times case" and "final-time case" respectively. We will also develop solvers for the Neumann boundary control of the heat equation.

We consider the development of block diagonal preconditioners for these problems to be used with MINRES, though of course related block triangular preconditioners could also be constructed. The goal of our solvers, in keeping with the work presented so far in this thesis, is for their performance to be independent of mesh-size h and regularization parameter β, as well as the time-step τ taken in the set-up of the problem. Devising solvers for these problems is a much more difficult task than doing so for their time-independent counterparts, as the matrix systems in question are of much larger dimension.

This chapter is structured as follows. In Section 8.1, we will derive robust preconditioners for the distributed control problems discussed above, as well as related boundary control problems. We again pay particular attention to the choice of Schur complement approximations. In Section 8.2, we discuss aspects of our solvers, including the potential for parallelization of our methods, and spectral properties of the preconditioned systems. In Section 8.3, we present numerical results to highlight the performance in practice of our solvers, and in Section 8.4 we make some concluding comments.

8.1 Preconditioning the Matrix Systems

In this section, we motivate preconditioners for a number of time-dependent optimal control problems, commencing with distributed control problems, and then moving on to Neumann boundary control problems.

8.1.1 Distributed Control

All-Times Case with Discretize-then-Optimize Approach

We start by examining the problem (8.1) with $J(y,u) = J_1(y,u)$. As demonstrated in Section 2.1.4, employing the discretize-then-optimize strategy results in the matrix system

$$\begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau\mathbf{z}_{1/2} \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}, \tag{8.2}$$

where $\mathcal{M}_{1/2}$, $\mathcal{M}$ and $\mathcal{K}$ are defined by (2.23), (2.22) and (2.21) respectively. Equation (8.2) is a saddle point system of the form (2.31), with

$$\Phi = \begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} \end{bmatrix}, \quad \Psi = \begin{bmatrix} \mathcal{K} & -\tau\mathcal{M} \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

To construct preconditioners for this problem, as usual we require good approximations of the $(1,1)$-block and Schur complement of the matrix system in question. Approximating the $(1,1)$-block is a trivial task, as it is a block diagonal matrix solely consisting of mass matrices. For our numerical experiments in this chapter we will employ lumped mass matrices, and so we may use diagonal solves to represent Φ^{-1} exactly. Such a strategy has long been used for solving physical problems using finite element methods (see [65] for instance), and involves taking a diagonal approximation of finite element mass matrices to make the problem structure simpler. However if we were to use consistent mass matrices instead, we may of course apply the Chebyshev semi-iteration method.

Our task then is to approximate the (negative) Schur complement of the system, which is given by

$$S = \frac{1}{\tau}\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}.$$

We wish to see if we may approximate S in a similar way as we did for the time-independent case. We therefore employ a similar matching strategy as in previous

chapters to attempt to "capture" both terms of the Schur complement, and so define

$$\widehat{S} = \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)\mathcal{M}_{1/2}^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)^T.$$

We see that, with this choice of approximation, it is not desirable to invert $\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}$ and its transpose exactly as this is essentially equivalent to solving the PDE directly, which is itself computationally expensive. Hence, in our solver we apply the inverse of $\widehat{S}$ using multigrid cycles for $\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}$ and its transpose. This involves using a multigrid process for each of the diagonal blocks $M + \tau K + \frac{\tau}{\sqrt{\beta}}M$. We will need to apply a few cycles of such a multigrid process N_t times to approximate the action of the inverse of $\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}$, and N_t times to approximate the action of the inverse of its transpose.

In conclusion, to solve the matrix system (8.2), we recommend using the MINRES method with the following preconditioner:

$$\widehat{\mathcal{P}} = \begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} & 0 \\ 0 & 0 & \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}}\mathcal{M}_{1/2}^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}}^T \end{bmatrix}.$$

We emphasize that we only require the storage of two matrices, M and K, to apply this iterative method. In Section 8.3, we provide numerical results to demonstrate the effectiveness of our proposed preconditioner.

All-Times Case with Optimize-then-Discretize Approach

We now wish to consider the iterative solution of the matrix system

$$\begin{bmatrix} \tau\mathcal{M}_0 & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau\mathbf{z}_0 \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}, \tag{8.3}$$

which corresponds to the optimize-then-discretize method for the all-times case. The matrix $\mathcal{M}_0$ is defined by (2.25). We also introduced this matrix system in Section 2.1.4. This is a saddle point system with

$$\Phi = \begin{bmatrix} \tau\mathcal{M}_0 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} \end{bmatrix}, \quad \Psi = \begin{bmatrix} \mathcal{K} & -\tau\mathcal{M} \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

However we immediately face a problem when we attempt to approximate it using our usual saddle point preconditioners, namely that the $(1,1)$-block of the matrix system is not invertible due to the zero block on the block diagonal. This means, furthermore, that the Schur complement of the matrix system (8.3) does not exist.

Our solution to this problem is to consider for our preconditioner a perturbation of the $(1,1)$-block of the matrix system to

$$\widetilde{\Phi} = \begin{bmatrix} \tau\mathcal{M}_0^\gamma & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} \end{bmatrix},$$

where

$$\mathcal{M}_0^\gamma = \begin{bmatrix} M & & & & \\ & M & & & \\ & & \ddots & & \\ & & & M & \\ & & & & \gamma M \end{bmatrix},$$

for a constant γ such that $0 < \gamma \ll 1$. We will return to the precise choice of γ in Section 8.2.1.

Perturbing the $(1,1)$-block of the matrix system in this way ensures invertibility of the perturbed block, and the existence of the Schur complement of the perturbed system

$$\widetilde{S} = \Psi\widetilde{\Phi}^{-1}\Psi^T = \frac{1}{\tau}\mathcal{K}\left(\mathcal{M}_0^\gamma\right)^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}.$$

Therefore, in our heuristic approach, we aim to develop a preconditioner for the matrix system $\begin{bmatrix} \widetilde{\Phi} & \Psi^T \\ \Psi & 0 \end{bmatrix}$ with the goal that this preconditioner will also be effective for the original system. We find that this is a highly effective strategy in practice.

Preconditioning the perturbed system is quite feasible. The matrix $\widetilde{\Phi}$ may be inverted exactly as we use lumped mass matrices (and if we were to use consistent mass matrices, we could use the Chebyshev semi-iteration method to approximate $\widetilde{\Phi}^{-1}$). For the Schur complement $\widetilde{S}$ of the perturbed system, we wish to make use of a similar matching strategy as for the discretize-then-optimize case.

To enable us to do this, we apply basic algebraic manipulation to $\widetilde{S}$ to arrive at

$$\widetilde{S} = \frac{1}{\tau}\mathcal{K}\left(\mathcal{M}_0^\gamma\right)^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\Gamma_1\left(\mathcal{M}_0^\gamma\right)^{-1}\Gamma_1,$$

where

$$\Gamma_1 = \begin{bmatrix} \sqrt{2}M & & & & \\ & M & & & \\ & & \ddots & & \\ & & & M & \\ & & & & \sqrt{2\gamma}M \end{bmatrix}.$$

We are now in a position to apply our usual strategy to capture both terms of the Schur complement in our approximation. We hence advocate

$$\widehat{S} = \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\Gamma_1\right)\left(\mathcal{M}_0^\gamma\right)^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\Gamma_1\right)^T$$

as an effective Schur complement approximation for the perturbed system. We believe this is a pragmatic choice for our preconditioner of the non-perturbed system as well, as the two systems are very similar in structure. Of course as in the previous section we do not advocate inverting $\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\Gamma_1$ or its transpose exactly; instead we will apply multigrid cycles to approximate the actions of the matrix inverses.

We therefore recommend the following preconditioner for the matrix system (8.3):

$$\widehat{\mathcal{P}} = \begin{bmatrix} \tau\mathcal{M}_0^\gamma & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} & 0 \\ 0 & 0 & \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\Gamma_1\right)_{\text{MG}}\left(\mathcal{M}_0^\gamma\right)^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\Gamma_1\right)_{\text{MG}}^T \end{bmatrix}.$$

In Section 8.3, we show numerical results illustrating the potency of this preconditioner.

Final-Time Case

The last distributed control problem we consider is the final-time case introduced in Section 2.1.4 – this leads to the matrix system

$$\begin{bmatrix} \mathcal{M}_1 & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{1/2} & -\tau\mathcal{M} \\ \mathcal{K} & -\tau\mathcal{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix}, \tag{8.4}$$

where $\mathcal{M}_1$ is defined by (2.27). This is a saddle point system with

$$\Phi = \begin{bmatrix} \mathcal{M}_1 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} \end{bmatrix}, \quad \Psi = \begin{bmatrix} \mathcal{K} & -\tau\mathcal{M} \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

However, as in the previous section, the $(1,1)$-block Φ is not invertible, as the matrix $\mathcal{M}_1$ is not invertible. We note that in fact $\mathcal{M}_1$ is highly degenerate, as every block diagonal entry is zero apart from the one corresponding to the final time-step.

We again consider a perturbation of the $(1,1)$-block to

$$\widetilde{\Phi} = \begin{bmatrix} \mathcal{M}_1^\gamma & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} \end{bmatrix},$$

where

$$\mathcal{M}_1^\gamma = \begin{bmatrix} \gamma M & & & & \\ & \gamma M & & & \\ & & \ddots & & \\ & & & \gamma M & \\ & & & & M \end{bmatrix},$$

with γ once again a small constant, the choice of which we discuss in Section 8.2.1.

As in the previous section, we wish to develop a preconditioner for the matrix system $\begin{bmatrix} \widetilde{\Phi} & \Psi^T \\ \Psi & 0 \end{bmatrix}$. As there, the matrix $\widetilde{\Phi}$ may be inverted exactly, leaving the main task as approximating the Schur complement of the perturbed system

$$\widetilde{S} = \mathcal{K}\left(\mathcal{M}_1^\gamma\right)^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M},$$

which by simple manipulation we may write as

$$\widetilde{S} = \mathcal{K}\left(\mathcal{M}_1^{\gamma}\right)^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\Gamma_2\left(\mathcal{M}_1^{\gamma}\right)^{-1}\Gamma_2,$$

where

$$\Gamma_2 = \begin{bmatrix} \sqrt{2\gamma}M & & & & \\ & \sqrt{\gamma}M & & & \\ & & \ddots & & \\ & & & \sqrt{\gamma}M & \\ & & & & \sqrt{2}M \end{bmatrix}.$$

We may now apply our matching strategy to arrive at the following Schur complement approximation for the perturbed system:

$$\widehat{S} = \left(\mathcal{K} + \sqrt{\frac{\tau}{\beta}}\Gamma_2\right)\left(\mathcal{M}_1^{\gamma}\right)^{-1}\left(\mathcal{K} + \sqrt{\frac{\tau}{\beta}}\Gamma_2\right)^T.$$

We believe this is also a sensible choice for our preconditioner of the non-perturbed system. Once again we will approximate $\mathcal{K} + \sqrt{\frac{\tau}{\beta}}\Gamma_2$ and its transpose by multigrid cycles in our preconditioner.

To summarize, we recommend the following preconditioner for the matrix system (8.4):

$$\widehat{\mathcal{P}} = \begin{bmatrix} \mathcal{M}_1^{\gamma} & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} & 0 \\ 0 & 0 & \left(\mathcal{K} + \sqrt{\frac{\tau}{\beta}}\Gamma_2\right)_{\text{MG}}\left(\mathcal{M}_1^{\gamma}\right)^{-1}\left(\mathcal{K} + \sqrt{\frac{\tau}{\beta}}\Gamma_2\right)_{\text{MG}}^T \end{bmatrix}.$$

In Section 8.3, we show numerical results to demonstrate the effectiveness of this preconditioner.

8.1.2 Time-Dependent Neumann Boundary Control

We now wish to consider the iterative solution of time-dependent Neumann boundary control problems, and see if our approach can yield effective preconditioning strategies once more. The problem we consider corresponds to the all-times case, and is written

as

$$\begin{aligned} \min_{y,u} \ & J_3(y,u) \\ \text{s.t} \quad & y_t - \nabla^2 y = f, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T], \\ & \frac{\partial y}{\partial n} = u, \quad \text{on } \partial\Omega, \\ & y = y_0, \quad \text{at } t = 0, \end{aligned}$$

where

$$J_3(y,u) = \int_0^T \int_\Omega (y(\mathbf{x},t) - \widehat{y}(\mathbf{x},t))^2 \, \mathrm{d}\Omega \mathrm{d}t + \frac{\beta}{2} \int_0^T \int_{\partial\Omega} (u(\mathbf{x},t))^2 \, \mathrm{d}s \mathrm{d}t.$$

We consider a discretize-then-optimize approach for this matrix system. This results in the Lagrangian

$$\mathcal{L}_{DTO} = \frac{\tau}{2}\mathbf{y}^T \mathcal{M}_{1/2}\mathbf{y} - \tau \mathbf{y}^T \mathbf{z}_{1/2} + \bar{C} + \frac{\beta\tau}{2}\mathbf{u}^T \mathcal{M}_{b,1/2}\mathbf{u} + \mathbf{p}^T(\mathcal{K}\mathbf{y} - \mathcal{N}\mathbf{u} - \mathbf{c}),$$

where $\mathcal{M}_{1/2}$ and $\mathbf{z}_{1/2}$ are as defined in Section 2.1.4, $\mathcal{K}$ is as defined at (2.21) except now with mass and stiffness matrices corresponding to the Neumann problem, and

$$\mathcal{M}_{b,1/2} = \begin{bmatrix} \frac{1}{2}M_b & & & & \\ & M_b & & & \\ & & \ddots & & \\ & & & M_b & \\ & & & & \frac{1}{2}M_b \end{bmatrix},$$

$$\mathcal{N} = \begin{bmatrix} N_b & & & & \\ & N_b & & & \\ & & \ddots & & \\ & & & N_b & \\ & & & & N_b \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} M\mathbf{y}_0 + \mathbf{f} \\ \mathbf{f} \\ \vdots \\ \mathbf{f} \\ \mathbf{f} \end{bmatrix},$$

with $\mathbf{y}_0$ and $\mathbf{f}$ corresponding to the initial condition and source term of the PDE respectively, and the boundary matrices M_b, N_b defined by (2.14), (2.13).

Differentiating with respect to $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{p}$ as usual gives the matrix system

$$\begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & \mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_{b,1/2} & -\tau\mathcal{N}^T \\ \mathcal{K} & -\tau\mathcal{N} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau\mathbf{z}_{1/2} \\ \mathbf{0} \\ \mathbf{c} \end{bmatrix}. \tag{8.5}$$

This is a saddle point system with

$$\Phi = \begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 \\ 0 & \beta\tau\mathcal{M}_{b,1/2} \end{bmatrix}, \quad \Psi = \begin{bmatrix} \mathcal{K} & -\tau\mathcal{N} \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

Therefore, when developing a preconditioner, we need to approximate Φ and $S = \Psi\Phi^{-1}\Psi^T$. Representing Φ is again trivial, as we use lumped mass matrices and so Φ may be inverted exactly.

We turn our attention to the Schur complement

$$S = \frac{1}{\tau}\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T,$$

for which we wish to develop good approximations.

One possible such approximation is given by

$$\begin{aligned} \widehat{S}_1 &= \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\widehat{\mathcal{M}}\right)\mathcal{M}_{1/2}^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\widehat{\mathcal{M}}\right)^T \\ &= \frac{1}{\tau}\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\widehat{\mathcal{M}}\mathcal{M}_{1/2}^{-1}\widehat{\mathcal{M}} + \frac{1}{\sqrt{\beta}}\left(\mathcal{K}\mathcal{M}_{1/2}^{-1}\widehat{\mathcal{M}} + \widehat{\mathcal{M}}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\right), \end{aligned}$$

where we make a careful choice of $\widehat{\mathcal{M}}$, similarly as for the time-independent boundary control problem in Section 4.1. As then, we wish to choose $\widehat{\mathcal{M}}$ such that the second term of $\widehat{S}_1$ and the second term of S approximately match, that is

$$\frac{\tau}{\beta}\widehat{\mathcal{M}}\mathcal{M}_{1/2}^{-1}\widehat{\mathcal{M}} \approx \frac{\tau}{\beta}\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T.$$

Now, as

$$\widehat{\mathcal{M}}\mathcal{M}_{1/2}^{-1}\widehat{\mathcal{M}} = \begin{bmatrix} 2\widehat{M}M^{-1}\widehat{M} & & & & & \\ & \widehat{M}M^{-1}\widehat{M} & & & & \\ & & \ddots & & & \\ & & & \widehat{M}M^{-1}\widehat{M} & & \\ & & & & 2\widehat{M}M^{-1}\widehat{M} \end{bmatrix},$$

$$\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T = \begin{bmatrix} 2N_bM_b^{-1}N_b^T & & & & & \\ & N_bM_b^{-1}N_b^T & & & & \\ & & \ddots & & & \\ & & & N_bM_b^{-1}N_b^T & & \\ & & & & 2N_bM_b^{-1}N_b^T \end{bmatrix},$$

we are reduced to choosing a matrix such that $\widehat{M}M^{-1}\widehat{M} = N_bM_b^{-1}N_b^T$. In Section 4.1, we established that a good choice for such a matrix is $\sqrt{h}M_\Gamma$, where $M_\Gamma = \text{blkdiag}(0, M_b)$ (assuming that the interior nodes are ordered first, followed by the boundary nodes). We therefore make this choice again, and select

$$\widehat{\mathcal{M}} = \sqrt{h}\begin{bmatrix} M_\Gamma & & & & \\ & M_\Gamma & & & \\ & & \ddots & & \\ & & & M_\Gamma & \\ & & & & M_\Gamma \end{bmatrix}.$$

We may similarly justify the alternative Schur complement approximation

$$\widehat{S}_2 = \frac{1}{\tau}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\widehat{\mathcal{M}}\right)\left(h\widehat{\mathcal{M}}_\Gamma\right)^{-1}\left(\mathcal{K} + \frac{\tau}{\sqrt{\beta}}\widehat{\mathcal{M}}\right)^T,$$

where $\widehat{\mathcal{M}}_\Gamma = \text{blkdiag}\left(\frac{1}{2}\widehat{M}_\Gamma, \widehat{M}_\Gamma, ..., \widehat{M}_\Gamma, \frac{1}{2}\widehat{M}_\Gamma\right)$, with $\widehat{M}_\Gamma$ again denoting a matrix containing the entries of M_Γ at boundary nodes, and entries of $\mathcal{O}(h^{d-1})$ on diagonal entries corresponding to interior nodes. This is analogous to the choice $\widehat{S}_2$ we made for the time-independent boundary control problem in Section 4.1, and can be justified using the observation that $h\widehat{\mathcal{M}}_\Gamma$ is spectrally equivalent to $\mathcal{M}_{1/2}$.

Putting all the pieces together, we recommend the following preconditioners for

the matrix system (8.5):

$$\begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{b,1/2} & 0 \\ 0 & 0 & \widehat{S}_1 \end{bmatrix}, \quad \begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{b,1/2} & 0 \\ 0 & 0 & \widehat{S}_2 \end{bmatrix},$$

where multigrid cycles are used to approximate inverses of matrices involving discrete elliptic operators where required. In Section 8.3, we present results which highlight the effectiveness of these approximations for a range of parameters.

8.2 Aspects of our Preconditioners

We now wish to provide a short discussion of three important points relating to our preconditioning strategies. The first note that we make concerns the selection of the perturbation parameter γ that we utilize in a number of our preconditioners for distributed control problems. Secondly, we discuss how the preconditioners may be modified with the goal of making our solvers parallelizable. Finally, we discuss spectral properties of the preconditioned systems using our methods.

8.2.1 Selection of Parameter γ

In two of the solvers for distributed control problems described above (relating to the all-times case using the optimize-then-discretize method, and the final-time case), we have used explicitly a small perturbation parameter γ. We now wish to detail precisely how we select this value.

In the final-time case, our selection is motivated by wishing both terms of the Schur complement of the perturbed system

$$S = \mathcal{K}(\mathcal{M}_1^{\gamma})^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}$$

to be "balanced" in some sense. In more detail, we wish to approximately balance the smallest eigenvalues of $\mathcal{K}(\mathcal{M}_1^{\gamma})^{-1}\mathcal{K}^T$ and $\tau\beta^{-1}\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}$. We simplify this task by replacing $\mathcal{K}$ by its block diagonal $\bar{\mathcal{L}} := \text{blkdiag}(L_d, ..., L_d)$, where $L_d = M + \sqrt{\beta}K$, and comparing the blocks in S as follows:

$$\gamma^{-1}h^{-2}L_d^2 \approx \tau\beta^{-1}h^2 I,$$

using the approximation $M \approx h^2 I$ for the 2D problem. Now, for $L_d^2 = \tau^2 K^2 + \tau K M + \tau M K + M^2$, the smallest eigenvalues will be of order $\tau^2 h^4$ (up to multiplicative constants). Hence, if we wish to balance both terms of S, we obtain

$$\gamma^{-1} h^{-2} \tau^2 h^4 \approx \tau \beta^{-1} h^2 \quad \Leftrightarrow \quad \gamma \approx \tau \beta.$$

We therefore use this reasoning to make the selection $\gamma = \tau\beta$ in our preconditioning strategy. We note that the same working can be done to motivate the same choice of γ in 3D.

For the all-times case being solved with the optimize-then-discretize approach, applying identical analysis to that above leads to the selection $\gamma = \beta$. However, on occasions we believe it is sensible to modify the analysis to make the choice $\gamma = \tau\beta$ for this problem as well, as this ensures that γ is small when the time-step is small.

We note that if we wished to solve variants of the time-dependent optimal control problems detailed here, such as a boundary control problem corresponding to the final-time case, we would need to incorporate perturbation parameters into the preconditioners in a similar way. In this case, we would recommend selecting these parameters in a similar way as described in this section.

8.2.2 Parallelization of Solvers

A subject area which is currently attracting much interest in scientific computing is that of parallel computing (see [47]). This involves constructing methods and algorithms such that operations within the method may be carried out concurrently, on many processors rather than just one. Developing such algorithms has the potential to drastically reduce the computation time to solve the problem in question. Parallelization has previously been considered in the context of PDE-constrained optimization problems in literature such as [13, 14, 31, 37].

Based on this, a reasonable question concerning our methods for time-dependent optimal control problems seems to be whether they may be parallelized over time. That is to say, would it be possible to take a time-dependent PDE-constrained optimization problem described in this chapter, involving a large number N_t of time-steps, and divide up the time interval into a number of smaller intervals such that each processor within a cluster is required to carry out computations resulting from a much smaller number of time-steps.

In order to predict whether this may be achieved for our solvers, the pertinent

question is whether our preconditioners could be applied in parallel. We answer this question for our recommended preconditioner

$$\widehat{\mathcal{P}} = \begin{bmatrix} \tau\mathcal{M}_{1/2} & 0 & 0 \\ 0 & \beta\tau\mathcal{M}_{1/2} & 0 \\ 0 & 0 & \frac{1}{\tau}\left(\mathcal{K}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}}\mathcal{M}_{1/2}^{-1}\left(\mathcal{K}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}}^{T} \end{bmatrix}$$

for the distributed control problem corresponding to the all-times case using the discretize-then-optimize method. However the reasoning introduced here carries over to the other problems considered in this chapter as well.

We first note that individual mass matrices within $\mathcal{M}_{1/2}$ may clearly be inverted in parallel. However, there is greater difficulty when seeking to apply multigrid to the matrix $\mathcal{K}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}$ or its transpose, as this involves solving systems of the form

$$\begin{bmatrix} L_d+\frac{\tau}{\sqrt{\beta}}M & & & \\ -M & L_d+\frac{\tau}{\sqrt{\beta}}M & & \\ & \ddots & \ddots & \\ & & -M & L_d+\frac{\tau}{\sqrt{\beta}}M \end{bmatrix}\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_{N_t} \end{bmatrix} \tag{8.6}$$

as well as the equivalent transpose system, at each MINRES iteration. The system (8.6) could not be solved in parallel, as information about $\mathbf{x}_{j-1}$ would be needed to solve for $\mathbf{x}_j$ (for $j = 2, ..., N_t$).

However, we find in results not presented in this chapter that using the similar Schur complement approximation

$$\widehat{S}_{\bar{\mathcal{L}}} = \frac{1}{\tau}\left(\bar{\mathcal{L}}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}}\mathcal{M}_{1/2}^{-1}\left(\bar{\mathcal{L}}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}\right)_{\text{MG}},$$

where once again $\bar{\mathcal{L}}$ consists of the block diagonals of $\mathcal{K}$, results in almost identical iteration numbers to those obtained when using the approximation $\widehat{S}$. This is unsurprising, as by inspection of the structure of $\mathcal{K}$ the dominant blocks all occur on the block diagonals, with the τK matrices strongly dominating the $-M$ terms unless τ is very small and h extremely large. Figure 8.1 demonstrates this dominance of the block diagonal terms.

Moreover, applying the inverse of $\widehat{S}_{\bar{\mathcal{L}}}$ is parallelizable, as $\bar{\mathcal{L}}+\frac{\tau}{\sqrt{\beta}}\mathcal{M}$ is a block diagonal matrix. We therefore advocate this as an effective strategy for developing

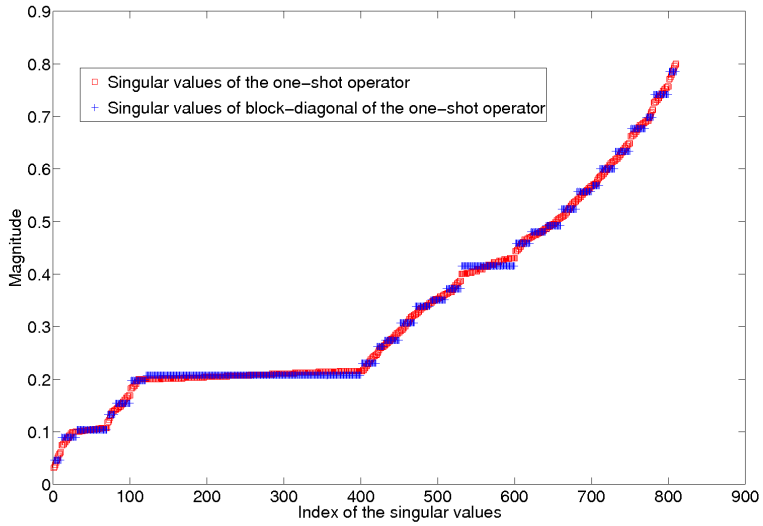


Figure 8.1: Singular values of the matrices $\mathcal{K}$ and $\bar{\mathcal{L}}$ for a small example.

parallel solvers for these problems.

With this approach, each processor would need only to store the comparatively small matrices M and K (as well as M_b and N_b for boundary control problems), a few (relatively small) vectors, and a number of scalars such as τ and β. We therefore conclude that our preconditioners can be feasibly modified to make their application parallelizable, meaning that the practical scope of our methods could potentially become much wider.

8.2.3 Spectral Properties of Preconditioned Matrices

We now briefly discuss the spectral properties of the preconditioned matrix systems generated using our methods, as such properties will indicate the effectiveness of our solvers. How clustered the eigenvalues of the preconditioned system are will depend on the effectiveness of our approximations of the $(1,1)$-block and Schur complement. Naturally, as the $(1,1)$-blocks of the matrix systems (or perturbed matrix systems) are all diagonal matrices, our "approximations" of these blocks are ideal ones. To justify the effectiveness of our solvers, we therefore consider the eigenvalues of the preconditioned Schur complements using our approximations.

Consider first the eigenvalues of the preconditioned Schur complement for the boundary control problem of Section 8.1.2 – we wish to argue in the same way as for the time-independent case in Section 4.1. As then, we present analysis for the

2D case, with the 3D case being similar. We hope to argue that the eigenvalues of $\widehat{S}_2^{-1}S$ are contained within bounds of $\mathcal{O}(1)$ (we may carry out a similar argument for $\widehat{S}_1^{-1}S$).

We therefore consider the Rayleigh quotient

$$\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_2\mathbf{v}} = \frac{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v} + \frac{\tau}{\beta}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}}{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\mathbf{v} + \frac{\tau}{\beta}\mathbf{v}^T\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}\mathbf{v} + \frac{2}{\sqrt{\beta}}\mathbf{v}^T\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}\mathbf{v}}.$$

Consider first the case where $\mathbf{v}\in\text{null}\big(\widehat{\mathcal{M}}\big)$. By the definition of $\widehat{\mathcal{M}}$, this would mean that $\mathbf{v} \in \text{null}\big(\text{blkdiag}\left(M_\Gamma,...,M_\Gamma\right)\big) = \text{null}\big(\text{blkdiag}\left(N_bM_b^{-1}N_b^T,...,N_bM_b^{-1}N_b^T\right)\big)$, and therefore that $\mathbf{v}\in\text{null}\big(\mathcal{N}^T\big)$ also. As a result,

$$\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}_2\mathbf{v}} = \frac{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}}{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\mathbf{v}} = \mathcal{O}(1),$$

using the spectral equivalence of $\mathcal{M}_{1/2}$ and $h\widehat{\mathcal{M}}_\Gamma$. In the case $\mathbf{v}\notin\text{null}\big(\widehat{\mathcal{M}}\big)$, we look at the Rayleigh quotient written in the form

$$\frac{1}{\frac{\tau^{-1}\mathbf{v}^T\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}} + \frac{1}{\sqrt{\beta}}\,\frac{\mathbf{v}^T\Big(\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}+\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\Big)\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}}}}.$$

Neglecting multiplicative constants for now, we see that $h\widehat{\mathcal{M}}_\Gamma\approx\mathcal{M}_{1/2}\approx h^2I$ and $\widehat{\mathcal{M}}_\Gamma\approx\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}\approx hI$, and so

$$\frac{\tau^{-1}\mathbf{v}^T\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}} = \mathcal{O}(1).$$

To simplify the analysis, we now once more approximate $\mathcal{K}$ by its block diagonal, that is $\bar{\mathcal{L}}\approx\mathcal{K}$. We again refer to Figure 8.1 for a demonstration of this feature.

We use this to approximate the quotient

$$\frac{1}{\sqrt{\beta}}\,\frac{\mathbf{v}^T\Big(\mathcal{K}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}+\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\mathcal{K}^T\Big)\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}}$$

by

$$\frac{1}{\sqrt{\beta}}\,\frac{\mathbf{v}^T\left(\bar{\mathcal{L}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\widehat{\mathcal{M}}+\widehat{\mathcal{M}}\big(h\widehat{\mathcal{M}}_\Gamma\big)^{-1}\bar{\mathcal{L}}\right)\mathbf{v}}{\tau^{-1}\mathbf{v}^T\bar{\mathcal{L}}\mathcal{M}_{1/2}^{-1}\bar{\mathcal{L}}\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{N}\mathcal{M}_{b,1/2}^{-1}\mathcal{N}^T\mathbf{v}}=:\frac{T_5}{T_6}.$$

Applying the same strategy as in Section 4.1 for the time-independent boundary control problem then gives that (neglecting multiplicative constants)

$$\frac{T_5}{T_6}=\frac{2\beta^{-1/2}\xi h^{-2}h^{1/2}h}{\tau^{-1}h^{-2}\xi^2+\tau\beta^{-1}h}=\frac{2\beta^{-1/2}h^{-1/2}\xi}{\tau^{-1}h^{-2}(\xi^2+\tau^2\beta^{-1}h^3)}=\frac{2\tau\beta^{-1/2}h^{3/2}\xi}{\xi^2+\tau^2\beta^{-1}h^3}=\frac{2\upsilon_\tau\xi}{\xi^2+\upsilon_\tau^2}\leq 1,$$

with $\upsilon_\tau=\tau\beta^{-1/2}h^{3/2}$, and ξ corresponding to the relevant eigenvalue of $\bar{\mathcal{L}}$. We can also mimic our argument of Section 4.1 for the time-independent boundary control case, to obtain that $\frac{T_5}{T_6}$ is typically non-negative. We may utilize these bounds for $\frac{T_5}{T_6}$ to obtain that

$$\lambda_{\min}(\widehat{S}_2^{-1}S)\geq\widetilde{c}_5,\quad \lambda_{\max}(\widehat{S}_2^{-1}S)\leq\widetilde{C}_5,$$

where $\widetilde{c}_5$ and $\widetilde{C}_5$ are positive constants independent of h, β and τ.

As for the time-independent boundary control case, we have provided a heuristic argument that the eigenvalues of the preconditioned Schur complement are bounded within an interval of $\mathcal{O}(1)$, but we are unable to rigorously pin the eigenvalues into a precise range.

Let us now consider a similar argument for the distributed control problems. We focus here on the all-times case using the discretize-then-optimize approach, but we may also apply this argument to the (perturbed) Schur complements for the other two distributed control problems we discussed.

Here, the relevant Rayleigh quotient is given by

$$\begin{aligned}\frac{\mathbf{v}^TS\mathbf{v}}{\mathbf{v}^T\widehat{S}\mathbf{v}}&=\frac{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\frac{\tau}{\beta}\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}}{\frac{1}{\tau}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\frac{\tau}{\beta}\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}+\frac{2}{\sqrt{\beta}}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}}\\&=\frac{1}{1+\frac{1}{\sqrt{\beta}}\,\frac{\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}+\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}+\tau\beta^{-1}\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}}},\end{aligned}$$

and so the quantity which we consider is

$$\frac{1}{\sqrt{\beta}} \frac{\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v} + \mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v}}{\tau^{-1}\mathbf{v}^T\mathcal{K}\mathcal{M}_{1/2}^{-1}\mathcal{K}^T\mathbf{v} + \tau\beta^{-1}\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}} = \frac{\mathbf{a}^T\mathbf{b} + \mathbf{b}^T\mathbf{a}}{\mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b}}, \tag{8.7}$$

where $\mathbf{a} = \tau^{-1/2}\mathcal{M}_{1/2}^{-1/2}\mathcal{K}^T\mathbf{v}$ and $\mathbf{b} = \beta^{-1/2}\tau^{1/2}\mathcal{M}_{1/2}^{-1/2}\mathcal{M}\mathbf{v}$. Using a similar argument as for convection-diffusion control in Theorem 13 of Chapter 5, we conclude that $\frac{\mathbf{a}^T\mathbf{b}+\mathbf{b}^T\mathbf{a}}{\mathbf{a}^T\mathbf{a}+\mathbf{b}^T\mathbf{b}} \leq 1$ (as this is equivalent to writing $\|\mathbf{a}-\mathbf{b}\|_2 \geq 0$). To demonstrate a lower bound, as for the boundary control problem above, let us approximate (8.7) by

$$\frac{1}{\sqrt{\beta}} \frac{\mathbf{v}^T\bar{\mathcal{L}}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v} + \mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\bar{\mathcal{L}}\mathbf{v}}{\tau^{-1}\mathbf{v}^T\bar{\mathcal{L}}\mathcal{M}_{1/2}^{-1}\bar{\mathcal{L}}\mathbf{v} + \tau\beta^{-1}\mathbf{v}^T\mathcal{M}\mathcal{M}_{1/2}^{-1}\mathcal{M}\mathbf{v}}, \tag{8.8}$$

of which the denominator is clearly strictly positive, since both the individual terms must be, and the numerator is equal to $\frac{2}{\sqrt{\beta}}\mathbf{v}^T\text{blkdiag}(2L_d, L_d, ..., L_d, 2L_d)\mathbf{v} > 0$, as L_d is positive definite. Combining these observations gives that the quantity (8.8) is positive. These arguments concerning bounds for (8.7) place the values of $\frac{\mathbf{v}^T S\mathbf{v}}{\mathbf{v}^T\widehat{S}\mathbf{v}}$, and hence the eigenvalues of $\widehat{S}^{-1}S$, within the range $\left[\frac{1}{2}, 1\right)$.

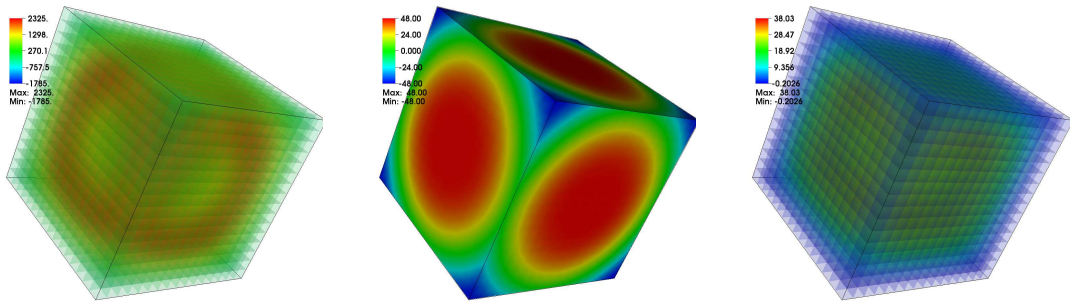
We may arrive at similar conclusions regarding the eigenvalues of the preconditioned Schur complements for the all-times case using the optimize-then-discretize approach, as well as the final-time case. We highlight once more that for these problems the Schur complements of the matrix systems do not exist, so an argument may only be applied to the perturbed system. However, as the numerical results of the next section indicate, our approach yields convergence in few iterations nonetheless.

8.3 Numerical Results

We have developed solvers for a range of time-dependent PDE-constrained optimization problems that involve storing only a few matrices which are very small compared to the matrix system we wish to solve. We now demonstrate the effectiveness of our solvers for each of the problems by carrying out numerical experiments.

All results shown are computed within the deal.II [4] framework using **Q1** finite element basis functions.[34] We use a smoothed aggregation algebraic multigrid routine to approximate the relevant matrices, within the Trilinos ML package [44]. We apply

[34]The author acknowledges that all figures and numerical results in this chapter were created using code written by Martin Stoll as part of a collaboration with the author.



(a) Control (b) Desired state (c) Computed state

Figure 8.2: Control, desired state and state for the all-times case example with $\beta = 10^{-4}$, at the 15-th time-step.

		Iterations			CPU Time		
h	DoF	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$
2^{-4}	$4,913$	10	12	12	2	2	2
2^{-5}	$35,937$	10	12	12	14	17	18
2^{-6}	$274,625$	10	12	12	148	171	170

Table 8.1: Number of Minres iterations and CPU times to solve the discretize-then-optimize formulation of the all-times case example, for a range of h and β. Also stated are the number of spatial degrees of freedom.

the Minres algorithm, and solve to a tolerance of 10^{-4}.[35] The experiments are carried out with $T = 1$ in all cases, and with $\tau = 0.05$ (corresponding to 20 time-steps). We take the domain Ω to be the unit cube $[0, 1]^3$.

We first consider the following distributed control problem corresponding to the all-times case:

$$\begin{aligned} &\min_{y,u} \; J_1(y,u) \\ &\text{s.t} \quad y_t - \nabla^2 y = u, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T], \\ &\qquad\qquad y = 0, \quad \text{on } \partial\Omega, \\ &\qquad\qquad y = 0, \quad \text{at } t = 0, \end{aligned}$$

[35]The computations were carried out on a Centos Linux machine with Intel(R) Xeon(R) CPU X5650, 2.67GHz CPUs and 48GB of RAM.

		Iterations			CPU Time		
h	DoF	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$
2^{-4}	$4,913$	12	10	8	3	2	1
2^{-5}	$35,937$	12	10	10	16	14	14
2^{-6}	$274,625$	14	10	10	196	152	147

Table 8.2: Number of Minres iterations and CPU times to solve the optimize-then-discretize formulation of the all-times case example, for a range of h and β. DoF is the number of variables in space only.

where

$$\widehat{y} = 64t \sin\left(2\pi\left(\left(x_1 - \frac{1}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^2 + \left(x_3 - \frac{1}{2}\right)^2\right)\right),$$

and with $\mathbf{x} = [x_1, x_2, x_3]^T$ denoting the spatial coordinates. We present an illustration of the computed state, computed control and desired state, at one particular time-step, in Figure 8.2. The iteration numbers and CPU times for solving the discretize-then-optimize formulation of this problem, using the preconditioner described in Section 8.1.1, are given in Table 8.1. The corresponding results for the optimize-then-discretize strategy, using the preconditioner derived in Section 8.1.1, are shown in Table 8.2. In all tables in this section, the degrees of freedom shown correspond to the number of points in the spatial discretization. The overall size of the matrix system being solved is much larger – for instance for distributed control problems the size is $3 \times N_t = 3 \times 20$ times this number. The largest matrix system we solve is thus $16,477,500 \times 16,477,500$.

To demonstrate the feasibility of our approach for the final-time case, we consider the following test problem for the final-time case of the distributed control problem:

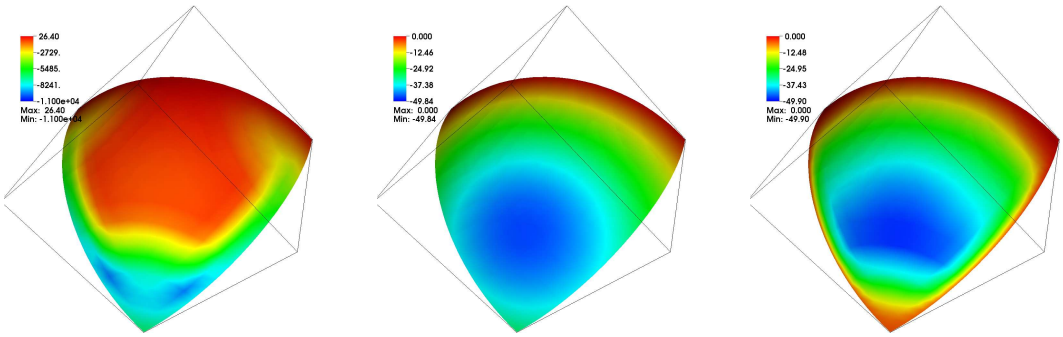
$$\begin{aligned} \min_{y,u} \ & J_2(y,u) \\ \text{s.t} \quad & y_t - \nabla^2 y = u, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T], \\ & y = 0, \quad \text{on } \partial\Omega, \\ & y = 0, \quad \text{at } t = 0, \end{aligned}$$

where

$$\widehat{y} = -64x_1 \exp\left(-\left(\left(x_1 - \frac{1}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^2 + \left(x_3 - \frac{1}{2}\right)^2\right)\right).$$

An illustration of the computed state, computed control and desired state, at one particular time-step, is shown in Figure 8.3. In Table 8.3 we present the iteration numbers and CPU times for solving this problem using the preconditioner described in Section 8.1.1.

Our results for these distributed control problems indicate that our preconditioners are highly robust with respect to h and β, with convergence achieved in low iteration numbers in all three cases, for all problems considered. Even when our solution strategy involves perturbing the original matrix system to create the preconditioner, this does not seem to affect the performance significantly.

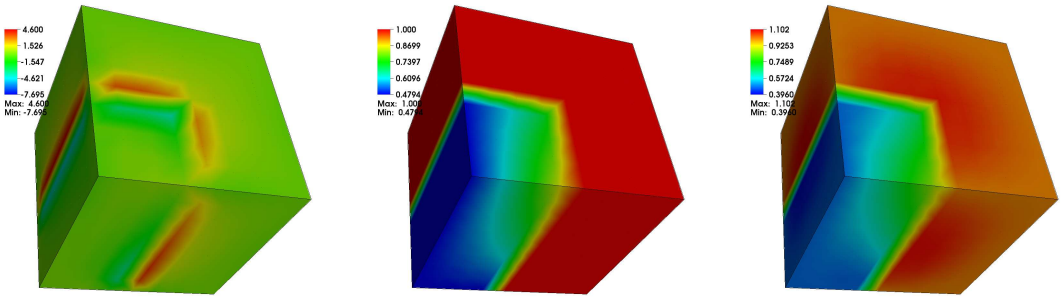


(a) Control (b) Desired state (c) Computed state

Figure 8.3: Control, desired state and state for the final-time case example with $\beta = 10^{-4}$, at the final time-step.

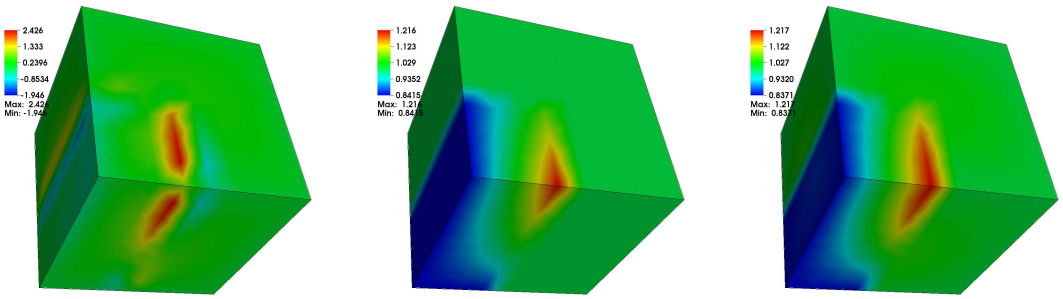
		Iterations			CPU Time		
h	DoF	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$
2^{-4}	4,913	14	13	9	9	9	6
2^{-5}	35,937	16	13	12	72	59	55
2^{-6}	274,625	16	15	13	624	677	531

Table 8.3: Number of MINRES iterations and CPU times to solve the final-time case example, for a range of h and β. DoF is the number of variables in space only.



(a) Control (b) Desired state (c) Computed state

Figure 8.4: Control, desired state and state for time-dependent boundary control example with $\beta = 10^{-6}$, at the 10-th time-step.



(a) Control (b) Desired state (c) Computed state

Figure 8.5: Control, desired state and state for time-dependent boundary control example with $\beta = 10^{-6}$, at the final time-step.

$\widehat{S}_1$		Iterations			CPU Time		
h	DoF	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$
2^{-4}	4,913	34	38	28	7	7	6
2^{-5}	35,937	38	48	38	49	62	48
2^{-6}	274,625	48	62	58	620	800	725
$\widehat{S}_2$		Iterations			CPU Time		
h	DoF	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$	$\beta = 10^{-2}$	$\beta = 10^{-4}$	$\beta = 10^{-6}$
2^{-4}	4,913	40	42	36	8	8	7
2^{-5}	35,937	50	59	42	65	73	54
2^{-6}	274,625	62	80	68	808	1002	855

Table 8.4: Number of Minres iterations and CPU times to solve the discretize-then-optimize formulation of the time-dependent boundary control example, for a range of h and β. Results are given when the Schur complement approximations $\widehat{S}_1$ and $\widehat{S}_2$ are used. DoF is the number of variables in space only.

To highlight the effectiveness of our approach for time-dependent boundary control problems, we also consider the following example:

$$\begin{aligned} \min_{y,u} \ & J_3(y,u) \\ \text{s.t} \quad & y_t - \nabla^2 y = 0, \quad \text{for } (\mathbf{x},t) \in \Omega \times [0,T], \\ & \frac{\partial y}{\partial n} = u, \quad \text{on } \partial\Omega, \\ & y = 0, \quad \text{at } t = 0, \end{aligned}$$

where

$$\widehat{y} = \begin{cases} \sin t + x_1 x_2 x_3 & \text{if } x_1 > \frac{1}{2} \text{ and } x_2 < \frac{1}{2}, \\ 1 & \text{otherwise.} \end{cases}$$

In Figures 8.4 and 8.5, we show the computed state, computed control and desired state for this test problem at different time-steps. In Table 8.4, we present iteration numbers and CPU times for solving this problem using the preconditioner described in Section 8.1.2, with Schur complement approximations $\widehat{S}_1$ and $\widehat{S}_2$. We see that the iteration numbers are higher than for the distributed control case, but very reasonable considering the added complexity of the problem. We observe that the Schur

complement approximation $\widehat{S}_1$ performs marginally better than $\widehat{S}_2$ for this problem.

Overall, we conclude that our preconditioning strategy for time-dependent optimal control problems is a highly effective one, and computational tests indicate that our preconditioners result in convergence of MINRES in a small number of iterations for a wide range of h, β and τ.

8.4 Summary

In this chapter, we have established that we may derive a strategy analogous to our approach for time-independent optimal control problems in order to develop fast iterative solvers for optimal control problems with an additional time-dependent element. As for time-independent problems, we were able to develop preconditioners for distributed and boundary control problems.

To do so in the distributed control case, we needed to take more care in generating our Schur complement approximation than for time-independent problems, so as to incorporate the time-stepping scheme involved. We also needed to include a perturbation of the $(1,1)$-block for some problems where this matrix was singular, and we were able to develop a heuristic approach to determine the optimal choice of the relevant perturbation parameter. For the Neumann boundary control case, we proposed Schur complement approximations using similar heuristics as for the time-independent problem. By building these strategies into our preconditioners, we were able to develop solvers robust with respect to h and β, as well as the time-step τ used.

Finding a way of dealing with such time-dependent optimal control problems can potentially open the door to finding methods for solving a wide range of complex physical problems, so we believe the work presented in this chapter is a crucial step towards this. An important additional aspect of this work is that the implementation required the storage of only a few matrices (M and K for distributed control problems, and additionally M_b and N_b for boundary control problems), which are very small compared to the matrix system as a whole. Furthermore, only a sparse matrix-vector multiply with the big matrix is required in the MINRES method we employ. The fact that in our method the large system exists "only in our head" leads to the possibility of parallelization of the solvers presented here – this is one major piece of future work which we wish to explore.

CHAPTER 9

Optimal Control of Chemical Processes

In this chapter, we turn our attention to a problem motivated by a real-world application, namely the optimal control of a reaction-diffusion problem arising from chemical processes.[36] The matrix systems that result from the numerical solution of this problem are considerably more complex than the systems which we have considered so far, and the methods needed to solve them will require components used to solve many of the problems previously discussed in this thesis, for instance nonlinear solvers, techniques for incorporating control constraints, and use of time-stepping schemes. Having introduced the problem that we will consider, and derived the resulting matrix systems, the iterative solution of which we seek, we explain how each of these components of our iterative solver should be applied. For brevity, we omit some of the technical details of our working, and refer the reader to [89] for the complete derivation.

This chapter is structured as follows. In Section 9.1, we state the optimal control problem to be considered, and derive the relevant matrix systems that we are required to solve. In Section 9.2, we describe the iterative solver and preconditioning strategy we use to solve these matrix systems. In Section 9.3, we present numerical results to verify the effectiveness of our recommended approach, and in Section 9.4, we make

[36]This chapter is based on the following paper, which is Ref. [89]:

J. W. Pearson, and M. Stoll, *Fast Iterative Solution of Reaction-Diffusion Control Problems Arising from Chemical Processes*, SIAM Journal on Scientific Computing, 35, pp.B987–B1009, 2013.

some concluding remarks.

9.1 Problem Formulation and Matrix Systems

The problem that we consider in this chapter is based on the framework introduced by Barthel, John and Tröltzsch in [5], Griesse in [49, 50], and Griesse and Volkwein in [51, 52]. The cost functional within the optimal control problem is given by

$$J(u,v,c) = \frac{\alpha_u}{2}\left\|u-u_Q\right\|^2_{L_2(Q)} + \frac{\alpha_v}{2}\left\|v-v_Q\right\|^2_{L_2(Q)} + \frac{\alpha_{TU}}{2}\left\|u(\mathbf{x},T)-u_\Omega\right\|^2_{L_2(\Omega)} + \frac{\alpha_{TV}}{2}\left\|v(\mathbf{x},T)-v_\Omega\right\|^2_{L_2(\Omega)} + \frac{\alpha_c}{2}\left\|c\right\|^2_{L_2(\partial Q)}.$$

Here we work on a space-time domain $Q := \Omega \times [0,T]$, where the spatial domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$. We define the space-time boundary by $\partial Q := \partial\Omega \times [0,T]$. The variables u and v represent concentrations of chemical reactants, and are the state variables of the problem.[37] The variable c is the control variable. We have four desired states in this problem: two on the entire space-time domain (u_Q and v_Q) and two solely defined at the final time (u_Ω and v_Ω). These quantities usually arise from measurement or experimental observation.

The PDE constraints subject to which $J(u,v,c)$ is minimized are given by the following reaction-diffusion system describing chemical reactions:

$$u_t - D_1\nabla^2 u + k_1 u = -\gamma_1 uv, \quad \text{in } Q, \tag{9.1}$$

$$v_t - D_2\nabla^2 v + k_2 v = -\gamma_2 uv, \quad \text{in } Q, \tag{9.2}$$

$$D_1\frac{\partial u}{\partial n} + b(\mathbf{x},t,u) = c, \quad \text{on } \partial\Omega, \tag{9.3}$$

$$D_2\frac{\partial v}{\partial n} + \widetilde{\epsilon} v = 0, \quad \text{on } \partial\Omega, \tag{9.4}$$

$$u(\mathbf{x},0) = u_0(\mathbf{x}), \quad \text{on } \Omega, \tag{9.5}$$

$$v(\mathbf{x},0) = v_0(\mathbf{x}), \quad \text{on } \Omega, \tag{9.6}$$

$$c_a \le c \le c_b, \quad \text{a.e. on } \partial\Omega. \tag{9.7}$$

In our problem set-up, the constants D_1, D_2, k_1, k_2, γ_1, γ_2, α_u, α_v, α_{TU}, α_{TV}, α_c and $\widetilde{\epsilon}$ are all non-negative; some constants such as D_1, D_2 and α_c must be strictly positive.

[37] We note the change in notation from previous chapters, where u represented a control variable – we make the change here to be consistent with the notation of the literature on this subject.

Equation (9.7) is an additional box constraint on the control variable expressed in terms of known functions c_a and c_b. For the remainder of this chapter, we assume that $b(\mathbf{x}, t, u) = 0$ and $\widetilde{\epsilon} = 0$, as done in [51]. We see from the structure of the above set-up that we are dealing with a boundary control problem.

Due to the wide potential applicability of problems of this form, it is highly desirable to investigate the numerical solution of such problems. To do so, we must first find a way of handling the nonlinearity of the PDE constraints. We may do this by using a Sequential Quadratic Programming (SQP) method, otherwise known as a Lagrange-Newton method. We first consider Lagrange-Newton iteration for the problem without control constraints – that is an optimal control problem with the state equations given by (9.1)–(9.6), but without the condition (9.7). In this case, we may use an optimize-then-discretize strategy for constructing the matrix systems, with the continuous Lagrangian given by

$$\begin{aligned}\mathcal{L}_{OTD}(u, v, c, p, q) = J(u, v, c) &+ \int_Q p_Q(u_t - D_1\nabla^2 u + k_1 u + \gamma_1 uv)\ \mathrm{d}Q \\ &+ \int_Q q_Q(v_t - D_2\nabla^2 v + k_2 v + \gamma_2 uv)\ \mathrm{d}Q \\ &+ \int_{\partial Q} p_{\partial Q}\left(D_1\frac{\partial u}{\partial n} - c\right)\ \mathrm{d}s + \int_{\partial Q} q_{\partial Q}\left(D_2\frac{\partial v}{\partial n}\right)\ \mathrm{d}s,\end{aligned}$$

where p and q are the two adjoint variables; we have split these up into interior and boundary parts (p_Q & $p_{\partial Q}$, and q_Q & $q_{\partial Q}$).

We now wish to generate optimality conditions from this continuous Lagrangian.[38] We will consider the case $\alpha_{TU} = \alpha_{TV} = 0$ from now on, but note that the case where these constants are non-zero is similar (this relates to the final-time case of the previous chapter). Differentiating the cost functional $\mathcal{L}_{OTD}$ with respect to the adjoint variables p and q gives us back the state equations (9.1)–(9.6). We may also

[38] As stated above, this is an optimize-then-discretize approach. We may also consider a discretize-then-optimize approach, and we again note that it is desirable to select discretization schemes such that the two approaches coincide.

differentiate $\mathcal{L}_{OTD}$ with respect to u and v to obtain the adjoint equations

$$\begin{aligned}
-p_t - D_1\nabla^2 p + k_1 p + \gamma_1 pv + \gamma_2 qv &= \alpha_u(u_Q - u), \quad \text{in } Q,\\
-q_t - D_2\nabla^2 q + k_2 q + \gamma_1 pu + \gamma_2 qu &= \alpha_v(v_Q - v), \quad \text{in } Q,\\
D_1\frac{\partial p}{\partial n} &= 0, \quad \text{on } \partial\Omega,\\
D_2\frac{\partial q}{\partial n} &= 0, \quad \text{on } \partial\Omega,\\
p(\mathbf{x}, T) &= 0, \quad \text{on } \Omega,\\
q(\mathbf{x}, T) &= 0, \quad \text{on } \Omega,
\end{aligned}$$

and differentiate $\mathcal{L}_{OTD}$ with respect to c to obtain the gradient equation

$$\alpha_c c - p = 0, \quad \text{in } Q.$$

As the state and gradient equations are a nonlinear system of equations, we now need to apply a method for linearizing the equations. In this work, we use a Newton iteration to do so. Let us write

$$u = \bar{u} + \delta u, \quad v = \bar{v} + \delta v, \quad c = \bar{c} + \delta c, \quad p = \bar{p} + \delta p, \quad q = \bar{q} + \delta q,$$

at each Newton step, where $\bar{u}$, $\bar{v}$, $\bar{c}$, $\bar{p}$, $\bar{q}$ denote the previous Newton iterates, and δu, δv, δc, δp, δq denote the updates in the solution at the particular Newton step. We may use this notation to write for the state equations that

$$\begin{aligned}
(\bar{u} + \delta u)_t - D_1\nabla^2(\bar{u} + \delta u) + k_1(\bar{u} + \delta u) + \gamma_1(\bar{u} + \delta u)(\bar{v} + \delta v) &= 0,\\
(\bar{v} + \delta v)_t - D_2\nabla^2(\bar{v} + \delta v) + k_2(\bar{v} + \delta v) + \gamma_2(\bar{u} + \delta u)(\bar{v} + \delta v) &= 0,\\
D_1\frac{\partial(\bar{u} + \delta u)}{\partial n} - (\bar{c} + \delta c) &= 0, \quad \text{on } \partial\Omega,\\
D_2\frac{\partial(\bar{v} + \delta v)}{\partial n} &= 0, \quad \text{on } \partial\Omega,\\
(\bar{u} + \delta u)(\mathbf{x}, 0) &= u_0(\mathbf{x}),\\
(\bar{v} + \delta v)(\mathbf{x}, 0) &= v_0(\mathbf{x}),
\end{aligned}$$

which after simplication gives that

$$
\begin{aligned}
(\delta u)_t - D_1\nabla^2(\delta u) + k_1(\delta u) + \gamma_1(\bar{v}(\delta u) + \bar{u}(\delta v)) &= -(\bar{u}_t - D_1\nabla^2\bar{u} + k_1\bar{u} + \gamma_1\bar{u}\bar{v}),\\
(\delta v)_t - D_2\nabla^2(\delta v) + k_2(\delta v) + \gamma_2(\bar{v}(\delta u) + \bar{u}(\delta v)) &= -(\bar{v}_t - D_2\nabla^2\bar{v} + k_2\bar{v} + \gamma_2\bar{u}\bar{v}),\\
D_1\frac{\partial(\delta u)}{\partial n} - \delta c = 0, &\quad \text{on } \partial\Omega,\\
D_2\frac{\partial(\delta v)}{\partial n} = 0, &\quad \text{on } \partial\Omega,\\
(\delta u)(\mathbf{x}, 0) = 0,&\\
(\delta v)(\mathbf{x}, 0) = 0,&
\end{aligned}
$$

neglecting second order terms in the Newton updates.

We may apply a similar strategy to the adjoint equations, to obtain that

$$
\begin{aligned}
-(\bar{p} + \delta p)_t - D_1\nabla^2(\bar{p} + \delta p) + k_1(\bar{p} + \delta p) + \gamma_1(\bar{p} + \delta p)(\bar{v} + \delta v)&\\
+\gamma_2(\bar{q} + \delta q)(\bar{v} + \delta v) = \alpha_u(u_Q - (\bar{u} + \delta u)),&\\
-(\bar{q} + \delta q)_t - D_2\nabla^2(\bar{q} + \delta q) + k_2(\bar{q} + \delta q) + \gamma_1(\bar{p} + \delta p)(\bar{u} + \delta u)&\\
+\gamma_2(\bar{q} + \delta q)(\bar{u} + \delta u) = \alpha_v(v_Q - (\bar{v} + \delta v)),&\\
D_1\frac{\partial(\bar{p} + \delta p)}{\partial n} = 0, &\quad \text{on } \partial\Omega,\\
D_2\frac{\partial(\bar{q} + \delta q)}{\partial n} = 0, &\quad \text{on } \partial\Omega,\\
(\bar{p} + \delta p)(\mathbf{x}, T) = 0,&\\
(\bar{q} + \delta q)(\mathbf{x}, T) = 0,&
\end{aligned}
$$

which gives after simple manipulation (again neglecting second order terms) that

$$\begin{aligned}
-(\delta p)_t - D_1\nabla^2(\delta p) + k_1(\delta p) + \gamma_1(\bar{p}(\delta v) + \bar{v}(\delta p)) + \gamma_2(\bar{q}(\delta v) + \bar{v}(\delta q)) + \alpha_u(\delta u) \\
= \alpha_u u_Q - (-\bar{p}_t - D_1\nabla^2\bar{p} + k_1\bar{p} + (\gamma_1\bar{p} + \gamma_2\bar{q})\bar{v} + \alpha_u\bar{u}), \\
-(\delta q)_t - D_2\nabla^2(\delta q) + k_2(\delta q) + \gamma_1(\bar{p}(\delta u) + \bar{u}(\delta p)) + \gamma_2(\bar{q}(\delta u) + \bar{u}(\delta q)) + \alpha_v(\delta v) \\
= \alpha_v v_Q - (-\bar{q}_t - D_2\nabla^2\bar{q} + k_2\bar{q} + (\gamma_1\bar{p} + \gamma_2\bar{q})\bar{u} + \alpha_v\bar{v}), \\
D_1\frac{\partial(\delta p)}{\partial n} = 0, \quad \text{on } \partial\Omega, \\
D_2\frac{\partial(\delta q)}{\partial n} = 0, \quad \text{on } \partial\Omega, \\
(\delta p)(\mathbf{x}, T) = 0, \\
(\delta q)(\mathbf{x}, T) = 0.
\end{aligned}$$

Finally, applying Newton iteration to the gradient equation is a simple process, and gives at each step that

$$\alpha_c(\delta c) - \delta p = -(\alpha_c\bar{c} - \bar{p}).$$

The equations resulting from the above Newton process may be summed up in continuous form (excluding boundary conditions of the PDEs) as the following matrix system at the $(k+1)$-th Newton step:

$$\mathcal{A}\begin{bmatrix} u_{(k+1)} \\ v_{(k+1)} \\ c_{(k+1)} \\ p_{(k+1)} \\ q_{(k+1)} \end{bmatrix} = \mathcal{A}\begin{bmatrix} u_{(k)} + \delta u \\ v_{(k)} + \delta v \\ c_{(k)} + \delta c \\ p_{(k)} + \delta p \\ q_{(k)} + \delta q \end{bmatrix} = \begin{bmatrix} \alpha_u u_Q + (\gamma_1 p_{(k)} + \gamma_2 q_{(k)})v_{(k)} \\ \alpha_v v_Q + (\gamma_1 p_{(k)} + \gamma_2 q_{(k)})u_{(k)} \\ 0 \\ \gamma_1 u_{(k)} v_{(k)} \\ \gamma_2 u_{(k)} v_{(k)} \end{bmatrix} =: \widetilde{\mathbf{b}},$$

where

$$\mathcal{A} = \begin{bmatrix} \alpha_u Id & \gamma_1 p_{(k)} + \gamma_2 q_{(k)} & 0 & \mathcal{L}'_u & \gamma_2 v_{(k)} \\ \gamma_1 p_{(k)} + \gamma_2 q_{(k)} & \alpha_v Id & 0 & \gamma_1 u_{(k)} & \mathcal{L}'_v \\ 0 & 0 & \alpha_c D_1^{-1} Id & -D_1^{-1} Id & 0 \\ \mathcal{L}_u & \gamma_1 u_{(k)} & -D_1^{-1} Id & 0 & 0 \\ \gamma_2 v_{(k)} & \mathcal{L}_v & 0 & 0 & 0 \end{bmatrix}, \tag{9.8}$$

with Id corresponding to the identity operator, and

$$\mathcal{L}_u = \frac{\partial}{\partial t} - D_1\nabla^2 + k_1 Id + \gamma_1 v_{(k)}, \quad \mathcal{L}'_u = -\frac{\partial}{\partial t} - D_1\nabla^2 + k_1 Id + \gamma_1 v_{(k)},$$
$$\mathcal{L}_v = \frac{\partial}{\partial t} - D_2\nabla^2 + k_2 Id + \gamma_2 u_{(k)}, \quad \mathcal{L}'_v = -\frac{\partial}{\partial t} - D_2\nabla^2 + k_2 Id + \gamma_2 u_{(k)}.$$

Here, $u_{(k)}$, $v_{(k)}$, $c_{(k)}$, $p_{(k)}$, $q_{(k)}$ denote the approximations of u, v, c, p, q at the k-th Newton step.

We may now discretize the system of equations (9.8) obtained at each Newton step. Doing this results in the following system at the $(k+1)$-th step:

$$\begin{bmatrix} \tau\widetilde{\mathcal{M}} & 0 & \widetilde{\mathcal{K}}^T \\ 0 & \alpha_c\tau D_1^{-1}\mathcal{M}_b & -\tau D_1^{-1}\widetilde{\mathcal{N}}^T \\ \widetilde{\mathcal{K}} & -\tau D_1^{-1}\widetilde{\mathcal{N}} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k+1)} \\ \mathbf{c}_{(k+1)} \\ \mathbf{p}_{(k+1)} \end{bmatrix} = \mathbf{b}, \tag{9.9}$$

where $\mathbf{y}_{(k+1)}$ relates to the discrete values of $u_{(k+1)}$ and $v_{(k+1)}$ at each time-step in turn, $\mathbf{p}_{(k+1)}$ similarly corresponds to the discrete values of $p_{(k+1)}$ and $q_{(k+1)}$ at each time-step, and $\mathbf{c}_{(k+1)}$ relates to the values of $c_{(k+1)}$ at each time-step. Here, the matrix $\widetilde{\mathcal{M}}$ is given as follows:

$$\widetilde{\mathcal{M}} = \begin{bmatrix} \widetilde{M}^{(1)} & & & & \\ & \widetilde{M}^{(2)} & & & \\ & & \ddots & & \\ & & & \widetilde{M}^{(N_t-1)} & \\ & & & & \widetilde{M}^{(N_t)} \end{bmatrix},$$

where

$$\widetilde{M}^{(j)} = \begin{bmatrix} \alpha_u M & \gamma_1 M_{p_j} + \gamma_2 M_{q_j} \\ \gamma_1 M_{p_j} + \gamma_2 M_{q_j} & \alpha_v M \end{bmatrix}, \quad j = 1, ..., N_t,$$

with M_{p_j} and M_{q_j} matrices containing terms of the form $\int_\Omega \bar{p}(\mathbf{x}, j\tau)\phi_i\phi_l \, \mathrm{d}\Omega$ and $\int_\Omega \bar{q}(\mathbf{x}, j\tau)\phi_i\phi_l \, \mathrm{d}\Omega$ at the (i,l)-th entries. Also, the matrix $\widetilde{\mathcal{K}}$, which corresponds to

the forward problem, is given by

$$\widetilde{\mathcal{K}} = \begin{bmatrix} \widetilde{L}^{(1)} & & & & \\ -\widetilde{M}_2 & \widetilde{L}^{(2)} & & & \\ & \ddots & \ddots & & \\ & & -\widetilde{M}_2 & \widetilde{L}^{(N_t-1)} & \\ & & & -\widetilde{M}_2 & \widetilde{L}^{(N_t)} \end{bmatrix},$$

where

$$\widetilde{L}^{(j)} = \begin{bmatrix} M + \tau(D_1 K + k_1 M + \gamma_1 M_{v_j}) & \tau\gamma_1 M_{u_j} \\ \tau\gamma_2 M_{v_j} & M + \tau(D_2 K + k_2 M + \gamma_2 M_{u_j}) \end{bmatrix},$$

$$\widetilde{M}_2 = \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix},$$

with M_{u_j} and M_{v_j} matrices containing entries of the form $\int_\Omega \bar{u}(\mathbf{x}, j\tau)\phi_i\phi_l \, \mathrm{d}\Omega$ and $\int_\Omega \bar{v}(\mathbf{x}, j\tau)\phi_i\phi_l \, \mathrm{d}\Omega$, and

$$\widetilde{\mathcal{N}} = \begin{bmatrix} N_b & & & & & & \\ & 0 & & & & & \\ & & N_b & & & & \\ & & & 0 & & & \\ & & & & \ddots & & \\ & & & & & N_b & \\ & & & & & & 0 \end{bmatrix},$$

with N_b as defined in (2.13). The matrix $\mathcal{M}_b$ is given by blkdiag $(M_b, M_b, ..., M_b, M_b)$, and consists of boundary mass matrices M_b defined by (2.14). Finally, the vector $\mathbf{b}$ corresponds to the discrete representation of

$$\begin{bmatrix} \alpha_u u_Q + (\gamma_1 p_{(k)} + \gamma_2 q_{(k)}) v_{(k)} \\ \alpha_v v_Q + (\gamma_1 p_{(k)} + \gamma_2 q_{(k)}) u_{(k)} \\ 0 \\ \gamma_1 u_{(k)} v_{(k)} \\ \gamma_2 u_{(k)} v_{(k)} \end{bmatrix}.$$

Incorporation of Control Constraints

Suppose now that we wish to additionally include the condition (9.7) in our model, that is, we impose an additional control constraint of the form

$$c_a \leq c \leq c_b.$$

For problems with this additional constraint, we wish to utilize the Moreau-Yosida regularization technique introduced in Section 2.1.3.

We therefore consider minimizing

$$J(u,v,c) + \frac{1}{2\epsilon}\left\|\min\{0, c-c_a\}\right\|^2_{L_2(\partial Q)} + \frac{1}{2\epsilon}\left\|\max\{0, c-c_b\}\right\|^2_{L_2(\partial Q)}.$$

At each SQP step, this involves solving matrix systems of the form

$$\begin{bmatrix} \tau\widetilde{\mathcal{M}} & 0 & \widetilde{\mathcal{K}}^T \\ 0 & \alpha_c\tau D_1^{-1}\mathcal{L}_b & -\tau D_1^{-1}\widetilde{\mathcal{N}}^T \\ \widetilde{\mathcal{K}} & -\tau D_1^{-1}\widetilde{\mathcal{N}} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_{(k)} \\ \mathbf{c}_{(k)} \\ \mathbf{p}_{(k)} \end{bmatrix} = \mathbf{b} \tag{9.10}$$

at the k-th step of the Active Set method (or semi-smooth Newton method) – see Section 2.1.3. Here,

$$\mathcal{L}_b = \begin{bmatrix} M_b + \epsilon^{-1}G_{\mathcal{A}^{(1)}}M_bG_{\mathcal{A}^{(1)}} & & \\ & \ddots & \\ & & M_b + \epsilon^{-1}G_{\mathcal{A}^{(N_t)}}M_bG_{\mathcal{A}^{(N_t)}} \end{bmatrix},$$

with $G_{\mathcal{A}^{(j)}}$, $j = 1, ..., N_t$, corresponding to the active set $\mathcal{A}_\pm$, in the notation of Section 2.1.3, at each time-step.

We therefore consider the iterative solution of the matrix system (9.10), as well as the system (9.9) for the case without control constraints. As in the previous chapter, the matrix systems involved can be extremely large, so the development of effective iterative methods is crucial to solve the systems efficiently.

9.2 Iterative Solution of Matrix Systems

Let us now examine how one might solve the matrix systems derived in the previous section using preconditioned iterative methods. We first consider developing

preconditioners for the matrix system (9.9), relating to the problem without control constraints. Once again, this system is of saddle point form, with

$$\Phi = \begin{bmatrix} \tau\widetilde{\mathcal{M}} & 0 \\ 0 & \alpha_c \tau D_1^{-1}\mathcal{M}_b \end{bmatrix}, \quad \Psi = \begin{bmatrix} \widetilde{\mathcal{K}} & -\tau D_1^{-1}\widetilde{\mathcal{N}} \end{bmatrix}, \quad \Theta = \begin{bmatrix} 0 \end{bmatrix}.$$

We highlight at this point that we use lumped mass matrices for our computations in this chapter.

Once more, the essential ingredients of a potent solver for the system (9.9) are good approximations of the $(1,1)$-block Φ and the Schur complement $S = \Psi\Phi^{-1}\Psi^T$. We will incorporate these approximations within block triangular preconditioners of the form

$$\widehat{\mathcal{P}}_2 = \begin{bmatrix} \widehat{\Phi} & 0 \\ \Psi & -\widehat{S} \end{bmatrix}.$$

For reasons discussed later in this section, the preconditioners we develop are unsuitable for use with MINRES or other methods for which block diagonal methods are appropriate. We therefore restrict our attention to block triangular preconditioners of the above form.

9.2.1 Approximation of the $(1,1)$-block

We first wish to develop an effective approximation of the $(1,1)$-block Φ. For this matrix system, our approximation is of the form $\widehat{\Phi} = \text{blkdiag}\left(\tau\widetilde{\mathcal{M}}_{\text{approx}}, \alpha_c\tau D_1^{-1}\mathcal{M}_b\right)$, where $\widetilde{\mathcal{M}}_{\text{approx}}$ denotes a suitable approximation of $\widetilde{\mathcal{M}}$. We note that $\alpha_c\tau D_1^{-1}\mathcal{M}_b$ is a diagonal matrix, and so is trivial to invert.

To form our approximation of $\widetilde{\mathcal{M}}$, we must construct approximations of $\widetilde{M}^{(j)}$ for $j = 1, ..., N_t$. Based on our discussions of saddle point theory in Section 2.2, and because we find this to be an effective choice in practice, we approximate $\widetilde{M}^{(j)}$ by

$$\begin{bmatrix} \alpha_u M - \alpha_v^{-1}(\gamma_1 M_{p_j} + \gamma_2 M_{q_j})M^{-1}(\gamma_1 M_{p_j} + \gamma_2 M_{q_j}) & 0 \\ \gamma_1 M_{p_j} + \gamma_2 M_{q_j} & \alpha_v M \end{bmatrix}.$$

This may be fed into our approximation $\widehat{\Phi}$. We believe that if we were to consider consistent mass matrices, we may use a similar approximation, but replace the mass matrix M by its diagonal where M^{-1} is taken.

9.2.2 Approximation of the Schur Complement

Our main task is now to devise an effective approximation of the Schur complement of the matrix system considered:

$$S = \frac{1}{\tau}\widetilde{\mathcal{K}}\widetilde{\mathcal{M}}^{-1}\widetilde{\mathcal{K}}^T + \frac{\tau}{\alpha_c D_1}\widetilde{\mathcal{N}}\mathcal{M}_b^{-1}\widetilde{\mathcal{N}}^T.$$

We motivate our approximation using a heuristic "matching" strategy that we have utilized in Chapters 4 and 8 for other boundary control problems.

Specifically, we attempt to generate an approximation of the form

$$\begin{aligned}\widehat{S} &= \frac{1}{\tau}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}\right)\widetilde{\mathcal{M}}^{-1}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}\right)^T \\ &= \frac{1}{\tau}\widetilde{\mathcal{K}}\widetilde{\mathcal{M}}^{-1}\widetilde{\mathcal{K}}^T + \frac{\tau}{\alpha_c}\widehat{\mathcal{M}}\widetilde{\mathcal{M}}^{-1}\widehat{\mathcal{M}} + \frac{1}{\sqrt{\alpha_c}}\left(\widetilde{\mathcal{K}}\widetilde{\mathcal{M}}^{-1}\widehat{\mathcal{M}} + \widehat{\mathcal{M}}\widetilde{\mathcal{M}}^{-1}\widetilde{\mathcal{K}}^T\right),\end{aligned}$$

where, as before, the second term of $\widehat{S}$ is sought to approximate the second term of S. Therefore, we require that

$$\frac{\tau}{\alpha_c}\widehat{\mathcal{M}}\widetilde{\mathcal{M}}^{-1}\widehat{\mathcal{M}} \approx \frac{\tau}{\alpha_c D_1}\widetilde{\mathcal{N}}\mathcal{M}_b^{-1}\widetilde{\mathcal{N}}^T. \tag{9.11}$$

We note that the left and right hand sides of (9.11) are both block diagonal matrices, and so the problem reduces to finding $\widehat{M}^{(1)}, \widehat{M}^{(2)}, ..., \widehat{M}^{(N_t)}$ such that $\widehat{\mathcal{M}} = \text{blkdiag}\left(\widehat{M}^{(1)}, 0, \widehat{M}^{(2)}, 0, ..., \widehat{M}^{(N_t)}, 0\right)$, and

$$\begin{aligned}&\begin{bmatrix} \widehat{M}^{(j)} & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \alpha_u M & \gamma_1 M_{p_j} + \gamma_2 M_{q_j} \\ \gamma_1 M_{p_j} + \gamma_2 M_{q_j} & \alpha_v M \end{bmatrix}^{-1}\begin{bmatrix} \widehat{M}^{(j)} & 0 \\ 0 & 0 \end{bmatrix} \\ &\qquad\qquad \approx \frac{1}{D_1}\begin{bmatrix} N_b M_b^{-1} N_b^T & 0 \\ 0 & 0 \end{bmatrix}, \quad j = 1, ..., N_t.\end{aligned} \tag{9.12}$$

We first wish to address the $\begin{bmatrix} \alpha_u M & \gamma_1 M_{p_j} + \gamma_2 M_{q_j} \\ \gamma_1 M_{p_j} + \gamma_2 M_{q_j} & \alpha_v M \end{bmatrix}^{-1}$ term. To do this we note that, given suitable invertibility conditions, the inverse of a general 2×2

block matrix is given by

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} (A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1} & -(A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1}A_{12}A_{22}^{-1} \\ -(A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}A_{21}A_{11}^{-1} & (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1} \end{bmatrix},$$

which may be easily checked. From this expression, we see that we can obtain an approximation of $\widehat{M}^{(j)}$ satisfying (9.12) by selecting the matrices $\widehat{M}^{(j)}$ to satisfy

$$\widehat{M}^{(j)}\Big(\alpha_u M - \alpha_v^{-1}\left(\gamma_1 M_{p_j} + \gamma_2 M_{q_j}\right) M^{-1}\left(\gamma_1 M_{p_j} + \gamma_2 M_{q_j}\right)\Big)^{-1}\widehat{M}^{(j)} \approx \frac{1}{D_1} M_\Gamma, \quad (9.13)$$

using the definition of M_Γ introduced in Chapter 4. Using the fact that all matrices in (9.13) are diagonal, we may re-write the expression as

$$\left(\widehat{M}^{(j)}\right)^2 \approx \frac{1}{D_1} A_0^{(j)} M_\Gamma, \quad A_0^{(j)} := \alpha_u M - \alpha_v^{-1}\left(\gamma_1 M_{p_j} + \gamma_2 M_{q_j}\right) M^{-1}\left(\gamma_1 M_{p_j} + \gamma_2 M_{q_j}\right).$$

Now, there are some issues here: firstly, the diagonal entries of $A_0^{(j)}$ may be negative, and secondly, the diagonal entries of M_Γ are zero for all interior nodes. Bearing in mind these considerations we postulate that a sensible choice of $\widehat{M}^{(j)}$ is a diagonal matrix with its diagonal entries $\widehat{m}_{ii}^{(j)}$ given by

$$\widehat{m}_{ii}^{(j)} = \begin{cases} \frac{1}{\sqrt{D_1}} |a_{0,ii}^{(j)}|^{1/2} m_{\Gamma,ii}^{1/2} & \text{if } i\text{-th node is on } \partial\Omega, \\ 0 & \text{otherwise.} \end{cases} \quad (9.14)$$

Here $a_{0,ii}^{(j)}$ represents the i-th diagonal entry of $A_0^{(j)}$, with $m_{\Gamma,ii}$ the i-th diagonal entry of M_Γ. We may thus use the choice of $\widehat{M}^{(j)}$ in (9.14) within our matrix $\widehat{\mathcal{M}}$. This gives an approximation $\widehat{S}$ of the Schur complement of the matrix system considered. We may therefore build our approximations $\widehat{\Phi}$ and $\widehat{S}$ into a block triangular preconditioner of the form $\widehat{\mathcal{P}}_2$.

As our preconditioner is not symmetric, nor necessarily positive definite, we cannot use it within the MINRES algorithm; we instead apply the BICG algorithm introduced in Section 2.3.7. As discussed in [89], there is no reason why we could not consider

instead approximations of S of the form

$$\widehat{S} = \frac{1}{\tau}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}_1\right)\widetilde{\mathcal{M}}^{-1}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}_2\right)^T,$$

where $\widehat{\mathcal{M}}_1 \neq \widehat{\mathcal{M}}_2$. However, we find that in practice our choice of $\widehat{S}$ derived in this section is a pragmatic one.

9.2.3 Preconditioning for Problems with Control Constraints

We now comment on preconditioning the matrix system (9.10), which pertains to the control constrained problem. The $(1,1)$-block may be approximated within our preconditioner in the same way as in the case without control constraints. We may view it in a similar way, as the only change in the matrix system when control constraints are introduced (the replacement of $\mathcal{M}_b$ by $\mathcal{L}_b$) does not introduce any extra difficulties, due to both matrices being diagonal and positive definite.

The extra difficulty arises from the fact that the Schur complement is changed – in the control constrained case this is given by

$$S = \frac{1}{\tau}\widetilde{\mathcal{K}}\widetilde{\mathcal{M}}^{-1}\widetilde{\mathcal{K}}^T + \frac{\tau}{\alpha_c D_1}\widetilde{\mathcal{N}}\mathcal{L}_b^{-1}\widetilde{\mathcal{N}}^T.$$

We again wish to devise an approximation of the form

$$\widehat{S} = \frac{1}{\tau}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}\right)\widetilde{\mathcal{M}}^{-1}\left(\widetilde{\mathcal{K}} + \frac{\tau}{\sqrt{\alpha_c}}\widehat{\mathcal{M}}\right)^T,$$

where this time

$$\frac{\tau}{\alpha_c}\widehat{\mathcal{M}}\widetilde{\mathcal{M}}^{-1}\widehat{\mathcal{M}} \approx \frac{\tau}{\alpha_c D_1}\widetilde{\mathcal{N}}\mathcal{L}_b^{-1}\widetilde{\mathcal{N}}^T.$$

Applying similar working as in the previous section gives us that we wish to find $\widehat{\mathcal{M}} = \text{blkdiag}\left(\widehat{M}^{(1)}, 0, \widehat{M}^{(2)}, 0, ..., \widehat{M}^{(N_t)}, 0\right)$ with

$$\widehat{M}^{(j)}A_0^{-(j)}\widehat{M}^{(j)} \approx \frac{1}{D_1}N_b\left(L_b^{(j)}\right)^{-1}N_b^T,$$

for $j = 1, ..., N_t$, where $L_b^{(j)} = M_b + \epsilon^{-1}G_{\mathcal{A}^{(j)}}M_bG_{\mathcal{A}^{(j)}}$ and $A_0^{-(j)} := \left(A_0^{(j)}\right)^{-1}$. We now make use of the fact that all the above matrices are diagonal, and denote the matrix

$N_b\big(L_b^{(j)}\big)^{-1}N_b^T$ by $L_\Gamma^{(j)}$ in our working. This leads to the desired approximation

$$\left(\widehat{M}^{(j)}\right)^2 \approx \frac{1}{D_1}A_0^{(j)}L_\Gamma^{(j)}.$$

Again noting that $A_0^{(j)}$ may contain negative entries on its diagonal, and that $L_\Gamma^{(j)}$ may contain zeros on its diagonal, we conclude that a good choice for $\widehat{M}^{(j)}$ is a diagonal matrix with diagonal entries given by

$$\widehat{m}_{ii}^{(j)} = \begin{cases} \frac{1}{\sqrt{D_1}}|a_{0,ii}^{(j)}|^{1/2}\big(l_{\Gamma,ii}^{(j)}\big)^{1/2} & \text{if } i\text{-th node is on } \partial\Omega, \\ 0 & \text{otherwise,} \end{cases}$$

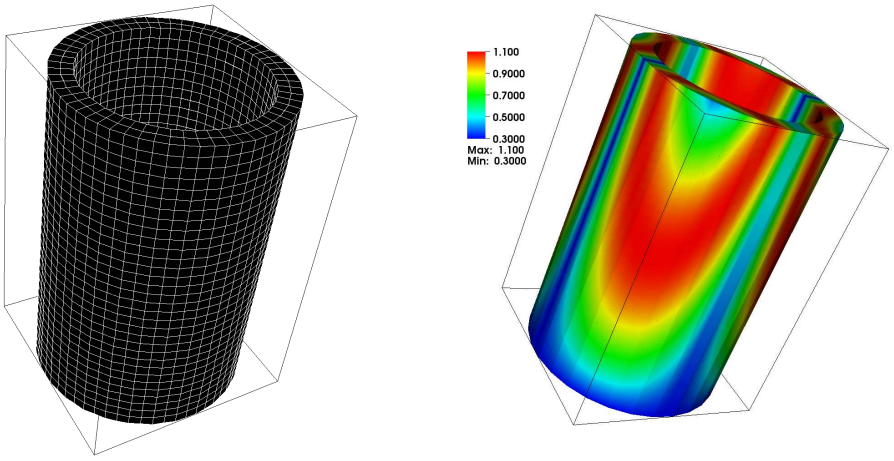
where here $l_{\Gamma,ii}^{(j)}$ denotes the i-th diagonal entry of $L_\Gamma^{(j)}$. We may build this approximation of $\widehat{M}^{(j)}$ into our Schur complement approximation $\widehat{S}$ for the control constrained case. We again incorporate our approximations $\widehat{\Phi}$ and $\widehat{S}$ into a preconditioned BiCG solver.

9.3 Numerical Results

In this section, we wish to carry out numerical experiments to demonstrate the performance of our proposed preconditioners. We again compute all results within the deal.II [4] framework using **Q1** finite element basis functions.[39] We once more use a smoothed aggregation algebraic multigrid routine to approximate the inverse of matrices representing elliptic operators, where appropriate, within the Trilinos ML package [44] – we take 6 V-cycles of this multigrid routine with 10 Chebyshev smoothing steps, as we attempt to minimize the sensitivity of the multigrid routine to parameter changes. We apply the BiCG algorithm, and typically solve to a tolerance of 10^{-4}, with the outer SQP iteration terminated whenever the relative change between two consecutive solutions is smaller than a given tolerance, typically 10^{-4} also.[40] The experiments are carried out with $T = 1$ in all cases, with $\tau = 0.05$ (corresponding to 20 time-steps).

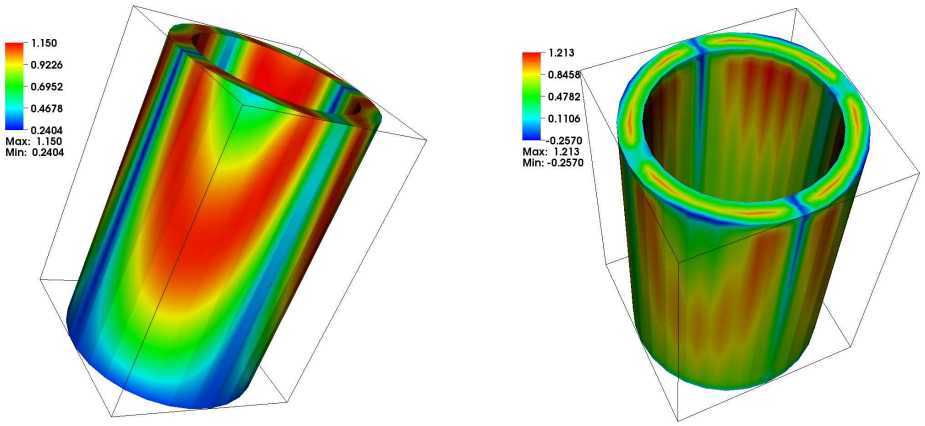
[39]The author acknowledges that all figures and numerical results in this section were created using code written by Martin Stoll as part of a collaboration with the author.

[40]The computations were carried out on a Centos Linux machine with Intel(R) Xeon(R) CPU X5650, 2.67GHz CPUs and 48GB of RAM.



(a) Domain (b) Desired state for first reactant

Figure 9.1: Cylindrical domain on which the first example was solved, and the desired state for the first reactant.



(a) Computed state for first reactant (b) Computed control

Figure 9.2: Computed state for the first reactant and computed control, at the 8-th time-step, with $\alpha_u = \alpha_v = 1$ and $\alpha_c = 10^{-5}$.

DoF	$\alpha_c = 10^{-3}$			$\alpha_c = 10^{-5}$		
	Time	SQP	Iterations	Time	SQP	Iterations
538,240	1,995	Step 1	17	1,726	Step 1	16
		Step 2	20		Step 2	16
		Step 3	20		Step 3	16
3,331,520	14,757	Step 1	28	14,904	Step 1	28
		Step 2	31		Step 2	27
		Step 3	29		Step 3	34

Table 9.1: BiCG iterations for each SQP step, and total CPU time, for the first example with varying values of mesh-size and α_c. DoF here represents the total dimension of the matrix system.

The first example on which we test our iterative method is the following problem:

$$\min_{u,v,c} \quad \frac{\alpha_u}{2} \|u - u_Q\|^2_{L_2(Q)} + \frac{\alpha_v}{2} \|v - v_Q\|^2_{L_2(Q)} + \frac{\alpha_c}{2} \|c\|^2_{L_2(\partial Q)} \tag{9.15}$$

such that

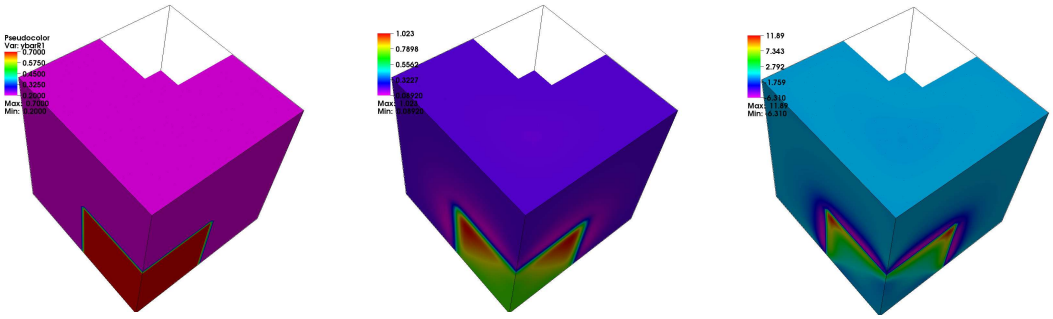
$$\begin{aligned} u_t - D_1 \nabla^2 u + k_1 u &= -\gamma_1 uv, \quad \text{in } Q, \\ v_t - D_2 \nabla^2 v + k_2 v &= -\gamma_2 uv, \quad \text{in } Q, \\ D_1 \frac{\partial u}{\partial n} &= c, \quad \text{on } \partial\Omega, \\ D_2 \frac{\partial v}{\partial n} + \widetilde{\epsilon} v &= 0, \quad \text{on } \partial\Omega, \\ u(\mathbf{x}, 0) &= 0, \quad \text{on } \Omega, \\ v(\mathbf{x}, 0) &= 0, \quad \text{on } \Omega, \end{aligned}$$

where the spatial domain Ω is a cylindrical shell domain with inner radius 0.8, outer radius 1 and height 3. The desired states are given by

$$u_Q = 2t|\sin(2x_1x_2x_3)| + 0.3, \quad v_Q = 0,$$

with u_Q shown in Figure 9.1 along with the spatial domain we work with. For this test we set the parameter values $D_1 = D_2 = k_1 = k_2 = 1$ and $\gamma_1 = \gamma_2 = 0.15$. The coordinate system is again denoted by $\mathbf{x} = [x_1, x_2, x_3]^T$.

In Figure 9.2, we show computed solutions for the first reactant u and the control



(a) Desired state

(b) Computed state for first reactant

(c) Computed control

Figure 9.3: Desired and computed states for the first reactant and computed control, at the 8-th time-step, with $\alpha_u = \alpha_v = 1$ and $\alpha_c = 10^{-5}$.

c. In Table 9.1, we provide the BiCG iteration numbers at each SQP step, as well as the total CPU time, to solve this problem for different problem sizes and values of α_c. All iteration numbers seem to be very reasonable given the complexity of the problem, and though there is some benign dependence on mesh-size (as for Neumann boundary control problems previously tested), our solver seems to perform well for a range of parameters. We note that, at present, we re-initialize the AMG preconditioner at each SQP step and for each time-step, which is an expensive process – we aim in future to reduce the CPU times significantly by finding alternative ways around this problem.

For our next experiment, we test a similar example, on a 'Hyper L' domain, which consists of the cube $[-1, 1]^3$, with the cube $(0, 1]^3$ removed. The desired states this time are given by

$$u_Q = \begin{cases} 0.7 & \text{if } \mathbf{x} \in \left[0, \frac{1}{2}\right]^3 \\ 0.2 & \text{otherwise} \end{cases}, \quad v_Q = 0.$$

Figure 9.3 gives solution plots for this problem, and in Table 9.2 we give numerical results for a range of mesh-sizes and values of α_c, with fixed parameters $D_1 = D_2 = k_1 = k_2 = 1$ and $\gamma_1 = \gamma_2 = 0.15$. In Table 9.3, we demonstrate the results of a more

DoF	$\alpha_c = 10^{-3}$			$\alpha_c = 10^{-5}$		
	Time	SQP	Iterations	Time	SQP	Iterations
60,920	369	Step 1	19	457	Step 1	23
		Step 2	20		Step 2	25
		Step 3	20		Step 3	25
382,840	2,624	Step 1	27	2,819	Step 1	29
		Step 2	30		Step 2	35
		Step 3	33		Step 3	33
2,670,200	19,128	Step 1	36	22,976	Step 1	46
		Step 2	44		Step 2	52
		Step 3	44		Step 3	53

Table 9.2: BiCG iterations for each SQP step, and total CPU time, for the second example with varying values of mesh-size and α_c. DoF here represents the total dimension of the matrix system.

Parameters	$\alpha_c = 10^{-3}$			$\alpha_c = 10^{-5}$		
	Time	SQP	Iterations	Time	SQP	Iterations
$D_1 = D_2 = 0.1$	1,744	Step 1	18	2,083	Step 1	19
		Step 2	20		Step 2	27
		Step 3	19		Step 3	20
$D_1 = D_2 = 100$	1,161	Step 1	16	1,783	Step 1	28
		Step 2	22		Step 2	33
$\gamma_1 = \gamma_2 = 0.05$	2,199	Step 1	22	2,426	Step 1	25
		Step 2	25		Step 2	29
		Step 3	25		Step 3	29
$\gamma_1 = \gamma_2 = 0.75$	3,796	Step 1	24	3,240	Step 1	20
		Step 2	36		Step 2	60
		Step 3	72		Step 3	32
$\text{tol} = 10^{-6}$	2,702	Step 1	27	3,226	Step 1	34
		Step 2	33		Step 2	38
		Step 3	33		Step 3	38
$\text{tol} = 10^{-8}$	3,289	Step 1	33	3,749	Step 1	39
		Step 2	39		Step 2	46
		Step 3	42		Step 3	46

Table 9.3: BiCG iterations for each SQP step, and total CPU time, for the parameter study example with fixed dimension 382,840.

detailed parameter study, by fixing the problem size and target states

$$u_Q = \begin{cases} 0.3 & \text{if } \mathbf{x} \in \left[0, \frac{1}{2}\right]^3 \\ 0.2 & \text{otherwise} \end{cases}, \quad v_Q = 0,$$

and varying a number of other parameters (α_c, D_1, D_2, γ_1 and γ_2, and the tolerance to which the BiCG method was run to). These results give further evidence of the robustness of our solver for a wide range of parameters involved. (We note again the large number of parameters within the problem set-up: h, τ, α_u, α_v, α_c, D_1, D_2, k_1, k_2, γ_1 and γ_2.) From experimental evidence, we find that the most difficult parameter regime occurs when γ_1 and γ_2 are large (and hence the $(1,1)$-block of the matrix system has large negative eigenvalues) and α_c is small (when the second term of our Schur complement approximation dominates, and the indefiniteness of the Schur complement is not captured by our approximation of it). We note also that when γ_1 and γ_2 are large, the convergence rate at different SQP steps may vary more than when γ_1 and γ_2 are small, as in this case the values of p and q at the previous SQP step determine the extent of the indefiniteness of the $(1,1)$-block.

DoF	$\alpha_c = 10^{-3}$			$\alpha_c = 10^{-5}$		
	Time	SQP	NM/∅ BiCG	Time	SQP	NM/∅ BiCG
60,920	1,066	Step 1	3/18	859	Step 1	3/22
		Step 2	3/21		Step 2	2/25.5
		Step 3	3/21			
382,840	5,498	Step 1	2/26	13,358	Step 1	5/28.6
		Step 2	2/36		Step 2	5/32.6
		Step 3	2/35		Step 3	5/32.8

Table 9.4: Number of semi-smooth Newton and average number of BiCG iterations for each SQP step, and total CPU time, for the control constrained example with varying values of mesh-size and α_c. DoF here represents the total dimension of the matrix system, and ∅ denotes an average number.

In order to highlight the effectiveness of our strategy when control constraints are present, we also carry out numerical tests on the problem (9.15) within the 'Hyper

DoF	$\epsilon = 10^{-2}$		$\epsilon = 10^{-4}$		$\epsilon = 10^{-6}$	
	SQP	NM/∅ BiCG	SQP	NM/∅ BiCG	SQP	NM/∅ BiCG
60,920	Step 1	3/32.3	Step 1	3/24.6	Step 1	3/20.6
	Step 2	3/36.3	Step 2	3/26.6	Step 2	3/22.3

Table 9.5: Number of semi-smooth Newton and average number of BiCG iterations at each SQP step for the control constrained example, with fixed dimension 60,920, $\alpha_c = 10^{-3}$, and varying values of penalty parameter ϵ. DoF here represents the total dimension of the matrix system, and ∅ denotes an average number.

L' domain, with

$$u_Q = t|\sin(2x_1x_2x_3)\cos(2x_1x_2x_3)|, \quad v_Q = 0,$$
$$D_1 = D_2 = k_1 = k_2 = 1, \quad \gamma_1 = \gamma_2 = 0.15,$$

but also with the additional control constraint

$$c \leq \frac{3}{2}.$$

In Table 9.4, we present results (including total CPU times, as well as the number of semi-smooth Newton steps to deal with the control constraint and BiCG iterations at each outer SQP step) for a range of mesh-sizes and values of α_c. In Table 9.5, we present results using our method for different values of the Moreau-Yosida regularization parameter ϵ. We observe low iteration numbers (both outer iterations and BiCG iterations) for a range of such problems. This indicates the potential of our approach for solving a further range of reaction-diffusion control problems with additional control constraints imposed.

9.4 Summary

In this chapter, we have been able to build on the knowledge we have acquired about the iterative solution of simpler PDE-constrained optimization problems, in order to tackle a problem motivated by a real-life application, namely that of the optimal control of chemical processes. To solve this problem, we have utilized and extended the building blocks introduced in nearly every previous chapter of this thesis – we have exploited fundamental saddle point theory as for other optimal control problems,

used a Schur complement approximation of the type first developed for distributed Poisson control problems and then extended to boundary control problems, dealt with nonlinearity in the underlying PDEs as for Navier-Stokes control, and incorporated a time-stepping scheme as for the optimal control of the heat equation.

Naturally, due to the difficulty of the problem being looked at in this chapter, and in particular the complexity in structure of the matrix systems being solved, our approach for this problem is based much more on heuristic argument than for the much simpler problems discussed earlier in this thesis. However, we were still able to obtain very encouraging numerical results using our proposed solvers for the matrix systems arising from this problem, with low iteration numbers achieved for a wide range of problems and parameters.

We therefore conclude that the methods we have constructed during the course of this thesis have the potential to tackle complex, real-world problems. This opens up the possibility of developing methods which may be used to create feasible and robust solvers for problems of great interest to the scientific community.

CHAPTER 10

Concluding Remarks and Outlook

In this thesis, we have aimed to develop fast and effective preconditioned iterative methods, using MINRES, Bramble-Pasciak CG, GMRES and BiCG, for problems in the field of PDE-constrained optimization. We have considered a range of such problems, and proposed preconditioned iterative methods which converged robustly with respect to all parameters involved in the problem set-up. The problems we tackled included the distributed, boundary and subdomain control of Poisson's equation, convection-diffusion control, Stokes and Navier-Stokes control, the optimal control of time-dependent problems involving the heat equation, and the optimal control of a reaction-diffusion control problem motivated by chemical processes.

We have shown that it is possible, using the theory of saddle point systems, to develop, for these problems, parameter-independent solvers which achieve rapid convergence and are extremely practical. This in turn requires accurate approximations of the $(1,1)$-block and Schur complement of the matrices involved – the Schur complement approximations were invariably the hardest component of the preconditioner to derive. In some sense the most crucial result presented in this thesis was Theorem 8 of Section 3.2 concerning our Schur complement approximation for Poisson control, as this result opened the door to many other effective approximations. From this, the results for convection-diffusion control, boundary control and subdomain control were motivated. Furthermore, the preconditioners we detailed for Stokes and Navier-Stokes control were derived using the Schur complement results for Poisson

control and convection-diffusion control, along with suitable commutator arguments. We have also extended the ideas behind these Schur complement approximations to a number of time-dependent optimal control problems. This family of preconditioners presents a framework and guidance for solving harder problems of this type.

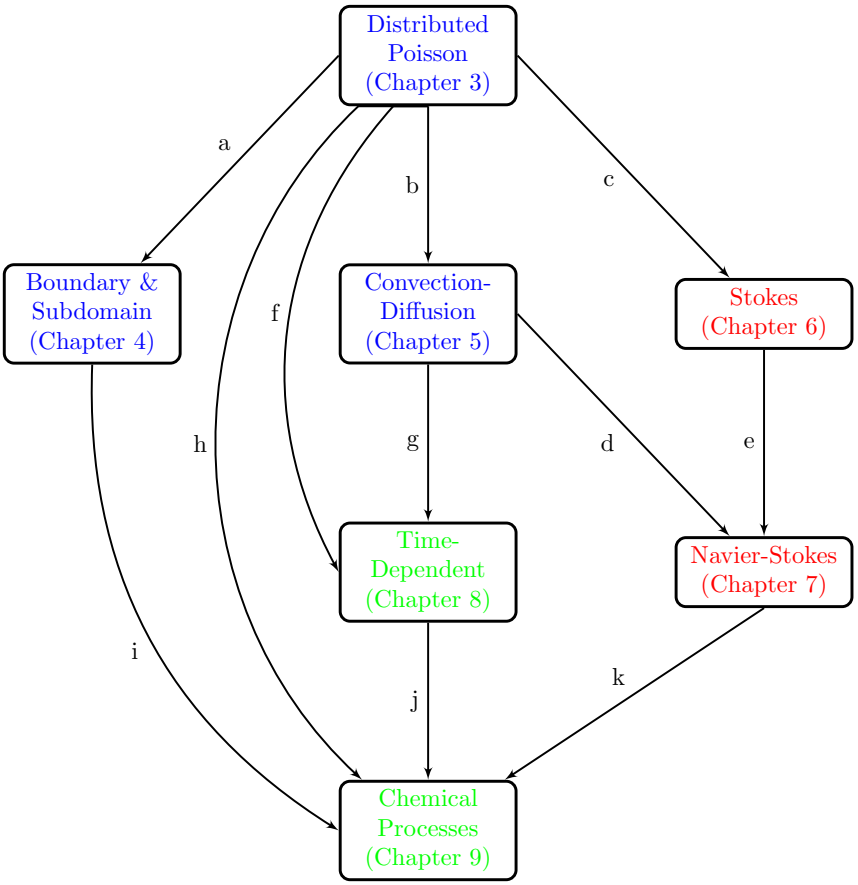


Figure 10.1: Flow chart outlining the problems of which we have sought the numerical solution in this thesis, and their links to each other from a preconditioning point-of-view.

Figure 10.1 summarizes the links we have observed between the methods for the various problems we have considered. We have seen, for instance, that the matrix systems for the distributed Poisson control problem and the analogous boundary control or subdomain control problems are very similar in structure, except that the blocks corresponding to the state and control variables are of different sizes for boundary and subdomain control. We thus use a very similar preconditioning strategy for each of these problems (hence link a), but are able to prove the effectiveness of our Schur complement approximation more easily for the distributed control problem, and also

observe lower iteration numbers. Also, for the convection-diffusion control problem, the same process was used to develop our proposed preconditioners and the same outcome achieved, but the proof of the Schur complement approximation required a slightly weaker assumption about the structure of the problem (link b). As we explored the Stokes control problem, we saw that we were able to exploit our methods for Poisson control to solve this problem as well, with the aid of additional commutator arguments (c). We dealt with the Navier-Stokes control problem in the same way as for Stokes control (e), but this time utilizing our approach for convection-diffusion control problems (d). We were able to solve time-dependent PDE-constrained optimization problems (in particular heat equation control problems) by motivating preconditioners of the same form as for the (time-independent) Poisson control problem (f), using a Schur complement approximation motivated similarly as for convection-diffusion control (g). The reaction-diffusion control problem we considered was of much more complex structure, but we were able to use many of the techniques we had explored for simpler problems. In particular, we utilized the saddle point structure and type of Schur complement approximation we applied to the distributed Poisson control problem (h), the theory of boundary control problems considered for Poisson control (i), the matrix structure and time-stepping methods used for heat equation control (j), and the embedding of our solvers into an outer iteration in order to deal with nonlinear terms as for Navier-Stokes control (k).

Figure 10.1 also highlights the three main stages in the development of our preconditioning methodologies. The problems coloured blue were solved using saddle point theory with carefully chosen Schur complement approximations. When solving the problems coloured red, we used the methods for the problems coloured blue, but also required tricks relating to the reordering of the matrix systems, as well as specialized commutator arguments. Finally, solving the problems coloured green needed knowledge previously acquired by solving the problems coloured blue and red, but also required time-stepping techniques and preconditioning strategies to deal with the increased size of the matrix system that resulted from the time-dependent nature of the problems.

In deriving our proposed methods for the problems we considered, the heuristic development of our preconditioners and the analytic proof of their effectiveness went hand-in-hand. As the problems became more complex, demonstrations of the effectiveness of the Schur complement approximation in particular became more heuristic, by necessity. It is possible that the frontier of what can be proved readily about the

effectiveness of preconditioners for such optimal control problems has been reached, and that if we were to tackle even harder problems of this type, more heuristic guidance will be required to derive these approximations. But I believe it will be useful in these cases to understand why the theory developed in this thesis *does not* hold; this will enable the understanding of which parameter regimes preconditioners will or will not be effective for, and why.

In the future, there are many variations on the problems discussed in this thesis which could be explored, using aspects of the methodology introduced here. For instance, a greater focus may be placed on boundary control problems as discussed in Section 4.1 and Chapter 9, as such problems are generally more physically realistic. We may also consider PDE-constrained optimization problems in which the functional contains norms which are not the L_2-norm (as done in [102] for example), problems in which the control is applied in only part of the domain, or problems where the "distance" between the state and desired state is only measured in some subdomain.

Apart from problems such as these, the main area which we believe should now be explored is that of developing feasible and fast iterative solvers for optimal control problems arising in real-world applications. The content of Chapter 9 was geared towards such an industrial application, with a view to demonstrating that our methodology may be applicable to such problems. There are a huge number of other PDE-constrained optimization problems that arise from real-world problems, such as those described in [59] and detailed in Section 2.1. Looking ahead, it seems that researchers in this field may now be in a position to derive iterative methods that can be used to solve such important problems, using in part the methodology and theory described in this thesis.

Bibliography

[1] S. Angenent, S. Haker, and A. Tannenbaum, *Minimizing Flows for the Monge-Kantorovich Problem*, SIAM Journal on Matrix Analysis and Applications, 35, pp.61–97, 2003.

[2] S. R. Arridge, *Optical Tomography in Medical Imaging*, Inverse Problems, 15, pp.R41–R93, 1999.

[3] O. Axelsson, and M. Neytcheva, *Eigenvalue Estimates for Preconditioned Saddle Point Matrices*, Numerical Linear Algebra with Applications, 13, pp.339–360, 2006.

[4] W. Bangerth, R. Hartmann, and G. Kanschat, *deal.II – A General-Purpose Object-Oriented Finite Element Library*, ACM Transactions on Mathematical Software, 33, Article 24, 2007.

[5] W. Barthel, C. John, and F. Tröltzsch, *Optimal Boundary Control of a System of Reaction Diffusion Equations*, Zeitschrift für Angewandte Mathematik und Mechanik (ZAMM), 90, pp.966–982, 2010.

[6] R. Becker, and M. Braack, *A Finite Element Pressure Gradient Stabilization for the Stokes Equations Based on Local Projections*, Calcolo, 38(4), pp.173–199, 2001.

[7] R. Becker, and B. Vexler, *Optimal Control of the Convection-Diffusion Equation Using Stabilized Finite Element Methods*, Numerische Mathematik, 106(3), pp.349–367, 2007.

[8] J. D. Benamou, and Y. Brenier, *A Computational Fluid Mechanics Solution to the Monge-Kantorovich Mass Transfer Problem*, Numerische Mathematik, 84, pp.375–393, 2000.

[9] M. Benzi, G. H. Golub, and J. Liesen, *Numerical Solution of Saddle Point Problems*, Acta Numerica, 14, pp.1–137, 2005.

[10] M. Benzi, E. Haber, and L. Taralli, *A Preconditioning Technique for a Class of PDE-Constrained Optimization Problems*, Advances in Computational Mathematics, 35, pp.149–173, 2011.

[11] M. Bergounioux, K. Ito, and K. Kunisch, *Primal-Dual Strategy for Constrained Optimal Control Problems*, SIAM Journal on Control and Optimization, 37(4), pp.1176–1194, 1999.

[12] D. P. Bertsekas, *Projected Newton Methods for Optimization Problems with Simple Constraints*, SIAM Journal on Control and Optimization, 20(2), pp.221–246, 1982.

[13] G. Biros, and O. Ghattas, *Parallel Lagrange-Newton-Krylov-Schur Methods for PDE-Constrained Optimization. Part I: The Krylov-Schur Solver*, SIAM Journal on Scientific Computing, 27(2), pp.687–713, 2005.

[14] G. Biros, and O. Ghattas, *Parallel Lagrange-Newton-Krylov-Schur Methods for PDE-Constrained Optimization. Part II: The Lagrange-Newton Solver and its Applications to Optimal Control of Steady Viscous Flows*, SIAM Journal on Scientific Computing, 27(2), pp.714–739, 2005.

[15] A. Borzi, *High-Order Discretization and Multigrid Solution of Elliptic Nonlinear Constrained Optimal Control Problems*, Journal of Computational and Applied Mathematics, 200, pp.67–85, 2007.

[16] A. Borzi, and K. Kunisch, *A Multigrid Scheme for Elliptic Constrained Optimal Control Problems*, Computational Optimization and Applications, 31, pp.309–333, 2005.

[17] A. Borzi, and V. Schulz, *Multigrid Methods for PDE Optimization*, SIAM Review, 51(2), pp.361–395, 2009.

[18] I. Bouchouev, and V. Isakov, *Uniqueness, Stability and Numerical Methods for the Inverse Problem that Arises in Financial Markets*, Inverse Problems, 15, pp.R95–R116, 1999.

[19] J. Boyle, M. D. Mihajlovic, and J. A. Scott, *HSL_MI20: An Efficient AMG Preconditioner for Finite Element Problems in 3D*, International Journal for Numerical Methods in Engineering, 82, pp.64–98, 2010.

[20] M. Braack, and E. Burman, *Local Projection Stabilization for the Oseen Problem and its Interpretation as a Variational Multiscale Method*, SIAM Journal on Numerical Analysis, 43(6), pp.2544–2566, 2006.

[21] J. H. Bramble, and J. E. Pasciak, *A Preconditioning Technique for Indefinite Systems Resulting from Mixed Approximations of Elliptic Problems*, Mathematics of Computation, 50(181), pp.1–17, 1988.

[22] A. Brandt, S. C. McCormick, and J. Ruge, *Algebraic Multigrid (AMG) for Sparse Matrix Equations*, Cambridge University Press, 1985.

[23] S. C. Brenner, and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, 2002.

[24] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, SIAM, 2000.

[25] J. Cahouet, and J.-P. Chabard, *Some Fast 3D Finite Element Solvers for the Generalized Stokes Problem*, International Journal for Numerical Methods in Fluids, 8, pp.869–895, 1988.

[26] R. Casanova, A. Silva, and A. R. Borges, *A Quantitative Algorithm for Parameter Estimation in Magnetic Induction Tomography*, Measurement Science and Technology, 15, pp.1412–1419, 2004.

[27] E. Casas, *Control of an Elliptic Problem with Pointwise State Constraints*, SIAM Journal on Control and Optimization, 24, pp.1309–1318, 1986.

[28] M. Cheney, D. Isaacson, and J. C. Newell, *Electrical Impedance Tomography*, SIAM Review, 41, pp.85–101, 1999.

[29] J. C. De Los Reyes, and R. Griesse, *State-Constrained Optimal Control of the Three-Dimensional Stationary Navier-Stokes Equations*, Journal of Mathematical Analysis and Applications, 343, pp.257–272, 2008.

[30] J. C. De Los Reyes, and F. Tröltzsch, *Optimal Control of the Stationary Navier-Stokes Equations with Mixed Control-State Constraints*, SIAM Journal on Control and Optimization, 46, pp.604–629, 2007.

[31] X. Du, M. Sarkis, C. E. Schaerer, and D. B. Szyld, *Inexact and Truncated Parareal-in-Time Krylov Subspace Methods for Parabolic Optimal Control Problems*, Electronic Transactions on Numerical Analysis, 40, pp.36–57, 2013.

[32] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Numerical Mathematics and Scientific Computation Series, Oxford Science Publications, 1989.

[33] B. Dupire, *Pricing with a Smile*, Risk, 7, pp.32–39, 1994.

[34] H. Egger, and H. W. Engel, *Tikhonov Regularization Applied to the Inverse Problem of Option Pricing: Convergence Analysis and Rates*, Inverse Problems, 21, pp.1027–1045, 2005.

[35] H. C. Elman, A. Ramage, and D. Silvester, *Algorithm 866: IFISS, a Matlab Toolbox for Modelling Incompressible Flow*, ACM Transactions on Mathematical Software, 33, pp.2–14, 2007.

[36] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*, Numerical Mathematics and Scientific Computation Series, Oxford University Press: New York, 2005.

[37] M. Fisher, J. Nocedal, Y. Trémolet, and S. J. Wright, *Data Assimilation in Weather Forecasting: A Case Study in PDE-Constrained Optimization*, Optimization and Engineering, 10(3), pp.409–426, 2009.

[38] D. A. Flanders, and G. Shortley, *Numerical Determination of Fundamental Modes*, Journal of Applied Physics, 21, pp.1326–1352, 1950.

[39] R. Fletcher, *Conjugate Gradient Methods for Indefinite Systems*, in *Numerical Analysis, Dundee 1975* (Proceedings), Lecture Notes in Mathematics, Volume 506, G. A. Watson (Editor), Springer Verlag, 1976.

[40] R. W. Freund, *Transpose-Free Quasi-Minimal Residual Methods for Non-Hermitian Linear Systems*, in *Recent Advances in Iterative Methods*, IMA Volume in Mathematics and its Applications, Volume 60, Springer, New York, pp.69–94, 1994.

[41] R. W. Freund, and N. M. Nachtigal, *QMR: A Quasi-Minimal Residual Method for Non-Hermitian Linear Systems*, Numerische Mathematik, 60, pp.315–341, 1991.

[42] M. R. Garvie, P. K. Maini, and C. Trenchea, *An Efficient and Robust Numerical Algorithm for Estimating Parameters in Turing Systems*, Journal of Computational Physics, 229, pp.7058–7071, 2010.

[43] M. R. Garvie, and C. Trenchea, *Identification of Space-Time Distributed Parameters in the Gierer-Meinhardt Reaction-Diffusion System*, submitted, 2011.

[44] M. W. Gee, C. M. Siefert, J. J. Hu, R. S. Tuminaro, and M. G. Sala, *ML 5.0 Smoothed Aggregation User's Guide*, Technical Report SAND2006-2649, Sandia National Laboratories, 2006.

[45] G. H. Golub, and R. S. Varga, *Chebyshev Semi-Iterative Methods, Successive Over-Relaxation Iterative Methods, and Second Order Richardson Iterative Methods, Part I*, Numerische Mathematik, 3, pp.147–156, 1961.

[46] G. H. Golub, and R. S. Varga, *Chebyshev Semi-Iterative Methods, Successive Over-Relaxation Iterative Methods, and Second Order Richardson Iterative Methods, Part II*, Numerische Mathematik, 3, pp.157–168, 1961.

[47] A. Grama, A. Gupta, G. Karypis, and V. Kumar, *Introduction to Parallel Computing*, Pearson Education Limited, Second Edition, 2003.

[48] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, SIAM, 1997.

[49] R. Griesse, *Parametric Sensitivity Analysis in Optimal Control of a Reaction-Diffusion System. I: Solution Differentiability*, Numerical Functional Analysis and Optimization, 25, pp.93–117, 2004.

[50] R. Griesse, *Parametric Sensitivity Analysis in Optimal Control of a Reaction-Diffusion System. II: Practical Methods and Examples*, Optimization Methods and Software, 19, pp.217–242, 2004.

[51] R. Griesse, and S. Volkwein, *Parametric Sensitivity Analysis for Optimal Boundary Control of a 3D Reaction-Diffusion System*, in G. Di Pillo, and M. Roma (editors), *Large-Scale Nonlinear Optimization, Proceedings of the Workshop Large Scale Nonlinear Optimization*, Erice, 2004; Nonconvex Optimization and its Applications, Volume 83, Springer, Berlin, 2006.

[52] R. Griesse, and S. Volkwein, *A Primal-Dual Active Set Strategy for Optimal Boundary Control of a Nonlinear Reaction-Diffusion System*, SIAM Journal on Control and Optimization, 44, pp.467–494, 2005.

[53] J.-L. Guermond, *Stabilization of Galerkin Approximations of Transport Equations by Subgrid Modeling*, Modélisation Mathématique et Analyse Numérique, 33(6), pp.1293–1316, 1999.

[54] J.-L. Guermond, and R. Pasquetti, *A Correction Technique for the Dispersive Effects of Mass Lumping for Transport Problems*, Computer Methods in Applied Mechanics and Engineering, 253, pp.186–198, 2013.

[55] M. Gunzberger, *Perspectives in Flow Control and Optimization*, SIAM, 1987.

[56] E. Haber, *A Parallel Method for Large Scale Time Domain Electromagnetic Inverse Problems*, Applied Numerical Mathematics: Transactions of IMACS, 58, pp.422–434, 2008.

[57] E. Haber, *Quasi-Newton Methods for Large-Scale Electromagnetic Inverse Problems*, Inverse Problems, 21, pp.305–317, 2004.

[58] E. Haber, and U. M. Ascher, *Preconditioned All-at-Once Methods for Large, Sparse Parameter Estimation Problems*, Inverse Problems, 17, pp.1847–1864, 2001.

[59] E. Haber, and L. Hanson, *Model Problems in PDE-Constrained Optimization*, Emory Technical Report TR-2007-009, 2007.

[60] W. Hackbusch, *Multigrid Methods and Applications*, Springer Series in Computational Mathematics, Volume 4, Springer-Verlag, Berlin, 1985.

[61] M. Heinkenschloss, and D. Leykekhman, *Local Error Estimates for SUPG Solutions of Advection-Dominated Elliptic Linear-Quadratic Optimal Control Problems*, SIAM Journal on Numerical Analysis, 47(6), pp.4607–4638, 2010.

[62] R. Herzog, and E. W. Sachs, *Preconditioned Conjugate Gradient Method for Optimal Control Problems with Control and State Constraints*, SIAM Journal on Matrix Analysis and Applications, 31, pp.2291–2317, 2010.

[63] M. R. Hestenes, and E. Stiefel, *Methods of Conjugate Gradients for Solving Linear Systems*, Journal of Research of the National Bureau of Standards, 49, pp.409–436, 1952.

[64] M. Hintermüller, K. Ito, and K. Kunisch, *The Primal-Dual Active Set Strategy as a Semismooth Newton Method*, SIAM Journal on Optimization, 13(3), pp.865–888, 2002.

[65] E. Hinton, T. Rock, and O. C. Zienkiewicz, *A Note on Mass Lumping and Related Processes in the Finite Element Method*, Earthquake Engineering & Structural Dynamics, 4(3), pp.245–249, 1976.

[66] M. Hinze, M. Köster, and S. Turek, *A Hierarchical Space-Time Solver for Distributed Control of the Stokes Equation*, Priority Programme 1253, Preprint Number SPP1253-16-01, 2008.

[67] M. Hinze, M. Köster, and S. Turek, *A Space-Time Multigrid Solver for Distributed Control of the Time-Dependent Navier-Stokes System*, Priority Programme 1253, Preprint Number SPP1253-16-02, 2008.

[68] M. Hinze, and K. Kunisch, *Second Order Methods for Optimal Control of Time-Dependent Fluid Flow*, SIAM Journal on Control and Optimization, 40, pp.925–946, 2001.

[69] M. Hinze, and R. Pinnau, *A Second Order Approach to Optimal Semiconductor Design*, Journal of Optimization Theory and Applications, 133, pp.179–200, 2007.

[70] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE Constraints*, Mathematical Modelling: Theory and Applications, Springer-Verlag, New York, 2009.

[71] T. J. R. Hughes, and A. Brooks, *Finite Element Methods for Convection Dominated Flows*, pp.120–131 of "A Multidimensional Upwind Scheme with no Crosswind Diffusion", Volume 34 of AMD Series, ASME, 1979.

[72] I. C. F. Ipsen, *A Note on Preconditioning Non-Symmetric Matrices*, SIAM Journal on Scientific Computing, 23(3), pp.1050–1051, 2001.

[73] K. Ito, and K. Kunisch, *Semi-Smooth Newton Methods for State-Constrained Optimal Control Problems*, Systems & Control Letters, 50(3), pp.221–228, 2003.

[74] C. John, and D. Wachsmuth, *Optimal Dirichlet Boundary Control of Navier-Stokes Equations with State Constraint*, Numerical Functional Analysis and Optimization, 30(11&12), pp.1309–1338, 2009.

[75] C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, 1990.

[76] A. Joshi, W. Bangerth, and E. M. Sevick-Muraca, *Non-Contact Fluorescence Optical Tomography with Scanning Patterned Illumination*, Optics Express, 14, pp.6516–6534, 2006.

[77] D. Kay, D. Loghin, and A. Wathen, *A Preconditioner for the Steady-State Navier-Stokes Equations*, SIAM Journal on Scientific Computing, 24(1), pp.237–256, 2002.

[78] M. V. Klibanov, and T. R. Lucas, *Numerical Solution of a Parabolic Inverse Problem in Optical Tomography Using Experimental Data*, SIAM Journal on Applied Mathematics, 59, pp.1763–1789, 1999.

[79] Y. A. Kuznetsov, *Efficient Iterative Solvers for Elliptic Finite Element Problems on Nonmatching Grids*, Russian Journal of Numerical Analysis and Mathematical Modelling, 10(3), pp.187–211, 1995.

[80] C. Lanczos, *An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators*, Journal of Research of the National Bureau of Standards, 45, pp.255–282, 1950.

[81] C. Lanczos, *Solution of Systems of Linear Equations by Minimized Iterations*, Journal of Research of the National Bureau of Standards, 49, pp.33–53, 1952.

[82] R. J. Leveque, *High-Resolution Conservative Algorithms for Advection in Incompressible Flow*, SIAM Journal on Numerical Analysis, 33, pp.627–665, 1996.

[83] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A Note on Preconditioning for Indefinite Linear Systems*, SIAM Journal on Scientific Computing, 21(6), pp.1969–1972, 2000.

[84] C. Orozco, and O. Ghattas, *Massively Parallel Aerodynamic Shape Optimization*, Computing Systems in Engineering, 1, pp.311–320, 1992.

[85] C. C. Paige, and M. A. Saunders, *Solutions of Sparse Indefinite Systems of Linear Equations*, SIAM Journal on Numerical Analysis, 12(4), pp.617–629, 1975.

[86] J. W. Pearson, *On the Role of Commutator Arguments in the Development of Parameter-Robust Preconditioners for Stokes Control Problems*, submitted to Electronic Transactions on Numerical Analysis, 2013.

[87] J. W. Pearson, *Preconditioned Iterative Methods for Navier-Stokes Control Problems*, submitted to SIAM Journal on Scientific Computing, 2013.

[88] J. W. Pearson, *A Radial Basis Function Method for Solving PDE-Constrained Optimization Problems*, to appear in Numerical Algorithms, available online, DOI: 10.1007/s11075-012-9675-6, 2012.

[89] J. W. Pearson, and M. Stoll, *Fast Iterative Solution of Reaction-Diffusion Control Problems Arising from Chemical Processes*, SIAM Journal on Scientific Computing, 35, pp.B987–B1009, 2013. Copyright © 2013, Society for Industrial and Applied Mathematics.

[90] J. W. Pearson, M. Stoll, and A. J. Wathen, *Preconditioners for State Constrained Optimal Control Problems with Moreau-Yosida Penalty Function*, to appear in Numerical Linear Algebra with Applications, available online, DOI: 10.1002/nla.1863, 2012. Publisher: John Wiley & Sons. Copyright © 2012 John Wiley & Sons Ltd. Reproduced with permission.

[91] J. W. Pearson, M. Stoll and A. J. Wathen, *Regularization-Robust Preconditioners for Time-Dependent PDE-Constrained Optimization Problems*, SIAM Journal on Matrix Analysis and Applications, 33(4), pp.1126–1152, 2012. Copyright © 2012, Society for Industrial and Applied Mathematics.

[92] J. W. Pearson, M. Stoll and A. J. Wathen, *Robust Iterative Solution of a Class of Time-Dependent Optimal Control Problems*, Proceedings in Applied Mathematics and Mechanics (PAMM), 12(1), pp.3–6, 2012. Publisher: WILEY-VCH Verlag GmbH & Co. Copyright © 2012 WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. Reproduced with permission.

[93] J. W. Pearson, and A. J. Wathen, *Fast Iterative Solvers for Convection-Diffusion Control Problems*, Electronic Transactions on Numerical Analysis, 40, pp.294–310, 2013.

[94] J. W. Pearson, and A. J. Wathen, *A New Approximation of the Schur Complement in Preconditioners for PDE-Constrained Optimization*, Numerical Linear Algebra with Applications, 19(5), pp.816–829, 2012. Publisher: John Wiley & Sons. Copyright © 2011 John Wiley & Sons Ltd. Reproduced with permission.

[95] V. Pillwein, and S. Takacs, *Computing Smoothing Rates of Collective Point Smoothers for Optimal Control Problems Using Symbolic Computation*, in U. Langer, and P. Paule (editors), *Numerical and Symbolic Scientific Computing: Progress and Prospects*, Springer Wien, 2011.

[96] M. Pošta, and T. Roubíček, *Optimal Control of Navier-Stokes Equations by Oseen Approximation*, Computers and Mathematics with Applications, 53, pp.569–581, 2007.

[97] A. Quarteroni, and G. Rozza, *Optimal Control and Shape Optimization of Aorto-Coronaric Bypass Anastomoses*, Mathematical Models and Methods in Applied Sciences, 13(12), pp.1801–1823, 2003.

[98] A. Ramage, *A Multigrid Preconditioner for Stabilised Discretisations of Advection-Diffusion Problems*, Journal of Computational and Applied Mathematics, 110, pp.187–203, 1999.

[99] T. Rees, *Preconditioning Iterative Methods for PDE Constrained Optimization*, DPhil Thesis, University of Oxford, 2010.

[100] T. Rees, H. S. Dollar, and A. J. Wathen, *Optimal Solvers for PDE-Constrained Optimization*, SIAM Journal on Scientific Computing, 32(1), pp.271–298, 2010.

[101] T. Rees, and M. Stoll, *Block Triangular Preconditioners for PDE-Constrained Optimization*, Numerical Linear Algebra with Applications, 17(6), pp.977–996, 2010.

[102] T. Rees, and M. Stoll, *A Fast Solver for an H_1 Regularized Optimal Control Problem*, Max Planck Institute Magdeburg Preprint MPIMD/12-06, 2012.

[103] T. Rees, M. Stoll, and A. J. Wathen, *All-at-Once Preconditioning in PDE-Constrained Optimization*, Kybernetika, 45, pp.341–360, 2010.

[104] T. Rees, and A. J. Wathen, *Preconditioning Iterative Methods for the Optimal Control of the Stokes Equations*, SIAM Journal on Scientific Computing, 33, pp.2903–2926, 2011.

[105] M. Rozložník, and V. Simoncini, *Krylov Subspace Methods for Saddle Point Problems with Indefinite Preconditioning*, SIAM Journal on Matrix Analysis and Applications, 24, pp.368–391, 2002.

[106] J. W. Ruge, and K. Stüben, *Algebraic Multigrid*, in *Multigrid Methods*, Frontiers of Applied Mathematics, Volume 3, SIAM, Philadelphia, pp.73–130, 1987.

[107] Y. Saad, *A Flexible Inner-Outer Preconditioned GMRES Algorithm*, SIAM Journal on Scientific and Statistical Computing, 14, pp.461–469, 1993.

[108] Y. Saad, and M. H. Schultz, *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM Journal on Scientific Computing, 7(3), pp.856–869, 1986.

[109] J. Schöberl, and W. Zulehner, *Symmetric Indefinite Preconditioners for Saddle Point Problems with Applications to PDE-Constrained Optimization Problems*, SIAM Journal on Matrix Analysis and Applications, 29(3), pp.752–773, 2007.

[110] D. Silvester, H. Elman, and A. Ramage, *Incompressible Flow and Iterative Solver Software (IFISS)*, version 3.1, `http://www.manchester.ac.uk/ifiss/`, 2011.

[111] D. Silvester, and A. Wathen, *Fast Iterative Solution of Stabilised Stokes Systems. Part II: Using General Block Preconditioners*, SIAM Journal on Numerical Analysis, 31(5), pp.1352–1367, 1994.

[112] P. Sonneveld, *CGS, a Fast Lanczos-Type Solver for Nonsymmetric Linear Systems*, SIAM Journal on Scientific and Statistical Computing, 10, pp.36–52, 1989.

[113] M. Stoll, *Solving Linear Systems Using the Adjoint*, DPhil Thesis, University of Oxford, 2008.

[114] M. Stoll, and A. Wathen, *All-at-Once Solution of Time-Dependent PDE-Constrained Optimization Problems*, Oxford Centre for Collaborative Applied Mathematics Technical Report 10/47, 2010.

[115] M. Stoll, and A. Wathen, *All-at-Once Solution of Time-Dependent Stokes Control*, Journal of Computational Physics, 232, pp.498–515, 2013.

[116] M. Stoll, and A. Wathen, *Combination Preconditioning and the Bramble-Pasciak$^+$ Preconditioner*, SIAM Journal on Matrix Analysis and Applications, 30(2), pp.582–608, 2008.

[117] M. Stoll, and A. Wathen, *Preconditioning for Partial Differential Equation Constrained Optimization with Control Constraints*, Numerical Linear Algebra with Applications, 19, pp.53–71, 2012.

[118] S. Takacs, and W. Zulehner, *Convergence Analysis of Multigrid Methods with Collective Point Smoothers for Optimal Control Problems*, Computing and Visualization in Science, 14, p.131–141, 2011.

[119] H. S. Thorne, *Properties of Linear Systems in PDE-Constrained Optimization. Part I: Distributed Control*, Rutherford Appleton Laboratory Technical Report RAL-TR-2009-017, submitted to SIAM Journal on Matrix Analysis and Applications, 2009.

[120] H. S. Thorne, *Properties of Linear Systems in PDE-Constrained Optimization. Part II: Neumann Boundary Control*, Rutherford Appleton Laboratory Technical Report RAL-TR-2009-018, submitted to SIAM Journal on Matrix Analysis and Applications, 2009.

[121] L. N. Trefethen, and D. Bau, III, *Numerical Linear Algebra*, SIAM, 1997.

[122] F. Tröltzsch, *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*, Graduate Series in Mathematics, American Mathematical Society, 2010.

[123] M. Ulbrich, *Constrained Optimal Control of Navier-Stokes Flow by Semismooth Newton Methods*, Systems & Control Letters, 48, pp.297–311, 2003.

[124] H. A. Van der Vorst, *Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems*, SIAM Journal on Scientific and Statistical Computing, 13(2), pp.631–644, 1992.

[125] A. J. Wathen, *Realistic Eigenvalue Bounds for the Galerkin Mass Matrix*, IMA Journal of Numerical Analysis, 7(4), pp.449–457, 1987.

[126] A. J. Wathen, and T. Rees, *Chebyshev Semi-Iteration in Preconditioning for Problems Including the Mass Matrix*, Electronic Transactions on Numerical Analysis, 34, pp.125–135, 2009.

[127] P. Wesseling, *An Introduction to Multigrid Methods*, Pure and Applied Mathematics (New York), John Wiley & Sons Ltd., Chichester, 1992.

[128] W. Zulehner, *Analysis of Iterative Methods for Saddle Point Problems: A Unified Approach*, Mathematics of Computation, 71, pp.479–505, 2002.

[129] W. Zulehner, *Nonstandard Norms and Robust Estimates for Saddle Point Problems*, SIAM Journal on Matrix Analysis and Applications, 32(2), pp.536–560, 2011.