Preconditioners for incompressible magnetohydrodynamics

by

Michael P. Wathen

M.Sci, Mathematics, University of Birmingham, 2012

M.Sc, Computer Science, The University of British Columbia, 2014

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate and Postdoctoral Studies

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

December 2018

© Michael P. Wathen 2018

The following individuals certify that they have read, and recommend to the
Faculty of Graduate and Postdoctoral Studies for acceptance, the dissertation entitled:

Preconditioners for incompressible magnetohydrodynamics

submitted by Michael Wathen in partial fulfillment of the requirements for

the degree of Doctor of Philosophy

in Computer Science

Examining Committee:

Chen Greif, Computer Science
Supervisor

Robert Bridson, Computer Science
Supervisory Committee Member

Uri Ascher, Computer Science
Supervisory Committee Member

Ian Mitchell, Computer Science
University Examiner

Brian Wetton
University Examiner

Hans De Sterck, Monash University
External Examiner

Abstract

The main goal of this thesis is to design efficient numerical solutions to incompressible magnetohydrodynamics (MHD) problems, with focus on the solution of the large and sparse linear systems that arise. The MHD model couples the Navier-Stokes equations that govern fluid dynamics and Maxwell's equations which govern the electromagnetic effects.

We consider a mixed finite element discretization of an MHD model problem. Upon discretization and linearization, a large block 4-by-4 nonsymmetric linear system needs to be (repeatedly) solved. One of the principal challenges is the presence of a skew-symmetric term that couples the fluid velocity with the magnetic field.

We propose two distinct preconditioning techniques. The first approach relies on utilizing and combining effective solvers for the mixed Maxwell and the Navier-Stokes sub-problems. The second approach is based on algebraic approximations of the inverse of the matrix of the linear system. Both approaches exploit the block structure of the discretized MHD problem. We perform a spectral analysis for ideal versions of the proposed preconditioners, and develop and test practical versions. Large-scale numerical results for linear systems of dimensions up to 10^7 in two and three dimensions validate the effectiveness of our techniques.

We also explore the use of the Conjugate Gradient (CG) method for saddle-point problems with an algebraic structure similar to the time-harmonic Maxwell problem. Specifically, we show that for a nonsingular saddle-point matrix with a maximally rank-deficient leading block, there are two sufficient conditions that allow for CG to be used.

An important part of the contributions of this thesis is the development of numerical software that utilizes state-of-the-art software packages. This software is highly modular, robust, and flexible.

Lay Summary

The goal of this thesis is to develop new, efficient solution techniques for large-scale multi-physics problems arising from fluid dynamics and electromagnetics. These problems appear in many physical applications, from glacial to astrophysical applications, and thus the development of models and simulations is of great importance.

For such applications, it is either impossible or impractical to consider a pen-and-paper approach, and hence, a computer-based solution is typically required. Large systems of unknowns describe the physical flow, and sophisticated computational solution techniques are needed.

We focus on developing modern solution algorithms and a large computer code that combines several state-of-the-art numerical software packages. Our algorithms exploit the underlying numerical and physical properties of the model. Our analysis illustrates favorable convergence properties, and by applying our algorithms to several challenging physical problems, we show high scalability for problems of tens of millions of unknowns.

Preface

This thesis describes results in three research articles:

1. M. Wathen, C. Greif, and D. Schötzau. Preconditioners for Mixed Finite Element Discretizations of Incompressible MHD Equations. *SIAM Journal on Scientific Computing*, 39(6):A2993–A3013, 2017

2. C. Greif, and M. Wathen. Conjugate Gradient for Nonsingular Saddle-Point Systems with a Maximally Rank-Deficient Leading Block. Under revision for Journal of Computational and Applied Mathematics (13 pages)

3. C. Greif, and M. Wathen. A Scalable Approximate Inverse-Based Preconditioner for Incompressible Magnetohydrodynamics. In final stages of preparation (15 pages)

Paper 1 has been published and is described in Chapter 2. Paper 3 is in the final stages of preparation and is presented in Chapter 3. Paper 2 is currently under revision and forms Chapter 4. We choose to present the papers in this order because paper 1 and paper 3 are strongly related to each other. I have taken a central role in the research, analysis of the methods and the generated data, and the methodology for all three pieces of work. In Paper 1, my two senior co-authors provided guidance on the discretization process and the numerical algorithms. Papers 2 and 3 were co-authored with my research supervisor. In all three papers, I am responsible for the numerical solution algorithms and techniques derived, which is the core of my thesis. I have received guidance and advice from my research supervisor. All three pieces of work have been written in majority by me with the assistance and editing of the other authors.

This work includes numerical software written solely by me, which effectively combines the finite element software *FEniCS* [77] in conjunction with the *PETSc4PY* package (Python interface for *PETSc* [7–9, 30]), the multigrid package *HYPRE* [37, 38], and the sparse direct solver *MUMPS* [4, 5]. These papers appear largely as they are published. However, for consistency purposes we have altered notation throughout the thesis.

Table of Contents

Abstract . . . . . **iii**

Lay Summary . . . . . **iv**

Preface . . . . . **v**

Table of Contents . . . . . **vi**

List of Tables . . . . . **ix**

List of Figures . . . . . **xii**

List of Programs . . . . . **xiii**

Acknowledgements . . . . . **xiv**

Dedication . . . . . **xv**

1 Introduction **1**

1.1 Model problem 1

1.1.1 Incompressible magnetohydrodynamics 2

1.1.2 Navier-Stokes equations 2

1.1.3 Maxwell's equations 3

1.2 Finite element discretization of PDEs 4

1.2.1 Laplacian example 5

1.2.2 Mixed discretizations of the model problems 6

1.3 Iterative solution of sparse linear systems 8

1.3.1 Krylov subspace methods 9

1.3.2 Preconditioning 12

1.3.3 Review of preconditioners for saddle-point systems 13

1.4 Numerical software 16

1.4.1 PDE discretization: *FEniCS* 17

1.4.2 Solution of the linear system: *PETSc* 18

1.5 Outline and contributions 21
1.6 Notation 23

2 Preconditioners for Mixed Finite Element Discretizations of Incompressible MHD Equations 25
2.1 Discretization 25
2.1.1 Mixed finite element approximation 25
2.1.2 Picard iteration 27
2.1.3 Decoupling 28
2.1.4 The linear systems 29
2.2 Review of preconditioning techniques for the sub-problems 31
2.2.1 Fluid flow preconditioner 31
2.2.2 Maxwell preconditioner 33
2.3 Preconditioners for the MHD system 34
2.3.1 Reordering 35
2.3.2 From an ideal to a practical preconditioner 38
2.3.3 Implementation 39
2.4 Numerical results 40
2.4.1 2D smooth solution 42
2.4.2 2D smooth solution parameter tests 42
2.4.3 2D smooth solution on L-shaped domain 45
2.4.4 2D singular solution on L-shaped domain 45
2.4.5 2D Hartmann flow 46
2.4.6 3D smooth solution 48

3 An Approximate Inverse-Based Preconditioner for Incompressible Magnetohydrodynamics 50
3.1 Newton's method discretization of the MHD model 50
3.2 A new formula for the inverse of the MHD coefficient matrix 52
3.3 A new approximate inverse-based preconditioner 54
3.3.1 A sparse approximation of the Schur complement 54
3.3.2 A practical preconditioner 55
3.3.3 Spectral analysis 56
3.4 A block triangular preconditioner 58
3.5 Numerical experiments 62
3.5.1 3D Cavity driven flow 63
3.5.2 Fichera corner 64
3.5.3 MHD generator 65

4 Conjugate gradient for nonsingular saddle-point systems with a maximally rank-deficient leading block 67
4.1 Problem statement 67
4.2 Krylov Subspace 69
4.3 Null-space decoupling 72
4.4 Eigenvalue Analysis 73
4.5 Numerical Experiments 75
4.5.1 Krylov Subspace Solver Test 78
4.5.2 Divergence and Non-Divergence Free Right-Hand-Side 78
4.5.3 Variable Coefficients 80
4.5.4 Fichera Corner Problem 81
4.5.5 Gear Domain 81

5 Conclusions and future work 84
5.1 Conclusions 84
5.2 Future work 84

Bibliography 86

List of Tables

Table 1.1 Notation for block systems 23
Table 1.2 Notation for superscripts 24
Table 1.3 Notation for subscripts 24

Table 2.1 Algebraic multiplicity of eigenvalues for preconditioned matrices associated with $\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}}$ and $\widetilde{\mathcal{M}}_{\mathrm{S}}^{\mathrm{MHD}}$. Note that $n_c = n_u - n_b + m_b$ . . . 38
Table 2.2 Orders of matrix entries for relevant discrete operators 38
Table 2.3 Solution methods for systems associated with separate block matrices within (2.35) 40
Table 2.4 2D smooth: Number of nonlinear iterations and number of iterations to solve the MHD system with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$. 43
Table 2.5 Number of nonlinear iterations for various values of ν with $\kappa = 1$ and $\nu_m = 10$. 44
Table 2.6 Average linear solver time for various values of ν with $\kappa = 1$ and $\nu_m = 10$. 44
Table 2.7 Number of nonlinear iterations for various values of κ with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-5, $\nu = 1$ and $\nu_m = 10$. 44
Table 2.8 Average linear solver time for various values of κ with $\nu = 1$ and $\nu_m = 10$. 45
Table 2.9 2D L-shaped: Number of nonlinear iterations and number of iterations to solve the MHD system with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$. The iteration was terminated before completion for $\ell = 9$ due to the computation reaching the prescribed time limit. 46
Table 2.10 2D singular solution on an L-shaped domain: Number of nonlinear iterations and number of iterations to solve the MHD system with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$. 47
Table 2.11 2D Hartmann: Number of nonlinear iterations and number of iterations to solve the MHD system with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 1000$ 48

Table 2.12 3D smooth: Number of nonlinear iterations and number of iterations to solve the MHD system with tol_{NL} = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$ 49

Table 3.1 Block coefficient matrices, their corresponding continuous operators and approximate norms for the Fluid matrices 51
Table 3.2 Block coefficient matrices, their corresponding continuous operators and approximate norms for the Magnetic matrices 52
Table 3.3 Block coefficient matrices, their corresponding continuous operators and approximate norms for the Newton matrices 52
Table 3.4 Solution method for block systems associated with the preconditioners 62
Table 3.5 3D Cavity Driven using both the approximate inverse and block triangular preconditioner with parameters $\kappa = 1$, $\nu = 1$, $\nu_m = 1$ and Ha = 1. 63
Table 3.6 3D Cavity Driven using both the approximate inverse and block triangular preconditioner with parameters κ = 1e1, ν = 1e-1, ν_m = 1e-1 and Ha = $\sqrt{1000}$ 64
Table 3.7 Numerical cost of using $\mathcal{M}_{\text{A}}^{\text{MHD}}$ and $\mathcal{M}_{\text{B}}^{\text{MHD}}$ for mesh level $\ell = 6$ in Table 3.6 65
Table 3.8 Fichera corner using the approximate inverse preconditioner. Setup 1: κ =1e1, ν =1e-2, ν_m =1e-2 and Ha=$\sqrt{1e5}$ and Setup 2: κ =1e1, ν =1e-2, ν_m =1e-3 and Ha=1000. 66
Table 3.9 MHD generator using the approximate inverse preconditioner with parameters $\kappa = 1$, ν = 1e-1, ν_m = 1e-1 and Ha = 10 66

Table 4.1 Eigenpairs and algebraic multiplicities for the preconditioned matrices $\widehat{\mathcal{M}}_{\text{I}}^{-1}K$ and $\mathcal{M}_{\text{I}}^{-1}K$. We use the notation $b_E \in \text{Null}(E)$, $b_N \in \text{Null}(N)$, $r_R \in \text{Null}(R)$, and $l = \text{Dim}(\text{Null}(R))$. 75
Table 4.2 Krylov solver test: time and iteration results for using one iterations of the preconditioner with FCG, MINRES and GMRES. The viscosity for these tests is ν_m = 1e-2 78
Table 4.3 Block preconditioner test: time and iteration results using the block diagonal ($\mathcal{M}_P$), block upper triangular ($\mathcal{M}_{\text{U}}$) and block lower triangular ($\mathcal{M}_{\text{L}}$) preconditioners with ν_m = 1e-2 79
Table 4.4 Divergence-free vs. non-divergence-free right-hand-side: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for divergence and non-divergence free right-hand-sides with ν_m =1e-2. . . 79
Table 4.5 Variable coefficients: time and iteration results using the block diagonal preconditioner, $\mathcal{M}$, for various different values of a. 80

Table 4.6 Variable coefficients: time and iteration results using the block diagonal and triangular preconditioners for $a = 100$. 81
Table 4.7 Fichera corner: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for various different values of ν_m. 82
Table 4.8 Gear domain: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for various different values of ν_m. 83

List of Figures

Figure 1.1 Unit square mesh generated by *FEniCS* 5

Figure 1.2 L-shaped domain generated using *mshr*. We locally refine cells that are a distance 0.4 from the origin. 19

Figure 2.1 Eigenvalues of preconditioned matrices for the smooth solution given in this section. The number of degrees of freedom for these matrices is 724. 43

Figure 3.1 Preconditioned eigenvalue plots for (a) the approximate inverse preconditioner in (3.12) and (b) the block triangular preconditioner in (3.21) using the smooth solution given (3.31). The dimension of the matrices in this example is 1399×1399. 59

Figure 4.1 Eigenvalue distribution of the preconditioned matrix $\widetilde{\mathcal{M}}_{\mathrm{I}}^{-1}K$ using randomly generated blocks with $n = 100$, $m = 20$, and $l = 5$ 76

Figure 4.2 Fichera corner domain for mesh level, $\ell = 1$ 81

Figure 4.3 Gear domain for mesh level, $\ell = 5$ 82

List of Programs

1.1 *mshr*: construction of a locally refined mesh on an L-shaped domain in Figure 1.1. The cells that are within a certain distance from the radius. 18
1.2 *FEniCS*: construction of the bilinear forms for the MHD model in (1.1) . 19
1.3 *PETSc*: our implementation of a block diagonal preconditioner. 21
1.4 *PETSc*: example of how using a Python class as a preconditioner 22

Acknowledgements

First and foremost I would like to thank my supervisor, Chen Greif. Your guidance, help, and encouragement has been invaluable throughout my time at UBC. To my supervisory committee, Uri Ascher and Robert Bridson – thank you for your time, support and helpful feedback.

There are too many friends that have supported me along the way but I would like to say a special thank you to Mike Mitchell, Michael Firmin, Anna Mittelholz, Anna Grau Galofre, Thibaut Astic, Rowan Cockett, Hannah Loch, Greg Bex, Lili Geelhand, Joli Fooken, Colin Rowell, Keelin Scully, Julie Nutini, Jilmarie Stephens, and Luna. You are all a little crazy, but I suppose that's why we have been friends for so many years. Next, I would like to thank two very important places, Bean Around The World and Boulevard coffee shops. I shudder to think how much I have spent at these places in the six years I have been in Vancouver but what I do know is that without either place this thesis wouldn't be what it is today.

Finally, I would like to thank my family for their unwavering support throughout my PhD. To Grandpa, I've enjoyed everyone of our "Skrype" talks, but now we'll have to make do with just telephone calls instead. To my parents, it's simple, this thesis wouldn't have happened without you.

To Granny and Elizabeth – the old and the new

Chapter 1

Introduction

The central topic of this thesis is the numerical solution of elliptic partial differential equations (PDEs) that model electromagnetics problems. Developing the continuous model, namely the PDE and its associated boundary conditions, requires significant physical and mathematical efforts. Once the continuous model is given, the main steps for the numerical solution may be described as follows:

1. discretization and linearization (when applicable);
2. solution of the linear system;
3. implementation.

Our interest is in the numerical solution of the problem, and we primarily focus on items 2 and 3.

In this introductory chapter, we first look at the model problems that we consider (Section 1.1). Following that, we briefly review finite element discretizations and relevant literature (Section 1.2). We then give an overview of iterative solution of large and sparse linear systems (Section 1.3). Next, we discuss numerical software for solving these problems (Section 1.4). Finally, we provide an outline and contributions of this thesis (Section 1.5).

1.1 Model problem

We consider the incompressible magnetohydrodynamics (MHD) model that describes the behavior of electrically conductive incompressible fluids (liquid metals, plasma, salt water, etc.) in an electromagnetic field. It has a number of important applications in technology and industry, along with geophysical and astrophysical applications. Some such applications are electromagnetic pumping, aluminum electrolysis the Earth's molten core, and solar flares. See [45, 111] for a comprehensive description of the physical problem.

The two key main components of the MHD model are the Navier-Stokes equations, which govern the fluid dynamics, and Maxwell's equations, which govern the electromagnetics. These two problems are strongly coupled in the MHD model.

1.1.1 Incompressible magnetohydrodynamics

Following the formulation considered in [49, 95], the governing equations for the steady-state incompressible MHD model are:

$$-\nu\,\Delta \boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p - \kappa\,(\nabla\times\boldsymbol{b})\times\boldsymbol{b} = \boldsymbol{f} \qquad \text{in } \Omega, \tag{1.1a}$$

$$\nabla\cdot\boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{1.1b}$$

$$\kappa\nu_m\,\nabla\times(\nabla\times\boldsymbol{b}) + \nabla r - \kappa\,\nabla\times(\boldsymbol{u}\times\boldsymbol{b}) = \boldsymbol{g} \qquad \text{in } \Omega, \tag{1.1c}$$

$$\nabla\cdot\boldsymbol{b} = 0 \qquad \text{in } \Omega. \tag{1.1d}$$

Here, Ω is a bounded Lipschitz polygonal or polyhedral domain of $\mathbb{R}^d$ for $d = 2, 3$ with boundary $\partial\Omega$. The unknowns are: the velocity $\boldsymbol{u}$, the hydrodynamic pressure p, the magnetic field $\boldsymbol{b}$ and the Lagrange multiplier r associated with the divergence constraint on the magnetic field. We comment on the role of r in Section 1.1.3. The functions $\boldsymbol{f}$ and $\boldsymbol{g}$ represent external forcing terms. The equations (1.1) are characterized by three dimensionless parameters: the hydrodynamic Reynolds number $\mathrm{Re} = 1/\nu$, the magnetic Reynolds number $\mathrm{Rm} = 1/\nu_m$, and the coupling number κ.

In the case of liquid metals, the ratio between magnetic viscosity, ν_m, and fluid viscosity, ν, tends to be small. For example, Mercury has a ratio of about 10^{-7} with $\nu_m \approx 10^4 - 10^5$, $\nu \approx 10^{-2} - 10^{-4}$ and $\kappa \approx 10^2 - 10^9$; cf. [6]. In [6] the authors define *strong coupling* for $10^2 \leq \kappa \leq 10^9$. For more discussion of the physical parameters we refer the reader to [6, 44, 45, 72, 90].

For simplicity, we consider homogeneous Dirichlet boundary conditions:

$$\boldsymbol{u} = \boldsymbol{0}, \quad \boldsymbol{n}\times\boldsymbol{b} = \boldsymbol{0}, \quad r = 0 \quad \text{on } \partial\Omega, \tag{1.2}$$

with $\boldsymbol{n}$ being the unit outward normal on $\partial\Omega$.

1.1.2 Navier-Stokes equations

The steady-state incompressible Navier-Stokes equations that govern incompressible fluid flow are given by:

$$-\nu\,\Delta \boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p = \boldsymbol{f} \qquad \text{in } \Omega, \tag{1.3a}$$

$$\nabla\cdot\boldsymbol{u} = 0 \qquad \text{in } \Omega. \tag{1.3b}$$

Again, here $\boldsymbol{u}$ and p are the velocity and pressure of the fluid, $\boldsymbol{f}$ is the forcing term and ν is the kinematic viscosity. For a classical overview of the Navier-Stokes equations we

refer the reader to [1], and for a comprehensive numerical study of this model, see [35].

One of the key numerical challenges in the approximation and numerical solution of the Navier-Stokes equations is the presence of the nonlinear convection term $(\boldsymbol{u} \cdot \nabla)\boldsymbol{u}$. This term requires the use of a nonlinear iteration scheme. The simplest common choice of a nonlinear solver is the linearly convergent Picard iteration. For a given initial guess $\boldsymbol{u}_0$, the approximate solution at the $k+1^{\text{st}}$ iteration, $\boldsymbol{u}_{k+1}$, is given by:

$$\begin{aligned} -\nu\,\Delta \boldsymbol{u}_{k+1} + (\boldsymbol{u}_k \cdot \nabla)\boldsymbol{u}_{k+1} + \nabla p &= \boldsymbol{f} \qquad \text{in } \Omega, \\ \nabla \cdot \boldsymbol{u}_{k+1} &= 0 \qquad \text{in } \Omega, \end{aligned}$$

where $k = 0, 1, \ldots$. Here, the scalar pressure, p, appears in linear form and is not iterated upon.

An alternative to the Picard iteration is the locally quadratically convergent Newton's method. That is, Newton's method will converge quadratically if the initial guess is "close enough" to the solution. To form a suitable initial guess, it is common to use the solution of the Picard scheme after a few iterations. Newton's method is highly effective, and is considered to be the method of choice for problems such as the Navier-Stokes equations. This scheme is slightly more elaborate than the Picard iteration; see [35, Section 8.2.2]. In Chapter 3, we will discuss Newton's method for the MHD model.

Upon discretization, the convection term $((\boldsymbol{u}\cdot\nabla)\boldsymbol{u})$ causes nonsymmetry or possibly skew-symmetry of the matrix within the discrete Navier-Stokes model. We also note that as $\nu \to 0$, convection dominates the diffusion terms $(\nu\Delta\boldsymbol{u})$, which causes turbulent (nonviscous) flow. This may cause significant difficulties both in terms of the solution method and the discretization used. See [35] for an extensive discussion.

1.1.3 Maxwell's equations

The second coupled subproblem in the MHD model (1.1) is the time-harmonic Maxwell equation in mixed form [51, 60, 78]. It is given as follows:

$$\nabla \times \nu_m \nabla \times \boldsymbol{b} + \nabla r = \boldsymbol{g} \qquad \text{in } \Omega \tag{1.5a}$$

$$\nabla \cdot \boldsymbol{b} = 0 \qquad \text{in } \Omega, \tag{1.5b}$$

where $\boldsymbol{b}$ is the magnetic field and r is the Lagrange multiplier associated with the divergence constraint on the magnetic field. The constant ν_m is the magnetic viscosity. We note that it is not necessary to introduce the Maxwell's equations with an incompressibility constraint and a Lagrange multiplier; see for example [60, 61, 88]. The following

tensor identities involving the curl operator are useful:

$$\nabla \cdot (\nabla \times \boldsymbol{a}) = 0 \quad \text{and} \quad \nabla \times (\nabla \boldsymbol{a}) = 0, \tag{1.6}$$

where $\boldsymbol{a}$ is an arbitrary vector field. Taking the divergence of (1.5a) and using (1.6), we obtain the Poisson problem

$$\Delta r = \nabla \cdot \boldsymbol{g} \quad \text{in } \Omega, \quad r = 0 \quad \text{on } \partial\Omega.$$

Since $\boldsymbol{g}$ is divergence-free in many physically relevant applications, the solution for r of the above Poisson problem is zero. Alas, from a numerical point of view the introduction of the Lagrange multiplier may be beneficial in terms of numerical stability; see [33, 49][1].

From (1.6), the kernel of the curl operator is the gradient operator. This may cause numerical difficulties due to the large dimension of the null-space of the discrete curl operator. Numerical methods often attempt to exploit this null-space. This is the approach we take.

1.2 Finite element discretization of PDEs

In this section, we briefly discuss the essential parts of the numerical discretization of PDEs of the kind we consider. Throughout, we consider boundary value problems and steady-state PDEs. We cover linear and nonlinear problems.

Given a continuous model expressed as a PDE, a typical numerical solution procedure includes a discretization process. We first generate a mesh. Two common choices of tessellation of this mesh are quadrilaterals or triangles. An example of a simple uniform square mesh with a triangular tessellation is given in Figure 1.1.

Once the mesh is generated, the next goal is to approximate the continuous problem by a discrete model. In practice this amounts to converting the differential equation to a difference equation. There are three common choices of discretization techniques: finite differences [79, 100], finite volumes [42, 70], and finite elements [17, 99]. We focus on finite element discretizations, which are based on considering solutions in weak form, expressed in terms of basis functions of a specific space. These methods have a rich mathematical theory behind the numerical approximation, and can relatively easily handle complex domains and unstructured meshes.

[1] The stability result is discussed in detail in [33, Section 3.3], where a nonzero wave number is incorporated into the equation and is then taken to zero. The Helmholtz decomposition can be used along with an inf-sup stability analysis.

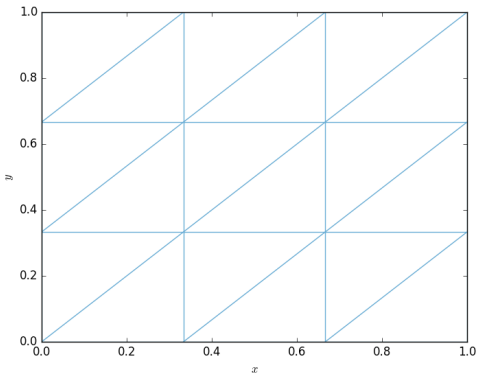


Figure 1.1: Unit square mesh generated by *FEniCS*

A general finite element method approach follows the steps given bellow:

1. Obtain a weak formulation
2. Choose an approximation space for the so-called trial functions
3. Choose an approximation space for the so-called test functions
4. Solve the linear system

The meaning of trial and test functions is illustrated in the example that follows.

1.2.1 Laplacian example

Here we present a finite element discretization of the "hello world" numerical test problem: Poisson's equation. For simplicity we consider homogeneous Neumann boundary conditions to form the problem as follows:

$$\Delta u = f \quad \text{in } \Omega \quad \text{with} \quad \frac{\partial u}{\partial n} = 0 \quad \text{on } \partial\Omega. \tag{1.7}$$

The standard weak form of the Poisson problem can be obtained by multiplying (1.7) by a smooth *test* function v to obtain:

$$-\int_\Omega v\Delta u = \int_\Omega vf. \tag{1.8}$$

The function u is considered the *trial* function. Using integration by parts and the divergence theorem, the left-hand-side of (1.8) can be written as:

$$-\int_\Omega v\Delta u = \int_\Omega \nabla u \cdot \nabla v - \int_\Omega \nabla \cdot (v\nabla u) = \int_\Omega \nabla u \cdot \nabla v - \int_{\partial\Omega} v\frac{\partial u}{\partial n},$$

where $\frac{\partial u}{\partial n}$ denotes the directional derivative in the direction of the normal component on the boundary. Using the homogeneous Neumann condition ($\frac{\partial u}{\partial n} = 0$ on $\partial\Omega$), the weak form is given by:

$$\int_\Omega \nabla u \cdot \nabla v = \int_\Omega v f. \tag{1.9}$$

The next step for a finite element discretization is to determine the approximation spaces for the trial (u) and test (v) functions. For a given suitable Sobolov space, V, consider a solution $u \in V$.

The most intuitive choice of approximation space for the trial function is the same as the test functions, thus, $v \in V$. This is called *Galerkin* finite elements. We define ϕ_i as basis functions of V and represent the test and trial functions with respect to this basis as:

$$v = \sum_{j=1}^{n} v_j \phi_j \quad \text{and} \quad u = \sum_{i=1}^{n} u_i \phi_i. \tag{1.10}$$

Substituting (1.10) into the weak form in (1.9) gives the matrix form as:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} v_j \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \, u_i = \sum_{j=1}^{n} v_j \int_\Omega \phi_j f. \tag{1.11}$$

We can re-write (1.11) as:

$$\sum_{i=1}^{n} \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \, u_i = \int_\Omega \phi_j f \quad \text{for } j = 1, 2, \ldots, n.$$

From this an $n \times n$ linear system is defined:

$$Ku = b,$$

where

$$K_{i,j} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \quad \text{and} \quad b_j = \int_\Omega \phi_j f.$$

Here K and b are known as the *stiffness matrix* and *load vector*, respectively.

1.2.2 Mixed discretizations of the model problems

It is common in many PDE settings that the continuous model has multiple unknowns. For example, the Navier-Stokes equations (1.3) has two variables, velocity and pressure. For such models it is common to consider different finite elements for each variable. This is called a mixed finite element discretization. This thesis will consider mixed finite element discretizations for electromagnetic and fluid flow problems [35, 95, 111].

In Section 1.1 we discussed the three model problems that are our focus. Research into the discretization of these model problems is a very active area, ranging from classical methods to new discretizations. Here we will give a brief overview of possible mixed discretization strategies for each problem.

Navier-Stokes equations

Computational fluid dynamics is of huge importance. Examples of applications are flood modeling, computational aerodynamics, glacial flows and biomedical engineering.

There are many choices for a mixed finite element discretization of the Navier-Stokes equations (1.3). For mixed discretizations it is important to consider what is known as stable discretizations. Suppose the weak formulation of the Navier-Stokes is defined using the finite-dimensional spaces $\boldsymbol{X}^h$ and M^h. Then for $(\boldsymbol{X}^h, M^h)$ to be a stable discretization the finite-dimensional spaces must satisfy the Ladyzhenskaya-Babuška-Brezzi (LBB) (or inf–sup) compatibility condition [19]. For a positive γ independent of the mesh size h, the LBB condition is:

$$\min_{q_h \neq \text{constant}} \max_{\boldsymbol{v}_h \neq \boldsymbol{0}} \frac{|(q_h, \nabla \cdot \boldsymbol{v}_h)}{\|\boldsymbol{v}_h\|_{1,\Omega} \|q_h\|_{0,\Omega}} \geq \gamma,$$

where $(\boldsymbol{v}_h, q_h) \in (\boldsymbol{X}^h, M^h)$, $\|\boldsymbol{v}_h\|_{1,\Omega} = (\int_\Omega \boldsymbol{v}_h \cdot \boldsymbol{v}_h + \nabla \boldsymbol{v}_h : \nabla \boldsymbol{v}_h)^{1/2}$, $\|q_h\|_{0,\Omega} = \|q - \frac{1}{|\Omega|}\int_\Omega q\|$, $\nabla \boldsymbol{u} : \nabla \boldsymbol{v} = \sum_{i,j=1}^{d} (\nabla \boldsymbol{u})_{ij} (\nabla \boldsymbol{v})_{ij}$ and d is the spatial dimension.

One of most widely used mixed element that is LBB stable is the Taylor-Hood mixed element [101]. The lowest order Taylor-Hood element is (P_2/P_1), where the velocity field is approximated with nodal quadratic polynomials and the pressure variable with nodal linear polynomials. This is the discretization we use throughout the thesis. We note, though, that low order stabilized elements are also widely used; see, for example, [35, Section 8.3].

Maxwell's equations

Similarly to the Navier-Stokes equations, the mixed Maxwell problem is discretized using a mixed finite element method. The mixed discretization of the Maxwell problem (1.5) has been extensively studied; see, for example [24, 84, 106]. It is well known that for nonconvex domains nodal approximations of the magnetic field, $\boldsymbol{b}$, often fail to capture the singular solution [28]. However, for smooth domains nodal approximations appear to be effective. Thus for the mixed formulation, the common choice of finite elements spaces would be to use $H(\text{curl})$ conforming elements for the magnetic field and H^1 elements for the multiplier. These are more natural for the approximation of the curl-

operator and can be applied in a seamless fashion, without penalty or stabilization, to problems that involve nonsmooth domains, singularities, and other challenging settings.

This leads to a mixed finite element discretization based on $H(\text{curl})$ conforming edge elements known as Nédélec [81] elements for $\boldsymbol{b}$ and nodal elements of equal order for r. We consider first order Nédélec elements for the magnetic field and linear nodal elements for the multiplier variables.

MHD model

The mixed finite element discretization of the MHD model incorporates many of the proposed techniques for the two subproblems. The common discretizations for the MHD model are derived from three-field formulation $(\boldsymbol{u}, p, \boldsymbol{b})$ or the four-field formulation given in (1.1).

For the three-field formulation of the MHD model, (1.1) is considered without the Lagrange multiplier. In [44, 53] the authors propose an exact penalty formulation of a stationary MHD model. The exact penalty formulation reduces the curl-curl operator in (1.1c) and the incompressibility condition in (1.1d) to a vector Laplacian using the vector identity:

$$\Delta a = \nabla(\nabla \cdot a) - \nabla \times (\nabla \times a)\,. \tag{1.12}$$

This discretization is based on standard nodal elements for the approximation of the magnetic field ($\boldsymbol{b}$). As with Maxwell's equations in isolation, nodal discretizations of the magnetic field are effective in smooth settings, but they often fail to capture singularities in nonsmooth domains; see [95] and the references therein.

In [95], the author introduces a mixed function space which utilizes H^1 elements for the velocity field and multiplier variable, L^2 elements for the pressure variables and $H(\text{curl})$ elements for the magnetic field. Therefore, the proposed mixed finite element discretization we use incorporates the two mixed discretizations given for the Navier-Stokes and Maxwell subproblems. This entails using the lowest order Taylor-Hood elements for the $(\boldsymbol{u}, p)$ and the lowest order mixed Nédélec approximation for $(\boldsymbol{b}, r)$.

1.3 Iterative solution of sparse linear systems

In Section 1.2, we saw that the finite element discretization leads to large and sparse linear systems. In this section, we discuss iterative solution methods for such linear systems.

Consider the linear system:[2]

$$Kx = b, \tag{1.13}$$

where K is a nonsingular matrix of dimensions $n \times n$ and b is a vector of length n. If the dimension n is sufficiently small, direct solvers are the method of choice. However, we are considering large and sparse linear systems arising from three-dimensional problems, for which direct methods (that is, techniques based on matrix decompositions) are infeasible to use due to computational time and memory requirements. Such systems therefore require an iterative solution method. The state-of-the-art class of iterative methods are Krylov subspace methods. There are many excellent books in this area; see, for example, [48, 93].

1.3.1 Krylov subspace methods

For a given initial guess, x_0, the initial residual of (1.13) is given as

$$r_0 = b - Kx_0.$$

The k-dimensional Krylov subspace is defined as:

$$\mathbb{K}_k(K; r_0) = \text{span}\{r_0, Kr_0, K^2 r_0, \cdots, K^{k-1} r_0\}.$$

Krylov subspace methods are based on finding an approximate solution, x_k, in the Krylov subspace:

$$K^{-1}b \approx x_k = x_0 + Q_k\, y_k, \tag{1.14}$$

where $Q_k \in \mathbb{R}^{n \times k}$ is a matrix whose columns form an orthogonal basis for the subspace $\mathbb{K}_k(K; r_0)$ and y_k is a vector of length k. The procedure that forms the orthogonal basis to the Krylov subspace is the Arnoldi process for nonsymmetric matrices and the Lanzcos process for symmetric matrices. Here we will give a brief outline of the Arnoldi and Lanzcos methods, which respectively form the foundation of the GMRES [94] and MINRES methods [83]. Both GMRES and MINRES are minimum residual methods; the approximate solution at the k^{th} iteration is given by the minimization problem:

$$\min_{x_k \in \mathbb{K}_k(K; r_0)} \|b - Kx_k\|_2. \tag{1.15}$$

GMRES and MINRES are methods that form an orthogonal basis of the Krylov subspace as a first step. Other methods such as BiCG [39], BiCGstab [104] and QMR [41], form a bi-orthogonal basis which relies on matrix-vector products using both K and

[2] In this section we switch to a slightly different notation, with x signifying the vector of unknowns.

K^T; we will not consider these methods.

The Arnoldi process is written as:

$$KQ_k = Q_{k+1}H_{k+1,k}, \tag{1.16}$$

where $H_{k+1,k} \in \mathbb{R}^{(k+1)\times k}$ is an upper Hessenberg matrix. Here, $Q_{k+1} \in \mathbb{R}^{n\times(k+1)}$ contains $(k+1)$ orthogonal vectors to the Krylov subspace and Q_k is the same matrix which contains the first k columns of Q_{k+1}. This is indeed the matrix referred to in (1.14). If we consider pre-multiplying (1.16) by Q_k^T then we obtain:

$$Q_k^T KQ_k = H_{k,k}, \tag{1.17}$$

where $H_{k,k}$ contains the first k rows of $H_{k+1,k}$. Thus, for the symmetric case it is clear that $H_{k,k}$ in (1.17) is symmetric. Therefore, we denote it as $T_{k+1,k}$ where $T_{k,k}$ is tridiagonal and $T_{k+1,k}$ has one extra row. This is called the Lanzcos process and is written as

$$KQ_k = Q_k T_{k+1,k}.$$

The tridiagonal matrix $T_{k+1,k}$ is significantly sparser than the upper Hessenberg matrix $H_{k+1,k}$, and indeed carrying out the Lanzcos process in the symmetric case is considerably cheaper than the Arnoldi process for the nonsymmetric case.

From now on we will refer to the Arnoldi process for nonsymmetric matrices, noting that the same can be done for the Lanczos process. Substituting (1.16) into (1.15), then y_k must solve the least-squares problem:

$$\begin{aligned} &\min_{y_k}\|b - K(x_0 + Q_k\, y_k)\|_2 = \min_{y_k}\|r_0 - KQ_k\, y_k\|_2 \\ &= \min_{y_k}\|r_0 - Q_{k+1}H_{k+1,k}\, y_k\|_2 = \min_{y_k}\|\beta\xi_1 - H_{k+1,k}\, y_k\|_2, \end{aligned} \tag{1.18}$$

where $\beta = \|r_0\|_2$ and ξ_1 is the first column of the identity matrix of dimension $k+1$. Thus, the solution y_k in (1.18) is now a least-squares problem. The common way of solving this is using a QR factorization of $H_{k+1,k}$. This QR factorization is done via Givens rotations in a sequential fashion, exploiting the structure to optimize the computations. For more details on this we refer the reader to [48, Chapter 2.4].

In summary, minimum residual methods are based on minimizing the residual with respect to $x_k \in \mathbb{K}_k(K; r_0)$. They start by constructing an orthogonal basis to the Krylov subspace. Using this, they compute the approximate iterates via small least-squares problems which use an upper Hessenberg matrix in the nonsymmetric case or a tridiagonal matrix in the symmetric case.

Eigenvalues

The following description follows the standard for any sparse linear algebra textbook, see for example [48, 93].

The solution at the k^{th} iteration is given in (1.14) where we recall that Q_k is made up of orthonormal vectors that span the Krylov subspace, thus,

$$x_k - x_0 \in \mathbb{K}_k(K; r_0) \quad \text{for } k = 1, 2, \ldots.$$

Therefore, for some coefficients β_i where $i = 0, 1, \ldots, k-1$, the residuals and the approximate solution iterates can be expressed as

$$x_k - x_0 = \sum_{i=0}^{k-1} \beta_i K^i r_0.$$

We can thus write the solution iterate in terms of a polynomial of K:

$$x_k = x_0 + q_{k-1}(K) r_0, \tag{1.19}$$

where q_{k-1} are polynomials of degree $k-1$. By premultiplying (1.19) by K and subtracting it from b we can represent the residual at the k^{th} iteration in terms of polynomials of the initial residual:

$$\begin{aligned} b - Kx_k &= b - Kx_0 + Kq_{k-1}(K) r_0, \\ r_k &= p_k(K) r_0, \end{aligned} \tag{1.20}$$

where $p_k(x) = 1 - xq_{k-1}(x)$ are polynomials of degree k with $p_k(0) = 1$. Since GMRES and MINRES are both minimum residual methods then:

$$\min_{x_k \in \mathbb{K}_k(K; r_0)} \|b - Kx_k\|_2 = \min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|p_k(K) r_0\|_2 \tag{1.21}$$

Suppose that K is diagonalizable, then

$$K = X \Lambda X^{-1}, \tag{1.22}$$

where X are the right eigenvectors of K and Λ is a diagonal matrix of eigenvalues of K. Substitution of (1.22) into (1.21) gives

$$\begin{aligned} \min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|p_k(K) r_0\| &= \min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|X p_k(\Lambda) X^{-1} r_0\| \\ &\leq \|X\| \|X^{-1}\| \min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|p_k(\Lambda)\| \|r_0\| \end{aligned} \tag{1.23}$$

In the symmetric case, X is orthogonal and therefore (1.23) reduces to

$$\min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|p_k(K)r_0\| \leq \min_{p_k \in \mathcal{P}_k,\ p_k(0)=1} \|p_k(\Lambda)\| \|r_0\|.$$

To obtain fast convergence, we require p_k to be small at the diagonal elements of Λ. This can be readily accomplished if K has a small number of eigenvalues, or if the eigenvalues are strongly clustered.

1.3.2 Preconditioning

We saw in Section 1.3.1 that Krylov subspace methods may rapidly converge if the eigenvalue distribution of K is favorable. However, in general, we have no control over the spectral distribution of K. Therefore, we introduce a matrix M which is known as a *preconditioner* to enable us to efficiently cluster eigenvalues and speed up convergence of the Krylov subspace solver. The preconditioned linear systems is given by

$$M^{-1}Kx = M^{-1}b, \tag{1.24}$$

where M is the preconditioner that approximates K to some degree. This is called left preconditioning because the inverse of the preconditioner is acting on the left.

Instead of left preconditioning in (1.24), it is possible to apply right preconditioning,

$$KM^{-1}y = b, \quad x = M^{-1}y,$$

or split preconditioning

$$M_1^{-1}KM_2^{-1}y = M_1^{-1}b, \quad x = M_2^{-1}y,$$

where $M = M_1M_2$. When either left, right or split preconditioning can be applied, there appears to be little evidence which technique speeds up the convergence of the Krylov subspace solver the most. However, there are differences among them in terms of the norm that is minimized and the ability to use flexible methods [92]. Henceforth, we will focus on left preconditioning.

Generally speaking, an effective preconditioner has the following properties:

1. the construction and application of M^{-1} is computationally efficient;
2. the eigenvalues of matrix $M^{-1}K$ are clustered.

It is often the case that it is infeasible to construct either M or $M^{-1}K$. Therefore, in practice, the iterative solver operates on K and applies M^{-1} at each iteration implicitly.

Thus, the action M^{-1} needs to be "fast" on a single vector which occurs within the iterative solver.

The simpler examples of preconditioners arise from sparse factorization or approximation of K. For example, these methods include the Jacobi preconditioner which takes the diagonal of K as the preconditioner and the Gauss–Seidel preconditioner which uses the lower triangular entries of K as the preconditioner. These methods can work well for a small class of problems; see for example [107]. However, in general the effectiveness of these preconditioners is limited. More sophisticated approaches are sought, such as incomplete factorizations, multigrid methods, and so on. For a full review of current preconditioning methods we refer the reader to [108].

Generally speaking, preconditioning can arguably be split into two main categories:

1. operator-based or PDE preconditioning,

2. "black-box" preconditioning.

Operator-based preconditioning typically arises from PDE based problems that utilize the underlying physical properties and discretization of the problem. The aim for such preconditioning approaches is to have scalable iterations. That is, as the mesh is refined and correspondingly the matrix dimensions increase, the number of iterations which the Krylov subspace solver takes to converge remains constant. In contrast, the "black-box" preconditioning approach is based more on an algebraic nature of the problem and constructs preconditioners using the matrix entries. Scalable "black-box" preconditioning approaches are typically harder to develop because there is no underlying application, and hence, there is less information on the matrix properties. In this thesis, we mainly consider operator-based preconditioning techniques.

1.3.3 Review of preconditioners for saddle-point systems

The mixed discretizations considered in this thesis lead to what is known as *saddle-point systems* [10, 108]. These linear systems are block structured and they feature two block variables, u and v. We call u and v the *primary* and *secondary* variables, respectively. For example, in the Navier-Stokes equations the primary variable is the fluid velocity and the secondary variable is the pressure of the fluid. These systems are comprised of block matrices of the form:

$$\underbrace{\begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix}}_{K} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}, \tag{1.25}$$

where $F \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{m\times n}$ with $m < n$. We consider both nonsingular and singular leading blocks, F.

We primarily consider two preconditioning methods which arise from block LDL decompositions of (1.25), or a regularized version thereof, with $-W$ in the trailing block. First, if $\text{rank}(F) = n$, then it is possible to directly invert F, and thus, an ideal preconditioner for (1.25) would be of the form:

$$M_1 = \begin{pmatrix} F & 0 \\ 0 & BF^{-1}B^T \end{pmatrix}, \tag{1.26}$$

where $BF^{-1}B^T$ is known as the dual Schur complement. It has been shown in [68, 80] that the preconditioned matrix $M_1^{-1}K$ has three distinct eigenvalues. If F is singular then it is not possible to consider preconditioners similar to (1.26). For such problems, one often considers preconditioners of the form:

$$M_2 = \begin{pmatrix} F + B^TW^{-1}B & 0 \\ 0 & W \end{pmatrix}, \tag{1.27}$$

where $F + B^TW^{-1}B$ is known as the primal Schur complement for the regularized saddle-point matrix

$$\begin{pmatrix} F & B^T \\ B & -W \end{pmatrix},$$

and W is an arbitrary symmetric positive definite matrix. There are various natural choices for W based of the underlying discrete operators; see, for example, [87]. The authors in [50] show that if $\text{rank}(F) = n - m$ then the preconditioned matrix $M_2^{-1}K$ has two distinct eigenvalues.

In general, the construction of both the primal and dual Schur complements will lead to dense matrices which are computationally impractical to form and subsequently solve for. Therefore, state-of-the-art methods aim to design effective sparse approximations to $F + B^TW^{-1}B$ or $BF^{-1}B^T$. These approximations are often based on mimicking the essential spectral properties of the Schur complement.

Navier-Stokes equations

The numerical solution of the Navier-Stokes problem is of great importance. Three-dimensional fluid applications often lead to systems in excess of 10's of millions of degrees of freedom. In Section 1.3.3 we mentioned how effective Schur complement based preconditioners can be.

For the Navier-Stokes problem, F in (1.25) is defined as discrete convection-diffusion

operator and thus rank$(F) = n$. Therefore, the aim is to approximate the dual Schur complement in (1.26).

The "SIMPLE" preconditioner in [103], uses the approximation $X = B\,\mathrm{diag}(F)^{-1}B^T$. This preconditioner is known to work well for moderate to large values of the kinematic viscosity. However, for small values of ν the preconditioned iterations degrade. Other "SIMPLE" type preconditioners use other algebraic approximations to F^{-1}.

In this thesis, we utilize the approaches from [35] where the authors introduce the pressure convection diffusion (PCD) and least-squares commutator (LSC) preconditioners. These approaches are based on minimizing a commutator operator; see Section 2.2.1 for more details.

An alternative to the Schur complement approximations presented in [35, 103] are monolithic multigrid methods. In [105] Vanka introduced a "all-at-once" multigrid approach to the Navier-Stokes problem.

Mixed Maxwell problem

Applications of computational electromagnetics, just like computational fluid dynamics, lead to large sparse systems and are widely used within industrial and geophysical applications.

For the mixed Maxwell problem, F in (1.25) is the discrete curl-curl operator and therefore rank$(F) = n - m$. Thus, in [51] that authors propose a preconditioner based on (1.27) where they introduce the approximation $F + B^T W^{-1} B \approx F + X$. Here X is a mass matrix (finite element identity operator), so that $F + X$ is a shifted curl-curl operator; see Section 2.2.2 for more details. Due to the large null-space of the discrete curl-curl operator, effective "black-box" scalable solvers which can be used in the Navier-Stokes case do not work as well. Thus, more specialized solvers are required.

There are several effective multigrid solvers designed specifically for shifted curl-curl operator. The method proposed in [59] is based on geometric multigrid, using specialized smoothers. More recent work has been done using algebraic multigrid for unstructured meshes or where a sequence of meshes is hard to construct; see [14, 43, 63, 88]. These methods use smoothers developed for the geometric multigrid case in [59]. The nodal auxiliary space preconditioner introduced in [61] has proven to be particularly effective. It reduces the solution of the shifted curl-curl problem to a nodal shifted vector Laplacian and scalar Laplacian solve. In Chapter 2, the numerical results are produced with an in-house implementation of [61] whereas for Chapters 3 and 4 we use the more efficient *Hypre* [37, 38] implementation. A full description of the implementation aspects of [61] can be found in [67, 72].

MHD model

The development of block preconditioning techniques for this model is relatively limited. A number of attempts have been made to develop scalable solution methods for the MHD model, with limited success [20–22, 66, 76, 96, 97]. We are particularly interested in the class of block preconditioning methods. These techniques are often based on effective Schur complement approximations [10, 50, 80]. Such methods are known to be robust with respect to mesh refinement and nondimensional parameters.

In recent years, the interest in the development of block preconditioning methods for the MHD model has increased; see [2, 29, 71, 85, 86, 109, 110]. The work that is the most relevant to ours is [85, 86]. The authors in those papers consider approaches based on utilizing effective preconditioners for the separate subproblems, and then incorporating the coupling terms. Our work in [110] capitalizes on a useful tensor identity introduced in [85, Equation (3.15)] and also uses block preconditioners. However, in contrast to [85] we formulate a different block matrix and use other approximations to the Schur complements. Similarly to the "all-at-once" multigrid for the Navier-Stokes equations in [105], the authors in [2] develop "all-at-once" multigrid methods for the MHD model. In [29], the aim is to use block factorizations to reduce the dimension of the model and thus develop block preconditioning techniques based on this reduced model.

So far there appear to be limited block preconditioning strategies that are fully scalable for large scale three-dimensional problems. This thesis introduces progress on this front.

1.4 Numerical software

The number of open-source numerical software packages that deal with the discretization and linear solution methods for PDE-based problems has risen dramatically in the last few decades. Theses packages are used in a variety of different settings; from the design of new and more complex solution methods for a variety of different models to users that require a "black-box" type solution method to PDEs.

Numerical software for a linear or linearized elliptic PDEs of the kind considered in this thesis may be split into the following components:

1. PDE discretization, including domain construction and mesh generation;

2. solution of the linear system.

1.4.1 PDE discretization: *FEniCS*

There are many open-source software packages that consider discretizations of PDEs – most of these are based on either finite elements or finite volumes. A few examples can be found in [3, 13, 25, 27, 54, 77] but by no means is this an exhaustive list. In this thesis, we limit our discussion to *FEniCS* [77], which is a finite element package.

The *FEniCS* project started in 2003 and the aim was create software that automates a finite element discretization and solution of differential equations. One of the main strengths of the *FEniCS* package is its easy-to-use code to create efficient large-scale numerical examples for any type of PDE-based problem. The main underlying code base for *FEniCS* is written in C++ and Python, with interfaces in both languages.

FEniCS supports a large range of different finite element function spaces, thus allowing it to be used for a variety of different applications, including electromagnetics and fluid flow problems, which are our focus of interest. We use *FEniCS* primarily for its discretization modules but note that it is possible to use its linear solver packages as well. For our specialized block-based linear solvers we use *PETSc*.

As mentioned in Section 1.2, the first step of numerical discretization is domain construction and mesh triangulation. Often domains of interest for PDEs are standard rectangles in 2D or boxes in 3D. Such domains are commonly written into the finite element discretization software. However, for certain physical applications, nonstandard domains and meshes are typically required.

For the more complex geometries and meshes, we use the *mshr* mesh generation software [65]. Using Constructive Solid Geometry (CSG) [40], *mshr* generates 2D and 3D meshes using both *CGAL* [102] and *Tetgen* [98] as the mesh generation backends. This enables the user to define several points as the domain and then *mshr* will form a quasi-uniform mesh triangularization from those points. It is possible to use the *CGAL* or *Tetgen* libraries in isolation. However, since *mshr* incorporates both and it links seamlessly into *FEniCS*, it is the natural choice of meshing software for *FEniCS* discretizations.

For complicated physical domains and meshes, local mesh refinement is often required. There are a number of different types of mesh refinements. These include graded mesh refinement, adaptive mesh refinement and cell-wise refinement; see, for example, [89]. In Program 1.1 we give an example of cell-wise mesh refinement using *mshr* and *FEniCS*. This sort of refinement is done by flagging the cells depending on their location in the mesh and refining them. For example, in Program 1.1 we flag the cells which are within a certain distance from the singular point and refine the mesh in that area.

One of the most common complex domains considered is an L-shaped domain; see

Figure 1.2. The mesh refinement introduced in Figure 1.2 is based on cell-wise mesh refinement around the singular point.

Program 1.1 *mshr*: construction of a locally refined mesh on an L-shaped domain in Figure 1.1. The cells that are within a certain distance from the radius.

```
from dolfin import Point, mesh
import mshr
import matplotlib.pylab as plt

# L-shaped domain and mesh generation
domain = mshr.Rectangle(Point(-1,-1), Point(1,1)) - mshr.Rectangle(Point(0,0),
 ↪  Point(1,1))
mesh = mshr.generate_mesh(domain, 32)

# Specifying local mesh refinement
cell_markers = CellFunction("bool", mesh)
cell_markers.set_all(False)
for cell in cells(mesh):
    p = cell.midpoint()
    if sqrt(p.x()**2+p.y()**2) < 0.4:
        cell_markers[cell] = True
mesh = refine(mesh, cell_markers)

plot(mesh, interactive=True)
plt.show()
```

Once the domain and mesh are constructed, the next step is to define the discretization used. We consider a mixed finite element approach for the MHD model (1.1). This thus requires the construction of the mixed function space and then the linear weak form (bilinear form) associated with the model problem. Program 1.2 demonstrates how *FEniCS* discretizes the complex MHD model in (1.1). The variational form we use for the MHD model is given in Section 2.1. The seamless fashion of implementing the variational form is one of the main strengths of the *FEniCS* software package, as evident from Program 1.2. The program constructs the bilinear form for the Navier-Stokes equations (1.3), the mixed Maxwell model (1.5) and the coupling terms defined in (1.1).

1.4.2 Solution of the linear system: *PETSc*

Three of the main and most widely used linear algebra software packages are *Trilinos* [56, 57], *Eigen* [52], and *PETSc* [7, 9, 30]. Theses three packages are all available in the C++ and Python programming languages. Each package incorporate additional external solvers (direct methods, multigrid methods, etc.) within their solver classes. This therefore allows users to incorporate state-of-the-art external software packages all within one framework. In this thesis, we are particularly interested in the development

Program 1.2 *FEniCS*: construction of the bilinear forms for the MHD model in (1.1)

```
(u, p, b, r) = TrialFunctions(W)
(v, q, c, s) = TestFunctions(W)

# the bilinear form for the mixed Maxwell's equations in (1.5)
Maxwell = kappa*nu_m*inner(curl(b),curl(c))*dx + inner(c,grad(r))*dx +
 ↪  inner(b,grad(s))*dx

# the bilinear form for the mixed Navier-Stokes equations in (1.4)
NavierStokes = nu*inner(grad(u), grad(v))*dx + inner((grad(u)*u_k),v)*dx +
 ↪  (1./2)*div(u_k)*inner(u,v)*dx - (1./2)*inner(u_k,n)*inner(u,v)*ds - div(v)*p*dx -
 ↪  div(u)*q*dx

# the bilinear form for the coupling terms of the MHD model in (1.1)
Coupling = kappa*inner(cross(v, b_k), curl(b))*dx - kappa*inner(cross(u, b_k),
 ↪  curl(c))*dx

# the bilinear form associated with the Newton linearization of the MHD model in
 ↪  (1.1)
Newton = inner((grad(u_k)*u), v)*dx + (1./2)*div(u)*inner(u_k, v)*dx -
 ↪  (1./2)*inner(u, n)*inner(u_k, v)*ds + kappa*inner(cross(v, b), curl(b_k))*dx -
 ↪  kappa*inner(cross(u_k, b), curl(c))*dx

# the bilinear form for the MHD model in (1.4)
MHD = Maxwell + NavierStokes + Coupling + Newton

# assembling MHD coeffi matrix
K = assemble(MHD)
```

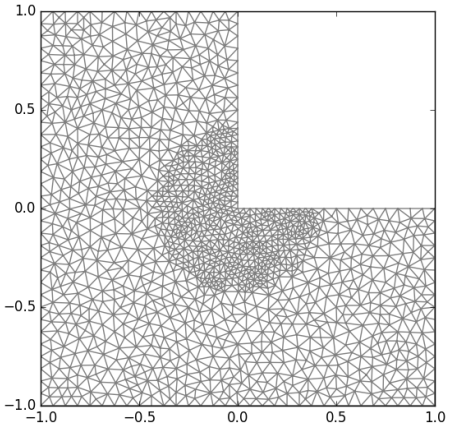


Figure 1.2: L-shaped domain generated using *mshr*. We locally refine cells that are a distance 0.4 from the origin.

of block-based preconditioning techniques. All three of these packages have block-based preconditioning classes, however within the Python interface, *PETSc* seemed the most

usable package.

The three key components which we use *PETSc* for are its library of Krylov subspace solvers, its interface with external packages, and the ability to apply block preconditioners in a seamless fashion. A few examples of external packages are:

- *Hypre* [37] - the LLNL preconditioner library.
- *MUMPS* [4, 5] - MUltifrontal Massively Parallel sparse direct Solver.
- *ParMeTiS* [64] - parallel graph partitioner.
- *PaStiX* [55] - a parallel LU and Cholesky solver package.
- SuiteSparse, including *KLU* [32], *UMFPACK* [31], and *CHOLMOD* [23] - sparse direct solvers, developed by Timothy A. Davis.
- *SuperLU* [73, 74] and SuperLU_Dist [112] - robust and efficient sequential and parallel direct sparse solves.
- *Trilinos/ML* [57] - ML: Multilevel Preconditioning Package. Sandia's main multigrid preconditioning package.

Several of these packages have been used throughout this project, e.g., *Hypre*, *MUMPS*, *PaStiX*, *ML* and *UMFPACK*. For the eventual software package that is being experimented with, we use *Hypre* and *MUMPS*. We use *Hypre* primarily as the multigrid package due to its large range of different smoother types and, particularly, its implementation of the auxiliary space preconditioner in [61]. The auxillary space preconditioner has been proven to be highly robust for shifted curl-curl problems, and thus, is an important component in the new preconditioning techniques we propose. *MUMPS* is a widely used direct solver for many problems and for exact inverses we use this software package.

The primary focus of this work is on the development of block-based linear solvers, and as such, it was necessary to use software features that support this methodology. *PETSc* has a builtin function known as `fieldsplit`, which is used to apply block-based preconditioners. A key feature of the application of our block-based preconditioning techniques is the ability to apply a nested set of linear system solves for an individual block. In this work, we use various Schur complement approximations that give rise to nested sets of matrix solves; see Section 1.3.3. It appears that this sort of nested solves is not well supported in `fieldsplit`, therefore, we create a Python class which can be used to define the inverse of block preconditioners. A generic example of the application of a block diagonal preconditioner is given in Programs 1.3 and 1.4. The

code examples demonstrate how it is possible to split the application of a block diagonal preconditioner of the form:

$$\begin{pmatrix} F & 0 \\ 0 & L \end{pmatrix}.$$

`ksp1` and `ksp2` in Program 1.3 are the action of F^{-1} and L^{-1}, respectively, and `IS` denotes the indices of the the two variables. Program 1.4 is set up so that a linear system $Kx = b$ is solved using FGMRES [92] with the application of the preconditioner done using a Python class.

Program 1.3 *PETSc*: our implementation of a block diagonal preconditioner.

```
class ApplyD(object):

    def __init__(self, ksp1, ksp2, IS):
        # ksp1 and ksp2 are the individual solvers for the (1,1) and (2,2) blocks,
 ↪   respectively
        self.ksp1 = ksp1
        self.ksp2 = ksp2
        # IS is the index set where IS[0] denotes the indices associated with the
 ↪   primary variable and IS[1] the secondary variable
        self.IS = IS

    def setUp(self, pc):
        A, P = pc.getOperators()
        self.B = A.getSubMatrix(self.IS[1], self.IS[0])

    def apply(self, pc, x, y):

        x1 = x.getSubVector(self.IS[0])
        y1 = x1.duplicate()
        x2 = x.getSubVector(self.IS[1])
        y2 = x2.duplicate()

        self.ksp1.solve(x1, y1)
        self.ksp2.solve(x2, y2)
        y.array = np.concatenate([y1.array, y2.array])
```

Using classes similar to the one in Program 1.3, it is possible to create sophisticated block preconditioners for the four-field MHD model considered in this work.

1.5 Outline and contributions

This thesis is organized into five chapters. In Chapter 2, we introduce a new block triangular preconditioner for the MHD model (1.1)–(1.2). We use a finite element approach based on Taylor-Hood elements [101] for $(\boldsymbol{u}, p)$ and the lowest order Nédélec [81] pair for $(\boldsymbol{b}, r)$ in (2.2). Using this discretization, we propose a preconditioning approach that

Program 1.4 *PETSc*: example of how using a Python class as a preconditioner

```
ksp = PETSc.KSP()
ksp.create(comm=PETSc.COMM_WORLD)
pc = ksp.getPC()

ksp.setType('fgmres')
pc.setType('python')
pc.setPythonContext(ApplyD(ksp1, ksp2, IS))

ksp.setTolerances(rtol=rtol, atol=atol, divtol=divtol, max_it=max_it)
ksp.setOperators(K)

ksp.setFromOptions()

ksp.solve(b, x)
```

combines well-known preconditioners for the two subproblems, together with a novel approach of dealing with coupling terms and approximating Schur complements. We analytically show that the proposed preconditioning technique effectively clusters eigenvalues of the preconditioned matrix. Finally, we numerically test this preconditioner and demonstrate its merits.

In Chapter 3, we derive a new formula for the inverse for the MHD matrix (2.8). The formula uses the nullity of both the coupling terms and curl-curl operator, which is the leading block of the Maxwell subproblem. Along with showing that the exact inverse formula has a number of zero blocks, we are able to efficiently approximate it using a sparse matrix, by dropping small entries of the inverse. We show several three-dimensional numerical results which illustrate the strong scalability of this approach.

In Chapter 4, we consider the use of conjugate gradient (CG) [58] for a class of indefinite matrices. The class of matrices considered are nonregularized or regularized nonsingular saddle-point problems with a highly rank deficient leading block. We show that there are two sufficient conditions for which CG can be used. The conditions rely on the null-space of the leading block of the saddle-point problem. We show that these conditions are perfectly satisfied by the Maxwell problem in (1.5). Our extensive numerical tests show that these conditions can be relaxed slightly but still enable the use of CG for the example we consider, namely the mixed Maxwell problem (1.5).

In the final chapter we offer concluding remarks and areas for future work.

The main contributions are:

1. We propose a new block preconditioner that utilizes state-of-the-art preconditioners for the Navier-Stokes and Maxwell subproblems. Using a result from [85] we form a practical version of it and test its robustness with respect to matrix dimension. The numerical results show strong scalability with respect to two-

dimensional problems and a small increase in iterations with respect to the mesh for three-dimensional problems[3].

2. We numerically show that for moderate nondimensional parameters it is possible to decouple the MHD model into its subproblems, either the Navier-Stokes and Maxwell problems or the Stokes and Maxwell problems.

3. We design a new approximate inverse-based preconditioner for the MHD matrix. To our knowledge, this is one of the only strongly scalable preconditioning approaches for large-scale three-dimensional problems with strong coupling.

4. Given a symmetric saddle-point matrix with a leading block that is maximally rank deficient which still leads to a nonsingular matrix, we show that there are sufficient conditions on the construction of a block diagonal and triangular preconditioner that enable the use of CG. This work builds upon and complements [36, 51, 72].

5. I have developed a large-scale code to solve the MHD problem (1.1)–(1.2) and its essential subproblems. This code combines several well-known open-source software packages to produce large-scale numerical examples within the Python programming language.

1.6 Notation

We follow the notation rules given in Tables 1.1 to 1.3. We will denote closely related matrices (either permutations of the original or an extra off diagonal block) with a tilde or hat.

Notation	Meaning
$\mathcal{K}$	coefficient matrix
$\mathcal{M}$	preconditioner
$\mathcal{S}$	Schur complement

Table 1.1: Notation for block systems

[3]Black-box preconditioners such as the Jacobi and Gauss-Seidel do not utilize the underlying physical properties of the PDE, and hence, the iteration increase is directly proportional to the mesh level.

Superscript	Meaning	Used for
NS	Navier-Stokes	$\mathcal{K}$, $\mathcal{M}$
S	Stokes	$\mathcal{K}$, $\mathcal{M}$
MX	Maxwell	$\mathcal{K}$, $\mathcal{M}$
MHD	Magnetohydrodynamics	$\mathcal{K}$, $\mathcal{M}$
C	Coupling for MHD	$\mathcal{K}$
N	(1,1) block rank deficient	$\mathcal{K}$

Table 1.2: Notation for superscripts

Subscript	Meaning	Used for
I	ideal	$\mathcal{M}$
P	practical	$\mathcal{M}$
S	Schur complement based	$\mathcal{M}$
A	Approximate inverse	$\mathcal{M}$
B	2-by-2 Block triangular (lower/upper)	$\mathcal{M}$

Table 1.3: Notation for subscripts

Chapter 2

Preconditioners for Mixed Finite Element Discretizations of Incompressible MHD Equations

This chapter consists of our new preconditioning results, which were published in the SIAM Journal of Scientific Computing [110]. The structure of the chapter is as follows. In Section 2.1, we introduce the finite element discretization, consider decoupling schemes for the nonlinear iterations, and discuss the resulting matrix systems. In Section 2.2, we review relevant preconditioners for the Navier-Stokes and Maxwell subproblems. Our new preconditioning approach for the fully coupled MHD discretization is introduced and analyzed in Section 2.3. We also give details of a scalable implementation of the proposed preconditioners in this section. In Section 2.4 we show a series of numerical results in two and three dimensions. The two-dimensional results show good scalability with respect to the mesh. However, for three-dimensional problems, we observe a small increase in iterations with respect to the mesh.

2.1 Discretization

In this section we specify the mixed finite element discretization for the incompressible MHD model (1.1)–(1.2), the nonlinear Picard iteration, decoupling approaches for the nonlinear problem, and the resulting linear systems arising in each iteration step.

2.1.1 Mixed finite element approximation

Our mixed finite element approximation is based on the variational formulation for (1.1)–(1.2) introduced and analyzed in [95]. It consists in finding a weak solution $(\boldsymbol{u}, p, \boldsymbol{b}, r)$

in the standard Sobolev spaces

$$\begin{aligned}
\boldsymbol{u} \in \boldsymbol{V} &= \big\{\boldsymbol{v} \in H^1(\Omega)^d \,:\, \boldsymbol{v} = \boldsymbol{0} \text{ on } \partial\Omega\big\},\\
p \in Q &= \{\, q \in L^2(\Omega) \,:\, (q,1)_\Omega = 0\},\\
\boldsymbol{b} \in \boldsymbol{C} &= \big\{\boldsymbol{c} \in L^2(\Omega)^d \,:\, \nabla \times \boldsymbol{c} \in L^2(\Omega)^{\bar{d}},\ \boldsymbol{n} \times \boldsymbol{c} = \boldsymbol{0} \text{ on } \partial\Omega\big\},\\
r \in S &= \{r \in H^1(\Omega) \,:\, r = 0 \text{ on } \partial\Omega\}.
\end{aligned} \tag{2.1}$$

Here and in the following, we write $(\cdot,\cdot)_\Omega$ for all L^2-inner products, and use $\bar{d} = 2d - 3$ to define the curls simultaneously for two- and three-dimensional vector fields [49].

Now let the domain Ω be divided into regular meshes $\mathcal{T}_h = \{K\}$ consisting of triangles ($d = 2$) or tetrahedra ($d = 3$) with mesh size h. We consider Taylor-Hood elements for $(\boldsymbol{u}, p)$ and the lowest order Nédélec pair for $(\boldsymbol{b}, r)$. The corresponding finite element spaces are:

$$\begin{aligned}
\boldsymbol{V}_h &= \{\, \boldsymbol{u} \in \boldsymbol{V} \,:\, \boldsymbol{u}|_K \in \mathcal{P}_2(K)^d,\, K \in \mathcal{T}_h \,\},\\
Q_h &= \{\, p \in Q \cap H^1(\Omega) \,:\, p|_K \in \mathcal{P}_1(K),\, K \in \mathcal{T}_h \,\},\\
\boldsymbol{C}_h &= \{\, \boldsymbol{b} \in \boldsymbol{C} \,:\, \boldsymbol{b}|_K \in \boldsymbol{R}_1(K),\, K \in \mathcal{T}_h \,\},\\
S_h &= \{\, r \in S \,:\, r|_K \in \mathcal{P}_1(K),\, K \in \mathcal{T}_h \,\}.
\end{aligned} \tag{2.2}$$

Here $\mathcal{P}_k(K)$ denotes the space of polynomials of total degree at most k on K and $\boldsymbol{R}_1(K) = \{\boldsymbol{a} + \boldsymbol{b} \times \boldsymbol{x} : \boldsymbol{a} \in \mathbb{R}^{\bar{d}},\, \boldsymbol{b} \in \mathbb{R}^d\}$ denotes the linear edge element space on K in terms of the position vector $\boldsymbol{x}$ on K.

With the finite element spaces in place, our mixed finite element approximation to problem (1.1)–(1.2) reads: find $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$ such that

$$\begin{aligned}
A(\boldsymbol{u}_h, \boldsymbol{v}) + O(\boldsymbol{u}_h; \boldsymbol{u}_h, \boldsymbol{v}) + C(\boldsymbol{b}_h; \boldsymbol{v}, \boldsymbol{b}_h) + B(\boldsymbol{v}, p_h) &= (\boldsymbol{f}, \boldsymbol{v})_\Omega,\\
B(\boldsymbol{u}_h, q) &= 0,\\
M(\boldsymbol{b}_h, \boldsymbol{c}) - C(\boldsymbol{b}_h; \boldsymbol{u}_h, \boldsymbol{c}) + D(\boldsymbol{c}, r_h) &= (\boldsymbol{g}, \boldsymbol{c})_\Omega,\\
D(\boldsymbol{b}_h, s) &= 0,
\end{aligned} \tag{2.3}$$

for all $(\boldsymbol{v}, q, \boldsymbol{c}, s) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$. Here, we denote by $(\cdot,\cdot)_\Omega$ the inner product in $L^2(\Omega)^d$. The bilinear forms A, B, M and D are given by

$$\begin{aligned}
A(\boldsymbol{u}, \boldsymbol{v}) &= \nu(\nabla \boldsymbol{u}, \nabla \boldsymbol{v})_\Omega, & B(\boldsymbol{u}, q) &= -(\nabla \cdot \boldsymbol{u},\, q)_\Omega,\\
M(\boldsymbol{b}, \boldsymbol{c}) &= \kappa\nu_m(\nabla \times \boldsymbol{b}, \nabla \times \boldsymbol{c})_\Omega, & D(\boldsymbol{b}, s) &= (\boldsymbol{b}, \nabla s)_\Omega.
\end{aligned}$$

Moreover, the trilinear forms O and C are defined as

$$O(\boldsymbol{w};\boldsymbol{u},\boldsymbol{v}) = \big((\boldsymbol{w}\cdot\nabla)\boldsymbol{u},\boldsymbol{v}\big)_\Omega + \frac{1}{2}\big((\nabla\cdot\boldsymbol{w})\boldsymbol{u},\boldsymbol{v}\big)_\Omega,$$
$$C(\boldsymbol{d};\boldsymbol{v},\boldsymbol{b}) = \kappa\,\big(\boldsymbol{v}\times\boldsymbol{d},\nabla\times\boldsymbol{b}\big)_\Omega.$$

The forms A, B, and O are associated with the variational formulation of the incompressible Navier-Stokes sub-problem, M and D with that of the Maxwell sub-problem in mixed form, and C is the coupling form which combines the two sub-problems into the full MHD system. We note that the convection form O appears in a standard skew-symmetric fashion. As a consequence, the discretization (2.3) is energy-stable without violating consistency.

The discrete problem (2.3) falls into the class of conforming mixed discretization studied in [95]. Hence, it is stable and has a unique solution for small data (i.e., for sufficiently large ν, ν_m, κ and forcing terms $\boldsymbol{f}$ and $\boldsymbol{g}$ with sufficiently small L^2-norms). Moreover, we have optimal-order error estimates in natural norms, both for smooth and non-smooth solutions. In particular, the strongest singularities of the curl-curl operator in non-convex domains are correctly captured and resolved.

Remark 1. *The pairs $\boldsymbol{V}_h \times Q_h$ and $\boldsymbol{C}_h \times S_h$ form standard and optimally convergent mixed discretizations for the fluid and magnetic equations in isolation. However, the approximation properties of these pairs are not properly matched for the fully coupled system* (2.3)*. Specifically, the optimal order $\mathcal{O}(h^2)$ of the Taylor-Hood spaces $\boldsymbol{V}_h \times Q_h$ (for the H^1-norm velocity errors and the L^2-norm pressure errors) are potentially reduced due to the coupling with the lower-order magnetic spaces $\boldsymbol{C}_h \times S_h$ in* (2.2)*. Nonetheless, we have chosen to work with the spaces in* (2.2) *due to computational considerations and availability of fast solvers. In particular, we avoid the need for stabilized or enriched fluid elements, and are able to use the well established auxiliary space preconditioner [61] for the lowest-order Nédélec pair.*

2.1.2 Picard iteration

A common choice for dealing with the nonlinearity within the incompressible Navier-Stokes equations in isolation is to perform Picard or Oseen iterations [35]. We adapt this approach for the fully coupled MHD system, and linearize around the current velocity and magnetic fields. Hence, given a current iterate $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$, we solve for updates

$(\delta \boldsymbol{u}_h, \delta p_h, \delta \boldsymbol{b}_h, \delta r_h)$ and introduce the next iterate by setting

$$\begin{aligned} \boldsymbol{u}_h &\to \boldsymbol{u}_h + \delta \boldsymbol{u}_h, \quad p_h \to p_h + \delta p_h, \\ \boldsymbol{b}_h &\to \boldsymbol{b}_h + \delta \boldsymbol{b}_h, \quad r_h \to r_h + \delta r_h. \end{aligned}$$

The updates $(\delta \boldsymbol{u}_h, \delta p_h, \delta \boldsymbol{b}_h, \delta r_h) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$ are found by solving the Picard system

$$\begin{aligned} A(\delta \boldsymbol{u}_h, \boldsymbol{v}) + O(\boldsymbol{u}_h; \delta \boldsymbol{u}_h, \boldsymbol{v}) + C(\boldsymbol{b}_h; \boldsymbol{v}, \delta \boldsymbol{b}_h) + B(\boldsymbol{v}, \delta p_h) &= R_u(\boldsymbol{u}_h, \boldsymbol{b}_h, p_h; \boldsymbol{v}), \\ B(\delta \boldsymbol{u}_h, q) &= R_p(\boldsymbol{u}_h; q), \\ M(\delta \boldsymbol{b}_h, \boldsymbol{c}) + D(\boldsymbol{c}, \delta r_h) - C(\boldsymbol{b}_h; \delta \boldsymbol{u}_h, \boldsymbol{c}) &= R_b(\boldsymbol{u}_h, \boldsymbol{b}_h, r_h; \boldsymbol{c}), \\ D(\delta \boldsymbol{b}_h, s) &= R_r(\boldsymbol{b}_h; s), \end{aligned} \tag{2.4}$$

for all $(\boldsymbol{v}, q, \boldsymbol{c}, s) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$. Note that this system is linearized around $(\boldsymbol{u}_h, \boldsymbol{b}_h)$. The right-hand side linear forms correspond to the residual at the current iteration $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$ and are defined by

$$\begin{aligned} R_u(\boldsymbol{u}_h, \boldsymbol{b}_h, p_h; \boldsymbol{v}) =& (\boldsymbol{f}, \boldsymbol{v})_\Omega - A(\boldsymbol{u}_h, \boldsymbol{v}) - O(\boldsymbol{u}_h; \boldsymbol{u}_h, \boldsymbol{v}) \\ & - C(\boldsymbol{b}_h; \boldsymbol{v}, \boldsymbol{b}_h) - B(\boldsymbol{v}, p_h), \\ R_p(\boldsymbol{u}_h; q) =& - B(\boldsymbol{u}_h, q), \\ R_b(\boldsymbol{u}_h, \boldsymbol{b}_h, r_h; \boldsymbol{c}) =& (\boldsymbol{g}, c)_\Omega - M(\boldsymbol{b}_h, \boldsymbol{c}) + C(\boldsymbol{b}_h; \boldsymbol{u}_h, \boldsymbol{c}) - D(\boldsymbol{c}, r_h), \\ R_r(\boldsymbol{b}_h; s) =& - D(\boldsymbol{b}_h, s), \end{aligned} \tag{2.5}$$

for all $(\boldsymbol{v}, q, \boldsymbol{c}, s) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$. For small data, the iteration (2.4) converges for any initial guess [95].

2.1.3 Decoupling

Let us consider two important cases where simplifications to the Picard iteration (2.4) can be used. We introduce the following variants, referred to as *magnetic decoupling* and *complete decoupling*.

As mentioned in the Introduction, [6] discusses the notion of strong coupling, according to the value of κ: cases where $\kappa < 100$ are considered to have weak coupling. Otherwise, the problem is treated as one with strong coupling. We have found this to be useful from a computational point of view, too. For $\kappa < 100$ we may converge to a solution by taking the coupling terms explicitly, i.e., we omit them in the LHS of (2.4). Therefore, for a given solution $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$, neglecting the coupling terms in (2.4),

results in solving for the updates $(\delta \boldsymbol{u}_h, \delta p_h, \delta \boldsymbol{b}_h, \delta r_h) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$ such that

$$\begin{aligned} A(\delta \boldsymbol{u}_h, \boldsymbol{v}) + O(\boldsymbol{u}; \delta \boldsymbol{u}_h, \boldsymbol{v}) + B(\boldsymbol{v}, \delta p_h) &= R_u(\boldsymbol{u}_h, \boldsymbol{b}_h, p_h; \boldsymbol{v}) \\ B(\delta \boldsymbol{u}_h, q) &= R_p(\boldsymbol{u}_h; q), \\ M(\delta \boldsymbol{b}_h, \boldsymbol{c}) + D(\boldsymbol{c}, \delta r_h) &= R_b(\boldsymbol{u}_h, \boldsymbol{b}_h, r_h; \boldsymbol{c}), \\ D(\delta \boldsymbol{b}_h, s) &= R_r(\boldsymbol{b}_h; s), \end{aligned} \tag{2.6}$$

where R_u, R_p, R_b and R_r are defined in (2.5). We call this *magnetic decoupling* (MD).

When we have both weak coupling and small convection terms in the system (2.4), the simplest strategy is to take all the nonlinear terms explicitly. This is the simplest technique, as it removes all nonlinear terms. For a given solution $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$, removing the coupling and convection terms in (2.4) results in solving for the updates $(\delta \boldsymbol{u}_h, \delta p_h, \delta \boldsymbol{b}_h, \delta r_h) \in \boldsymbol{V}_h \times Q_h \times \boldsymbol{C}_h \times S_h$ such that

$$\begin{aligned} A_h(\delta \boldsymbol{u}_h, \boldsymbol{v}) + B(\boldsymbol{v}, \delta p_h) &= R_u(\boldsymbol{u}_h, \boldsymbol{b}_h . p_h; \boldsymbol{v}) \\ B(\delta \boldsymbol{u}_h, q) &= R_p(\boldsymbol{u}_h; q), \\ M(\delta \boldsymbol{b}_h, \boldsymbol{c}) + D(\boldsymbol{c}, \delta r_h) &= R_b(\boldsymbol{u}_h, \boldsymbol{b}_h, r_h; \boldsymbol{c}), \\ D(\delta \boldsymbol{b}_h, s) &= R_r(\boldsymbol{b}_h; s), \end{aligned} \tag{2.7}$$

where again R_u, R_p, R_b and R_r are given in (2.5). We call this *complete decoupling* (CD).

2.1.4 The linear systems

For the matrix representation of (2.4)–(2.5), we introduce the basis function for the finite element spaces in (2.2):

$$\begin{aligned} \boldsymbol{V}_h &= \operatorname{span}\langle \boldsymbol{\psi}_j \rangle_{j=1}^{n_u}, \qquad & Q_h &= \operatorname{span}\langle \alpha_i \rangle_{i=1}^{m_u}, \\ \boldsymbol{C}_h &= \operatorname{span}\langle \boldsymbol{\phi}_j \rangle_{j=1}^{n_b}, \qquad & S_h &= \operatorname{span}\langle \beta_i \rangle_{i=1}^{m_b}. \end{aligned}$$

The aim is to find the coefficient vectors $u = (u_1, \ldots, u_{n_u}) \in \mathbb{R}^{n_u}$, $p = (p_1, \ldots, p_{m_u}) \in \mathbb{R}^{m_u}$, $b = (b_1, \ldots, b_{n_b}) \in \mathbb{R}^{n_b}$, and $r = (r_1, \ldots, r_{m_b}) \in \mathbb{R}^{m_b}$ of the finite element functions $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$ in terms of the chosen bases. To this end, we define the following stiffness

matrices and load vectors:

$$\begin{aligned}
A_{i,j} &= A(\boldsymbol{\psi}_j, \boldsymbol{\psi}_i), && 1 \le i, j \le n_u, \\
B_{i,j} &= B(\boldsymbol{\psi}_j, \alpha_i), && 1 \le i \le m_u,\ 1 \le j \le n_u, \\
M_{i,j} &= M(\boldsymbol{\phi}_j, \boldsymbol{\phi}_i), && 1 \le i, j \le n_b, \\
D_{i,j} &= D(\boldsymbol{\phi}_j, \beta_i), && 1 \le i \le m_b,\ 1 \le j \le n_b, \\
f_i &= (\boldsymbol{f}, \boldsymbol{\psi}_i)_\Omega, && 1 \le i \le n_u, \\
g_i &= (\boldsymbol{g}, \boldsymbol{\phi}_i)_\Omega, && 1 \le i \le n_b.
\end{aligned}$$

We define the stiffness matrices for the two nonlinear forms, O and C, with respect to the current finite element iterates $\boldsymbol{u}_h \in \boldsymbol{V}_h$ and $\boldsymbol{b}_h \in \boldsymbol{C}_h$ and their associated coefficient vectors u and b as

$$\begin{aligned}
O(u)_{i,j} &= O(\boldsymbol{u}_h; \boldsymbol{\psi}_j, \boldsymbol{\psi}_i), && 1 \le i, j \le n_u, \\
C(b)_{i,j} &= C(\boldsymbol{b}_h; \boldsymbol{\psi}_j, \boldsymbol{\phi}_i), && 1 \le i \le n_b,\ 1 \le j \le n_u.
\end{aligned}$$

We denote by (u, p, b, r) and $(\delta u, \delta p, \delta b, \delta r)$ the coefficient vectors associated with $(\boldsymbol{u}_h, p_h, \boldsymbol{b}_h, r_h)$ and $(\delta \boldsymbol{u}_h, \delta p_h, \delta \boldsymbol{b}_h, \delta r_h)$, respectively. Then it can be readily seen that the Picard iteration (2.4) amounts to solving the matrix system

$$\begin{pmatrix} A + O(u) & B^T & C(b)^T & 0 \\ B & 0 & 0 & 0 \\ -C(b) & 0 & M & D^T \\ 0 & 0 & D & 0 \end{pmatrix} \begin{pmatrix} \delta u \\ \delta p \\ \delta b \\ \delta r \end{pmatrix} = \begin{pmatrix} r_u \\ r_p \\ r_b \\ r_r \end{pmatrix}, \tag{2.8}$$

with

$$\begin{aligned}
r_u &= f - Au - O(u)u - C(b)^T b - B^T p, \\
r_p &= -Bu, \\
r_b &= g - Mu + C(b)b - D^T r, \\
r_r &= -Db.
\end{aligned} \tag{2.9}$$

At each nonlinear iteration, the right hand side vectors and matrices $O(u)$ and $C(b)$ in (2.8), (2.9) must be assembled with the solution coefficient vectors (u, p, b, r) of the current iterate. Here, the matrix A is symmetric positive definite, $O(u)$ is non-symmetric and $-C(b)$, $C(b)^T$ appear in a skew symmetric fashion. We also note that M is symmetric positive semidefinite with nullity m_b corresponding to the dimension of the scalar space of the discrete gradients, see [78]. In the sequel, we shall often omit

the dependence of $O(u)$ and $C(b)$ on u and b, respectively, and write O and C.

The linear system associated with the *magnetic decoupling* scheme in (2.6) then is:

$$\left(\begin{array}{cc|cc} A+O & B^T & 0 & 0 \\ B & 0 & 0 & 0 \\ \hline 0 & 0 & M & D^T \\ 0 & 0 & D & 0 \end{array}\right) \begin{pmatrix} \delta u \\ \delta b \\ \delta p \\ \delta r \end{pmatrix} = \begin{pmatrix} r_u \\ r_b \\ r_p \\ r_r \end{pmatrix}, \tag{2.10}$$

with the right-hand side quantities as defined in (2.9). While still non-symmetric, the system decouples into a Navier-Stokes block and a Maxwell block. Thus, allowing the problems to be solved with well known specifically designed preconditioners and possibly in parallel.

The linear system connected with the *complete decoupling* scheme in (2.7) is:

$$\left(\begin{array}{cc|cc} A & B^T & 0 & 0 \\ B & 0 & 0 & 0 \\ \hline 0 & 0 & M & D^T \\ 0 & 0 & D & 0 \end{array}\right) \begin{pmatrix} \delta u \\ \delta b \\ \delta p \\ \delta r \end{pmatrix} = \begin{pmatrix} r_u \\ r_b \\ r_p \\ r_r \end{pmatrix}, \tag{2.11}$$

again with the right-hand side quantities as defined in (2.9). The system is now symmetric and decouples into a linear Stokes problem and a Maxwell problem. Then, we may apply MINRES to both of the sub-problems.

2.2 Review of preconditioning techniques for the sub-problems

The linear systems in Section 2.1.3 are associated with a few important sub-problems, as discussed. These are the Navier-Stokes, Stokes, and Maxwell problems. In this section we review preconditioners for each of these sub-problems.

2.2.1 Fluid flow preconditioner

For the *magnetic decoupling* scheme (2.10), the governing equations for the fluid flow are the Navier-Stokes equations. Their associated discretized and linearized operator is given by

$$\mathcal{K}^{\mathrm{NS}} = \begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix}, \tag{2.12}$$

with $F = A + O$. One of the principal preconditioning approaches in the literature is based on approximations to the Schur complement. Using [35, 80], we look at preconditioners of the form

$$\mathcal{M}_{\mathrm{I}}^{\mathrm{NS}} = \begin{pmatrix} F & B^T \\ 0 & -S \end{pmatrix}, \tag{2.13}$$

where $S = BF^{-1}B^T$. In practice, the leading block, F, and the Schur complement, S, are approximated by linear operators that are easier to invert. Two common choices for an approximation to the Schur complement, S, are the least squares commutator (LSC) and pressure-convection diffusion (PCD). The approximations are based on finding an operator $\mathcal{F}_p$ which minimizes the commutator:

$$\mathcal{E} = \mathcal{B}\mathcal{F} - \mathcal{F}_p\mathcal{B}^T \quad \Longrightarrow \quad (\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T)^{-1} \approx (\mathcal{B}\mathcal{B}^T)^{-1}\mathcal{F}_p, \tag{2.14}$$

where $\mathcal{F}$ and $\mathcal{B}$ are the continuous convection-diffusion and divergence operators, respectively. Loosely speaking, one can split up LSC and PCD in the following two ways:

PCD: Works with the continuous operator (2.14) and tries to find $\mathcal{F}_p$ such that $\mathcal{E}$ is small. Discretization is done once the continuous operator $\mathcal{F}_p$ is found.

LSC: Discretizes the left equation in (2.14) straight away and then finds the discrete analog to $\mathcal{F}_p$, which effectively minimizes the discretized commutator.

We use the PCD approximation developed in [35] which has proven to be robust with respect to viscosity, different choices of mixed finite elements and type of mesh triangulation (i.e., squares or triangles in 2D). The approximation is based on

$$S = BF^{-1}B^T \approx A_p\, F_p^{-1} Q_p, \tag{2.15}$$

where the matrix A_p is the pressure Laplacian, F_p is the pressure convection-diffusion operator and Q_p is the pressure mass matrix:

$$\begin{aligned} (A_p)_{i,j} &= (\nabla\alpha_j, \nabla\alpha_i)_\Omega && 1 \le i,j \le m_u, \\ (F_p)_{i,j} &= \nu(A_p)_{i,j} + (\boldsymbol{u}_h \cdot \nabla\alpha_j, \alpha_i)_\Omega, && 1 \le i,j \le m_u, \\ (Q_p)_{i,j} &= (\alpha_j, \alpha_i)_\Omega, && 1 \le i,j \le m_u, \end{aligned}$$

where $\boldsymbol{u}_h \in \boldsymbol{V}_h$ is the given velocity field in the current iteration step. Note that A_p and F_p are well-defined since we work with continuous pressure elements. An effective implementation of this preconditioning approach is discussed in Section 2.3.3.

For the *complete decoupling* scheme (2.11), the governing equations for the fluid flow

are the Stokes equations. Their discrete form is given by the matrix

$$\mathcal{K}^{\mathrm{S}} = \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}. \tag{2.16}$$

Here we opt to use a standard block diagonal (i.e., with B^T zero in (2.13)) and symmetric positive definite preconditioner of the form

$$\mathcal{M}_{\mathrm{P}}^{\mathrm{S}} = \begin{pmatrix} A & 0 \\ 0 & \widehat{S} \end{pmatrix}.$$

A natural choice for $\widehat{S}$ is $\widehat{S} = \frac{1}{\nu} Q_p$, where Q_p is the pressure mass matrix in (2.15). It is important here to note that the system is completely decoupled, hence, using a symmetric positive definite preconditioner allows use of MINRES and therefore short recurrences within the Krylov solver. This preconditioner allows one to obtain mesh-independent convergence rates; see [35].

2.2.2 Maxwell preconditioner

A key part of each decoupling scheme is an efficient preconditioner for the discrete Maxwell sub-problem, whose associated matrix is given by

$$\mathcal{K}^{\mathrm{MX}} = \begin{pmatrix} M & D^T \\ D & 0 \end{pmatrix}. \tag{2.17}$$

In [51], it was shown that an ideal block diagonal positive definite preconditioner is

$$\mathcal{M}_{\mathrm{I}}^{\mathrm{MX}} = \begin{pmatrix} M + D^T L^{-1} D & 0 \\ 0 & L \end{pmatrix}. \tag{2.18}$$

Here L is the scalar Laplacian on S_h defined as

$$L_{i,j} = (\nabla \beta_j, \nabla \beta_i)_\Omega, \qquad 1 \leq i, j \leq m_b. \tag{2.19}$$

Using this preconditioner yields precisely two distinct eigenvalues, 1 and -1, hence a symmetric preconditioned Krylov solver such as MINRES will converge within two iterations in the absence of roundoff errors.

Remark 2. *Given that the MHD problem is nonsymmetric, and hence we would have to use a nonsymmetric solver anyway, one may be tempted to ask whether it would make sense to incorporate the (1,2) block of the coefficient matrix* (2.17) *into the precondi-*

tioner, namely replace (2.18) *by*

$$\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MX}} = \begin{pmatrix} M + D^T L^{-1} D & D^T \\ 0 & L \end{pmatrix}.$$

Interestingly, it turns out that there is no advantage in doing so in terms of eigenvalue distribution and consequently, the convergence of the iterative solver. The preconditioned eigenvalues of $(\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MX}})^{-1}\mathcal{K}^{\mathrm{MX}}$ *are* 1 *and* $\frac{1\pm\sqrt{5}}{2}$*, that is there are three of them, whereas the block diagonal preconditioner* (2.18) *gives rise to two eigenvalues. Therefore, the additional computational cost entailed in a matrix-vector product with* D^T *does not translate into a benefit in terms of iteration counts.*

Inverting the (1,1) block of the preconditioner,

$$M_L = M + D^T L^{-1} D, \tag{2.20}$$

is typically computationally prohibitive. Let X be the scalar mass matrix on $\boldsymbol{C}_h$, defined as

$$X_{i,j} = (\boldsymbol{\phi}_j, \boldsymbol{\phi}_i)_\Omega, \qquad 1 \le i, j \le n_b. \tag{2.21}$$

Then, it has been shown in [51, Theorem 3.3] that M_L and $M + X$ are spectrally equivalent. Hence, we may use the preconditioner

$$\mathcal{M}_{\mathrm{P}}^{\mathrm{MX}} = \begin{pmatrix} M_X & 0 \\ 0 & L \end{pmatrix}, \qquad \text{where} \qquad M_X = M + X. \tag{2.22}$$

The preconditioner $\mathcal{M}_{\mathrm{P}}^{\mathrm{MX}}$ still has rather attractive spectral properties; in particular, the preconditioned operator $(\mathcal{M}_{\mathrm{P}}^{\mathrm{MX}})^{-1}\mathcal{K}^{\mathrm{MX}}$ has the two eigenvalues 1 and -1 with algebraic multiplicity m_b each, and the rest of the eigenvalues are bounded by a constant independent of the mesh size; cf. [51]. We thus will use $\mathcal{M}_{\mathrm{P}}^{\mathrm{MX}}$ as a preconditioner for the Maxwell sub-problem. As discussed in Section 2.3.3, we shall use certain approximations to M_X and L to achieve maximal scalability with respect to computing time and problem size.

2.3 Preconditioners for the MHD system

In this section, we propose a preconditioning approach for the discrete MHD system (2.8), which is based on keeping the coupling matrix C in the preconditioner, and on applying the preconditioners discussed in Sections 2.2.1 and 2.2.2 to the Navier-Stokes and Maxwell sub-problems. Leaving the coupling terms in and applying these

known preconditioners to each of the sub-problems in the MHD system yields the ideal preconditioner:

$$\mathcal{M}_{\mathrm{I}}^{\mathrm{MHD}} = \left(\begin{array}{cc|cc} F & B^T & C^T & 0 \\ 0 & -S & 0 & 0 \\ \hline -C & 0 & M_L & 0 \\ 0 & 0 & 0 & L \end{array}\right),$$

where M_L and S are defined as the Schur complement approximations in (2.20) and (2.15), respectively.

2.3.1 Reordering

We note that by reordering the blocks in $\mathcal{M}_{\mathrm{I}}^{\mathrm{MHD}}$, such that the solution vector is of the following form (u, b, p, r), we obtain a 2×2 block triangular preconditioner of the form:

$$\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MHD}} = \left(\begin{array}{cc|cc} F & C^T & B^T & 0 \\ -C & M_L & 0 & 0 \\ \hline 0 & 0 & -S & 0 \\ 0 & 0 & 0 & L \end{array}\right). \tag{2.23}$$

If we are to use this preconditioner, then we must reorder the linear system accordingly:

$$\left(\begin{array}{cc|cc} F & C^T & B^T & 0 \\ -C & M & 0 & D^T \\ \hline B & 0 & 0 & 0 \\ 0 & D & 0 & 0 \end{array}\right) \begin{pmatrix} \delta u \\ \delta b \\ \delta p \\ \delta r \end{pmatrix} = \begin{pmatrix} r_u \\ r_b \\ r_p \\ r_r \end{pmatrix}, \tag{2.24}$$

with the right-hand side quantities as defined in (2.9). Let us denote by $\mathcal{K}^{\mathrm{MHD}}$ the coefficient matrix defined in (2.24). From this point on, we consider the reordered system.

The computational bottleneck is solving systems associated with the matrix

$$\begin{pmatrix} F & C^T \\ -C & M_L \end{pmatrix}, \tag{2.25}$$

in the $(1, 1)$ block matrix of (2.23). To invert the matrix in (2.25), we apply a block triangular preconditioner based on the Schur complement given by

$$\begin{pmatrix} F + M_C & C^T \\ 0 & M_L \end{pmatrix}, \qquad \text{where} \qquad M_C = C^T M_L^{-1} C.$$

Using the above approximation in $\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MHD}}$ yields:

$$\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}} = \begin{pmatrix} F + M_C & C^T & B^T & 0 \\ 0 & M_L & 0 & 0 \\ 0 & 0 & -S & 0 \\ 0 & 0 & 0 & L \end{pmatrix}. \tag{2.26}$$

To analyze the spectral properties of $\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}}$, we refer to vectors $b \in \mathrm{null}(M)$ as discrete gradients. With the discrete Helmholtz decomposition, it follows that for each $b \in \mathrm{null}(M)$ there is a unique $r \in \mathbb{R}^{m_b}$ such that $b = Gr$ for a discrete gradient matrix $G \in \mathbb{R}^{n_b \times m_b}$; cf. [51, Section 2]. Hence, $\dim(\mathrm{null}(M)) = m_b$. The following result holds true.

Theorem 1. *The matrix $(\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$ has an eigenvalue $\lambda = 1$ with algebraic multiplicity of at least $n_b + n_c$ where n_c is the dimension of the nullspace of $C = C(b)$ and an eigenvalue $\lambda = -1$ with algebraic multiplicity of at least m_b. The dimension of the nullspace of C is $n_c = n_u - n_b + m_b$. The corresponding eigenvalue-eigenvector $(\lambda, \boldsymbol{x})$ pairs are:*

$$\lambda = 1, \quad \boldsymbol{x}^T = (u_c^T, b^T, (-S^{-1}Bu_c)^T, (L^{-1}Db)^T),$$

with $u_c \in \mathrm{null}(C)$ and $b \in \mathbb{R}^{n_b}$ arbitrary, and

$$\lambda = -1, \quad \boldsymbol{x}^T = (0, b_g^T, 0, (-L^{-1}Db_g)^T),$$

with $b_g = Gr$ a discrete gradient for $r \in \mathbb{R}^{m_b}$.

Proof. The corresponding eigenvalue problem is

$$\begin{pmatrix} F & C^T & B^T & 0 \\ -C & M & 0 & D^T \\ B & 0 & 0 & 0 \\ 0 & D & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ b \\ p \\ r \end{pmatrix} = \lambda \begin{pmatrix} F + M_C & C^T & B^T & 0 \\ 0 & M_L & 0 & 0 \\ 0 & 0 & -S & 0 \\ 0 & 0 & 0 & L \end{pmatrix} \begin{pmatrix} u \\ b \\ p \\ r \end{pmatrix}.$$

The four block rows of the generalized eigenvalue problem can be written as

$$\begin{aligned} (1-\lambda)(Fu + B^T p + C^T b) - \lambda C^T (M + D^T L^{-1} D)^{-1} Cu &= 0, && (2.27) \\ -Cu + (1-\lambda)Mb - \lambda D^T L^{-1} Db + D^T r &= 0, && (2.28) \\ Bu &= -\lambda S\, p, && (2.29) \\ Db &= \lambda L r. && (2.30) \end{aligned}$$

If $\lambda = 1$, (2.27) is satisfied if

$$C^T(M + D^T L^{-1} D)^{-1} C u = 0.$$

This only happens when $u \in \text{Null}(C)$. Using u_c to denote a nullspace vector of C then (2.29) simplifies to:

$$p = -S^{-1} B u_c.$$

Equation (2.30) leads to $r = L^{-1} D b$. If this holds, (2.28) is readily satisfied. Therefore, $(u_c^T, b^T, (-S^{-1} B u_c)^T, (L^{-1} D b)^T)$ is an eigenvector corresponding to $\lambda = 1$. There exist n_c linearly independent such vectors u and n_b linearly independent such vectors b. Hence, it follows that $\lambda = 1$ is an eigenvalue with algebraic multiplicity of at least $n_u + m_b$.

If $\lambda = -1$, (2.30) leads to $r = -L^{-1} D b$. Substituting it into (2.28), we obtain $Cu = Mb$. If $b = b_g$ is a discrete gradient then $Mb = 0$ and $C^T b = 0$. If we take $u = 0$, then $Cu = 0$ and the requirement $Cu = Mb$ is satisfied. If $u = 0$ and $b = b_G$ is a discrete gradient, equation (2.27) becomes $B^T p = 0$. Since B has full row rank, this implies $p = 0$. Therefore, if $b = b_g$ is a discrete gradient, then $(0, b_g^T, 0, (-L^{-1} D b_g)^T)$ is an eigenvector corresponding to $\lambda = -1$. There are m_b discrete gradients. Therefore $\lambda = -1$ is an eigenvalue with algebraic multiplicity at least m_b. □

Remark 3. *In the spirit of Remark 2, looking at* (2.26) *one may ask whether incorporating* D^T *into the preconditioner, such that*

$$\widetilde{\mathcal{M}}_{\text{S}}^{\text{MHD}} = \begin{pmatrix} F + M_C & C^T & B^T & 0 \\ 0 & M_L & 0 & D^T \\ 0 & 0 & -S & 0 \\ 0 & 0 & 0 & L \end{pmatrix},$$

may generate a slightly better eigenvalue distribution for the preconditioned system. Consistently with Remark 2, it turns out that doing so does not practically generate an improvement of the eigenvalue distribution. From Table 2.1, we see that both preconditioned systems yield exactly the same number of eigenvalues. However, $(\mathcal{M}_{\text{S}}^{\text{MHD}})^{-1}\mathcal{K}^{\text{MHD}}$ *only has* 2 *distinct eigenvalues whereas* $(\widetilde{\mathcal{M}}_{\text{S}}^{\text{MHD}})^{-1}\mathcal{K}^{\text{MHD}}$ *has 3 distinct eigenvalues. Thus, the insertion of* D^T *does not theoretically decrease the number of iterations for a Krylov subspace method to converge; this has been confirmed by numerical experiments. Since incorporating* D^T *into the preconditioner slightly increases the cost of a single iteration, we opt for using* $\mathcal{M}_{\text{S}}^{\text{MHD}}$ *rather than* $\widetilde{\mathcal{M}}_{\text{S}}^{\text{MHD}}$.

	$(\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$	$(\widetilde{\mathcal{M}}_{\mathrm{S}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$
1	$n_b + n_c$	n_u
-1	m_b	0
$\frac{1+\sqrt{5}}{2}$	0	m_b
$\frac{1-\sqrt{5}}{2}$	0	m_b
Total	$n_u + 2m_b$	$n_u + 2m_b$

Table 2.1: Algebraic multiplicity of eigenvalues for preconditioned matrices associated with $\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}}$ and $\widetilde{\mathcal{M}}_{\mathrm{S}}^{\mathrm{MHD}}$. Note that $n_c = n_u - n_b + m_b$

2.3.2 From an ideal to a practical preconditioner

We now consider further simplifications of $\mathcal{M}_{\mathrm{S}}^{\mathrm{MHD}}$ in (2.26), to make the preconditioner computationally feasible. Effective sparse approximations are required for the relevant Schur complements that arise. We use the approximations in Sections 2.2.1 and 2.2.2 for S and M_L. For approximating M_C, we follow a similar approach to that taken in [85, Section 3.1]. For a given magnetic field $\boldsymbol{b}$, let $\mathcal{C}_{\boldsymbol{b}}$ be the continuous differential operator analogue of $M_C = C^T M_L^{-1} C$:

$$\begin{aligned}\mathcal{C}_{\boldsymbol{b}}\boldsymbol{u} &= \kappa\Big(\nabla\times\big((\kappa\nu_m\nabla\times\nabla\times+\nabla\Delta^{-1}\nabla\cdot)^{-1}\kappa\nabla\times(\boldsymbol{u}\times\boldsymbol{b})\big)\Big)\times\boldsymbol{b}\\ &= \kappa^2\nabla\times\big(\kappa\nu_m\nabla\times\nabla\times+\nabla\Delta^{-1}\nabla\,\cdot\big)^{-1}\nabla\times(\boldsymbol{u}\times\boldsymbol{b})\times\boldsymbol{b}.\end{aligned} \tag{2.31}$$

The discretization of (2.31) is

$$\mathcal{C}_b u = \kappa^2 G\big(\kappa\nu_m G^T G + D^T L^{-1} D\big)^{-1} G^T (u\times b)\times b, \tag{2.32}$$

where G is a discrete curl matrix and u and b are vectors of velocity and magnetic coefficients, respectively.

Discrete Operator	Order
G	$\mathcal{O}(h^{-1})$
L	$\mathcal{O}(h^{-2})$
D	$\mathcal{O}(h^{-1})$

Table 2.2: Orders of matrix entries for relevant discrete operators

In Table 2.2 we state the order of the discrete differential operators that are involved in (2.32). We observe that the discrete curl-curl matrix, $G^T G$, contains entries of magnitude $\mathcal{O}(h^{-2})$ and $D^T L^{-1} D$ is order $\mathcal{O}(1)$. Thus for small h and moderate values

of κ and ν_m the curl-curl matrix will be the dominant term. Therefore we consider:

$$\mathcal{C}_b u \approx \kappa \nu_m^{-1} G\big(G^T G\big)^{-1} G^T (u \times b) \times b.$$

Furthermore, $G\big(G^T G\big)^{-1} G^T$ is an orthogonal projector onto the range space of G^T, hence, it acts as an identity operator within that space. We therefore use the approximation

$$\mathcal{C}_b u \approx \kappa {\nu_m}^{-1} b \times (u \times b). \tag{2.33}$$

From (2.33), $\mathcal{C}_b$ can approximated by a scaled mass matrix determined by the coefficients of the magnetic field b. We thus approximate M_C by a scaled mass matrix, which we denote as Q_S, and whose elements are:

$$(Q_S)_{i,j} = \kappa \nu_m^{-1} (\boldsymbol{b}_h \times \boldsymbol{\psi}_j, \boldsymbol{b}_h \times \boldsymbol{\psi}_i)_\Omega, \qquad 1 \le i, j \le n_u. \tag{2.34}$$

Combining the sparse Schur complement approximations in (2.15), (2.22) and (2.34) in the MHD preconditioner (2.26) gives the approximate preconditioner:

$$\mathcal{M}_{\mathrm{P}}^{\mathrm{MHD}} = \begin{pmatrix} F + Q_S & C^T & B^T & 0 \\ 0 & M + X & 0 & 0 \\ 0 & 0 & -A_p F_p^{-1} Q_p & 0 \\ 0 & 0 & 0 & L \end{pmatrix}. \tag{2.35}$$

In the transition from the ideal preconditioner (2.23) to the practical preconditioner (2.35), some spectral clustering is inevitably lost. But as we show, the spectral structure of the preconditioned matrix associated with $\mathcal{M}_{\mathrm{P}}^{\mathrm{MHD}}$ is still very appealing. We illustrate this in Section 2.4.

2.3.3 Implementation

So far we have introduced the matrix systems with possible preconditioners, but have not discussed practical implementation considerations. One of our main goals is to provide a fully scalable solution method. To this end, we will consider mesh-independent solvers for the separate block matrices within the preconditioners. Table 2.3 outlines the methods we use to solve the systems associated with Q_p, A_p, $F + Q_s$, $M + X$ and L which are the block diagonal matrices in (2.35). Let us provide a few additional comments:

1. For solving systems involving Q_p, we have experimentally observed that scaling by a multiplicative scalar α smaller than 1 results in a slight decrease of iteration

Matrix	Implementation method
Q_p	diagonal of αQ_p where $\alpha = 0.75$
A_p	single AMG V-cycle
$F + Q_s$	single AMG V-cycle
$M + X$	AMG method developed in [61]
L	single AMG V-cycle

Table 2.3: Solution methods for systems associated with separate block matrices within (2.35)

counts, especially in the 3D case. In our numerical experiments we provide results that correspond to using $\alpha = 0.75$.

2. For solving systems involving $M + X$, the method developed in [61] aims to overcome issues with standard AMG methods for the discrete curl-curl operator by using an auxiliary space approach [113] for $H(\text{curl})$ finite element discretizations of elliptic problems. The construction of the auxiliary space multigrid (ASM) preconditioner relies on three additional matrices. First, the discrete Nédélec interpolation operator $P \in \mathbb{R}^{n_b \times d\, m_b}$ (where d is the spatial dimension), the discrete gradient operator $G \in \mathbb{R}^{n_b \times m_b}$ and mass matrix defined on the scalar space S_h as Q; see [67, 72].

2.4 Numerical results

This section examines the efficiency of our preconditioning approaches to the MHD model (1.1)-(1.2). The auxiliary space multigrid is used as a preconditioner within a Conjugate Gradient solver. Since this practically means that the Krylov subspace associated with the preconditioned iterations varies in every outer solve, a flexible solver for nonsymmetric matrices is required. We chose to use Flexible GMRES (FGMRES) [92]. In all experiments, unless otherwise stated, we use a 2-norm absolute tolerance of 1e-4 for the nonlinear solver and relative error of 1e-5 for both FMGRES and the auxiliary space multigrid [61].

All numerical experiments have been carried out using the finite element software *FEniCS* [77] in conjunction with the `PETSc4PY` package (Python interface for `PETSc` [7, 9]) and the multigrid package `HYPRE` [38].

We test our methods on problems with inhomogeneous Dirichlet boundary conditions in the hydrodynamic variables, even though the analysis has been carried out solely for the homogeneous Dirichlet case. Other boundary conditions may be handled

by our finite element framework, and our preconditioning approaches can be extended accordingly.

In the subsequent tables we use the following notation:

- $\text{time}_{\text{solve}}$ is the average FGMRES time to solve systems associated with $\mathcal{K}^{\text{MHD}}$;
- time_{NL} is the total nonlinear solve time (including all matrix assembles, FGMRES solves and updates to nonlinear iteration);
- it_{NL} is the number of nonlinear iterations;
- $\text{it}^{*}_{\text{av}}$ is the average number of FGMRES where $*$ is either I or D. Superscript I denotes an approximate/iterative application of the preconditioner; for example, this could be a multigrid solve. See Table 2.3 for a complete list of methods used to solve systems associated with the block diagonal matrices of the preconditioner. Superscript D denotes a direct solve for block diagonal matrices of the preconditioner.

The time and nonlinear iteration columns ($\text{time}_{\text{solve}}$, time_{NL}, and it_{NL}) are computed using an iterative application of the preconditioner ($\text{it}^{\text{I}}_{\text{av}}$). We run the program twice, once for $\text{it}^{\text{I}}_{\text{av}}$ and once for $\text{it}^{\text{D}}_{\text{av}}$.

Mesh sequences. In our tests, we consider sequences of uniformly refined simplicial grids (i.e., triangles in 2D and tetrahedra in 3D). We define ℓ to be the mesh level, such that there are 2^{ℓ} edges along each boundary. In our tables, we usually show the grid level ℓ in the first column. We also note that DoFs (in the second column), refers to the total number of degrees of freedom of the linear system.

Stopping criteria. Throughout, we enforce the following nonlinear stopping criteria for the updates:

$$\|\delta u\|_2 + \|\delta p\|_2 + \|\delta b\|_2 + \|\delta r\|_2 < \text{tol}_{\text{NL}},$$

where $\|\cdot\|_2$ is the absolute error in the 2-norm of a vector and tol_{NL} =1e-4.

Initial guess tolerance. For all numerical experiments, we formulate the initial guess by iteratively solving a Stokes problem and mixed Maxwell problem in isolation. We choose the a tight Krylov 2-norm relative tolerance as 1e-10 to ensure that the approximations of the inhomogeneous boundary conditions are sufficiently accurate; see [109, Chapter 2.5]. This is to ensure the accuracy of the initial solution on the boundaries since subsequent Picard iterations are solved for homogeneous boundary conditions and hence, any errors in the initial guess will be carried throughout Picard iteration.

2.4.1 2D smooth solution

The first example considered is a simple domain with a structured mesh. We use a unit square domain, $\Omega = [0,1]^2$. We take $\nu = \kappa = 1$, $\nu_m = 10$; then the source terms $\boldsymbol{f}$, $\boldsymbol{g}$ and inhomogeneous Dirichlet boundary conditions on $\partial\Omega$ are defined from the analytical solution:

$$\begin{aligned}
\boldsymbol{u}(x,y) &= \begin{pmatrix} xy\exp(x+y) + x\exp(x+y) \\ -xy\exp(x+y) - y\exp(x+y) \end{pmatrix}, \\
p(x,y) &= \exp(y)\sin(x), \\
\boldsymbol{b}(x,y) &= \begin{pmatrix} \exp(x+y)\cos(x) \\ \exp(x+y)\sin(x) - \exp(x+y)\cos(x) \end{pmatrix}, \\
r(x,y) &= x\sin(2\pi x)\sin(2\pi y).
\end{aligned}$$

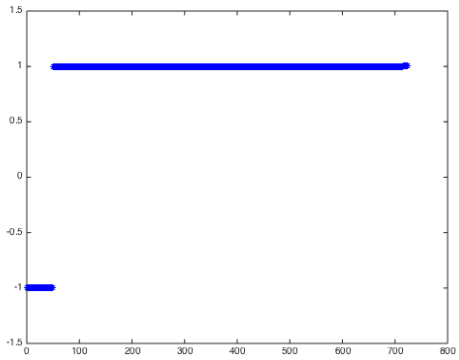
We note that $r \neq 0$ in this example. Indeed, while the right-hand-side is often divergence free in applications, which leads to an identically zero Lagrange multiplier, for testing purposes we set r as above and test convergence to the exact solution.

To illustrate the eigenvalue distribution, we compute the eigenvalues of the preconditioned matrix $\left(\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MHD}}\right)^{-1}\mathcal{K}^{\mathrm{MHD}}$ in Figure 2.1a and $(\mathcal{M}_{\mathrm{P}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$ in Figure 2.1b. We note that there are no imaginary parts of the eigenvalues for Figure 2.1a. From Figure 2.1b we see that we still observe strong clustering of eigenvalues around 1 and -1 with only a few complex conjugate pairs. Thus, the spectral structure is still rather appealing in terms of eigenvalue clustering. Therefore, the preconditioner $\mathcal{M}_{\mathrm{P}}^{\mathrm{MHD}}$ seems a viable approximation to $\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MHD}}$.

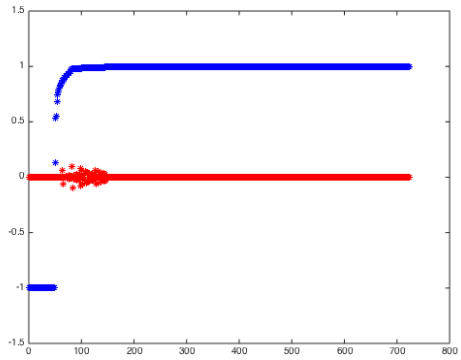
To test the scalability of our method, we considered a uniformly refined sequence of meshes. The results are presented in Table 2.4. The iterations appear to remain fairly constant with the increasing mesh level for both $\mathrm{it}_{\mathrm{av}}^{D}$ and $\mathrm{it}_{\mathrm{av}}^{\mathrm{I}}$.

2.4.2 2D smooth solution parameter tests

We next test the performance of the three nonlinear iteration schemes *Picard* (P), *magnetic decoupling* (MD) and *complete decoupling* (CD) introduced in Section 2.1. The convergence of the nonlinear iterations is likely to be affected by the parameter setup of the problem, i.e., by the values of the fluid viscosity (ν), the magnetic viscosity (ν_m) and the coupling number (κ). By varying κ and ν, we examine the robustness of the three schemes with respect to these parameters. If the nonlinear iterations do not converge then this is denoted by "-" in the tables.



(a) Eigenvalues of the preconditioned matrix $\left(\widetilde{\mathcal{M}}_{\mathrm{I}}^{\mathrm{MHD}}\right)^{-1}\mathcal{K}_{\mathrm{P}}^{\mathrm{MH}}$ associated with the ideal preconditioner.



(b) Eigenvalues of the preconditioned matrix $(\mathcal{M}_{\mathrm{P}}^{\mathrm{MHD}})^{-1}\mathcal{K}_{\mathrm{P}}^{\mathrm{MH}}$ associated with the practical preconditioner. The eigenvalues in this case are complex; the blue curves represent their real parts, and the red curves represent their imaginary parts.

Figure 2.1: Eigenvalues of preconditioned matrices for the smooth solution given in this section. The number of degrees of freedom for these matrices is 724.

ℓ	DoFs	$\mathrm{time}_{\mathrm{solve}}$	$\mathrm{time}_{\mathrm{NL}}$	$\mathrm{it}_{\mathrm{NL}}$	$\mathrm{it}_{\mathrm{av}}^{\mathrm{I}}$	$\mathrm{it}_{\mathrm{av}}^{D}$
4	3,556	0.33	2.7	7	24.4	20.1
5	13,764	1.11	9.2	7	25.9	20.4
6	54,148	4.48	37.2	7	27.1	20.9
7	214,788	20.32	166.4	7	28.4	21.4
8	855,556	94.29	762.0	7	31.3	21.8
9	3,415,044	486.53	3835.0	7	34.3	-
10	13,645,828	2231.71	17944.6	7	34.0	-

Table 2.4: 2D smooth: Number of nonlinear iterations and number of iterations to solve the MHD system with $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$.

Viscosity test

As a first test we consider varying the fluid viscosity, ν, for $\mathrm{tol}_{\mathrm{NL}}$ = 1e-4, $\kappa = 1$ and $\nu_m = 10$. The nonlinear iteration results are shown in Table 2.5 and the average linear solve times are shown in Table 2.6.

As the fluid viscosity (ν) decreases, the fluid flow equations (1.1a) and (1.1b) become more convection-dominated. Thus we see that the (CD) scheme breaks down for small ν, as in this decoupling scheme the convection term is taken explicitly. On the other hand, the Picard and (MD) schemes perform similarly. We note that for smaller ν both (P) and (MD) have trouble converging without a sufficiently refined mesh. From

		$\nu = 1$			$\nu = 0.1$			$\nu = 0.01$		
ℓ	DoFs	(P)	(MD)	(CD)	(P)	(MD)	(CD)	(P)	(MD)	(CD)
4	3,556	5	5	9	7	7	-	11	11	-
5	13,764	5	5	9	15	7	-	11	11	-
6	54,148	5	5	9	7	7	-	13	11	-
7	214,788	5	5	9	7	7	-	11	11	-
8	855,556	5	5	9	7	7	-	11	11	-

Table 2.5: Number of nonlinear iterations for various values of ν with $\kappa = 1$ and $\nu_m = 10$.

		$\nu = 1$			$\nu = 0.1$			$\nu = 0.01$		
ℓ	DoFs	(P)	(MD)	(CD)	(P)	(MD)	(CD)	(P)	(MD)	(CD)
4	3,556	0.04	0.04	0.03	0.08	0.04	-	0.04	0.04	-
5	13,764	0.18	0.17	0.14	0.97	0.18	-	0.19	0.18	-
6	54,148	0.90	0.91	0.67	0.95	0.92	-	2.05	0.93	-
7	214,788	4.48	4.48	3.49	4.68	4.57	-	5.02	4.48	-
8	855,556	25.52	22.81	16.67	25.63	23.03	-	25.01	22.61	-

Table 2.6: Average linear solver time for various values of ν with $\kappa = 1$ and $\nu_m = 10$.

Table 2.6 we see that the direct solve for the Picard (P) system and magnetic decoupling are very similar, but as expected the complete decoupling solve time is shorter.

Coupling number test

		$\kappa = 1$			$\kappa = 10$			$\kappa = 100$		
ℓ	DoFs	(P)	(MD)	(CD)	(P)	(MD)	(CD)	(P)	(MD)	(CD)
4	3,556	5	5	9	7	10	17	8	-	-
5	13,764	5	5	9	7	10	18	8	-	-
6	54,148	5	5	9	7	10	18	8	-	-
7	214,788	5	5	9	7	10	18	8	-	-
8	855,556	5	5	9	7	10	18	8	-	-

Table 2.7: Number of nonlinear iterations for various values of κ with tol_{NL} = 1e-5, $\nu = 1$ and $\nu_m = 10$.

The next parameter test examines the effects of the coupling terms in the three

		$\kappa = 1$			$\kappa = 10$			$\kappa = 100$		
ℓ	DoFs	(P)	(MD)	(CD)	(P)	(MD)	(CD)	(P)	(MD)	(CD)
4	3,556	0.06	0.06	0.04	0.06	0.06	0.03	0.06	-	-
5	13,764	0.23	0.23	0.21	0.24	0.25	0.19	0.24	-	-
6	54,148	1.12	1.08	0.80	1.01	1.05	0.79	0.98	-	-
7	214,788	4.80	4.92	3.65	5.05	4.92	3.74	5.52	-	-
8	855,556	25.21	25.93	18.03	25.49	26.03	18.49	26.75	-	-

Table 2.8: Average linear solver time for various values of κ with $\nu = 1$ and $\nu_m = 10$.

nonlinear iteration schemes. We expect the Picard scheme to outperform the other schemes for large values of κ. The results in Table 2.7 show that this is indeed the case. Both the (MD) and (CD) schemes completely break down for $\kappa \geq 100$. This is the point at which the Picard iteration (P) becomes the most viable option. Altogether, as expected, the full Picard iteration is more robust than (CD) and (MD). We see a similar trend with the timing results in Table 2.8 as we saw in the timing results for the viscosity in Table 2.6.

2.4.3 2D smooth solution on L-shaped domain

To further test the robustness of our preconditioning technique, we will consider non-convex domains. We first consider a problem with a smooth solution in the L-shaped domain $\Omega = (-1,-1)^2 \setminus ([0,0) \times (1,1])$. We prescribe the same analytical solution as in Section 2.4.1. The results in Table 2.9 show very good scalability with respect to mesh refinement for direct solves of the preconditioner. On the other hand, we report that when using iterative inner solvers with a reasonably loose tolerance, the iterations dramatically deteriorate and scalability is lost. We speculate that the AMG solver cannot handle well the non-convexity of the domain. In such cases it may be necessary to apply a rather strict convergence tolerance.

2.4.4 2D singular solution on L-shaped domain

We next consider the model singular solution from [49] on the L-shaped domain $\Omega = (-1,1)^2 \setminus ([0,1) \times (-1,0])$. That is, taking $\nu = \kappa = 1$ and $\nu_m = 10$, we set the forcing terms and the boundary conditions such that the analytic solution is given by the strongest corner singularities of the underlying Stokes and Maxwell operators subject to the boundary conditions (1.2) on the two inner sides meeting at the reentrant corner.

ℓ	DoFs	it_{NL}	$\text{it}_{\text{av}}^{D}$
5	12,880	5	24.4
6	51,678	5	26.0
7	203,712	5	27.4
8	809,705	5	29.6
9	3,219,082	-	-

Table 2.9: 2D L-shaped: Number of nonlinear iterations and number of iterations to solve the MHD system with $\text{tol}_{\text{NL}} = 1\text{e-}4$, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$. The iteration was terminated before completion for $\ell = 9$ due to the computation reaching the prescribed time limit.

In polar coordinates (ρ, ϕ) at the origin, the fluid solution $(\boldsymbol{u}, p)$ is then of the form

$$\boldsymbol{u}(\rho, \phi) = \rho^{\lambda} \boldsymbol{\Psi}_u(\phi), \qquad p(\rho, \phi) = \rho^{\lambda - 1} \Psi_p(\phi), \tag{2.36}$$

with the singular exponent $\lambda \approx 0.54448373678246$ and where $(\boldsymbol{\Psi}_u, \Psi_p)$ are smooth functions in the angle ϕ. Similarly, the magnetic pair $(\boldsymbol{b}, r)$ (with $\nabla \cdot \boldsymbol{b} = 0$ and $\nabla \times \boldsymbol{b} = 0$) is of the form

$$\boldsymbol{b}(\rho, \phi) = \rho^{-1/3} \boldsymbol{\Psi}_b(\phi), \qquad r \equiv 0, \tag{2.37}$$

Detailed expressions of the fluid components $(\boldsymbol{u}, p)$ and the magnetic field $\boldsymbol{b}$ can be found in [49, Section 5.2]. Using Nédélec elements allows us to properly capture this singular solution, whereas applying standard nodal elements for $\boldsymbol{b}$ fail to do so. As with the smooth solution on an L-shaped domain in Section 2.4.3, we only consider direct applications of the preconditioner. The results are shown in Table 2.10. We draw similar conclusions with respect to mesh independence. Overall, the iterations show good scalability.

2.4.5 2D Hartmann flow

As a final 2D numerical example, we consider the two-dimensional Hartmann flow problem, which involves a steady unidirectional flow in the channel $\Omega = (0, 10) \times (-1, 1)$ under the constant transverse magnetic field $\boldsymbol{b}_D = (0, 1)$ on $\partial\Omega$. We impose the analytical solution given in [49, Section 5.3] with Dirichlet boundary conditions for $\boldsymbol{u}$ on the

ℓ	DoFs	it_{NL}	$\text{it}_{\text{av}}^{D}$
3	740	4	13.8
4	2,724	4	14.5
5	10,436	4	15.8
6	40,836	4	17.5
7	161,540	4	18.5
8	642,564	4	20.0
9	2,563,076	4	21.8

Table 2.10: 2D singular solution on an L-shaped domain: Number of nonlinear iterations and number of iterations to solve the MHD system with tol_{NL} = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$.

entire boundary $\partial\Omega$. The MHD solution then takes the form:

$$\begin{aligned} \boldsymbol{u}(x,y) &= (u(y),0), \quad p(x,y) = -Gx + p_0(y), \\ \boldsymbol{b}(x,y) &= (b(y),1), \quad r(x,y) \equiv 0. \end{aligned} \tag{2.38}$$

The exact solution is given by (2.38) with

$$\begin{aligned} u(y) &= \frac{G}{\nu \text{Ha} \tanh(\text{Ha})} \left(1 - \frac{\cosh(y\text{Ha})}{\cosh(\text{Ha})}\right), \\ b(y) &= \frac{G}{\kappa} \left(\frac{\sinh(y\text{Ha})}{\sinh(\text{Ha})} - y\right), \\ p_0(y) &= -\frac{G^2}{2\kappa} \left(\frac{\sinh(y\text{Ha})}{\sinh(\text{Ha})} - y\right)^2, \end{aligned}$$

where $\text{Ha} = \sqrt{\frac{\kappa}{\nu\nu_m}}$ is the Hartmann number. We impose inhomogeneous Dirichlet boundary conditions from the exact solutions.

The results are reported in Table 2.11. We observe that for the two-dimensional Hartmann flow example the solver appears to accomplish an excellent degree of scalability. For the last two mesh levels the preconditioner approximation becomes better in terms of lower iterations, $\text{it}_{\text{av}}^{I}$. More exploration of the conditioning of the problem and the norm of the (preconditioned) residual is needed for fully understanding the reason for this; the large size of the problems presents a challenge in fully exploring this.

ℓ	DoFs	$\text{time}_{\text{solve}}$	time_{NL}	it_{NL}	it^I_{av}	it^D_{av}
2	1,212	0.66	1.58	2	18.0	13.5
3	4,500	1.71	3.82	2	17.5	13.0
4	17,316	5.13	11.04	2	17.5	12.5
5	67,908	21.06	44.73	2	18.5	12.5
6	268,932	97.16	204.36	2	19.0	13.0
7	1,070,340	447.66	935.03	2	19.0	12.5
8	4,270,596	921.90	1001.37	1	8.0	7.0
9	17,060,868	1459.82	1778.95	1	3.0	-

Table 2.11: 2D Hartmann: Number of nonlinear iterations and number of iterations to solve the MHD system with tol_{NL} = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 1000$

2.4.6 3D smooth solution

We next consider a 3D example with a smooth solution on $\Omega = [0,1]^3$. Let $\nu = \kappa = 1$, $\nu_m = 10$ and let the analytical solution be given by:

$$\begin{aligned}
\boldsymbol{u}(x,y,z) &= \begin{pmatrix} -xy\exp(x+y+z) + xz\exp(x+y+z)) \\ xy\exp(x+y+z) - yz\exp(x+y+z) \\ -xz\exp(x+y+z) + yz\exp(x+y+z) \end{pmatrix}, \\
p(x,y,z) &= \exp(x+y+z)\sin(y), \\
\boldsymbol{b}(x,y,z) &= \begin{pmatrix} -\exp(x+y+z)\sin(y) + \exp(x+y+z)\sin(z) \\ xy\exp(x+y+z) - yz\exp(x+y+z) \\ -\exp(x+y+z)\sin(x) + \exp(x+y+z)\sin(y) \end{pmatrix}, \\
r(x,y,z) &= \sin(2\pi x)\sin(2\pi y)\sin(2\pi z).
\end{aligned}$$

Then the source terms $\boldsymbol{f}$ and $\boldsymbol{g}$ and inhomogeneous boundary conditions are defined from the analytical solution. The corresponding results are shown in Table 2.12. We observe good scalability when we consider direct solves for the preconditioner. However, the average iterations degrade when using multigrid methods to solve the preconditioner and we do not obtain full scalability with respect to the three-dimensional results.

ℓ	DoFs	$\text{time}_{\text{solve}}$	time_{NL}	it_{NL}	$\text{it}_{\text{av}}^{\text{I}}$	$\text{it}_{\text{av}}^{D}$
1	527	0.03	0.9	4	18.8	18.0
2	3,041	0.22	3.5	3	26.7	22.3
3	20,381	1.77	26.6	3	37.0	24.7
4	148,661	22.11	237.0	3	40.7	26.0
5	1,134,437	206.43	2032.7	3	44.3	-
6	8,861,381	2274.28	19662.0	3	50.0	-

Table 2.12: 3D smooth: Number of nonlinear iterations and number of iterations to solve the MHD system with tol_{NL} = 1e-4, $\kappa = 1$, $\nu = 1$ and $\nu_m = 10$

Chapter 3

An Approximate Inverse-Based Preconditioner for Incompressible Magnetohydrodynamics

This chapter is currently unpublished and presents a new block approximate inverse preconditioner for the MHD model (1.1)–(1.2). The structure of the chapter is a follows. In Section 3.1, we introduce the Newton system for the MHD model. In Section 3.2 we derive a new formula for the inverse of the MHD matrix. In Section 3.3, we form an effective sparse approximation to the inverse and analyze spectral properties for an ideal case. A new block triangular approach is derived and analyzed in Section 3.4. Finally, we present several numerical experiments which demonstrate the strong scalability of this approach in Section 3.5.

3.1 Newton's method discretization of the MHD model

For this Chapter, we consider two non-linear iteration schemes, namely, Picard iteration and Newton's method for the MHD model (1.1)–(1.2). Using the notation

$$\begin{aligned} \boldsymbol{u}_{k+1} &= \boldsymbol{u}_k + \delta\boldsymbol{u}, \quad p_{k+1} = p_k + \delta p, \\ \boldsymbol{b}_{k+1} &= \boldsymbol{b}_k + \delta\boldsymbol{b}, \quad r_{k+1} = r_k + \delta r. \end{aligned}$$

where $*_{k+1}$ is the $k+1^{\text{th}}$ iteration of either the Picard or Newton's iteration, then the linearized system becomes

$$\begin{aligned} -\nu\Delta\delta\boldsymbol{u} + (\boldsymbol{u}_k\cdot\nabla)\delta\boldsymbol{u} + \alpha(\delta\boldsymbol{u}\cdot\nabla)\boldsymbol{u}_k + \nabla\delta p - \kappa(\nabla\times\delta\boldsymbol{b})\times\boldsymbol{b}_k - \alpha\kappa(\nabla\times\boldsymbol{b}_k)\times\delta\boldsymbol{b} &= \boldsymbol{r}_u, \\ \nabla\cdot\delta\boldsymbol{u} &= r_p, \\ \kappa\nu_m\nabla\times(\nabla\times\delta\boldsymbol{b}) + \nabla r - \kappa\nabla\times(\delta\boldsymbol{u}\times\boldsymbol{b}_k) - \kappa\alpha\nabla\times(\boldsymbol{u}_k\times\delta\boldsymbol{b}) &= \boldsymbol{r}_b, \\ \nabla\cdot\delta\boldsymbol{b} &= r_r, \end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{r}_u =& \boldsymbol{f} - \left[-\nu\,\Delta\boldsymbol{u}_k + (\boldsymbol{u}_k\cdot\nabla)\boldsymbol{u}_k + \nabla p_k - \kappa\,(\nabla\times\boldsymbol{b}_k)\times\boldsymbol{b}_k\right],\\
r_p =& -\nabla\cdot\boldsymbol{u}_k,\\
\boldsymbol{r}_b =& \boldsymbol{g} - \left[\kappa\nu_m\,\nabla\times(\nabla\times\boldsymbol{b}_k) + \nabla r - \kappa\,\nabla\times(\boldsymbol{u}_k\times\boldsymbol{b}_k)\right],\\
r_r =& -\nabla\cdot\boldsymbol{b}_k.
\end{aligned}$$

Here we note that the Picard system is for $\alpha = 0$ and the Newton system is $\alpha = 1$.

We consider a finite element discretization of the MHD model (1.1)–(1.2) where the hydrodynamic unknowns ($\boldsymbol{u}$ and p, respectively) are discretized with any stable mixed finite elements and the magnetic and multiplier unknowns are discretized though a mixed edge and nodal element pair. Using the same format as [110], we use Taylor-Hood elements [101] for $(\boldsymbol{u}, p)$ and the lowest order Nédélec [81] pair for $(\boldsymbol{b}, r)$. Upon discretization, we obtain the following matrix system to solve:

$$\begin{pmatrix} F+\alpha F_{\mathrm{NT}} & B^T & C^T+\alpha C_{\mathrm{NT}}^T & 0\\ B & 0 & 0 & 0\\ -C & 0 & M+\alpha M_{\mathrm{NT}} & D^T\\ 0 & 0 & D & 0 \end{pmatrix} \begin{pmatrix} \delta u\\ \delta p\\ \delta b\\ \delta r \end{pmatrix} = \begin{pmatrix} r_u\\ r_p\\ r_b\\ r_r \end{pmatrix}, \tag{3.1}$$

with

$$\begin{aligned}
r_u &= f - Fu_k - C^T b_k - B^T p_k,\\
r_p &= -Bu_k,\\
r_b &= g - Mu_k + Cb_k - D^T r_k,\\
r_r &= -Db_k.
\end{aligned}$$

Similarly to [86], the individual matrix blocks in (3.1) are defined in Tables 3.1 to 3.3.

Matrix-vector product	Continuous operator	Approximate norm
$F\,\delta u$	$\nu\Delta\,\delta\boldsymbol{u}_k + (\boldsymbol{u}_k\cdot\nabla)\delta\boldsymbol{u}_k$	$\nu/h^2 + \|\boldsymbol{u}_k\|/h$
$B^T\,\delta p$	$\nabla\delta p$	$1/h$
$B\,\delta u$	$\nabla\cdot\delta\boldsymbol{u}$	$1/h$
$C^T\,\delta u$	$-\kappa\,(\nabla\times\delta\boldsymbol{b})\times\boldsymbol{b}_k$	$\kappa\|\boldsymbol{b}_k\|/h$

Table 3.1: Block coefficient matrices, their corresponding continuous operators and approximate norms for the Fluid matrices

The recent work in [86] expanded the class of block preconditions to the 4×4 block formulation of the incompressible MHD model. Using the formulation in [95], the

Matrix-vector product	Continuous operator	Approximate norm
$M\,\delta b$	$\kappa\nu_m\,\nabla\times(\nabla\times\delta\boldsymbol{b})$	$\kappa\nu_m/h^2$
$D^T\,\delta r$	∇r	$1/h$
$D\,\delta b$	$\nabla\cdot\delta\boldsymbol{b}$	$1/h$
$-C\,\delta b$	$-\kappa\,\nabla\times(\delta\boldsymbol{u}\times\boldsymbol{b}_k)$	$\|\boldsymbol{b}_k\|/h$

Table 3.2: Block coefficient matrices, their corresponding continuous operators and approximate norms for the Magnetic matrices

Matrix-vector product	Continuous operator	Approximate norm
$F_{\mathrm{NT}}\,\delta u$	$(\delta\boldsymbol{u}\cdot\nabla)\boldsymbol{u}_k$	$\|\nabla\boldsymbol{u}_k\|$
$C_{\mathrm{NT}}^T\,\delta u$	$-\kappa\,(\nabla\times\boldsymbol{b}_k)\times\delta\boldsymbol{b}$	$\kappa\|\nabla\times\boldsymbol{b}_k\|$
$M_{\mathrm{NT}}\,\delta b$	$-\kappa\,\nabla\times(\boldsymbol{u}_k\times\delta\boldsymbol{b})$	$\|\boldsymbol{u}_k\|/h$

Table 3.3: Block coefficient matrices, their corresponding continuous operators and approximate norms for the Newton matrices

authors derive a block triangular preconditioning approach based on Schur complement approximations as well an additive mass matrix shift to the curl-curl operator. The results show a moderate increase in linear iterations with respect to mesh refinement. Our approach looks at deriving an approximate inverse preconditioner for the same 4×4 block system.

3.2 A new formula for the inverse of the MHD coefficient matrix

To preserver the null space properties of the block system, we will derive the new approximate inverse preconditioner for the Picard iteration, $\alpha = 0$. In practice, the numerical results produced in this paper will be done using the Newton nonlinear iteration scheme. Let us denote by $\mathcal{K}^{\mathrm{MHD}}$ the coefficient matrix in the MHD model (3.1) and write it as:

$$\mathcal{K}^{\mathrm{MHD}} = \begin{pmatrix} \mathcal{K}^{\mathrm{NS}} & (\mathcal{K}^{\mathrm{C}})^T \\ -\mathcal{K}^{\mathrm{C}} & \mathcal{K}^{\mathrm{M}} \end{pmatrix},$$

where $\mathcal{K}^{\mathrm{NS}}$ is the Navier-Stokes sub-problem, $\mathcal{K}^{\mathrm{C}}$ is the block for the coupling and $\mathcal{K}^{\mathrm{M}}$ is the Maxwell sub-problem:

$$\mathcal{K}^{\mathrm{NS}} = \begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix}, \quad \mathcal{K}^{\mathrm{M}} = \begin{pmatrix} M & D^T \\ D & 0 \end{pmatrix},$$
$$(\mathcal{K}^{\mathrm{C}})^T = \begin{pmatrix} C^T & 0 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad \mathcal{K}^{\mathrm{C}} = \begin{pmatrix} C & 0 \\ 0 & 0 \end{pmatrix}.$$

Then, by [10, Equation (3.4)], the inverse is given by

$$(\mathcal{K}^{\mathrm{MHD}})^{-1} = \begin{pmatrix} (\mathcal{K}^{\mathrm{NS}})^{-1} + (\mathcal{K}^{\mathrm{NS}})^{-1}(\mathcal{K}^{\mathrm{C}})^T\mathcal{S}^{-1}\mathcal{K}^{\mathrm{C}}(\mathcal{K}^{\mathrm{NS}})^{-1} & -(\mathcal{K}^{\mathrm{NS}})^{-1}(\mathcal{K}^{\mathrm{C}})^T\mathcal{S}^{-1} \\ -\mathcal{S}^{-1}\mathcal{K}^{\mathrm{C}}(\mathcal{K}^{\mathrm{NS}})^{-1} & \mathcal{S}^{-1} \end{pmatrix}, \tag{3.2}$$

where $\mathcal{S}$ denotes the Schur complement,

$$\mathcal{S} \;=\; \mathcal{K}^{\mathrm{M}} + \mathcal{K}^{\mathrm{C}}(\mathcal{K}^{\mathrm{NS}})^{-1}(\mathcal{K}^{\mathrm{C}})^T. \tag{3.3}$$

The inverses $(\mathcal{K}^{\mathrm{NS}})^{-1}$ and $\mathcal{S}^{-1}$ appear multiple times in (3.2), and we now derive explicit formulas that further reveal their block structure. Notably, using results that have appeared in [36], we show that $\mathcal{S}^{-1}$ has a zero (2,2) block, and can be expressed in terms of a free matrix parameter. Let us write

$$(\mathcal{K}^{\mathrm{NS}})^{-1} = \begin{pmatrix} K_1 & K_2 \\ K_3 & K_4 \end{pmatrix}. \tag{3.4}$$

We then have the following useful result.

Theorem 2. *Let $\mathcal{K}^{\mathrm{NS}}$ be written in block form as in* (3.4)*. Then*

$$\mathcal{S}^{-1} = \begin{pmatrix} M_F^{-1}(I - D^TW^{-1}G^T) & GW^{-1} \\ W^{-1}G^T & 0 \end{pmatrix}, \tag{3.5}$$

where W is a (free) symmetric positive definite matrix,

$$M_F = M + D^TW^{-1}D + CK_1C^T \quad \textit{and} \quad G = M_F^{-1}D^T.$$

Proof. Writing out all the matrices involved in formula (3.3) for $\mathcal{S}$, we have

$$\mathcal{S} = \begin{pmatrix} M + CK_1C^T & D^T \\ D & 0 \end{pmatrix}.$$

Since the discrete gradient operator is the null space of M and C^T we have

$$\dim(\text{null}(M + CK_1C^T)) = m_b,$$

where m_b is defined as the number of rows of the magnetic discrete divergence matrix D. Thus the (1,1) block of the Schur complement has the maximum nullity which still leads to a non-singular saddle point system. Therefore using [36, equation (3.6)], the inverse of the Schur complement is given by (3.5). □

Using the inverse formula (3.2) and (3.5) together gives the exact expression for the inverse of (3.1) as

$$(\mathcal{K}^{\text{MHD}})^{-1} = \begin{pmatrix} K_1 - K_1VK_1 & K_2 - K_1VK_2 & -K_1C^TM_F^{-1}H & 0 \\ K_3 - K_3VK_1 & K_4 - K_3VK_2 & -K_3C^TM_F^{-1}H & 0 \\ M_F^{-1}CK_1 & M_F^{-1}CK_2 & M_F^{-1}H & GW^{-1} \\ 0 & 0 & W^{-1}G^T & 0 \end{pmatrix}. \quad (3.6)$$

where $V = C^TM_F^{-1}C$ and $H = I - D^TW^{-1}G^T$.

3.3 A new approximate inverse-based preconditioner

In this section we aim to sparsify the inverse formula in (3.6) exploiting the null-space properties and the approximate order of the block system.

3.3.1 A sparse approximation of the Schur complement

Recall that K_1 is the (1, 1) block matrix of the inverse of the Navier-Stokes sub-problem. Thus forming it is very computationally costly and will often yield a dense matrix. Therefore, CK_1C^T is a major bottleneck within M_F which is a integral part of the Schur complement, $\mathcal{S}$. Defining

$$M_F = M_W + CK_1C^T$$

where $M_W = M + D^TW^{-1}D$, then using a generalization of the Sherman-Morrison-Woodbury theorem, the Binomial inverse formula we can re-write M_F as

$$M_F^{-1} = M_W^{-1} - M_W^{-1}CK_1(K_1 - K_1C^TM_W^{-1}CK_1)^{-1}K_1C^TM_W^{-1}. \quad (3.7)$$

Using (3.7) for G in (3.5) we obtain

$$\begin{aligned} G &= (M_W^{-1} - M_W^{-1}CK_1(K_1 - K_1C^TM_W^{-1}CK_1)^{-1}K_1C^TM_W^{-1})D^T, \\ &= M_W^{-1}D^T - M_W^{-1}CK_1(K_1 - K_1C^TM_W^{-1}CK_1)^{-1}K_1C^TM_W^{-1}D^T = M_W^{-1}D^T, \end{aligned}$$

since $M_W^{-1}D^T$ (being a definition of the discrete gradients from [36, Proposition 3.6]) is the null space matrix of M and C^T.

In order to approximate M_F^{-1} we again use the Binomial inverse formula and note that the approximation orders for C, M_W and K_1 are $\mathcal{O}(h^{-1})$, $\mathcal{O}(h^2)$ and $\mathcal{O}(h^2)$, respectively. Therefore it can be seen that:

$$M_W^{-1} \approx \mathcal{O}(h^2) \quad \text{and} \quad M_W^{-1}CK_1(K_1 - K_1C^TM_W^{-1}CK_1)^{-1}K_1C^TM_W^{-1} \approx \mathcal{O}(h^4).$$

Thus, we use the approximation

$$M_F^{-1} \approx M_W^{-1}.$$

The final step is to consider what matrix to use for W. From [36] we take $DG = W$, recalling that G is the matrix of null vectors of M. Since G is made up of discrete gradients then from [51, Proposition 2.2], W is defined to be the scalar Laplacian. Also, as shown in [51], the vector mass matrix, X, is spectrally equivalent to $D^TW^{-1}D$. Therefore, we set W to be the scalar Laplacian and therefore change the notation to L. Using these two results and the observation that a multiplication of the Schur complement involves multiplications of the leading block with M ($G^TM = 0$), then the simplified inverse Schur complement becomes:

$$\mathcal{S}^{-1} \approx \mathcal{S}_{\mathrm{P}}^{-1} = \begin{pmatrix} M_X^{-1} & GL^{-1} \\ L^{-1}G^T & 0 \end{pmatrix}, \tag{3.8}$$

where $M_X = M + X$ and $G = M_X^{-1}D^T$.

3.3.2 A practical preconditioner

We observe that the application of the inverse, (3.6), involves multiplications of $H = I - D^TW^{-1}G^T$ with M where G^T is the null-space of M. Thus, in a similar fashion to (3.8), we can reduce H to the identity. Also, we note that

$$K_iVK_j \gtrsim \mathcal{O}(h^3)$$

for any $i, j = 1, 2, 3, 4$. Hence, removing these terms we form the first step for the approximation of (3.6) as:

$$(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1} = \begin{pmatrix} K_1 & K_2 & -K_1 C^T M_X^{-1} & 0 \\ K_3 & K_4 & -K_3 C^T M_X^{-1} & 0 \\ M_X^{-1} C K_1 & M_X^{-1} C K_2 & M_X^{-1} & G L^{-1} \\ 0 & 0 & L^{-1} G^T & 0 \end{pmatrix}. \tag{3.9}$$

The final step to approximate (3.9) is to consider the inverse of the Navier-Stokes system $\mathcal{K}^{\mathrm{NS}}$. For this we return to the exact inverse formula of a block matrix in (3.2). Applying this to the Navier-Stokes system gives the precise expression for the inverse

$$(\mathcal{K}^{\mathrm{NS}})^{-1} = \begin{pmatrix} F^{-1} - F^{-1} B^T S_{\mathrm{NS}}^{-1} B F^{-1} & F^{-1} B^T S_{\mathrm{NS}}^{-1} \\ S_{\mathrm{NS}}^{-1} B F^{-1} & -S_{\mathrm{NS}}^{-1} \end{pmatrix}. \tag{3.10}$$

Practically, we use the pressure-convection diffusion (PCD) approximation developed in [35]. Substituting (3.10) into the expression for $(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}$ in (3.9) gives

$$(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1} = \begin{pmatrix} K & F^{-1} B^T S_{\mathrm{NS}}^{-1} & -K C^T M_X^{-1} & 0 \\ S_{\mathrm{NS}}^{-1} B F^{-1} & -S_{\mathrm{NS}}^{-1} & -S_{\mathrm{NS}}^{-1} B F^{-1} C^T M_X^{-1} & 0 \\ M_X^{-1} C K & M_X^{-1} C F^{-1} B^T S_{\mathrm{NS}}^{-1} & M_X^{-1} & G L^{-1} \\ 0 & 0 & L^{-1} G^T & 0 \end{pmatrix}, \tag{3.11}$$

where

$$K = F^{-1} - F^{-1} B^T S_{\mathrm{NS}}^{-1} B F^{-1}.$$

As with the approximation of M_F^{-1} in Section 3.3, we consider the approximate orders of the individual blocks of (3.11). Removing the $\mathcal{O}(h^3)$ terms in the (1,3) and (3,1) blocks of (3.11) yields the approximation:

$$(\mathcal{M}_{\mathrm{A}}^{\mathrm{MHD}})^{-1} = \begin{pmatrix} K & F^{-1} B^T S_{\mathrm{NS}}^{-1} & 0 & 0 \\ S_{\mathrm{NS}}^{-1} B F^{-1} & -S_{\mathrm{NS}}^{-1} & -S_{\mathrm{NS}}^{-1} B F^{-1} C^T M_X^{-1} & 0 \\ 0 & M_X^{-1} C F^{-1} B^T S_{\mathrm{NS}}^{-1} & M_X^{-1} & G L^{-1} \\ 0 & 0 & L^{-1} G^T & 0 \end{pmatrix}. \tag{3.12}$$

3.3.3 Spectral analysis

To analyze the spectral properties of our approximate inverse preconditioner, we investigate the eigenspectrum of $(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1} \mathcal{K}^{\mathrm{MHD}}$ where $(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}$ is given in (3.11).

We consider the $\mathcal{O}(h^3)$ approximate inverse, (3.11), to simplify the complex eigenvalue analysis. The form of $(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$ is given by:

$$(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}} = \begin{pmatrix} I_{\mathrm{u}} + KC^T M_L^{-1} C & 0 & KC^T(I - M_L^{-1}M) & 0 \\ S_{\mathrm{NS}}^{-1} BF^{-1}C^T M_L^{-1} C & I_{\mathrm{p}} & S_{\mathrm{NS}}^{-1} BF^{-1}C^T(I - M_L^{-1}M) & 0 \\ 0 & 0 & M_L^{-1}(M + CKC^T) & G \\ 0 & 0 & 0 & I_{\mathrm{r}} \end{pmatrix}. \tag{3.13}$$

where M_L is the exact primal Schur complement $M + DL^{-1}D^T$. Let us introduce a few identities utilizing the null space properties of C^T and M_L. Proposition 1 is particularly useful to simplify (3.13).

Proposition 1. *The following relations hold:*

(i) $C^T(I - M_L^{-1}M) = 0$,

(ii) $C^T(I + M_L^{-1}M) = 2C^T$

Proof. Since C^T and M_L have the same null space (space of discrete gradients), by using the Helmholtz decomposition

$$b = d + \nabla\,\phi,$$

we obtain

$$C^T(I - M_L^{-1}M)b = C^T(I - M_L^{-1}M)d,$$

where $d \notin \mathrm{Null}(M)$. Since $M_L^{-1}M$ is zero on the null space of M and an identity on the range space of M, then

$$(I - M_L^{-1}M) \in \mathrm{Null}(M).$$

Therefore, identity (i) holds due to the fact the null spaces of C^T and M are the same. Using similar arguments, identity (ii) can also be shown to be true. □

Using Proposition 1 simplifies (3.13) to:

$$(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}} = \left(\begin{array}{cc|cc} I_{\mathrm{u}} + KC^T M_L^{-1} C & 0 & 0 & 0 \\ S_{\mathrm{NS}}^{-1} BF^{-1}C^T M_L^{-1} C & I_{\mathrm{p}} & 0 & 0 \\ \hline 0 & 0 & M_L^{-1}(M + CKC^T) & G \\ 0 & 0 & 0 & I_{\mathrm{r}} \end{array}\right), \tag{3.14}$$

where we recall that $G = M_L^{-1}D^T$ and $K = F^{-1} - F^{-1}B^T S_{\mathrm{NS}}^{-1} BF^{-1}$.

Theorem 3. *The matrix* $(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$ *has an eigenvalue* $\lambda = 1$ *of algebraic multiplicity at least* $n_u - n_b + 3m_b + m_u$. *The corresponding eigenvectors* $\{v_i\}_{i=1}^{n_u-n_b+3m_b+m_u}$ *are given by*

$$v_i = (u_i,\, p_i,\, b_i,\, r_i)$$

where $u_i \in Null(C),\ b_i \in Null(M)$ *and* p_i *and* r_i *free.*

Proof. Since (3.14) is block diagonal where the two blocks are also triangular, the generalized eigenvalue problem

$$(\widetilde{\mathcal{M}}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}v = \lambda v,$$

where $v = (u,\, p,\, b,\, r)$ can be simplified to solving:

$$\begin{aligned}
\lambda u &= (I_{\mathrm{u}} + KC^T M_L^{-1} C)u, && (3.15)\\
\lambda p &= p, && (3.16)\\
\lambda b &= M_L^{-1}(M + CKC^T)b, && (3.17)\\
\lambda r &= r. && (3.18)
\end{aligned}$$

Consider $\lambda = 1$, then trivially (3.16) and (3.18) are automatically satisfied. Taking $u \in \mathrm{Null}(C)$ and $b \in \mathrm{Null}(M + CKC^T)$ then (3.15) and (3.17) also hold. Since the null space of M and C^T are the same we choose $b \in \mathrm{Null}(M)$. □

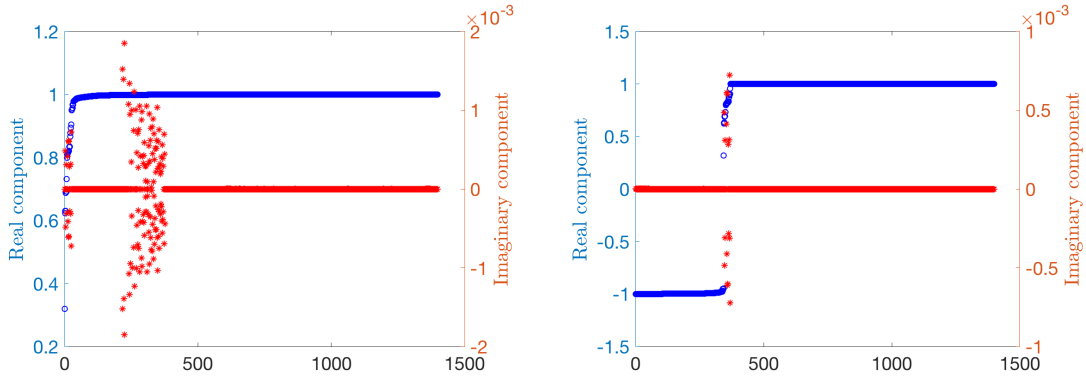
Figure 3.1a shows the preconditioned eigenvalues for $(\mathcal{M}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}$ in (3.12) using the PCD approximation (from [35]) for the fluid Schur complement and the vector mass matrix approximation of the magnetic Schur complement. Here we see that the clustering around the eigenvalue $\lambda = 1$ is very strong.

3.4 A block triangular preconditioner

As well as the approximate inverse preconditioner, we introduce a Schur complement based preconditioner for (3.1). Again we will consider the simpler Picard case, $\alpha = 0$, but in practice use the Newton nonlinear scheme. We follow the well known setting of [62, 80] for experimental comparison. Let us define $\widetilde{\mathcal{M}}_{\mathrm{B}}^{\mathrm{MHD}}$ as:

$$\widetilde{\mathcal{M}}_{\mathrm{B}}^{\mathrm{MHD}} = \begin{pmatrix} \mathcal{K}^{\mathrm{NS}} & (\mathcal{K}^{\mathrm{C}})^T \\ 0 & -\mathcal{S} \end{pmatrix}. \tag{3.19}$$

From [62, 80] the preconditioned matrix, $(\widetilde{\mathcal{M}}_{\mathrm{B}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$, has precisely two eigenvalues ± 1 and is diagonalizable. We would therefore expect an appropriate Krylov



(a) Real (blue) and imaginary (red) parts of eigenvalues of preconditioned matrix $(\mathcal{M}_{\mathrm{A}}^{\mathrm{MHD}})^{-1}\,\mathcal{K}^{\mathrm{MHD}}$.

(b) Real (blue) and imaginary (red) parts of eigenvalues of preconditioned matrix $(\mathcal{M}_{\mathrm{B}}^{\mathrm{MHD}})^{-1}\,\mathcal{K}^{\mathrm{MHD}}$.

Figure 3.1: Preconditioned eigenvalue plots for (a) the approximate inverse preconditioner in (3.12) and (b) the block triangular preconditioner in (3.21) using the smooth solution given (3.31). The dimension of the matrices in this example is 1399×1399.

subspace solver to converge within two iterations in exact precision. To use $\widetilde{\mathcal{M}}_{\mathrm{B}}^{\mathrm{MHD}}$ as a preconditioner, we require a direct Navier-Stokes solve and a Schur complement solve.

The direct solve for the Navier-Stokes system is too costly, so we approximate $\mathcal{K}^{\mathrm{NS}}$ with the Schur complement system:

$$\mathcal{M}_{\mathrm{I}}^{\mathrm{NS}} = \begin{pmatrix} F & B^T \\ 0 & -S_{\mathrm{NS}} \end{pmatrix}, \tag{3.20}$$

where $S_{\mathrm{NS}} = BF^{-1}B^T$ is the fluid Schur complement. Using (3.20), we obtain the more practical preconditioner

$$\mathcal{M}_{\mathrm{B}}^{\mathrm{MHD}} = \begin{pmatrix} \mathcal{M}_{\mathrm{NS}} & \mathcal{K}^{\mathrm{C}} \\ 0 & -\mathcal{S} \end{pmatrix}. \tag{3.21}$$

Theorem 4. *The matrix $(\mathcal{M}_{\mathrm{B}}^{\mathrm{MHD}})^{-1}\mathcal{K}^{\mathrm{MHD}}$ has an eigenvalue $\lambda = 1$ of algebraic multiplicity at least n_u, and an eigenvalue $\lambda = -1$ of algebraic multiplicity at least n_b. The corresponding (known) eigenvectors are given as follows:*

$\lambda = 1$: with eigenvectors $\{v_i\}_{i=1}^{n_b - m_b}$ and $\{v_j\}_{j=n_b-m_b+1}^{n_u}$, as follows:

$$v_i = (u_i, -S^{-1}Bu_i, b_i, 0) \quad \text{and} \quad v_j = (u_j, -S^{-1}Bu_j, 0, 0),$$

where $b_i \in null(D) \neq 0$, $Cu_i = (2M + CK_1C^T)b_i$ and $u_j \in null(C)$.

$\lambda = -1$*: with eigenvectors* $\{v_i\}_{i=1}^{n_b-m_b}$ *and* $\{v_j\}_{j=n_b-m_b+1}^{n_b}$*, as follows:*

$$v_i = (u_i, 0, b_i, r_i) \quad \textit{and} \quad v_j = (0, 0, b_j, r_j), \tag{3.22}$$

where $u_i \in$ *null*$(B) \neq 0$*,* $Fu_i + C^T b_i = 0$*,* $b_j \in$ *null*(M)*,* r_i *and* r_j *free.*

Proof. The corresponding eigenvalue problem is

$$\begin{pmatrix} F & B^T & C^T & 0 \\ B & 0 & 0 & 0 \\ -C & 0 & M & D^T \\ 0 & 0 & D & 0 \end{pmatrix} \begin{pmatrix} u \\ p \\ b \\ r \end{pmatrix} = \lambda \begin{pmatrix} F & B^T & C^T & 0 \\ 0 & -S_{\mathrm{NS}} & 0 & 0 \\ 0 & 0 & -(M + K_C) & -D^T \\ 0 & 0 & -D & 0 \end{pmatrix} \begin{pmatrix} u \\ p \\ b \\ r \end{pmatrix},$$

where $K_C = CK_1C^T$. The four block rows of the generalized eigenvalue problem can be written as

$$\begin{aligned} (1-\lambda)(Fu + B^T p + C^T b) &= 0, && (3.23)\\ Bu &= -\lambda S_{\mathrm{NS}}\, p, && (3.24)\\ (1+\lambda)(Mb + D^T r) + \lambda CK_1C^T b - Cu &= 0, && (3.25)\\ (1+\lambda)Db &= 0. && (3.26) \end{aligned}$$

If $\lambda = 1$, (3.23) is automatically satisfied. Equation (3.24) simplifies to:

$$p = -S_{\mathrm{NS}}^{-1} Bu.$$

From (3.26) we have $Db = 0$, hence, $b \in \mathrm{null}(D)$. Let us take $r = 0$, then (3.25) yields

$$Cu = (2M + CK_1C^T)b. \tag{3.27}$$

Case 1: Consider $0 \neq b \in \mathrm{null}(D)$, then $Cu = (2M + CK_1C^T)b$. Since, rank of C and $(2M + CK_1C^T)$ is $n_b - m_b$, then the condition (3.27) has at least $n_b - m_b$ linearly independent eigenvectors.

Case 2: Consider $b = 0$, then we have that $Cu = 0$. Hence, u must be in the null space of C. Since

$$\dim(\mathrm{null}(C)) \;=\; n_u - n_b + m_b,$$

this accounts for $n_u - n_b + m_b$ such eigenvectors.

Therefore $\lambda = 1$ is an eigenvalue with algebraic multiplicity at least n_u.

If $\lambda = -1$, (3.26) is satisfied, hence, r is free. Simplifying (3.25) obtains

$$CK_1C^Tb + Cu = 0. \tag{3.28}$$

Let us take $u \in \text{null}(B)$, then $p = 0$ and the condition for b is

$$Fu + C^Tb = 0. \tag{3.29}$$

Under the condition that $u \in \text{null}(B)$, (3.29) satisfies the equality (3.28).

Case 1: Consider $u \in \text{null}(B)$ and $u \neq 0$, then from (3.29) we have $u = -F^{-1}C^Tb$. Since the rank of C^T is $n_b - m_b$ and F is full rank, then there are only $n_b - m_b$ such linearly independent b's that determine u. Hence, for this case we obtain at least $n_b - m_b$ such eigenvectors.

Case 2: Consider $u = 0$, then for (3.29) to hold $C^Tb = 0$. Therefore, we take $b \in \text{null}(C^T)$. Since, the null space of C^T is made up of discrete gradients then

$$\dim(\text{null}(C^T)) \;=\; m_b.$$

Therefore $\lambda = -1$ is an eigenvalue with algebraic multiplicity at least n_b.

□

Remark 4. *Note that in* (3.22)*,* $\{u_i\}$ *is a* subset *of the null vectors of* B*.*

An approximation of the fluid Schur complement, S_{NS}, is needed to create a practical preconditioner. For S_{NS} we will use the PCD preconditioner developed in [35]. The approximation is based on

$$S_{\text{NS}} = BF^{-1}B^T \approx A_p\, F_p^{-1} Q_p,$$

where the matrix A_p is the pressure Laplacian, F_p is the pressure convection-diffusion operator and Q_p is the pressure mass matrix.

$\mathcal{M}_{\text{B}}^{\text{MHD}}$ is defined in (3.21), but in practice we use the approximation for the inverse of the Schur complement from Section 3.3 which is

$$\mathcal{S}^{-1} \approx \mathcal{S}_{\text{P}}^{-1} = \begin{pmatrix} M_X^{-1} & GL^{-1} \\ L^{-1}G^T & 0 \end{pmatrix}.$$

The eigenvalues of the preconditioned matrix $(\mathcal{M}_{\text{B}}^{\text{MHD}})^{-1}\mathcal{K}^{\text{MHD}}$ are represented in Figure 3.1b. As with the eigenvalues for the approximate inverse preconditioned matrix we

see a small degradation of the eigenvalue clusters. However, we still see strong clustering around 1 and -1.

3.5 Numerical experiments

In this section, we present several 3D numerical results to illustrate the performance of our preconditioning approaches. We use *FEniCS* [77], a finite element software package, to create the matrix system and `PETSc` [7, 9]) and `HYPRE` [38] to solve the resulting system.

In line with [86], we set the non-linear stopping tolerance to 1e-4 and the linear solve tolerance as 1e-3. For all experiments we us FGMRES [92] and the Newton ($\alpha = 1$) nonlinear iteration scheme. This means that for every solve associated with F and multiplication associated with C^T we replace them with $\hat{F} = F + F_{\text{NT}}$ and $\hat{C}^T = C^T + C^T_{\text{NT}}$. Table 3.4 details the methods which we use to solve the systems associated with the block preconditioner.

Matrix	Implementation method
Q_p	single AMG V-cycle
A_p	single AMG V-cycle
$\hat{F}$	Preconditioned AMG GMRES with tolerance 1e-2
$M + X$	AMG method developed in [61] with tolerance 1e-2
L	single AMG V-cycle

Table 3.4: Solution method for block systems associated with the preconditioners

We use the **Notation**: ℓ is the mesh level, DoF is the total degrees of freedom, time is the average solve time to solve systems associated with $\mathcal{K}^{\text{MHD}}$ at each nonlinear iteration, it_{NL} is the number of nonlinear/Newton iterations to reach the stopping criteria, it_{O} is the average number of linear/FGMRES iterations to solve $\mathcal{K}^{\text{MHD}}$, it_{MX} is the average of CG/Auxiliary Space iterations to solve $M + X$ and it_{F} is the average of GMRES iterations to solve $\hat{F}$. Adding an A or B superscript to time or it_* denotes whether we use the approximate inverse or block preconditioner, respectively

3.5.1 3D Cavity driven flow

The first example we consider is the classic lid driven cavity problem [35]. The problem is designed by the following Dirichlet boundary conditions:

$$\begin{aligned} &\mathbf{u} = (1,0,0) \quad \text{on} \quad z = 1, \\ &\mathbf{u} = (0,0,0) \quad \text{on} \quad x = \pm 1,\ y = \pm 1,\ z = -1, \\ &\mathbf{n} \times \mathbf{b} = \mathbf{n} \times \mathbf{b}_N \quad \text{on} \quad \partial\Omega, \\ &r = 0 \quad \text{on} \quad \partial\Omega, \end{aligned} \tag{3.30}$$

where $\mathbf{b}_N = (-1, 0, 0)$.

Scalability results

Tables 3.5 and 3.6 show the time and iteration results using the approximate inverse and block triangular preconditioners. We can see that the approximate inverse preconditioner exhibits near perfect scaling with respect to the FGMRES iterations for both tables, whereas the FGMRES iterations for the block triangular preconditioner increase each mesh level for the harder problem, Ha $= \sqrt{1000}$. For the easier parameter setup, Table 3.5, we can see that both preconditioners exhibit scalable iterations. We note that for the latest mesh level ($\ell = 6$) for both Tables 3.5 and 3.6 the approximate inverse preconditioner yields a faster solution method. In fact, for the more difficult problem (Ha $= \sqrt{1000}$) the approximate inverse preconditioner is almost exactly two times quicker than the block triangular precondition for $\ell = 6$.

ℓ	DoFs	time$^{\mathrm{A}}$	$it^{\mathrm{A}}_{\mathrm{NL}}$	$it^{\mathrm{A}}_{\mathrm{O}}$	$it^{\mathrm{A}}_{\mathrm{MX}}$	$it^{\mathrm{A}}_{\mathrm{F}}$	time$^{\mathrm{B}}$	$it^{\mathrm{B}}_{\mathrm{NL}}$	$it^{\mathrm{B}}_{\mathrm{O}}$	$it^{\mathrm{B}}_{\mathrm{MX}}$	$it^{\mathrm{B}}_{\mathrm{F}}$
1	14,012	1.18	3	10.0	1.70	1.61	0.81	4	21.5	2.00	1.96
2	28,436	2.79	3	10.0	1.78	1.65	1.93	4	21.5	2.00	1.96
3	64,697	11.43	3	9.7	1.85	1.78	6.90	4	21.0	2.00	2.00
4	245,276	34.64	4	11.0	2.11	1.85	13.65	3	18.3	2.00	2.00
5	937,715	255.25	4	10.5	2.21	1.95	135.88	3	19.0	2.47	2.00
6	5,057,636	1979.22	3	9.7	2.71	2.11	2273.48	3	22.3	3.00	2.50

Table 3.5: 3D Cavity Driven using both the approximate inverse and block triangular preconditioner with parameters $\kappa = 1$, $\nu = 1$, $\nu_m = 1$ and Ha $= 1$.

Numerical cost of preconditioners

From the definition of the approximate inverse preconditioner it is obvious that each iteration is more expensive than the block triangular preconditioner. However, as seen

ℓ	DoFs	time$^{\mathrm{A}}$	$it^{\mathrm{A}}_{\mathrm{NL}}$	$it^{\mathrm{A}}_{\mathrm{O}}$	$it^{\mathrm{A}}_{\mathrm{MX}}$	$it^{\mathrm{A}}_{\mathrm{F}}$	time$^{\mathrm{B}}$	$it^{\mathrm{B}}_{\mathrm{NL}}$	$it^{\mathrm{B}}_{\mathrm{O}}$	$it^{\mathrm{B}}_{\mathrm{MX}}$	$it^{\mathrm{B}}_{\mathrm{F}}$
1	14,012	7.58	4	57.0	1.96	2.00	5.58	4	146.2	2.00	2.00
2	28,436	22.21	4	56.2	1.98	2.00	14.88	4	147.2	2.00	2.00
3	64,697	65.95	4	56.0	1.99	2.00	47.79	4	154.2	2.00	2.00
4	245,276	271.48	4	56.0	2.15	2.00	205.63	4	160.5	2.23	2.00
5	937,715	1255.15	4	55.5	2.79	2.01	1003.92	4	168.8	2.91	2.00
6	5,057,636	17656.36	4	58.5	2.99	2.07	35563.15	4	217.5	3.03	2.03

Table 3.6: 3D Cavity Driven using both the approximate inverse and block triangular preconditioner with parameters $\kappa =$ 1e1, $\nu =$ 1e-1, $\nu_m =$ 1e-1 and Ha $= \sqrt{1000}$

in Table 3.6 the scalability of this preconditioner for harder and larger problems yields smaller timing results.

Table 3.7 shows the number of solves and matrix-vector products associated with the individual block matrices that make up the approximate inverse and the block triangular preconditioners for mesh level $\ell = 6$ in Table 3.6. We can see that the approximate inverse preconditioner has a smaller number of solves for both systems associated with vector ($\hat{F}$ and M_X) and scalar (A_p, Q_p and L) valued matrices. Also, the total number of matrix-vector products is less for the approximate inverse preconditioner. This is reflected in the time it takes to solve this system, where the approximate inverse preconditioner is faster than the block triangular one. We also note that the iterations in Table 3.6 remain constant for the approximate inverse preconditioner compared to the block triangular one. Therefore, for harder problems on larger meshes it seems that the approximate inverse preconditioner seems more efficient with respect to time and iterations.

3.5.2 Fichera corner

The second solution we consider is a smooth solution on a non-convex domain. Specifically, the domain is a cube missing a corner and is defined as $\Omega = (0,1)^3/[0.5,1) \times [0.5,1) \times [0.5,1)$. The exact solution is

$$\begin{aligned} &\mathbf{u} = \nabla \times (u_1, u_1, u_1) \quad \text{on} \quad \Omega, \\ &p = xyz(x-1)(y-1)(z-1)\exp(x) \quad \text{on} \quad \Omega \\ &\mathbf{b} = \nabla \times (b_1, b_1, b_1) \quad \text{on} \quad \Omega, \\ &r = xyz(x-1)(y-1)(z-1)\exp(x+y+z) \quad \text{on} \quad \Omega, \end{aligned} \tag{3.31}$$

Linear operation	Number of operations $\mathcal{M}_{\mathrm{A}}^{\mathrm{MHD}}$	$\mathcal{M}_{\mathrm{B}}^{\mathrm{MHD}}$
$\hat{F}^{-1}$	4*58.5 = 234.0	1*217.5 = 217.5
A_p^{-1}	3*58.5 = 175.5	1*217.5 = 217.5
Q_p^{-1}	3*58.5 = 175.5	1*217.5 = 217.5
M_X^{-1}	2*58.5 = 117.0	1*217.5 = 217.5
L^{-1}	2*58.5 = 117.0	1*217.5 = 217.5
$\hat{C}^T$ or C	2*58.5 = 117.0	1*217.5 = 217.5
B^T or B	4*58.5 = 234.0	1*217.5 = 217.5
G^T or G	2*58.5 = 117.0	2*217.5 = 435.0
Average total FMGRES iteration time	17656.36	35563.15

Table 3.7: Numerical cost of using $\mathcal{M}_{\mathrm{A}}^{\mathrm{MHD}}$ and $\mathcal{M}_{\mathrm{B}}^{\mathrm{MHD}}$ for mesh level $\ell = 6$ in Table 3.6

where

$$u_1 = x^2y^2z^2(x-1)^2(y-1)^2(z-1)^2\cos(x),$$
$$b_1 = x^2y^2z^2(x-1)^2(y-1)^2(z-1)^2\sin(y),$$

which defines the inhomogeneous Dirichlet boundary conditions and forcing terms $\boldsymbol{f}$ and $\boldsymbol{g}$.

Table 3.8 shows the timing and iteration results for the following two setups:

- Setup 1: κ = 1e1, ν = 1e-2, ν_m = 1e-2 and Ha = $\sqrt{1e5}$,
- Setup 2: κ = 1e1, ν = 1e-2, ν_m = 1e-3 and Ha = 1000.

We can see that for Setup 1 (Ha = $\sqrt{1e5}$) that the outer FGMRES iterations remain constant. However, when considering Setup 2 (Ha = 1000) we start to see a large degradation in terms of the iteration counts. This is not unexpected for such problems.

3.5.3 MHD generator

This is a problem that describes unidirectional flow in a duct which induces an electromagnetic field. We consider the channel $[0,5]\times[0,1]\times[0,1]$. On the left and right boundaries we enforce the boundary condition $\mathbf{u} = (1,0,0)$ and on the other walls a no slip boundary condition is applied. Defining $\delta = 0.1$, $b_0 = 1$, $x_{\mathrm{on}} = 2$ and $x_{\mathrm{off}} = 2.5$ the boundary condition associated with the magnetic unknowns is $\mathbf{n}\times\mathbf{b} = \mathbf{n}\times(0,\mathbf{b}_y,0)$ where

$$\mathbf{b}_y = \frac{b_0}{2}\left[\tanh\left(\frac{x-x_{\mathrm{on}}}{\delta}\right) - \tanh\left(\frac{x-x_{\mathrm{off}}}{\delta}\right)\right].$$

ℓ	DoFs	Setup 1					Setup 2				
		time$^{\mathrm{A}}$	$it^{\mathrm{A}}_{\mathrm{NL}}$	$it^{\mathrm{A}}_{\mathrm{O}}$	$it^{\mathrm{A}}_{\mathrm{MX}}$	$it^{\mathrm{A}}_{\mathrm{F}}$	time$^{\mathrm{A}}$	$it^{\mathrm{A}}_{\mathrm{NL}}$	$it^{\mathrm{A}}_{\mathrm{O}}$	$it^{\mathrm{A}}_{\mathrm{MX}}$	$it^{\mathrm{A}}_{\mathrm{F}}$
1	34,250	15.64	4	29.2	2.56	2.00	102.34	7	211.7	2.04	2.00
2	57,569	30.41	4	29.2	2.74	2.00	242.27	7	237.0	2.10	2.00
3	89,612	52.90	4	28.8	2.91	2.00	440.66	7	252.1	2.15	2.00
4	332,744	232.23	4	27.8	2.96	2.00	2361.45	7	294.3	2.40	2.00
5	999,269	1026.31	4	27.8	2.98	2.95	8657.89	7	303.9	2.76	2.12
6	5,232,365	11593.47	5	28.6	2.99	3.00	111675.33	7	321.4	2.93	2.48

Table 3.8: Fichera corner using the approximate inverse preconditioner. Setup 1: κ =1e1, ν =1e-2, ν_m =1e-2 and Ha=$\sqrt{1e5}$ and Setup 2: κ =1e1, ν =1e-2, ν_m =1e-3 and Ha=1000.

The timing and iteration results for the approximate inverse preconditioner are represented in Table 3.9. From the table we can see that the iteration counts decrease as the problem gets larger. We suspect this is due to the fact the the mesh size h is becoming small enough for mesh level $\ell \geq 3$ that the fluid and magnetic viscosities are correctly capture on these meshes.

ℓ	DoFs	time$^{\mathrm{A}}$	$it^{\mathrm{A}}_{\mathrm{NL}}$	$it^{\mathrm{A}}_{\mathrm{O}}$	$it^{\mathrm{A}}_{\mathrm{MX}}$	$it^{\mathrm{A}}_{\mathrm{F}}$
1	2,199	1.50	3	172.7	1.50	1.92
2	13,809	13.32	3	108.0	1.54	1.99
3	96,957	260.81	4	105.2	1.95	2.00
4	724,725	1693.26	3	70.7	1.98	2.79
5	5,600,229	8515.71	3	68.0	2.10	2.64

Table 3.9: MHD generator using the approximate inverse preconditioner with parameters $\kappa = 1$, ν = 1e-1, ν_m = 1e-1 and Ha = 10

Chapter 4

Conjugate gradient for nonsingular saddle-point systems with a maximally rank-deficient leading block

The final chapter is currently under revision for publication in the Journal of Computational and Applied Mathematics. In Section 4.1, we introduce the saddle-point matrix and the specific properties considered. In Section 4.2 we derive conditions under which this saddle-point system can be solved using preconditioned conjugate gradient. In Section 4.3, we consider a null-space decoupling of (4.1). In Section 4.4 we analyze the spectral properties of the block preconditioners. Section 4.5 presents several numerical results for the mixed Maxwell problem (1.5) using the preconditioned conjugate gradient method.

4.1 Problem statement

In this chapter, we derive two sufficient conditions that allow the use of the conjugate gradient (CG) [58] method to solve an indefinite saddle-point system with a maximally rank-deficient leading block. Our work builds on [36], where the authors developed an indefinite approximate inverse preconditioner for such problems. We expand this to consider the family of block diagonal and block triangular preconditioners from [50, 51].

Consider the regularized saddle-point system

$$\underbrace{\begin{pmatrix} N & E^T \\ E & -R \end{pmatrix}}_{\mathcal{K}^{\mathrm{N}}} \begin{pmatrix} b \\ r \end{pmatrix} = \begin{pmatrix} h \\ 0 \end{pmatrix}, \tag{4.1}$$

where $N \in \mathbb{R}^{n\times n}$, $E \in \mathbb{R}^{m\times n}$ and $R \in \mathbb{R}^{m\times m}$ with $m < n$. In many situations we have $R = 0$, however for some PDE discretizations, using a symmetric positive semi-definite

matrix $R \neq 0$ can be utilized as a stabilization procedure. We focus our investigation on symmetric positive semi-definite matrices N. In particular, we are interested in maximally rank-deficient leading blocks that still yield to a nonsingular block matrix, $\mathcal{K}^{\mathrm{N}}$. For $R = 0$, we require the following properties to ensure (4.1) is invertible:

$$\operatorname{rank}(N) = n - m, \quad \operatorname{rank}(E) = m, \quad \text{and} \quad \ker(N) \cap \ker(E) = \{0\}.$$

For $R \neq 0$, the requirement on $\operatorname{rank}(E)$ can be relaxed.

Applying CG to indefinite linear systems has been extensively considered; see for example [11, 12, 18, 34, 47, 69, 75, 91]. In [75], the authors show that negating the second block row of a symmetric saddle-point matrix obtains the property that the new saddle-point matrix is real positive. The authors use this property to derive conditions for positive definiteness with respect to a certain bilinear form. The authors in [11, 12, 34] show that for the class of constraint-based preconditioners the preconditioned matrix has all positive eigenvalues. Thus, even though the preconditioner and saddle-point matrix are indefinite, CG can still be used. Finally, in [18, 69] the authors use a non-standard inner product and [47, 91] start (and remaining) on a certain manifold, both to ensure that CG is possible.

In this work, we consider the class of maximally rank-deficient regularized and unregularized saddle-point systems. The class of preconditioners we use do not rely on the spectral structure of the preconditioner matrix, and thus, do not lead to similar techniques as mentioned above. Therefore, the approach we take relies on the construction of the Krylov subspace:

$$\mathbb{K}_k(\mathcal{M}^{-1}\mathcal{K}, r_0) = \operatorname{span}\{r_0,\ \mathcal{M}^{-1}\mathcal{K}\, r_0,\ \ldots,\ (\mathcal{M}^{-1}\mathcal{K})^{k-1}\, r_0\},$$

where r_0, $\mathcal{K}$ and $\mathcal{M}$ are the preconditioned initial residual, the coefficient matrix, and the preconditioner, respectively. If $\mathcal{K}$ and $\mathcal{M}$ are both SPD then clearly it is possible to use a solver such as CG for SPD matrices. In this work $\mathcal{K}$ is indefinite and $\mathcal{M}$ is either SPD or structurally nonsymmetric (block triangular preconditioners), thus $\mathcal{M}^{-1}\mathcal{K}$ will not be SPD or even real positive. However, by explicitly forming the Krylov subspace we show that for two sufficient conditions the Krylov subspace is constructed using products of SPD and SPSD matrices arising from individual blocks of $\mathcal{K}$ and $\mathcal{M}$. These conditions rely heavily on the maximal nullity of the leading block of (4.1). Thus, even though $(\mathcal{M}^{-1}\mathcal{K})^k$ is not SPD, the application of $(\mathcal{M}^{-1}\mathcal{K})^k r_0$ can be defined as a product of SPD and SPSD matrices on the preconditioned residual. This therefore means that we can use CG even though the preconditioned system has both positive and negative eigenvalues and it is also structurally nonsymmetric.

As we show in subsequent sections, the above properties are critical for accomplishing our main goal stated, which is to establish conditions under which CG may be used, despite the matrix $\mathcal{K}^{\mathrm{N}}$ being indefinite. We accompany our analytical observations with numerical experiments.

4.2 Krylov Subspace

In [50], the authors show that an ideal preconditioner for saddle-point systems of the form (4.1) with a maximally rank-deficient leading block is the augmented preconditioner:

$$\begin{pmatrix} N + E^T U^{-1} E & 0 \\ 0 & U \end{pmatrix}, \tag{4.2}$$

where U is an arbitrary symmetric positive-definite matrix. The authors of [47] promote the use of projected conjugate gradient.

Our goal is to find conditions on an approximation Q to the augmented term, $E^T U^{-1} E$, such that we can use preconditioned CG. We consider a preconditioner of the following form:

$$\mathcal{M}_P = \begin{pmatrix} N + Q & 0 \\ 0 & U \end{pmatrix},$$

where Q is chosen so that $N + Q$ is SPD. It is possible to use CG to solve (4.1) when the preconditioned matrix $\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}$ is symmetrizable and its symmetrized version is positive definite. Given an initial preconditioned residual r_0, the Krylov subspace is given by:

$$\mathbb{K}_k(\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}, r_0) = \mathrm{span}\{r_0,\ \mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}\, r_0,\ \ldots,\ (\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}})^{k-1}\, r_0\}. \tag{4.3}$$

Considering a zero initial guess, the initial preconditioned residual is given by:

$$r_0 = \mathcal{M}_P^{-1} \begin{pmatrix} h \\ 0 \end{pmatrix} = \begin{pmatrix} (N + Q)^{-1} h \\ 0 \end{pmatrix}. \tag{4.4}$$

To construct (4.3), an essential step is multiplication with the preconditioned matrix, which is given by:

$$\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}} = \begin{pmatrix} N + Q & 0 \\ 0 & U \end{pmatrix}^{-1} \begin{pmatrix} N & E^T \\ E & -R \end{pmatrix} = \begin{pmatrix} (N + Q)^{-1} N & (N + Q)^{-1} E^T \\ U^{-1} E & -U^{-1} R \end{pmatrix}.$$

Theorem 5. *Given a symmetric indefinite block* 2×2 *matrix and symmetric positive-*

definite preconditioner

$$\mathcal{K}^{\mathrm{N}} = \begin{pmatrix} N & E^T \\ E & -R \end{pmatrix} \quad \textit{and} \quad \mathcal{M}_P = \begin{pmatrix} N+Q & 0 \\ 0 & U \end{pmatrix},$$

assuming a zero initial guess, for the preconditioned residual r_0 *given in* (4.4)*, the resulting multiplications with the preconditioned matrices can be simplified as follows:*

$$[\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}]^i\, r_0 = \begin{pmatrix} [(N+Q)^{-1}N]^i(N+Q)^{-1}h \\ 0 \end{pmatrix} \quad \textit{for} \quad i = 0,\ 1,\ \ldots \tag{4.5}$$

as long as the following conditions hold:

$$Z^T h = 0, \tag{4.6a}$$

$$NZ = 0, \tag{4.6b}$$

where $Z = (N+Q)^{-1}E^T$*.*

Proof. The proof follows by induction. For $i = 0$ we have r_0, and for $i = 1$ we have

$$\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}\, r_0 = \begin{pmatrix} (N+Q)^{-1}N(N+Q)^{-1}h \\ U^{-1}E(N+Q)^{-1}h \end{pmatrix}. \tag{4.7}$$

Thus by condition (4.6a) we see that (4.7) holds for $i = 1$. Now assuming

$$[\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}]^{i-1} r_0 = \begin{pmatrix} [(N+Q)^{-1}N]^{i-1}(N+Q)^{-1}h \\ 0 \end{pmatrix}, \tag{4.8}$$

it readily follows that

$$\begin{aligned}
[\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}]^i r_0 &= [\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}][\mathcal{M}_P^{-1}\mathcal{K}^{\mathrm{N}}]^{i-1} r_0 \\
&= \begin{pmatrix} (N+Q)^{-1}N & (N+Q)^{-1}E^T \\ U^{-1}E & -U^{-1}R \end{pmatrix} \begin{pmatrix} [(N+Q)^{-1}N]^{i-1}(N+Q)^{-1}h \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} [(N+Q)^{-1}N]^i(N+Q)^{-1}h \\ U^{-1}E(N+Q)^{-1}N[(N+Q)^{-1}N]^{i-2}(N+Q)^{-1}h \end{pmatrix} \\
&= \begin{pmatrix} [(N+Q)^{-1}N]^i(N+Q)^{-1}h \\ 0 \end{pmatrix},
\end{aligned}$$

where in the last step we used (4.6b). This completes the proof. □

From (4.6b) in Theorem 5 it readily follows that the null-space of N is defined as $Z = (N+Q)^{-1}E^T$. Multiplying by $N+Q$ we can express the approximation Q in terms of E^T and the null-space of N.

Corollary 1. *To satisfy* (4.6b) *in Theorem 5, Q must be chosen such that:*

$$E^T = QZ.$$

Remark 5. *In Theorem 6, we considered left preconditioning. However, for right preconditioning the preconditioned matrix is $\mathcal{K}^{\mathrm{N}}\mathcal{M}_P^{-1}$ and the associated Krylov subspace would be*

$$\mathbb{K}_k(\mathcal{K}^{\mathrm{N}}\mathcal{M}_P^{-1}, r) = span\{r,\ \mathcal{K}^{\mathrm{N}}\mathcal{M}_P^{-1}\, r,\ \ldots,\ (\mathcal{K}^{\mathrm{N}}\mathcal{M}_P^{-1})^{k-1}\, r\},$$

where r is the initial residual for a zero initial guess. Using the the conditions (4.6) *one can prove a similar result to Theorem 5, that is:*

$$[\mathcal{K}^{\mathrm{N}}\mathcal{M}_P^{-1}]^i\, r = \begin{pmatrix} [N(N+Q)^{-1}]^i h \\ 0 \end{pmatrix} \quad \text{for } i = 0, 1, \ldots.$$

Thus, it is possible to use CG with left or right preconditioning.

Remark 6. *The matrix Q does not necessarily have to be positive definite, for the conditions in* (4.6) *to be satisfied. For example, if $Q = E^T U^{-1} E$ then X is defined as:*

$$Z = (N + E^T U^{-1} E)^{-1} E^T. \tag{4.9}$$

In [36, Proposition 3.6] the authors show that if Z is defined as in (4.9)*, then $NZ = 0$. Thus, condition* (4.6b) *in Theorem 5 is automatically satisfied.*

We now show that we are not restricted to the block diagonal preconditioner to enable the use of CG.

Proposition 2. *Consider the upper and lower triangular preconditioners:*

$$\widetilde{\mathcal{M}}_{\mathrm{P}} = \begin{pmatrix} N+Q & E^T \\ 0 & U \end{pmatrix} \quad \text{and} \quad \widehat{\mathcal{M}}_{\mathrm{P}} = \begin{pmatrix} N+Q & 0 \\ D & U \end{pmatrix}.$$

Then

$$[(\overset{*}{\mathcal{M}})^{-1}\mathcal{K}^{\mathrm{N}}]^k\, r_0 = \begin{pmatrix} [(N+Q)^{-1}N]^k (N+Q)^{-1} h \\ 0 \end{pmatrix},$$

where $*$ denotes upper or lower triangular preconditioner, holds under the same conditions as in Theorem 5.

The proof follows by induction. The preconditioned matrices are

$$\widetilde{\mathcal{M}}_{\mathrm{P}}^{-1}\mathcal{K}^{\mathrm{N}} = \begin{pmatrix} (N+Q)^{-1}N - (N+Q)^{-1}E^TU^{-1}E & (N+Q)^{-1}E^T \\ U^{-1}E & -U^{-1}R \end{pmatrix};$$
$$\widehat{\mathcal{M}}_{\mathrm{P}}^{-1}\mathcal{K}^{\mathrm{N}} = \begin{pmatrix} (N+Q)^{-1}N & (N+Q)^{-1}E^T \\ U^{-1}E - U^{-1}E(N+Q)^{-1}N & U^{-1}E(N+Q)^{-1}E^T - U^{-1}R \end{pmatrix},$$

and it follows that

$$[(\overset{*}{\mathcal{M}})^{-1}\mathcal{K}^{\mathrm{N}}]^k\, r_0 = \begin{pmatrix} [(N+Q)^{-1}N]^k(N+Q)^{-1}h \\ 0 \end{pmatrix},$$

where $*$ denotes the upper or lower triangular preconditioner.

4.3 Null-space decoupling

Consider the saddle-point form of (4.1) with $R = 0$. Since the null-space of N is of dimension m, there is a matrix $Z \in \mathbb{R}^{n\times m}$ whose columns form a linearly independent basis for the null-space of N. Using this null-space matrix, it is possible to decouple the saddle-point system (4.1) by multiplying the first block row with Z^T to obtain:

$$Z^TB^Tr = Z^Th.$$

The solution procedure for (4.1) becomes:

$$Z^TE^Tr = Z^Th, \tag{4.10a}$$
$$Nb = g - E^Tr \quad \text{with} \quad Eb = 0. \tag{4.10b}$$

The solution of (4.10) is unique if and only if EZ is nonsingular and N is positive definite on the null-space of E.

To aid discussion of the null-space decoupling, we introduce a Helmholtz-type decomposition similar to [36]. The intersection of the null-spaces of N and E are zero, thus:

$$\ker(N) \oplus \ker(E) = \mathbb{R}^n. \tag{4.11}$$

This decomposition provides key insights into the null-space decoupling of (4.1). Using (4.11) we write the primary variable solution

$$b = b_N + b_E$$

where $b_N \in \ker(N)$ and $b_E \in \ker(E)$.

Using [36, Proposition 3.6] and [36, Theorem 3.5], there exists a formulation of the null-space of N such that $U = EZ$ is SPD which allows the use of CG. From the Helmholtz-type decomposition, (4.10b) can be written as:

$$Nb_E = f - E^T r \quad \text{with} \quad Eb_N = 0.$$

Since $b_N \in \ker(N)$ then $Eb_N \neq 0$ unless $b_N = 0$. Thus the decoupled system becomes

$$Z^T E^T r = Z^T h \quad \text{and} \quad Nb_E = h - E^T r. \tag{4.12}$$

Since N is positive definite on the null-space of E, it is possible to use a CG-type solver for the second equation in (4.12). In practice, since N is singular, one may apply a null-space method, which amounts to applying CG on:

$$V^T N V b_E = V^T h \quad \text{where} \quad EV = 0.$$

We note that a null-space matrix V does not need to be explicitly constructed; see [46].

We conclude this section with a few comments for the case when $Z^T h = 0$. In this setting ($\dim(\text{null}(N)) = m$), it is very common that the secondary variables, r, arise from a Lagrange multiplier formulation (4.1); see for example [33, 49] for the mixed formulation of Maxwell's equations and [16] for norm minimization problems with equality constraints. Thus, the construction of h is often independent of the secondary variables which leads to $Z^T h = 0$ for these sorts of formulations. Thus, the solution to (4.10a) is zero and then the solution to (4.12) becomes:

$$Nb_E = h. \tag{4.13}$$

Considering a preconditioner of the form $N + Q$ for (4.13) would form the same block Krylov subspace defined in (4.5) where $r = 0$.

For a non-zero r, the discussion around (4.12) may indicate that the condition (4.6a) might be relaxed for the preconditioned saddle-point system.

4.4 Eigenvalue Analysis

We now analyze the ideal forms of the preconditioner where we do not use the approximation Q but rather the augmented term $E^T U^{-1} E$. We will consider the upper block

triangular preconditioner:

$$\widetilde{\mathcal{M}}_{\mathrm{I}} = \begin{pmatrix} N + E^T U^{-1} E & E^T \\ 0 & U \end{pmatrix}.$$

Extension to block diagonal and block lower triangular can easily be done using the same techniques.

Theorem 6. *The preconditioned matrix $\widetilde{\mathcal{M}}_{\mathrm{I}}^{-1} K$ has eigenvalue $\lambda = 1$ with algebraic multiplicity $n - m$ and eigenvalue $\lambda = \frac{1 \pm \sqrt{5}}{2}$ with algebraic multiplicity $m - l$ for each eigenvalue, where $l = Dim(Null(R))$. The corresponding eigenpairs $(\lambda, \boldsymbol{x})$ are:*

$$\lambda = 1, \quad \boldsymbol{x} = (b_D, 0), \quad \textit{and} \quad \lambda = \frac{1 \pm \sqrt{5}}{2}, \quad \boldsymbol{x} = (b_N, r_R),$$

where $b_D \in Null(D)$, $b_N \in Null(N)$, and $r_R \in Null(R)$.

Proof. The corresponding generalized eigenvalue problem is given by:

$$\begin{pmatrix} N & E^T \\ E & -R \end{pmatrix} \begin{pmatrix} b \\ r \end{pmatrix} = \lambda \begin{pmatrix} N + E^T U^{-1} E & E^T \\ 0 & U \end{pmatrix} \begin{pmatrix} b \\ r \end{pmatrix}. \tag{4.14}$$

Since W and Q are SPD and SPSD, respectively, $\lambda W + Q$ is nonsingular for all $\lambda \neq 0$. Substituting $r = (\lambda W + R)^{-1} D b$ into the first block row of (4.14), we obtain:

$$(1 - \lambda) N b + (1 - \lambda) E^T (\lambda W + R)^{-1} D b = \lambda E^T U^{-1} E b.$$

From this, it follows immediately that if $b \in \mathrm{Null}(D)$ then $\lambda = 1$ is an eigenvalue.

Now, assume that $\lambda \neq 1$ and $Db \neq 0$. Substituting $Db = (\lambda W + R) r$ into (4.14) gives:

$$(\lambda - 1) N b + (\lambda^2 + \lambda - 1) E^T r + \lambda E^T U^{-1} R r = 0.$$

Thus by setting $b \in \mathrm{Null}(N)$, and $r \in \mathrm{Null}(R)$, we have $\lambda = \frac{1 \pm \sqrt{5}}{2}$. □

From Theorem 6 it follows that if $R = 0$, we have the full spectrum of the preconditioned matrix, with just three distinct eigenvalues that have high algebraic multiplicities.

Corollary 2. *For the non-regularized saddle-point form of (4.1), with $R = 0$, $\widetilde{\mathcal{M}}_{\mathrm{I}}^{-1} K$ has eigenvalue $\lambda = 1$ with algebraic multiplicity $n - m$ and eigenvalues $\lambda_{\pm} = \frac{1 \pm \sqrt{5}}{2}$ with algebraic multiplicities m.*

Proposition 3, given below, outlines the algebraic multiplicities of eigenvalues and their corresponding eigenvectors for block diagonal and block lower triangular precondition-

ers. The full eigenvalue analysis for these preconditioners is omitted since it is very similar to what we show in Theorem 6.

Proposition 3. *Consider the preconditioners*

$$\mathcal{M}_{\mathrm{I}} = \begin{pmatrix} N + E^T U^{-1} E & 0 \\ 0 & U \end{pmatrix} \quad \textit{and} \quad \widehat{\mathcal{M}}_{\mathrm{I}} = \begin{pmatrix} N + E^T U^{-1} E & 0 \\ D & U \end{pmatrix}.$$

Then the eigenpairs and algebraic multiplicities of the preconditioned matrices are given in Table 4.1.

Preconditioned matrix	Eigenvalue	Eigenvector	Algebraic multiplicity
$\mathcal{M}_{\mathrm{I}}^{-1}K$	1	(b, r_R)	$n-l$
$\mathcal{M}_{\mathrm{I}}^{-1}K$	-1	(b_N, r_R)	$m-l$
$\widehat{\mathcal{M}}_{\mathrm{I}}^{-1}K$	1	$(b_E, 0)$	$n-m$
$\widehat{\mathcal{M}}_{\mathrm{I}}^{-1}K$	$\frac{1+\sqrt{5}}{2}$	(b_N, r_R)	$m-l$
$\widehat{\mathcal{M}}_{\mathrm{I}}^{-1}K$	$\frac{1-\sqrt{5}}{2}$	(b_N, r_R)	$m-l$

Table 4.1: Eigenpairs and algebraic multiplicities for the preconditioned matrices $\widehat{\mathcal{M}}_{\mathrm{I}}^{-1}K$ and $\mathcal{M}_{\mathrm{I}}^{-1}K$. We use the notation $b_E \in \mathrm{Null}(E)$, $b_N \in \mathrm{Null}(N)$, $r_R \in \mathrm{Null}(R)$, and $l = \mathrm{Dim}(\mathrm{Null}(R))$.

In Figure 6 we show an example of the eigenvalue distribution of the preconditioned matrix.

4.5 Numerical Experiments

In this section we use the time-harmonic Maxwell equation in mixed form [51, 78] to illustrate our findings. The continuous problem is given as follows:

$$\begin{aligned} \nabla \times \nu_m \nabla \times \boldsymbol{b} + \nabla r &= \boldsymbol{g} && \text{in } \Omega \\ \nabla \cdot \boldsymbol{b} &= 0 && \text{in } \Omega, \end{aligned} \tag{4.15}$$

where $\boldsymbol{b}$ is the magnetic field and r is the Lagrange multiplier associated with the divergence constraint on the magnetic field. The constant ν_m is the magnetic viscosity. This model is well understood and gives rise, upon finite element discretization, to a symmetric system; see [51, 72]. It also arises in coupled electromagnetic flows such as magnetohydrodynamics; see [86, 110] and the references therein.

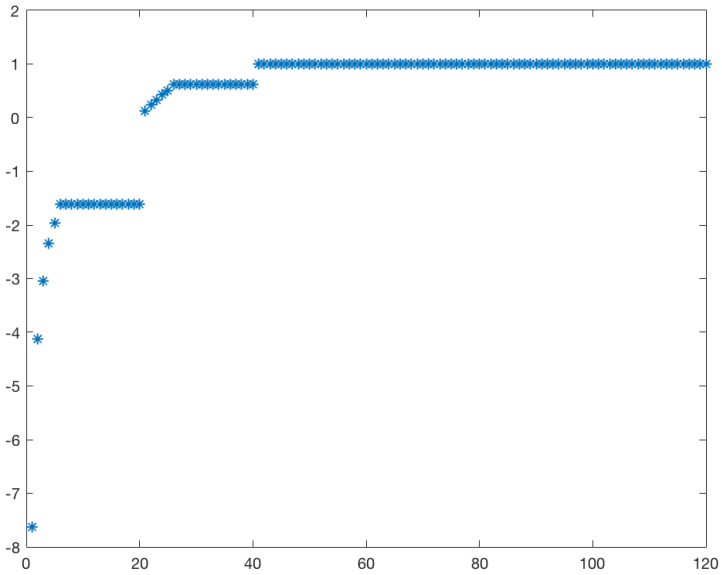


Figure 4.1: Eigenvalue distribution of the preconditioned matrix $\widetilde{\mathcal{M}}_{\mathrm{I}}^{-1}K$ using randomly generated blocks with $n = 100$, $m = 20$, and $l = 5$

We consider a finite element discretization of (4.15) which uses Nédélec elements for the magnetic field and nodal elements for the multiplier variable. See [51, 72] for more details. Upon discretization, N, E and E^T are defined to be the discrete curl-curl, magnetic gradient and magnetic divergence operators, respectively. Thus, the null space of N is the discrete gradient operator and has dimension m. The resulting discretization of (4.15) falls into the class of saddle-point systems with a maximally rank-deficient leading block. We note that a sparse construction of the discrete gradient operator for first order Nédélec elements is possible [72, Section 2].

In [51] it was shown that the vector mass matrix and scalar Laplacian are appropriate choices for Q and U, respectively. Since U is now the scalar Laplacian, use the notation $U = L$. We now show that for these choices, the conditions (4.6a)–(4.6b) in Theorem 5 hold. In [51, Proposition 2.2] the authors prove the identity

$$E^T = QZ,$$

where we recall that E^T and Z are the magnetic divergence and discrete gradient operators, respectively. Thus, by Corollary 1, (4.6b) holds. Typically, the right-hand-side is divergence-free for physical applications; see [49, 110]. Since Z is the null-space

operator of the curl-curl matrix, Z^T is a discrete divergence operator. Hence, the divergence-free condition (4.6a) holds.

Therefore, for the Maxwell problem of the form (4.15) the conditions (4.6a)–(4.6b) hold and enable the use of CG with either a block diagonal, upper or lower block triangular preconditioner. The coefficient matrix and block diagonal preconditioner for the mixed Maxwell problem in (1.5) are:

$$\mathcal{K} = \begin{pmatrix} N & E^T \\ E & 0 \end{pmatrix} \quad \text{and} \quad \mathcal{M} = \begin{pmatrix} N+Q & 0 \\ 0 & L \end{pmatrix},$$

where we recall that N, E, E^T, Q, and L are the curl-curl operator, magnetic divergence operator, a gradient operator, the vector mass matrix, and scalar Laplacian, respectively.

Let us consider a 3D example with a smooth solution on $\Omega = [0,1]^3$. The analytical solution is set to be

$$\begin{aligned} \boldsymbol{b}(x,y,z) &= \begin{pmatrix} -\exp(x+y+z)\sin(y) + \exp(x+y+z)\sin(z) \\ xy\exp(x+y+z) - yz\exp(x+y+z) \\ -\exp(x+y+z)\sin(x) + \exp(x+y+z)\sin(y) \end{pmatrix} \\ r(x,y,z) &= 0. \end{aligned} \tag{4.16}$$

Then the source terms $\boldsymbol{g}$ in (4.15) and inhomogeneous boundary conditions are defined from the analytical solution. By construction, we have ensured that the right-hand-side is divergence-free. We use this setup as a basis for all numerical experiments and outline any alterations (e.g., the domain or multiplier variable) we make for the specific example.

In each experiment we use a tolerance of 1e-6 for the outer flexible CG (FCG) iteration [82]. For the inner solves ($N+Q$ and L), we use the auxiliary space preconditioner of [61]. This entails a CG solve for $N+Q$ and L with a tolerance of 1e-3 for each. In the subsequent tables we use the following notation:

- ℓ: mesh level;
- DoFs: number of degrees of freedom for Magnetic field plus the multiplier variables;
- Time: total time for FCG to converge for the specified tolerance;
- it: number of outer FCG iterations to converge for the specified tolerance;
- it_1: number of inner CG/Auxiliary Space iterations for $N+Q$;

- it_2: number of inner CG/Auxiliary Space iterations for L.

4.5.1 Krylov Subspace Solver Test

The first numerical experiment we consider is to test the robustness and performance of FCG compared to other Krylov subspace methods. We set the inner tolerance to be 1e-4 and test against MINRES [83] and GMRES [94]. The results are shown in Table 4.2. We observe that the iteration counts are similar for all three methods tested, and FCG is slightly superior in terms of computational time. Given that the cost of single iterations is lower for FCG compared to MINRES, and given that the memory consumption of FCG is significantly lower than that of GMRES, we conclude that FCG is the most effective method of the three, although not by a significant margin.

		FCG				MINRES				GMRES			
ℓ	DoFs	Time	it	it_1	it_2	Time	it	it_1	it_2	Time	it	it_1	it_2
3	4913	0.06	14	2.4	1.4	0.07	13	3.0	1.9	0.07	12	3.0	1.9
4	35,937	0.70	14	2.7	1.7	0.72	13	3.6	1.9	0.78	13	3.5	2.1
5	274,625	5.82	13	3.1	2.2	7.42	14	3.9	2.9	7.57	13	3.9	2.8
6	2,146,689	61.78	15	3.3	2.1	60.36	12	4.2	2.8	78.06	14	4.2	2.9
7	16,974,593	580.22	14	3.9	2.5	601.85	13	4.5	2.9	712.28	13	4.9	2.9

Table 4.2: Krylov solver test: time and iteration results for using one iterations of the preconditioner with FCG, MINRES and GMRES. The viscosity for these tests is $\nu_m =$ 1e-2

Let us now consider preconditioned CG with diagonal and upper/lower triangular preconditioners. The results are given in Table 4.3. From Theorem 5 and Proposition 2, we have shown that the Krylov subspace for both the diagonal and block triangular preconditioners are the same. Thus we would expect the number of iterations for FCG to converge to be identical. From the table, we can see that the block diagonal preconditioner preforms slightly better with respect to the outer FCG iterations. We have shown that the preconditioner with the exact augmented term ($Q = E^T U^{-1} E$) has two distinct eigenvalues for the preconditioned matrix, whereas for the block triangular preconditioner we have three distinct eigenvalues for the preconditioned matrix. This may explain the slight difference in performance.

4.5.2 Divergence and Non-Divergence Free Right-Hand-Side

Let us now consider divergence and non-divergence free right-hand-sides. The results are given in Table 4.4. We observe that there is almost no difference in the iteration results

		$\mathcal{M}_P$				$\mathcal{M}_{\mathrm{U}}$				$\mathcal{M}_{\mathrm{L}}$			
ℓ	DoFs	Time	it	it_1	it_2	Time	it	it_1	it_2	Time	it	it_1	it_2
2	729	0.01	12	1.8	1.0	0.01	13	1.7	1.1	0.01	13	1.7	1.1
3	4,913	0.07	14	2.3	1.4	0.05	13	2.2	1.5	0.05	13	2.1	1.6
4	35,937	0.64	14	2.5	1.7	0.57	14	2.5	1.7	0.56	14	2.5	1.7
5	274,625	5.95	14	2.7	1.7	6.11	16	2.6	1.6	5.41	14	2.6	1.8
6	2,146,689	56.92	14	2.9	1.8	60.92	16	2.9	1.7	59.89	16	2.9	1.7
7	16,974,593	574.34	14	3.3	2.3	589.14	16	3.2	2.2	569.86	16	3.2	2.1

Table 4.3: Block preconditioner test: time and iteration results using the block diagonal ($\mathcal{M}_P$), block upper triangular ($\mathcal{M}_{\mathrm{U}}$) and block lower triangular ($\mathcal{M}_{\mathrm{L}}$) preconditioners with ν_m = 1e-2

between the divergence and non-divergence free right-hand-side solution. This shows that even though our analysis requires a divergence-free right-hand side, in practice the solver is robust with respect to right-hand side vectors that violate this condition. We do not have a theoretical justification for this; if the right-hand-side is not divergence free then term (4.7) does not vanish and the construction of the Krylov subspace is significantly more involved. In that case, there is no easy way to discern algebraic structure and proceed with deriving results such as (4.5) in Theorem 5. However, in Section 4.3 we show that it is possible to decouple such saddle point systems into two SPD linear systems without the divergence constraint being satisfied. Therefore, Theorem 5 only provides sufficient conditions and we suspect that the divergence-free condition can be relaxed in practice.

		Divergence Free				Non-divergence Free			
l	DoFs	Time	it	it_1	it_2	Time	it	it_1	it_2
2	729	0.01	12	1.5	0.9	0.01	12	1.5	0.9
3	4,913	0.06	14	2.3	1.4	0.06	14	2.3	1.4
4	35,937	0.64	14	2.5	1.7	0.64	14	2.5	1.7
5	274,625	7.41	14	2.7	1.7	7.41	14	2.7	1.7
6	2,146,689	55.02	14	2.9	1.8	55.02	14	2.9	1.8
7	16,974,593	544.91	14	3.3	2.3	644.91	14	3.3	2.6

Table 4.4: Divergence-free vs. non-divergence-free right-hand-side: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for divergence and non-divergence free right-hand-sides with ν_m =1e-2.

4.5.3 Variable Coefficients

We now let the magnetic viscosity be defined as:

$$\nu_m = \begin{cases} 1/a & \text{if } x < 0.5 \text{ and } y < 0.5 \text{ and } z < 0.5, \\ 1/2a & \text{if } x > 0.5 \text{ and } y < 0.5 \text{ and } z < 0.5, \\ 1/3a & \text{if } x < 0.5 \text{ and } y > 0.5 \text{ and } z < 0.5, \\ 1/4a & \text{if } x > 0.5 \text{ and } y > 0.5 \text{ and } z < 0.5, \\ 1/5a & \text{if } x < 0.5 \text{ and } y < 0.5 \text{ and } z > 0.5, \\ 1/6a & \text{if } x > 0.5 \text{ and } y < 0.5 \text{ and } z > 0.5, \\ 1/7a & \text{if } x < 0.5 \text{ and } y > 0.5 \text{ and } z > 0.5, \\ 1/8a & \text{otherwise,} \end{cases}$$

where a is a constant. The results are shown in Table 4.5 and 4.6. The results in Table 4.5 show iteration and timing results for the block diagonal preconditioner for various values of a. We can see from the table that as a increases then the number of outer FCG iterations increase but the inner Auxiliary Space iterations seem relatively constant, with only a slight increase going through the levels. For $a = 10$ or 100 the outer FCG iterations remain approximately constant but for $a = 1000$ the outer iterations start to degrade slightly. Table 4.6 shows the iteration and timing comparisons between the block diagonal and triangular preconditioners for $a = 100$. Again, we see that the outer iteration FCG iterations appear to be slightly better for the block diagonal preconditioner than the triangular versions. We also note that solve times for the triangular versions are significantly higher, especially for the larger mesh levels. This is due to the extra multiplications with E or E^T for the block triangular preconditioners. Thus, it is beneficial to use the block diagonal preconditioner.

		$a = 10$			$a = 100$			$a = 1000$		
ℓ	DoFs	Time	it	it_1/it_2	Time	it	it_1/it_2	Time	it	it_1/it_2
2	729	0.01	14	1.9/0.9	0.02	32	1.4/0.9	0.06	71	0.9/0.6
3	4,913	0.10	15	2.3/1.5	0.18	38	2.0/1.3	0.36	98	1.4/1.2
4	35,937	0.71	15	2.8/1.7	1.51	39	2.2/1.6	3.23	110	1.8/1.5
5	274,625	6.46	15	3.0/1.9	13.96	40	2.6/1.7	34.18	116	2.2/1.6
6	2,146,689	63.63	15	3.8/1.9	132.51	40	2.9/1.8	328.45	116	2.4/1.8
7	16,974,593	581.03	15	3.9/2.2	1306.93	40	3.3/2.3	3267.59	122	2.7/2.3

Table 4.5: Variable coefficients: time and iteration results using the block diagonal preconditioner, $\mathcal{M}$, for various different values of a.

		$\mathcal{M}$			$\mathcal{M}_{\mathrm{U}}$			$\mathcal{M}_{\mathrm{L}}$		
ℓ	DoFs	Time	it	$\mathrm{it}_1/\mathrm{it}_2$	Time	it	$\mathrm{it}_1/\mathrm{it}_2$	Time	it	$\mathrm{it}_1/\mathrm{it}_2$
2	729	0.02	32	1.4/0.9	0.01	32	1.3/0.8	0.02	34	1.3/0.9
3	4,913	0.18	38	2.0/1.3	0.15	38	1.9/1.3	0.14	38	1.8/1.3
4	35,937	1.51	39	2.2/1.6	1.43	39	2.3/1.6	1.43	40	2.1/1.6
5	274,625	13.96	40	2.6/1.7	14.62	41	2.6/1.7	14.64	41	2.4/1.7
6	2,146,689	132.51	40	2.9/1.8	147.63	41	3.0/1.8	150.59	43	2.9/1.8
7	16,974,593	1306.93	40	3.3/2.3	1557.95	42	3.4/2.4	1572.68	44	3.2/2.3

Table 4.6: Variable coefficients: time and iteration results using the block diagonal and triangular preconditioners for $a = 100$.

4.5.4 Fichera Corner Problem

For our next numerical illustration, we consider the same exact solution in (4.16), however the domain will be a cube missing a corner. That is, the domain is $\Omega = (-1,1)^3/[0,1) \times [0,1) \times [0,1)$ with local refinement in the corner. A visualization of this domain is given in Figure 4.2. We investigate how the magnetic viscosity affects the

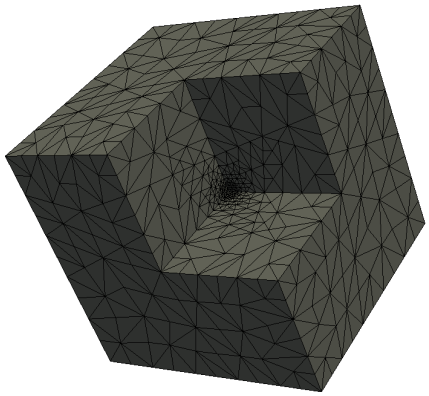


Figure 4.2: Fichera corner domain for mesh level, $\ell = 1$

iterations on such a domain. Table 4.7 presents the results. The table shows that as the magnetic viscosity decreases the number of outer FCG iterations increase. As with the previous example, inner Auxiliary Space iterations appear to be scalable.

4.5.5 Gear Domain

For our final experiment, we consider the same exact solution in (4.16) however with an quasi-uniform 3-dimensional gear as the domain. The domain is bounded in $\Omega =$

		ν_m =1e-1				ν_m =1e-2				ν_m =1e-3			
ℓ	DoFs	Time	it	it_1	it_2	Time	it	it_1	it_2	Time	it	it_1	it_2
1	14,636	0.41	13	3.8	2.0	0.73	31	3.7	2.3	1.69	80	3.9	2.6
2	111,315	3.84	13	4.4	2.5	7.23	31	4.2	3.0	17.01	81	4.5	3.4
3	869,397	45.74	13	5.9	2.7	85.37	32	5.2	3.3	181.89	72	5.0	3.6
4	6,874,601	614.06	15	8.1	3.4	1027.89	31	6.4	3.7	2248.88	69	6.4	4.2

Table 4.7: Fichera corner: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for various different values of ν_m.

$[-1, 1] \times [-1, 1] \times [0, -0.2]$. A visualization of this domain is given in Figure 4.3.

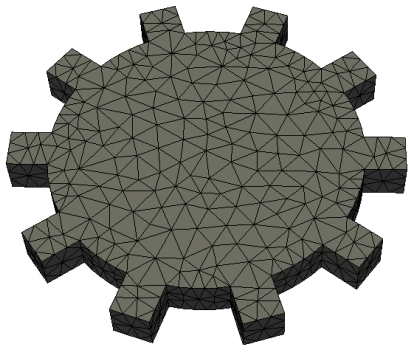


Figure 4.3: Gear domain for mesh level, $\ell = 5$

Table 4.8 shows iteration and timing results for the constant coefficients. From the table we can see that as we decrease ν_m the outer FCG iterations increase slightly but appear to remain constant with respect to the mesh size. For this example, the inner iterations are starting to increase slightly more every mesh level. However, the number of iterations are relatively small and still give a fast solution procedure.

		ν_m =1e-1				ν_m =1e-2				ν_m =1e-3			
ℓ	DoFs	Time	it	it_1	it_2	Time	it	it_1	it_2	Time	it	it_1	it_2
6	16,680	0.23	8	2.2	1.4	0.41	19	2.6	1.5	0.83	42	2.3	1.4
7	106,940	2.62	10	4.0	2.0	4.70	21	4.2	2.5	9.08	44	3.0	2.3
8	774,871	69.85	10	8.8	2.5	77.25	17	5.2	3.2	173.12	46	4.4	3.7
9	5,950,932	1387.91	10	17.2	4.5	1554.78	18	10.2	5.1	3227.96	48	7.8	6.0

Table 4.8: Gear domain: time and iteration results using the block diagonal preconditioner, $\mathcal{M}_P$, for various different values of ν_m.

Chapter 5

Conclusions and future work

5.1 Conclusions

We have derived two block preconditioning approaches for the MHD model (1.1)–(1.2). We have also shown that for a specific class of symmetric saddle-point problems, it is possible to use preconditioned CG as the Krylov subspace solver.

The block triangular preconditioner we derived in Chapter 2 demonstrates good eigenvalue clustering, as shown in the spectral analysis Section 2.3. We show numerical scalability for 2D problems. However for 3D problems, we observe a small increase in iterations per mesh level.

Our second block preconditioning approach is the block approximate inverse preconditioner given in Chapter 3. Using the results in [36], we derive a new inverse formula for the MHD model (1.1)–(1.2). Considering the orders of the discrete operators in the exact inverse, we form an effective sparse approximation. We show that for an ideal case the eigenvalues are very strongly clustered around 1. Developing robust solvers for this problem is a highly challenging task, and we believe that our approach shows promise in terms of its ability to strongly scale and tackle large-scale 3D problems with high Hartman numbers.

The final piece of work considered is the use of preconditioned CG for nonsingular saddle-point problems with a maximally rank deficient leading block. We show that it is possible to use the symmetric block diagonal or nonsymmetric upper/lower block triangular preconditioners given they satisfy two algebraic conditions. Thus, for the mixed Maxwell problem in (1.5) we show that for the well-known preconditioners in [51, 72] it is possible to use preconditioned CG for this indefinite saddle-point system.

5.2 Future work

1. Throughout this thesis we have considered Taylor-Hood elements [101] for velocity/pressure variables and the lowest order Nédélec [81] pair for magnetic/multiplier variables. This is by no means the only choice of finite element discretization for this model. For example, the authors in [49] use the formulation in [95] which

leads to divergence-free Brezzi–Douglas–Marini (BDM) elements [19, 26] for the velocity elements. We leave the exploration of other finite element discretizations for the MHD model as a topic of future work.

2. The numerical examples focus solely on Dirichlet boundary conditions for the MHD model. To incorporate alternative more complex boundary conditioners (Neumann or Robin boundary conditions), it may be necessary to alter the Schur complement approximations in a similar fashion to [15] and [35, Chapter 9] to correctly capture these settings.

3. Unlike block triangular preconditioners, it is possible to solve for each block of the approximate inverse preconditioner independently of the others. In particular, we note that each block column of the approximate inverse preconditioner in (3.12) has some repetition for the sequence of systems which are solved. Therefore, one could solve each block column of (3.12) in parallel. This may greatly reduce the computational cost for the application of the inverse preconditioner to 1 or 2 vector solves per block column, which is comparable to the block triangular preconditioners.

4. During the "sparsification" of the inverse formula for the MHD model in Chapter 3, we drop block matrices based on the mesh order. However, depending on the magnitude of the non-dimensional parameters (ν, ν_m or κ) different "sparsification" may be used. Thus, it would be possible to design a preconditioning approach based on non-dimensional parameters setup for a specific problem in addition to a "black-box" approach.

5. For the Maxwell example in Chapter 4, we solve the preconditioner to a tolerance. It may be possible to reduce the overall computational time by loosening the linear solver's tolerance and applying an inexact solution procedure.

Bibliography

[1] D. J. Acheson. *Elementary fluid dynamics*. Oxford Applied Mathematics and Computing Science Series. The Clarendon Press, Oxford University Press, New York, 1990.

[2] J. H. Adler, T. R. Benson, E. C. Cyr, S. P. MacLachlan, and R. S. Tuminaro. Monolithic multigrid methods for two-dimensional resistive magnetohydrodynamics. *SIAM Journal on Scientific Computing*, 38(1):B1–B24, 2016.

[3] G. Alzetta, D. Arndt, W. Bangerth, V. Boddu, B. Brands, D. Davydov, R. Gassmoeller, T. Heister, L. Heltai, K. Kormann, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. The `deal.II` library, version 9.0. *Journal of Numerical Mathematics*, 2018, accepted.

[4] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[5] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156, 2006.

[6] F. Armero and J. C. Simo. Long-term dissipativity of time-stepping algorithms for an abstract evolution equation with applications to the incompressible MHD and Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 131(1):41–90, 1996.

[7] S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, and H. Zhang. PETSc User's Manual. Technical Report ANL-95/11 - Revision 3.4, Argonne National Laboratory, 2013.

[8] S. Balay, W. D. Gropp, L. Cu. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *"Modern Software Tools in Scientific Computing"*, pages 163–202. Birkhäuser Press, 1997.

[9] S. M. Balay, S. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, and H. Zhang. PETSc Web page. http://www.mcs.anl.gov/petsc, 2014.

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

[11] L. Bergamaschi, M. Ferronato, and G. Gambolati. Mixed Constraint Preconditioners for the iterative solution of FE coupled consolidation equations. *Journal of Computational Physics*, 227(23):9885 – 9897, 2008.

[12] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli. Inexact constraint preconditioners for linear systems arising in interior point methods. *Computational Optimization and Applications*, 36:137–147, 2007.

[13] M. Blatt, A. Burchardt, A. Dedner, C. Engwer, J. Fahlke, B. Flemisch, C. Gersbacher, C. Gräser, F. Gruber, C. Grüninger, D. Kempf, R. Klöfkorn, T. Malkmus, S. Müthing, M. Nolte, M. Piatkowski, and O. Sander. The Distributed and Unified Numerics Environment, Version 2.4. *Archive of Numerical Software*, 4(100):13–29, 2016.

[14] P. B. Bochev, C. J. Garasi, J. J. Hu, A. C. Robinson, and R. S. Tuminaro. An improved algebraic multigrid method for solving Maxwell's equations. *SIAM Journal on Scientific Computing*, 25(2):623–642, 2003.

[15] N. Bootland, A. Bentley, C. Kees, and A. Wathen. Preconditioners for Two-Phase Incompressible Navier-Stokes Flow. *arXiv preprint arXiv:1710.08779*, 2017.

[16] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[17] D. Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, third edition, 2007.

[18] 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):1–17, 1988.

[19] F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Element Methods*. Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[20] L. Chacón. An optimal, parallel, fully implicit newton–krylov solver for three-dimensional viscoresistive magnetohydrodynamics. *Physics of Plasmas*, 15(5):056103, 2008.

[21] L. Chacón. Scalable parallel implicit solvers for 3D magnetohydrodynamics. *Journal of Physics: Conference Series*, 125(1):012041, 2008.

[22] L. Chacón, D. A. Knoll, and J. M. Finn. An Implicit, Nonlinear Reduced Resistive MHD Solver. *Journal of Computational Physics*, 178(1):15 – 36, 2002.

[23] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate. *ACM Transactions on Mathematical Software*, 35(3):22:1–22:14, October 2008.

[24] Z. Chen, Q. Du, and J. Zou. Finite element methods with matching and non-matching meshes for Maxwell equations with discontinuous coefficients. *SIAM Journal on Numerical Analysis*, 37(5):1542–1570, 2000.

[25] Clawpack Development Team. Clawpack software, 2018. Version 5.5.0.

[26] B. Cockburn, G. Kanschat, and D. Schötzau. A note on discontinuous Galerkin divergence-free solutions of the Navier-Stokes equations. *Journal of Scientific Computing*, 31(1-2):61–73, 2007.

[27] R. Cockett, S. Kang, L. J. Heagy, A. Pidlisecky, and D. W. Oldenburg. Simpeg: An open source framework for simulation and gradient based parameter estimation in geophysical applications. *Computers & Geosciences*, 85:142–154, 2015.

[28] M. Costabel and M. Dauge. Singularities of electromagnetic fields in polyhedral domains. *Archive for Rational Mechanics and Analysis*, 151(3):221–276, 2000.

[29] E. C. Cyr, J. N. Shadid, R. S. Tuminaro, R. P. Pawlowski, and L. Chacón. A new approximate block factorization preconditioner for two-dimensional incompressible (reduced) resistive MHD. *SIAM Journal on Scientific Computing*, 35(3):B701–B730, 2013.

[30] L. D. Dalcin, R. R. Paz, P. A. Kler, and A. Cosimo. Parallel distributed computing using python. *Advances in Water Resources*, 34(9):1124 – 1139, 2011. New Computational Methods and Software Tools.

[31] T. A. Davis. A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2):165–195, June 2004.

[32] T. A. Davis and E. Palamadai Natarajan. Algorithm 907: KLU, A Direct Sparse Solver for Circuit Simulation Problems. *ACM Transactions on Mathematical Software*, 37(3):36:1–36:17, September 2010.

[33] L. Demkowicz and L. Vardapetyan. Modelling of electromagnetic absorption/scattering problems using *hp*-adaptive finite elements. *Computer Methods in Applied Mechanics and Engineering*, 152(1):103–124, 1998.

[34] C. Durazzi and V. Ruggiero. Indefinitely preconditioned conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numerical Linear Algebra with Applications*, 10(8):673–688, 2003.

[35] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*. Oxford University Press, second edition, 2014.

[36] R. Estrin and C. Greif. On Nonsingular Saddle-point Systems with a Maximally Rank Deficient Leading Block. *SIAM Journal on Matrix Analysis and Applications*, 36(2):367–384, 2015.

[37] R. Falgout, A. Barker, H. Gahvari, T. Kolev, R. Li, D. Osei-Kuffuor, J. Schroder, P. Vassilevski, L. Wang, and U. M. Yang. *hypre: High Performance Preconditioners*. Lawrence Livermore National Laboratory. `http://www.llnl.gov/CASC/hypre/`.

[38] R. D. Falgout and U. Yang. *hypre*: A library of high performance preconditioners. In *Computational Science — ICCS 2002*, volume 2331 of *Lecture Notes in Computer Science*, pages 632–641. Springer Berlin Heidelberg, 2002.

[39] R. Fletcher. Conjugate gradient methods for indefinite systems. In G.Alistair Watson, editor, *Numerical Analysis*, volume 506 of *Lecture Notes in Mathematics*, pages 73–89. Springer Berlin Heidelberg, 1976.

[40] J. D. Foley. Constructive Solid Geometry. *Computer Graphics: Principles and Practice, Addison-Wesley Professional*, pages 557–558, 1996.

[41] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitian linear systems. *Numerische Mathematik*, 60(3):315–339, 1991.

[42] R. Eymardand T. Gallouët and R. Herbin. Finite volume methods. *Handbook of numerical analysis*, 7:713–1018, 2000.

[43] 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.

[44] J.-F. Gerbeau. A stabilized finite element method for the incompressible magnetohydrodynamic equations. *Numerische Mathematik*, 87(1):83–111, 2000.

[45] J.-F. Gerbeau, C. Le. Bris, and T. Lelièvre. *Mathematical Methods for the Magnetohydrodynamics of Liquid Metals*. Oxford University Press, 2006.

[46] N. Gould, M. Hribar, and J. Nocedal. On the solution of equality constrained quadratic programming problems arising in optimization. *SIAM Journal on Scientific Computing*, 23(4):1376–1395, 2001.

[47] N. Gould, D. Orban, and T. Rees. Projected Krylov methods for saddle-point systems. *SIAM Journal on Matrix Analysis and Applications*, 35(4):1329–1343, 2014.

[48] A. Greenbaum. *Iterative methods for solving linear systems*, volume 17 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[49] C. Greif, D. Li, D. Schötzau, and X. Wei. A mixed finite element method with exactly divergence-free velocities for incompressible magnetohydrodynamics. *Computer Methods in Applied Mechanics and Engineering*, 199(45-48):2840–2855, 2010.

[50] C. Greif and D. Schötzau. Preconditioners for saddle point linear systems with highly singular (1, 1) blocks. *Electronic Transactions on Numerical Analysis, Special Volume on Saddle Point Problems*, 22:114–121, 2006.

[51] C. Greif and D. Schötzau. Preconditioners for the discretized time-harmonic Maxwell equations in mixed form. *Numerical Linear Algebra with Applications*, 14(4):281–297, 2007.

[52] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[53] M. D. Gunzburger, A. J. Meir, and J. S. Peterson. On the existence, uniqueness, and finite element approximation of solutions of the equations of stationary, incompressible magnetohydrodynamics. *Mathematics of Computation*, 56(194):523–563, 1991.

[54] F. Hecht. New development in FreeFem++. *Journal of Numerical Mathematics*, 20(3-4):251–265, 2012.

[55] P. Hénon, P. Ramet, and J. Roman. PaStiX: A Parallel Sparse Direct Solver Based on a Static Scheduling for Mixed 1D/2D Block Distributions. In *Proceedings of the 15 IPDPS 2000 Workshops on Parallel and Distributed Processing*, IPDPS '00, pages 519–527, London, UK, 2000. Springer-Verlag.

[56] M. Heroux, R. Bartlett, V. Howle, R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, and A. Williams. An Overview of Trilinos. Technical Report SAND2003-2927, Sandia National Laboratories, 2003.

[57] M. A. Heroux and J. M. Willenbring. Trilinos Users Guide. Technical Report SAND2003-2952, Sandia National Laboratories, 2003.

[58] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[59] R. Hiptmair. Multigrid method for Maxwell's equations. *SIAM Journal on Numerical Analysis*, 36(1):204–225, 1999.

[60] R. Hiptmair. Finite elements in computational electromagnetism. *Acta Numerica*, 11:237–339, 2002.

[61] R. Hiptmair and J. Xu. Nodal auxiliary space preconditioning in $H(\mathrm{curl})$ and $H(\mathrm{div})$ spaces. *SIAM Journal on Numerical Analysis*, 45(6):2483–2509, 2007.

[62] I. Ipsen. A note on preconditioning nonsymmetric matrices. *SIAM Journal on Scientific Computing*, 23(3):1050–1051, 2001.

[63] J. Jones and B. Lee. A multigrid method for variable coefficient Maxwell's equations. *SIAM Journal on Scientific Computing*, 27(5):1689–1708, 2006.

[64] G. Karypis and K. Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM Journal of Scientific Computing*, 20(1):359–392, December 1998.

[65] B. Kehlet. Mshr - Mesh Generation in FEniCS. Talk at FEniCS'14 workshop in Paris, June 2014.

[66] R. Keppens, G. Tóth, M. A. Botchev, and A. van der Ploeg. Implicit and semi-implicit schemes: algorithms. *International Journal for Numerical Methods in Fluids*, 30(3):335–352, 1999.

[67] T. V. Kolev and P. Vassilevski. Parallel auxiliary space AMG for $H(\mathrm{curl})$ problems. *Journal of Computational Mathematics*, 27(5):604–623, 2009.

[68] J. Korzak. Eigenvalue relations and conditions of matrices arising in linear programming. *Computing. Archives for Scientific Computing*, 62(1):45–54, 1999.

[69] P. Krzyzanowski. On block preconditioners for saddle point problems with singular or indefinite (1, 1) block. *Numerical Linear Algebra with Applications*, 18:123–140, 2011.

[70] R. J. LeVeque. *Finite volume methods for hyperbolic problems*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, 2002.

[71] D. Li. *Numerical Solution of the Time-Harmonic Maxwell equations and Incompressible Magnetohydrodynamics Problems*. PhD thesis, University of British Columbia, 2010.

[72] D. Li, C. Greif, and D. Schötzau. Parallel numerical solution of the time-harmonic Maxwell equations in mixed form. *Numerical Linear Algebra with Applications*, 19(3):525–539, 2012.

[73] X. S. Li. An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Transactions on Mathematical Software*, 31(3):302–325, 2005.

[74] X.S. Li, J. W. Demmel, J. R. Gilbert, L. Grigori, M. Shao, and I. Yamazaki. SuperLU Users' Guide. Technical Report LBNL-44289, Lawrence Berkeley National Laboratory, 1999.

[75] J. Liesen and B. N. Parlett. On nonsymmetric saddle point matrices that allow conjugate gradient iterations. *Numerische Mathematik*, 108(4):605–624, 2008.

[76] P. T. Lin, J. N. Shadid, R. S. Tuminaro, M. Sala, G. L. Hennigan, and R. P. Pawlowski. A parallel fully coupled algebraic multilevel preconditioner applied to multiphysics PDE applications: drift-diffusion, flow/transport/reaction, resistive MHD. *International Journal for Numerical Methods in Fluids*, 64(10-12):1148–1179, 2010.

[77] A. Logg, K. A. Mardal, and G. N. Wells, editors. *Automated solution of differential equations by the finite element method*, volume 84 of *Lecture Notes in Computational Science and Engineering*. Springer, Heidelberg, 2012.

[78] P. Monk. *Finite Element Methods for Maxwell's Equations*. Oxford University Press, 2003.

[79] K. W. Morton and D. F. Mayers. *Numerical solution of partial differential equations*. Cambridge University Press, Cambridge, second edition, 2005. An introduction.

[80] 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):1969–1972, 2000.

[81] J. C. Nédélec. Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 35(3):315–341, 1980.

[82] Y. Notay. Flexible Conjugate Gradients. *SIAM Journal on Scientific Computing*, 22(4):1444–1460, 2000.

[83] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[84] I. Perugia, D. Schötzau, and P. Monk. Stabilized interior penalty methods for the time-harmonic Maxwell equations. *Computer Methods in Applied Mechanics and Engineering*, 191(41-42):4675–4697, 2002.

[85] E. G. Phillips, H. C. Elman, E. C. Cyr, J. N. Shadid, and R. P. Pawlowski. A block preconditioner for an exact penalty formulation for stationary MHD. *SIAM Journal on Scientific Computing*, 36(6):B930–B951, 2014.

[86] E. G. Phillips, H. C. Elman, E. C. Cyr, J. N. Shadid, and R. P. Pawlowski. Block Preconditioners for Stable Mixed Nodal and Edge finite element Representations of Incompressible Resistive MHD. *SIAM Journal on Scientific Computing*, 38(6):B1009–B1031, 2016.

[87] C. E. Powell and D. Silvester. Optimal preconditioning for Raviart-Thomas mixed formulation of second-order elliptic problems. *SIAM Journal on Matrix Analysis and Applications*, 25(3):718–738, 2003.

[88] S. Reitzinger and J. Schöberl. An algebraic multigrid method for finite element discretizations with edge elements. *Numerical Linear Algebra with Applications*, 9(3):223–238, 2002.

[89] M. Rivara. Mesh Refinement Processes Based on the Generalized Bisection of Simplices. *SIAM Journal on Numerical Analysis*, 21(3):604–613, 1984.

[90] P. H. Roberts. *An Introduction to Magnetohydrodynamics*. Longmans, London, 1967.

[91] M. Rozložník and V. Simoncini. Krylov Subspace Methods for Saddle Point Problems with Indefinite Preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 24(2):368–391, 2002.

[92] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[93] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, Second edition, 2003.

[94] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[95] D. Schötzau. Mixed finite element methods for stationary incompressible magneto–hydrodynamics. *Numerische Mathematik*, 96(4):771–800, 2004.

[96] J. N. Shadid, R. P. Pawlowski, J. W. Banks, L. Chacón, P. T. Lin, and R. S. Tuminaro. Towards a scalable fully-implicit fully-coupled resistive MHD formulation with stabilized FE methods. *Journal of Computational Physics*, 229(20):7649–7671, 2010.

[97] J. N. Shadid, R. P. Pawlowski, E. C. Cyr, R. S. Tuminaro, L. Chacón, and P. D. Weber. Scalable implicit incompressible resistive MHD with stabilized FE and fully-coupled Newton-Krylov-AMG. *Computer Methods in Applied Mechanics and Engineering*, 304:1–25, 2016.

[98] H. Si. TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator. *ACM Transactions on Mathematical Software*, 41(2):11:1–11:36, February 2015.

[99] G. Strang and G. Fix. *An analysis of the finite element method*. Wellesley-Cambridge Press, Wellesley, MA, second edition, 2008.

[100] J. C. Strikwerda. *Finite difference schemes and partial differential equations*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 2004.

[101] C. Taylor and P. Hood. A numerical solution of the Navier-Stokes equations using the finite element technique. *Computers and Fluids*, 1(1):73–100, 1973.

[102] The CGAL Project. *CGAL User and Reference Manual*. CGAL Editorial Board, 4.12.1 edition, 2018.

[103] M. ur Rehman, C. Vuik, and G. Segal. SIMPLE-type preconditioners for the Oseen problem. *International Journal for Numerical Methods in Fluids*, 61(4):432–452, 2009.

[104] 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):631–644, 1992.

[105] S. P. Vanka. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *Journal of Computational Physics*, 65(1):138–158, 1986.

[106] L. Vardapetyan and L. Demkowicz. *hp*-adaptive finite elements in electromagnetics. *Computer Methods in Applied Mechanics and Engineering*, 169(3-4):331–344, 1999.

[107] A. J. Wathen. Realistic eigenvalue bounds for the Galerkin mass matrix. *IMA Journal of Numerical Analysis*, 7(4):449–457, 1987.

[108] A. J. Wathen. Preconditioning. *Acta Numerica*, 24:329–376, 2015.

[109] M. Wathen. Iterative Solution of a Mixed Finite Element Discretisation of an Incompressible Magnetohydrodynamics Problem. Master's thesis, University of British Columbia, 2014.

[110] M. Wathen, C. Greif, and D. Schötzau. Preconditioners for Mixed Finite Element Discretizations of Incompressible MHD Equations. *SIAM Journal on Scientific Computing*, 39(6):A2993–A3013, 2017.

[111] X. Wei. *Mixed discontinuous Galerkin finite element methods for incompressible magnetohydrodynamics*. PhD thesis, University of British Columbia, 2011.

[112] S. L. Xiaoye and J. W. Demmel. SuperLU_DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems. *ACM Transactions on Mathematical Software*, 29(2):110–140, June 2003.

[113] J. Xu. The auxiliary space method and optimal multigrid preconditioning techniques for unstructured grids. *Computing*, 56(3):215–235, 1996.