Multilevel and Algebraic Multigrid Methods for the Higher Order Finite Element Analysis of Time Harmonic Maxwell's Equations

Ali Aghabarati

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada
October 2013

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dedicated to My Family

Abstract

The Finite Element Method (FEM) applied to wave scattering and quasi-static vector field problems in the frequency domain leads to sparse, complex-symmetric, linear systems of equations. For large problems with complicated geometries, most of the computer time and memory used by FEM goes to solving the matrix equation. Krylov subspace methods are widely used iterative methods for solving large sparse systems. They depend heavily on preconditioning to accelerate convergence. However, application of conventional preconditioners to the "curl-curl" operator which arises in vector electromagnetics does not result in a satisfactory performance and specialized preconditioning techniques are required.

This thesis presents effective Multilevel and Algebraic Multigrid (AMG) preconditioning techniques for *p*-adaptive FEM analysis. In *p*-adaption, finite elements of different polynomial orders are present in the mesh and the system matrix can be structured into blocks corresponding to the orders of the basis functions. The new preconditioners are based on a *p*-type multilevel Schwarz (*p*MUS) approximate inversion of the block structured system. A V-cycle multilevel correction starts by applying Gauss-Seidel to the highest block level, then the next level down, and so on. On the other side of the V, Gauss-Seidel iterations are applied in the reverse order. At the bottom of the cycle is the lowest order system, which is usually solved exactly with a direct solver. The proposed alternative is to use Auxiliary Space Preconditioning (ASP) at the lowest level and continue the V-cycle downwards, first into a set of auxiliary, node-based spaces, then through a series of progressively smaller matrices generated by an Algebraic Multigrid (AMG). The algebraic coarsening approach is especially useful for problems with fine geometric details, requiring a very large mesh in which the bulk of the elements remain at low order.

In addition, for wave problems, a "shifted Laplace" technique is applied, in which part of the ASP/AMG algorithm uses a perturbed, complex frequency. A significant convergence acceleration is achieved. The performance of Krylov algorithms is further enhanced during *p*-adaption by incorporation of a deflation technique. This projects out from the preconditioned system the eigenvectors corresponding to the smallest eigenvalues. The construction of the deflation subspace is based on efficient estimation of the eigenvectors from information obtained when solving the first problem in a *p*-adaptive sequence.

Extensive numerical experiments have been performed and results are presented for both wave and quasi-static problems. The test cases considered are complicated to solve and the numerical results show the robustness and efficiency of the new preconditioners. Deflated Krylov methods preconditioned with the current Multilevel/ASP/AMG approach are always considerably faster than the reference methods and speedups of up to 10 are achieved for some test problems.

Résumé

La méthode des éléments finis (FEM) appliquée à la dispersion des ondes et aux problèmes de champ de vecteurs quasi-statique dans le domaine fréquentiel mène à des systèmes d'équations linéaires rares, symétriques-complexes. Pour de grands problèmes ayant des géométries complexes, la plupart du temps et de la mémoire d'ordinateur utilisé par FEM va à la résolution de l'équation de la matrice. Les méthodes itératives de Krylov sont celles largement utilisées dans la résolution de grands systèmes creux. Elles dépendent fortement des préconditionnement qui accélèrent la convergence. Toutefois, l'application de préconditionnements conventionnels à l'opérateur "rot-rot" qui surgit en électromagnétisme vectoriel n'aboutit pas à des résultats satisfaisants et des techniques de préconditionnement spécialisés sont exigées.

Cette thèse présente des techniques de préconditionnement efficaces multiniveau et multigrilles algébrique (AMG) pour l'analyse *p*-adaptative FEM. Dans la *p*-adaptation, des éléments finis de différents ordres polynomiaux sont présents dans le maillage et la matrice du système peut être structurée en blocs correspondant aux ordres des fonctions de base. Les nouveaux préconditionneurs sont basés sur un type d'inversion approximative à multiniveau *p* Schwarz (*p*MUS) du système structuré de bloc. Une correction à niveaux multiples en cycle V débute par l'application de Gauss-Seidel au niveau du bloc le plus élevé, suivi par le niveau inférieur, et ainsi de suite. De l'autre côté du V, des itérations de Gauss-Seidel sont appliquées en ordre inverse. Au bas du cycle se trouve le système d'ordre le plus bas, qui est habituellement résolu exactement avec un solveur direct. L'alternative proposée est d'utiliser l'espace auxiliaire de préconditionnement (ASP) au niveau le plus bas et de poursuivre le cycle en V vers le bas, d'abord en un ensemble d'auxiliaires, basé sur les espacements de nœuds, à travers une série de plus en plus petites de matrices générées par un multigrille algébrique (AMG). L'approche de grossissement algébrique est particulièrement utile aux problèmes ayant de fins détails géométriques, nécessitant une très grande maille dans laquelle la majeure partie des éléments restent à un niveau plus bas.

En outre, pour des problèmes d'onde, la technique "décalé Laplace" est appliquée, dans laquelle une partie de l'algorithme ASP/AMG utilise une fréquence complexe

perturbée. Une accélération de la convergence significative est atteinte. La performance des algorithmes de Krylov est davantage renforcée au cours du *p*-adaptation par l'incorporation d'une technique de déflation. Cette saillie fait dépasser hors du système préconditionné, les vecteurs propres correspondants aux plus petites valeurs propres. La construction du sous-espace de déflation est basée sur une estimation efficace des vecteurs propres à partir d'informations obtenues lors de la résolution du premier problème dans une séquence *p*-adaptatif.

Des expériences numériques approfondies ont été effectuées et les résultats sont présentés à la fois aux problèmes d'onde et quasi-statiques. Les cas de test sont considérés comme compliqués à résoudre et les résultats numériques montrent la robustesse et l'efficacité des nouveaux préconditionnements. Les méthodes de Krylov de déflation préconditionnés par l'approche multiniveaux/ASP/AMG actuelle sont toujours considérablement plus rapides que les méthodes de référence et des accélérations allant jusqu'à 10 sont atteintes pour certains problèmes de test.

Acknowledgments

First and foremost, I need to thank my supervisor Prof. Jon P Webb, for his continuous guidance and support throughout my program. His wide knowledge and generous insight gave me significant help during this research work. His pro-active and well-organized attitude in scientific research also set a great example for me. Without his help the completion of this thesis would not have been possible and I wish to express my sincere gratitude.

I am also grateful to my Ph.D. committee members: Prof. Dennis Giannacopoulos and Prof. Roni Khazaka, for their valuable advice, suggestions and the time they devoted to my research work. I also wish to extend my thanks to Prof. Romanus Dyczij-Edlinger, Prof. David Lowther, Prof. Hani Mitri and Prof. Richard Rose for accepting to review this thesis as members of the jury. I am confident that I will benefit a great deal from the comments and suggestions.

During my study at McGill, I had the privilege of working closely with many friends. My sincere appreciation goes to Dr. Hussein Moghnieh, Dr. Ali Bostani, Dr. Maryam Mehri Dehnavi, Maryam Golshayan, Ali Akbarzadeh Sharbaf, Moein Nazari, Evgeny Kirshin, Adrian Ngoly, Tapabrata Mukherjee, Min Li, Jian Wang, Sajid Hussain, Armin Salimi and Rodrigo Pedrosa. I would like to collectively thank them for making the Computational Electromagnetics Laboratory a great working environment.

Last but not least, special thanks to my family and beloved one who provided so much support and encouragement throughout the years. Great thanks must go to my mother for giving me constant love and always praying for me; and my father for all that he has done for me. Thanks to my brothers and sister for the encouragement and support they extended to me throughout my life.

Research Publications

- A. Aghabarati, J. P. Webb, "An algebraic Multigrid Method for the Finite Element Analysis of Large Scattering Problems", *IEEE Transactions on Antennas and Propagation*, vol. 61 no. 02, Feb 2013.
- A. Aghabarati, J. P. Webb, "Multilevel Methods for p-adaptive Finite Element Analysis of Electromagnetic Scattering", *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 11, Nov 2013.
- A. Aghabarati, J. P. Webb, "Multilevel Preconditioning for Time-harmonic Eddy Current Problems Solved with Hierarchical Finite Elements", *IEEE Transactions on Magnetics,* Accepted for publication on Jun 14, 2013.
- A. Aghabarati, J. P. Webb, "Auxiliary Space Preconditioning for Hierarchical Elements", *11th International Conference on Finite Elements for Microwave Engineering – FEM2012*, June 4-6, 2012, Estes Park, Colorado, USA.
- A. Aghabarati, J. P. Webb, "Multilevel Preconditioning for Time-harmonic Eddy Current Problems Solved with Hierarchical Finite Elements", *19th Conference on the Computation of Electromagnetic Field – COMPUMAG,* 30 June - 4 July, 2013, Budapest, Hungary.

Table of Content

Page

List of Symbols

Symbols

$\mathbb{R}^3$	:	Three dimensional Euclidean space
$\mathcal{L}_2(\Omega)$, $\boldsymbol{\mathcal{L}}_2(\Omega)$	:	Spaces of square integrable scalar and vector fields over Ω
$\mathcal{H}^1(\Omega)$, $\boldsymbol{\mathcal{H}}^1(\Omega)$	:	Spaces of scalar and vector fields over Ω
$\boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$	:	Function space $\{\boldsymbol{v} \in \boldsymbol{\mathcal{L}}_2(\Omega), \nabla \times \boldsymbol{v} \in \boldsymbol{\mathcal{L}}_2(\Omega)\}$
$\mathscr{g}$	:	Member of $\mathcal{H}^1(\Omega)$
$\boldsymbol{u}$	:	Member of $\boldsymbol{\mathcal{H}}^1(\Omega)$
$\boldsymbol{v}_{\mathrm{sol}}$, $\boldsymbol{v}$	:	Members of $\boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$
$\boldsymbol{V}^h$	:	FE space for test or trail functions
N	:	FE space of piecewise linear, scalar functions , $N \subset \mathcal{H}^1(\Omega)$)
N^3	:	FE space of piecewise vector nodal functions, $N^3 \subset \boldsymbol{\mathcal{H}}^1(\Omega)$
$\widetilde{\boldsymbol{V}}_p$	:	FE space of $(p-1)^{\text{th}}$ order gradient and p^{th} order rotational basis
$\boldsymbol{V}^p$	:	FE space of p^{th} order polynomial, $\boldsymbol{V}^p \subset \boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$
Φ, $\widetilde{\Phi}$, $\widetilde{\Psi}$	:	Members of space N
$\widetilde{\boldsymbol{\omega}}$, $\widetilde{\mathbf{E}}$	:	Members of space N^3
$\mathbf{v}$, $\mathbf{v}_s$	:	Members of space $\boldsymbol{V}^1$
$\mathbf{W}_i^e$	:	ith basis function defined over element e
E_i^e	:	Coefficient of the ith basis function over element e
$\mathbf{v}$	:	Vector field quantity
$\mathbf{B}$	:	Magnetic flux density (T)
$\mathbf{H}$	:	Magnetic field (A/m)
$\mathbf{E}$	:	Electric field (V/m)
$\mathbf{D}$	:	Electric flux density ($\mathrm{C/m^2}$)
ρ	:	Electric charge density ($\mathrm{C/m^3}$)
$\boldsymbol{\mathcal{J}}$	:	Electric current density ($\mathrm{A/m^2}$)
$\boldsymbol{\mathcal{J}}_c$	:	Impressed currents ($\mathrm{A/m^2}$)
$\boldsymbol{\mathcal{J}}_i$	:	Induced conduction currents ($\mathrm{A/m^2}$)
ϕ	:	Magnetic scalar potential ($\mathrm{A/m^2}$)
$\mathbf{T}$	:	Magnetic vector potential (A/m)
ω	:	Angular frequency of the wave
k_0	:	Wavenumber
λ_0	:	Wavelength
$\hat{\sigma}$	:	Tensor of electric conductivity (S/m)
$\hat{\mu}$	:	Tensor of magnetic permeability (H/m)
μ_0	:	Magnetic permeability of vacuum ($\mu_0 = 4\pi \times 10^{-7}\mathrm{H/m}$)
$\hat{\mu}_r$	:	Tensor of relative magnetic permeability ($\hat{\mu}_r = \hat{\mu}/\mu_0$)
$\hat{\varepsilon}$	:	Tensor of electric permittivity (F/m)
ε_0	:	Electric permittivity of vacuum ($\varepsilon_0 = 8.854187817 \times 10^{-17}$ F/m)
$\hat{\varepsilon}_r$	:	Tensor of relative electric permittivity ($\hat{\varepsilon}_r = \hat{\varepsilon}/\varepsilon_0$)
S_{pp}	:	S parameters for waveguide structure

Ω	:	Bounded open set of $\mathbb{R}^3$
Ω_c	:	Eddy current conducting region of Ω
Ω_o	:	Non-conducting region of Ω
Γ	:	Governing PDE domain boundary ($\partial\Omega$)
Γ_D	:	Dirichlet boundary of domain Ω
Γ_N	:	Neumann boundary of domain Ω
Γ_o	:	Robin-type boundary of domain Ω
$\Gamma^{(q)}$	:	Port boundary of domain Ω
$\mathbf{a}_n$	:	Outward unit normal on boundary Γ
$\mathcal{T}^h$	:	Tetrahedral mesh discretization
$\mathcal{T}^0, \mathcal{T}^{-1}, \ldots$	:	(virtual) FE grids
h	:	Grid size
h_{min}	:	Minimum edge length of $\mathcal{T}^h$
h_{avg}	:	Average edge length of $\mathcal{T}^h$
h_{max}	:	Maximum edge length of $\mathcal{T}^h$
n_n^f, n_n^c	:	number of nodes in the original (fine) and virtual (coarse) meshes
n_e^f	:	Number of edges in the original mesh
n_t^f	:	Number of tetrahedra in the original mesh
(x_i, y_i, z_i)	:	Point Coordinates of node i
ξ_i	:	Barycentric co-ordinates
A, b	:	FE matrix and corresponding right-hand side
n	:	Number of rows in matrix A
m	:	Number of non-zeros in matrix A
$\mathcal{A}^p$	:	Block structured FE matrix corresponding to highest level of p
$\mathcal{A}_i^{-q}$	:	Matrix operators at level $-q$ for nodal system i
A_{mn}	:	Sub-matrix of block structure matrix $\mathcal{A}^p$ in m^{th} partition of rows and n^{th} partition of columns
$\mathcal{S}^1$	:	Stiffness matrix for the first order FE modeling
$\mathcal{M}^1$	:	Mass matrix for the first order FE modeling
$\mathcal{C}^1$	:	Matrix related to the contribution of boundaries in the first order FE modeling
$\mathcal{D}^1$	:	Diagonal part of $\mathcal{A}^1$
$\mathcal{L}^1$	:	Strict lower triangular part of $\mathcal{A}^1$
$A_{TT}, A_{\Omega\Omega}, A_{T\Omega}, A_{\Omega T}$	:	Sub-matrices of partitioned FE matrix at lowest order in $\mathbf{T}-\Omega$ method
M	:	Preconditioning Matrix for A
$\mathcal{M}^p$	:	Multilevel Schwarz preconditioner (two level) for $\mathcal{A}^p$
$R(A)$	:	Relaxation operator for matrix A
$R_f(A)$	:	Backward Gauss-Seidel relaxation operator for matrix A
$R_b(A)$	:	Forward Gauss-Seidel relaxation operator for matrix A
$R_{bf}(A)$	:	SSOR relaxation operator for matrix A
$\mathcal{B}_a^1, \mathcal{B}_v^1, \mathcal{B}_w^1$	:	Additive, multiplicative V-cycle and multiplicative W-cycle approximate inversions for matrix $\mathcal{A}^1$

$\mathcal{B}_i^0$	:	Approximate inversion for matrix $\mathcal{A}_i^0$
P^l	:	Prolongation mapping operator connecting levels l and $l-1$
P_i^{-q}	:	Prolongation mapping operator connecting levels $-q+1$ and $-q$ for nodal system i
G	:	Discrete gradient matrix
$[\Pi_x \quad \Pi_y \quad \Pi_z]$	:	Discrete Nedelec interpolation operator
Π_n, Π_Ω	:	Discrete mapping operator
$\mathbb{C}^n$	:	Space of n complex numbers
x^p	:	Block structured unknown vector corresponding to highest level of p
x_p	:	The vector for p^{th} partition of x^p
n^p	:	Dimensions of vector x^p
n_p	:	Dimensions of vector x_p
(μ_i, v_i)	:	Pair of eigenvalue-eigenvector
γ	:	Multigrid Complex shift coefficient
$\mathcal{K}_m$	:	Krylov subspace of dimension m
V_m	:	Subspace of Lanczos vectors
W, W^s	:	Krylov deflation subspaces
ε_K	:	Krylov convergence threshold
k	:	Size of deflation subspace

Abbreviations

3D	:	3-dimensional
AMG	:	Algebraic Multigrid
MG	:	Multigrid
CEM	:	Computational Electromagnetics
COCG	:	Conjugate Orthogonal Conjugate Gradient
CO-MINRES	:	Conjugate Orthogonal Minimal Residual
DOF	:	Degree Of Freedom
FEM	:	Finite Element Method
GMG	:	Geometric Multigrid
GMRES	:	Generalized Minimal Residual
HPD	:	Hermitian Positive Definite
IC	:	Incomplete Cholesky
PDE	:	Partial Deferential Equation
*p*MUS	:	*p*-type Multilevel Schwarz
SLP	:	Shifted Laplace Preconditioner

Operators

grad, ∇	:	Gradient
curl	:	Curl
div	:	Divergence
$\cdot$	:	Scalar product
$\times$	:	Vector product
∂	:	Boundary operator
$\| \quad \|_K$	:	Norm on the domain K
$\oplus$	:	Direct sum

List of Figures

Figure **Page**

List of Tables

Table **Page**

CHAPTER 1

Introduction

1.1 Background and Motivation

In many branches of physics and engineering, numerical simulations are used to study complex phenomena, either to gain insight or as part of a design process. Computational Electromagnetics (CEM) is an important addition to practical experiments and analytical descriptions. With the increases in computer power and improved algorithms of the last decades, the role of CEM in understanding the behaviour of electromagnetic fields in complex structures has become more significant.

The basis for the mathematical analysis and numerical treatment of electromagnetic phenomena is Maxwell's Partial Differential Equations (PDEs). The Finite Element Method (FEM) has been established as a powerful tool for the numerical solution of PDEs. In this method, based on the variational formulation of partial differential equations, the computational domain is discretized into finite elements, and the solution is approximated in a function space with finite dimension.

The original problem can then be transformed into a finite dimensional problem, and the solution can be obtained by solving a matrix equation.

With the introduction of the Tangential Vector Finite Element Method (TVFEM) [1], the FEM became a standard numerical modeling approach for 3D, vector electromagnetic fields in a variety of applications, from power frequencies to microwaves, and beyond. The FEM formulation for Maxwell's "curl-curl" PDE provides great flexibility in dealing with geometrical complexity, varying material coefficients and boundary conditions.

In spite of the great achievements for developing efficient CEM tools around FEM, analysis of modern real-life problems can still face great challenges. One main difficulty comes from the application of conventional matrix methods in FEM modeling of electrically large and geometrically complex components. Large-scale simulations impose great challenges for conventional matrix solvers, because of the high computational resources required. Consequently, it is of great interest to develop more efficient matrix methods for FEM simulation of vector electromagnetic fields.

1.2 Problem Statement and Outline

When the materials are linear and the sources time-harmonic, phasor analysis is possible. Discretization of the "curl-curl" PDE then gives rise to a linear matrix equation in which the matrix is sparse and complex-symmetric (for reciprocal materials). For complicated and large problems, commonly a very large number of finite elements is needed to achieve the required accuracy and resolution, especially for the simulation of wave problems. Therefore, the matrix problems to be solved in the simulations are generally very large, and as a consequence expensive to solve. If a matrix is large, alternatives to direct solution methods are needed, because the storage and speed become limiting factors. Iterative methods for solving linear systems use a computationally cheap process to find increasingly better approximations of a system's solution, and are widely used instead of direct methods.

One popular class of iterative methods builds a subspace, called a Krylov subspace, spanned by increasing powers of the matrix applied to a certain vector. The solution is extracted from the Krylov subspace. Improving the efficiency of Krylov

subspace methods decreases the costs of simulations and is therefore an important research area.

The robustness and efficiency of these methods can be increased using preconditioning techniques. Preconditioning is a way to change the linear system so that the solution remains the same but it is easier to solve iteratively. Development of preconditioners is therefore a very active research area. However, for the curl-curl problem, finding appropriate preconditioning techniques is difficult, for reasons given in Chapter 3. The main difficulty stems from the non-trivial, large kernel of the curl-operator and the existence of low-energy nearly irrotational fields in the curl-curl problem. This can lead to a very ill-conditioned system of equations that yields poor convergence with standard preconditioners.

In this work, we are concerned with efficient numerical solution of linear systems that arise from high-order FE analysis of wave scattering and quasi-static magnetic fields, in the frequency domain. Hierarchical high-order elements are popular for modeling electromagnetic problems, due to their error convergence rates and their block matrix structure [2] [3]. In this context, effective Multilevel/Algebraic Multigrid preconditioners are proposed for hierarchical systems. The algorithm combines correction ideas from *p*-type multiplicative Schwarz with algebraic multigrid for the lowest order system. The chosen problem is solved over different levels of representation and separate corrections in each level are computed and then connected appropriately to result in a cost-effective overall approximation. In particular, the lowest order correction and corresponding space splitting respects the properties of the "curl-curl" operator.

Another specific approach for increasing the efficiency of Krylov subspace methods is deflation acceleration. In recent years, this method has been researched extensively in combination with the Krylov iterative methods [4] [5]. Deflation has been shown to be very effective for problems with multiple right-hand sides. In this thesis, deflation is instead applied to the sequence of growing problems that arise in the *p*-adaptive solution of electromagnetic problems. To asses this option, the effects of deflation on convergence behaviour and overall speed of systems are investigated. Several test problems are considered in combination with popular Krylov methods.

The structure of this report is as follows. There follows an overview of present direct and iterative solution methods. Chapter 2 contains the finite element formulations and origin of the linear system of equations when simulating wave scattering and quasi-static magnetic field problems, together with a description of higher order FE modeling. This is followed by Chapter 3, describing the multilevel and algebraic multigrid preconditioning approaches. In Chapter 4, iterative Krylov method and some theoretical aspects of deflation are presented. In Chapter 5, the results for numerical experiments with several test cases can be found. Chapter 6 gives the main conclusions and recommendations.

1.3 Overview of Present Solution Methods

Discretization of the time-harmonic vector wave equation by the finite element (FE) method results in the linear system

$$Ax = b, \qquad A \in \mathbb{C}^{n \times n} \tag{1.1}$$

where n is the number of unknowns. The matrix A is sparse, symmetric ($A = A^T$) and, in general, complex. Furthermore, depending on the frequency and the boundary conditions, matrix A can be indefinite, meaning that the real part of the eigenvalues of A lie in both positive and negative halves of the complex plane. In this section some methods for solving the linear system are briefly discussed. There are in general two broad classes of methods to solve a linear system: direct and iterative.

1.3.1 Direct vs. Iterative Methods.

Direct methods are basically derived from the Gaussian elimination process. They are well-known for their robustness in solving general problems. However, Gaussian elimination can be unfavourable for sparse linear systems. During the elimination process, zero elements in the structure of the sparse matrix may be filled by non-zero values. This is called fill-in and gives rise to two complications. The first is extra memory required to store the additional non-zero entries, and the second is extra computational work during the elimination process.

During the past few decades, several packages for efficient direct solution of sparse linear systems have been introduced. The primary methods used are: left/right

looking, frontal and multifrontal. The ordering strategies that are usually applied for exploiting the sparsity are: minimum degree and its variants, nested one-way dissection, permutation to block triangular form and profile/bandwidth reduction [6].

UMFPACK is a set of highly efficient routines for solving sparse linear equations. It is based on the Unsymmetric-pattern MultiFrontal method that first finds a column pre-ordering for reducing the fill-in. It scales and analyzes the matrix, and then automatically selects one of two strategies for pre-ordering the rows and columns: unsymmetric and symmetric. Once the strategy is selected, the factorization of the matrix is broken down into the factorization of a sequence of dense rectangular frontal matrices. The frontal matrices are related to each other by a supernodal column elimination tree, in which each node in the tree represents one frontal matrix [7] [8].

For two dimensional FE problems the work and storage required for direct methods grows moderately with n as the number of elements grows, when appropriate reordering strategies are applied to exploit the sparsity. In efficient cases, the work and storage required are $O(n^{3/2})$ and $O(nlogn)$, respectively [9].

On the other hand, for three dimensional FE problems, the work and storage required by direct methods grow much faster, which makes them less attractive. For multifrontal methods, with a nested dissection reordering strategy, the work and storage are $O(n^2)$ and $O(n^{4/3})$, respectively [9]. Therefore, for very large problems in the three dimensional case, iterative methods are usually more efficient.

1.3.2 Iterative Methods and Preconditioning

Iterative approaches in general and Krylov subspace methods in particular are powerful techniques for solving large problems. The solution of the linear system is obtained in a series of steps consisting primarily of matrix-vector multiplication, starting from a given initial solution. A matrix-vector multiplication is a relatively cheap process, requiring only $O(n)$ arithmetic operations per iteration. If the method converges after a small enough number of iterations, it is very efficient.

Krylov methods are widely used for solving linear systems of equations. Examples are the Conjugate Gradient (CG) method [10], the Minimal Residual method (MINRES) [11] and the Generalized Minimal Residual (GMRES) method [12]. The principles of Krylov methods are described in Chapter 4. They

require little storage and for well-conditioned problems, converge in a relatively small number of iterations compared to n.

The vector wave ("curl-curl") equation, on the other hand, is known as a problem for which iterative methods typically result in an extremely slow convergence [13] [14]. In Chapter 3, the main reasons related to the slow convergence of the system are explained. However, with a proper remedy, i.e., a good preconditioner, an efficient iterative method can be designed.

The convergence of Krylov iterative methods is closely related to the condition number of the matrix A. The convergence then can be accelerated by incorporating appropriate preconditioners. The purpose of preconditioning is to improve the condition of the coefficient matrix. Suppose that we have a matrix M whose inverse is easily computed or approximated. Instead of $Ax = b$, we solve iteratively the following equivalent, preconditioned system:

$$M^{-1}Ax = M^{-1}b \tag{1.2}$$

A preconditioned Krylov subspace method can therefore be defined by adding the action of M^{-1} to the original algorithm.

One important aspect of solving (1.2) is that the convergence rate depends on the condition number of $M^{-1}A$, and not on that of A. Therefore, in order to have an improved convergence the preconditioned system must have a smaller condition number than A. In general, M should be chosen such that $M^{-1}A$ is close to the identity.

1.3.3 Review of Preconditioning Methods

Among popular preconditioners, here we mention some of the most widely used. Note that the methods described in this section may also be used in a standalone iteration for solving (1.1), but the rate at which the solution converges depends greatly on the spectrum of the matrix A. Indeed, standalone application of these methods may even fail to converge. They are most appropriate when combined with a Krylov solver to transform the coefficient matrix into one with a more favourable spectrum.

a) Incomplete LU Decomposition

One frequently used preconditioner for A can be obtained by approximately decomposing A into LU factors [15], where L and U are lower and upper triangular matrices, respectively. This is achieved by applying an incomplete LU (ILU) factorization to A. The degree of approximation depends on the number of fill-in elements allowed in the L and U factors.

In general, two different approaches are proposed, one in which fill-in is only allowed in predetermined locations, and another in which the values of the matrix entries are considered.

The simplest method is $ILU(0)$, in which a nonzero entry of L and U is only allowed where the original matrix A has a nonzero. This simple approach allows for a very efficient implementation [15]. A more accurate approximation can be obtained by increasing the level of fill-in. In one method which is structure oriented, more off-diagonals are added to the L and U factors. It is denoted as $ILU(level)$, which $level > 0$ is an integer indicating the number of off-diagonals allowed in the L and U factors. For many problems, the size of the elements decreases with the level number and in practice the number of levels is kept low. This method results in a structured matrix for L and U.

The second approach, which is value oriented, is to define a drop tolerance for the fill-in. If during an LU factorization the value of an element falls below a prescribed small tolerance, say eps, this element is set to zero. This incomplete LU decomposition is usually denoted as $ILU(eps)$. More detailed discussion on incomplete LU preconditioners can be found in [15].

Recently, the algebraic multilevel inverse-based ILU preconditioner was proposed for solving indefinite sparse FEM linear systems [16]. By employing a graph partition technique, this method reorders the original FEM matrix into a hierarchical multilevel structure. An inverse-based ILU dropping strategy is adopted to construct a robust preconditioner for indefinite systems.

b) Basic Relaxation Methods

Fixed-point iterations are among the basic iterative methods for solving linear systems. They can also be used as a simple preconditioning approach. Well-known relaxation methods are based on the matrix splitting :

$$A = D + L + L^T \tag{1.3}$$

in which D is the diagonal of A , L its strict lower part, and L^T its strict upper part. After substituting (1.3) into (1.1), we have

$$(D + L + L^T)x = b \tag{1.4}$$

Different iteration methods can be distinguished by the way the above splitting is reformulated. The forward Gauss-Seidel method can be defined as:

$$(D + L)x_{i+1} = -L^T x_i + b \tag{1.5}$$

where x_i and x_{i+1} are the solution vectors at iteration i and $i + 1$, respectively. In (1.5), a triangular system must be solved, since $D + L$ is the lower triangular part of A.

$$x_{i+1} = -(D + L)^{-1}L^T x_i + (D + L)^{-1}b \tag{1.6}$$

The above Gauss-Seidel sweep is called *forward Gauss-Seidel*, since the entries of the approximate solution x_{i+1} are obtained in forward manner, starting at the beginning of the vector. Similarly, a backward Gauss-Seidel iteration can be defined as

$$x_{i+1} = -(D + L^T)^{-1}Lx_i + (D + L^T)^{-1}b \tag{1.7}$$

Applying only one iteration of the forward or backward relaxations with zero initial guess corresponds to the following preconditioning matrix M

$$M^{-1} = R_f(A) \triangleq (D + L)^{-1} \tag{1.8}$$

and

$$M^{-1} = R_b(A) \triangleq (D + L^T)^{-1} \tag{1.9}$$

Overrelaxation is based on the splitting

$$\beta A = (D + \beta L) + (\beta L^T - (1 - \beta)D) \tag{1.10}$$

where β is a real number, $0 < \beta < 2$, and the corresponding forward Successive Over Relaxation *(SOR)* method is given by the recursion

$$(D + \beta L)x_{i+1} = (-\beta L^T + (1 - \beta)D)x_i + \beta b \tag{1.11}$$

A symmetric SOR, known as SSOR, consists of a forward SOR step followed by a backward SOR step. Choosing the relaxation parameter β to be 1, it reduces to

$$x_{i+1} = (D + L)^{-1}L^T x_i + (D + L^T)^{-1}D(D + L)^{-1}b \tag{1.12}$$

The application of SSOR method as a preconditioner can be described by the following matrix:

$$M^{-1} = R_{bf}(A) \triangleq (D + L^T)^{-1}D(D + L)^{-1} \tag{1.13}$$

c) Multigrid Methods

Multigrid and multilevel methods are well-established approaches for solving linear systems and their robustness is proven when applied to elliptic equations with self-adjoint operators [17]. The robustness of multigrid methods for solving the linear system comes from efficient interplay of two steps: smoothing and coarse level correction.

The classical relaxation methods described in the previous section have strong damping, or smoothing, effects on the high frequency parts of the error, i.e., the errors that correspond to large eigenvalues and which, for FE problems, are usually rapidly varying in space. The error which is not efficiently reduced by a smoothing operator can then be approximated by a coarser system. The basic principle of multigrid is to project the remaining errors after smoothing to a coarse system. The projected error may contain again high frequency components with respect to the coarser system. A multilevel scheme can be used to continue this process to coarser and coarser grids. On the coarsest grid a direct solver is usually applied.

If a hierarchy of nested grids is available, multigrid methods are most effective. The method is then known as geometric multigrid (GMG) [18] [19] [20], and it exhibits fast convergence independent of grid size h [21]. On the other hand, there are situations in which GMG can not be easily applied, e.g., when the FE mesh discretization provides no hierarchy of nested meshes, or when the coarsest mesh is too big for a direct solver to solve. Algebraic multigrid methods (AMGs) were developed to overcome the limitations of GMG and are not constrained in this way [21] [22] [23] [24] [25] [26]. AMG operates more on the level of the matrix than

of the underlying nested FE meshes. The coarsening process is based on mapping operators which are obtained in a purely algebraic way. We restrict ourselves to Algebraic Multigrid (AMG); GMG is beyond the scope of this thesis.

d) Domain Decomposition

For very large and complicated problems, one major difficulty in solving the problem as one large computational domain comes from the multiscale nature of the geometry [33]. The coexistence of electrically large and electrically small fine features can result in an ill-conditioned matrix equation. In this case, the decomposition of the large problem into smaller ones might make its solution feasible and also facilitate computation on a parallel architecture. Domain Decomposition is a class of methods based on this idea that makes large-scale computations possible and also takes advantage of parallel machine architectures [30]. It is inherently parallel, an important consideration in keeping with current trends in computer architecture.

In domain decomposition methods (DDMs), the computational domain is decomposed into overlapping or nonoverlapping subdomains. The PDE is then discretized on each subdomain. At the interfaces between adjacent subdomains, proper boundary conditions called *transmission conditions* are imposed to enforce the continuity of the electromagnetic fields. After that, the basic idea in DDM is to find the solution by solving each domain, and then exchange the solutions in the interface between neighbouring domains. The transmission conditions are imposed iteratively and the subdomains communicate with each other until a certain accuracy in the entire solution has been achieved.

DDMs have good parallelization properties since the domain problems can be solved separately in parallel. Several classes of DDM are proposed for analyzing electromagnetic problems based on overlapping and non-overlapping methods [27] [28] [30] [31]. Some of the early work on DDMs for solving the vector wave equation are due to Lee *et al.* [29] [30], using nonoverlapping domains. Memory requirement can be greatly reduced for problems with repetitions or symmetries. The non-overlapping and non-conforming DDM was introduced in [30] through the introduction of additional surface unknowns at the interface, and extended in [31] [32] [33] for further computational efficiency. Meshes at the interfaces are allowed to be non-matching, leading to considerable efficiency in mesh generation for

complex geometries. The wellposedness is ensured by incorporating the consistency condition in the form of the second order transition condition (complex Robin condition) at the interfaces.

When the transmission conditions are properly devised, DDM can become an effective preconditioner, especially at the initial stage of reducing the residual. The convergence of the algorithm might exhibit slow-down behaviour for small residuals or may deteriorate with the increase of the problem size or the number of subdomains [34]. Recently, a global plane wave deflation technique is utilized to derive a global coarse grid preconditioner and overcome some of the convergence issues for rectangular subdomains [34].

CHAPTER 2

The Finite Element Method for Maxwell's Equations

We begin our discussion in this chapter with an introduction to the Maxwell's partial differential equations (PDEs) in Section 2.1 and derivation of the second order vector wave equation in Section 2.2. The type of boundary conditions considered are explained in Section 2.3. An introduction to relevant function spaces used in the FEM is presented in Section 2.4. The process of replacing the Maxwell's PDE by its discrete formulation and different finite element formulations considered in this thesis can be found in Section 2.5. The chapter ends with a discussion of higher order FEM modeling and adaptivity for obtaining better accuracy and efficiency.

2.1 Maxwell's Equations

The behaviour of electromagnetic fields is governed by Maxwell's equations. These equations consist of pairs of coupled PDEs for the electric and magnetic fields that uniquely define them. When the fields are harmonically oscillating in time with a single frequency, the equations are reduced to their time-harmonic forms:

$$\nabla \times \mathbf{E} = -j\omega\mathbf{B} \tag{2.1}$$

$$\nabla \times \mathbf{H} = j\omega\mathbf{D} + \boldsymbol{\mathcal{J}} \tag{2.2}$$

$$\nabla \cdot \mathbf{D} = \rho \tag{2.3}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{2.4}$$

where $\mathbf{E} = (E_x, E_y, E_z)$ and $\mathbf{H} = (H_x, H_y, H_z)$ stand for the phasor electric and magnetic fields and $\mathbf{D} = (D_x, D_y, D_z)$ and $\mathbf{B} = (B_x, B_y, B_z)$ are the phasor electric and magnetic flux densities. Furthermore, $\boldsymbol{\mathcal{J}}$ denotes the electric current density, which is the summation of the current impressed by an external source $\boldsymbol{\mathcal{J}}_i$ and induced conduction currents $\boldsymbol{\mathcal{J}}_c = \hat{\sigma}\mathbf{E}$ inside the conductor regions, i.e., $\boldsymbol{\mathcal{J}} = \boldsymbol{\mathcal{J}}_c + \boldsymbol{\mathcal{J}}_i$. The volume charge density is denoted by ρ in (2.3).

The constitutive relations for linear and anisotropic media are:

$$\mathbf{B} = \hat{\mu}\mathbf{H} \tag{2.5}$$

$$\mathbf{D} = \hat{\varepsilon}\mathbf{E} \tag{2.6}$$

$$\boldsymbol{\mathcal{J}}_c = \hat{\sigma}\mathbf{E} \tag{2.7}$$

where the dielectric permittivity $\hat{\varepsilon} = \varepsilon_0\hat{\varepsilon}_r$, the conductivity $\hat{\sigma}$, and the magnetic permeability $\hat{\mu} = \mu_0\hat{\mu}_r$ are assumed to be known symmetric tensors.

2.2 Second Order PDE: the Vector Wave Equation

Maxwell's equations can be formulated into a second order PDE for the **E** or **H** fields by using (2.1) to (2.4) and the constitutive relations (2.5) to (2.7). The derived vector wave equation is often referred to as the curl-curl equation.

Starting from equations (2.1) and (2.2) and substituting equations (2.5) through (2.7) and then taking curl of both sides give

$$\nabla \times \hat{\mu}^{-1}\nabla \times \mathbf{E} = -j\omega\nabla \times \mathbf{H} \tag{2.8}$$

$$\nabla \times \hat{\varepsilon}^{-1}\nabla \times \mathbf{H} = j\omega\nabla \times \mathbf{E} + \nabla \times (\hat{\varepsilon}^{-1}\hat{\sigma}\mathbf{E} + \hat{\varepsilon}^{-1}\boldsymbol{\mathcal{J}}_i) \tag{2.9}$$

The conductivity $\hat{\sigma}$ is often included in the complex permittivity parameter $\hat{\varepsilon}$**,** but for the results of this section we avoid this inclusion. The substitution for $\nabla \times \mathbf{H}$ in (2.8) and $\nabla \times \mathbf{E}$ in (2.9) from equations (2.2) and (2.1), respectively, leads to

$$\nabla \times \hat{\mu}^{-1} \nabla \times \mathbf{E} - \omega^2 \hat{\varepsilon} \mathbf{E} + j\omega \hat{\sigma} \mathbf{E} = -j\omega \boldsymbol{J}_i \tag{2.10}$$

$$\nabla \times \hat{\varepsilon}^{-1} \nabla \times \mathbf{H} - \omega^2 \hat{\mu} \mathbf{H} - \nabla \times \hat{\varepsilon}^{-1} \hat{\sigma} \mathbf{E} = \nabla \times (\hat{\varepsilon}^{-1} \boldsymbol{J}_i) \tag{2.11}$$

For nonconductive regions, $\hat{\sigma}$ is zero and equations (2.10) and (2.11) reduce to

$$\nabla \times \hat{\mu}^{-1} \nabla \times \mathbf{E} - \omega^2 \hat{\varepsilon} \mathbf{E} = -j\omega \boldsymbol{J}_i \tag{2.12}$$

$$\nabla \times \hat{\varepsilon}^{-1} \nabla \times \mathbf{H} - \omega^2 \hat{\mu} \mathbf{H} = \nabla \times (\hat{\varepsilon}^{-1} \boldsymbol{J}_i) \tag{2.13}$$

For source-free problems, $\boldsymbol{J}_i$ is also zero and the wave equations can be written as

$$\nabla \times \hat{\mu}^{-1} \nabla \times \mathbf{E} - \omega^2 \hat{\varepsilon} \mathbf{E} = \mathbf{0} \tag{2.14}$$

$$\nabla \times \hat{\varepsilon}^{-1} \nabla \times \mathbf{H} - \omega^2 \hat{\mu} \mathbf{H} = \mathbf{0} \tag{2.15}$$

Equations (2.14) and (2.15) can be cast as a general equation of the form

$$\nabla \times \hat{p}^{-1} \nabla \times \mathbf{v} - k_0^2 \hat{q} \mathbf{v} = \mathbf{0} \tag{2.16}$$

where $\mathbf{v}$ is a general field denoting either $\mathbf{E}$ or $\mathbf{H}$. In the above, the wave number is defined as $k_0 = \omega\sqrt{\varepsilon_0 \mu_0} = 2\pi/\lambda_0$, where λ_0 is the free-space wavelength; and $\hat{p}$ and $\hat{q}$ represent the relative material tensors which are different in each case and are summarized as follow:

- $\mathbf{v}$ stands for $\mathbf{E}$ when $\hat{p} = \hat{\mu}_r$ and $\hat{q} = \hat{\varepsilon}_r$.
- $\mathbf{v}$ stands for $\mathbf{H}$ when $\hat{p} = \hat{\varepsilon}_r$ and $\hat{q} = \hat{\mu}_r$.

2.3 Boundary Conditions

In this section several frequently used boundary conditions for the normal or the tangential components of the electromagnetic fields are presented. The boundary conditions usually depend on the specific application. Here, the focus is on the standard boundary conditions for both $\mathbf{E}$ and $\mathbf{H}$ fields: Dirichlet and Neumann. As well, two specific boundary conditions for the $\mathbf{E}$ field are introduced: impedance and waveguide port constraints.

2.3.1 Dirichlet Boundary Condition

According to this boundary condition, the tangential part of the field on the surface is constrained to a defined value. Constraining the tangential part of field $\mathbf{E}$ to

zero represents a Perfect Electric Conductor (PEC), i.e., we assume $\mathbf{E}_t = 0$ on PEC. The PEC condition is in particular suitable for modeling metallic domains. For the magnetic field, forcing the tangential component to zero ($\mathbf{H}_t = 0$) represents a Perfect Magnetic Conductor (PMC). PMC can model materials with very high permeability, where one can assume a vanishing tangential magnetic field. A surface with a Dirichlet condition is denoted Γ_D.

2.3.2 Neumann Boundary Condition

For a surface Γ_N on which the Neumann boundary condition holds, the tangential part of the curl of the field is zero. The condition is same as Perfect Magnetic Conductors (PMC) for the electric field ($\mathbf{a}_n \times (\nabla \times \mathbf{E}) = 0$) and PEC for the magnetic field ($\mathbf{a}_n \times (\nabla \times \mathbf{H}) = 0$).

2.3.3 Impedance Boundary Condition for the Electric Field

Reflection of the electric field at an interface can be modeled by impedance boundary conditions. These are Robin-type boundary conditions relating the tangential magnetic and electric fields by specifying a surface impedance parameter. When the impedance is chosen to be the intrinsic impedance of free space, it can be interpreted as a simple Absorbing Boundary Condition (ABC) [35]. It is particularly suitable for defining a free-space scattering problem surrounded by a truncation surface Γ_o. If there is an incident wave whose electric and magnetic fields are $\mathbf{E}^i$ and $\mathbf{H}^i$ respectively, the following condition is applied to the enclosing surface Γ_o:

$$\frac{j}{k_0\eta_0}(\nabla \times \mathbf{E})_t = -\frac{1}{\eta_0}\mathbf{E} \times \mathbf{a}_n + \mathbf{H}_t^i + \frac{1}{\eta_0}\mathbf{E}^i \times \mathbf{a}_n \tag{2.20}$$

where η_0 is the intrinsic impedance of free space and $\mathbf{a}_n$ is the unit normal outward from the domain. The quantity on the left-hand side is proportional to the tangential part of the magnetic field and the first term on the right-hand side is proportional to the tangential electric field.

2.3.4 Port Boundary Condition for the Electric Field

This is the boundary condition for the electric field at a port of an N port microwave junction, where it connects to a uniform waveguide or transmission line.

The excitation takes the form of a wave in mode k of the waveguide, incident at port p. Let the transverse electric and magnetic fields of mode l incident at port q be $\mathbf{e}_l^{(q)}$, $\mathbf{h}_l^{(q)}$, respectively. The boundary condition on each port surface $\Gamma^{(q)}$ is [36]:

$$\frac{j}{k_0\eta_0}\hat{\mu}_r^{-1}(\nabla\times\mathbf{E})_t = -\sum_{l=1}^{\infty} V_l^{(q)}(\mathbf{E})\,\mathbf{h}_l^{(q)} + 2\delta_{pq}\mathbf{h}_k^{(p)} \tag{2.21}$$

Here $V_l^{(q)}$ is the normalized voltage of mode l, a linear functional of the transverse electric field at the port:

$$V_l^{(q)}(\mathbf{E}) = \int_{\Gamma^{(q)}} \mathbf{h}_l^{(q)} \times \mathbf{E}\cdot\mathbf{a}_n dS \tag{2.22}$$

2.4 Function Spaces

Here the function spaces for the variational formulation of the partial differential equations under consideration, $\mathcal{H}^1(\Omega)$ and $\boldsymbol{\mathcal{H}}(\mathrm{curl};\Omega)$, are introduced. These spaces will play an important role in finite dimensional approximations of the Maxwell's equation PDEs with the gradient and curl operators, as well as design and analysis of preconditioners presented later. Assuming that the Maxwell problems (2.14) and (2.15) are posed on a fixed three-dimensional domain $\Omega \subset \mathbb{R}^3$, we define:

$$\mathcal{L}_2(\Omega) := \{u, \int_\Omega |u|^2 dx < \infty\} \tag{2.23}$$

to be the space of complex-valued, square integrable scalar fields over Ω. We also denote $\boldsymbol{\mathcal{L}}_2(\Omega) = [\mathcal{L}_2(\Omega)]^3$ to be the space of square integrable vector fields over Ω. In addition:

$$\mathcal{H}^1(\Omega) := \{\varphi \in \mathcal{L}_2(\Omega), \nabla\varphi \in \mathcal{L}_2(\Omega)\} \tag{2.24}$$

$$\boldsymbol{\mathcal{H}}^1(\Omega) := [\mathcal{H}^1(\Omega)]^3 \tag{2.25}$$

$$\boldsymbol{\mathcal{H}}(\mathrm{curl};\Omega) := \{\boldsymbol{v} \in \boldsymbol{\mathcal{L}}_2(\Omega), \nabla\times\boldsymbol{v} \in \boldsymbol{\mathcal{L}}_2(\Omega)\} \tag{2.26}$$

The above Hilbert spaces are equipped with corresponding inner products. For (2.24), we define

$$(\varphi,\psi)_1 = \int_\Omega \nabla\varphi\cdot\nabla\psi^H dx + (\varphi,\psi)_0 \tag{2.27}$$

in which

$$(u, v)_0 = \int_\Omega u v^H dx \tag{2.28}$$

In these relations, the complex-conjugate transpose of v is denoted by v^H. We also need to define the inner product for $\boldsymbol{\mathcal{H}}(\mathrm{curl}; \Omega)$ in (2.26) as:

$$(\boldsymbol{u}, \boldsymbol{v})_{\mathrm{curl}} = \int_\Omega \nabla \times \boldsymbol{u} \cdot \nabla \times \boldsymbol{v}^H dx + (\boldsymbol{u}, \boldsymbol{v})_0 \tag{2.29}$$

where

$$(\boldsymbol{u}, \boldsymbol{v})_0 = \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v}^H dx \tag{2.30}$$

The norms induced by (2.28), (2.27) and (2.29) are denoted by $\|\cdot\|_0$, $\|\cdot\|_1$, $\|\cdot\|_{\mathrm{curl}}$, respectively.

In order to construct conforming finite element methods for finding the electric field by solving (2.14), the discrete space is chosen as a subspace of

$$\boldsymbol{\mathcal{H}}_0(\mathrm{curl}; \Omega) := \{\boldsymbol{u} \in \boldsymbol{\mathcal{L}}_2(\Omega), \nabla \times \boldsymbol{u} \in \boldsymbol{\mathcal{L}}_2(\Omega), (\mathbf{a}_\mathrm{n} \times \boldsymbol{u})|_{\Gamma_{PEC}} = 0\} \tag{2.31}$$

Conforming finite element spaces can be constructed by requiring the tangential components are continuous across element interfaces. This ensures that the resulting global finite element functions are in $\boldsymbol{\mathcal{H}}(\mathrm{curl}; \Omega)$. Note that the normal components of functions need not be continuous.

2.5 Finite Element Formulations

In this section, the finite element formulations used for solving wave scattering problems and quasi-static magnetic field problems are explained in detail.

2.5.1 Finite Element E formulation for the Wave Equation

Figure 2.1 shows the domain Ω of a scattering problem with various types of boundary condition, as introduced in Section 2.3. In the framework of FEM, the infinite-dimensional space $\boldsymbol{\mathcal{H}}_0(\mathrm{curl}; \Omega)$ is replaced by a finite-dimensional subspace, which yields the discrete variational problem. In order to apply the FEM, the domain is discretized, and the field is represented by basis functions defined on the individual elements. A tetrahedral mesh of the domain Ω, with tetrahedra of maximum edge length h, is denoted $\mathcal{T}^h$.

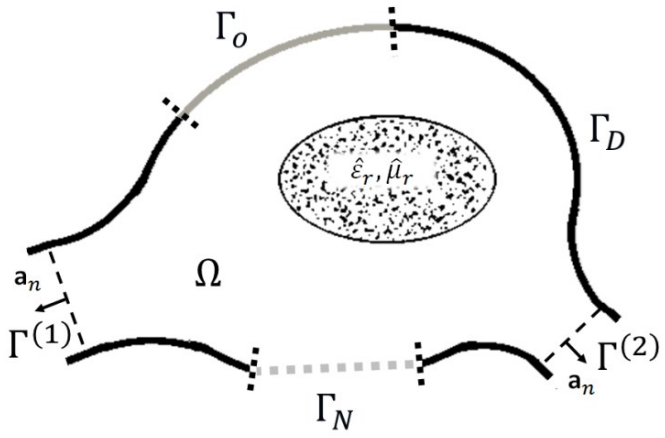


Figure 2.1: Illustration of a typical electric field problem.

Assuming that the space spanned by the basis functions is $\boldsymbol{V}^h$, the Galerkin projection method then provides a general technique for construction of the discrete approximations to the variational problem. The finite-dimensional weak statement corresponding to (2.14) is [36]:

$$\text{Find } \mathbf{E} \in \boldsymbol{V}^h \subset \boldsymbol{\mathcal{H}}_0(\text{curl};\Omega): \quad a(\mathbf{E},\mathbf{w}) = b(\mathbf{w}) \qquad \forall \mathbf{w} \in \boldsymbol{V}^h \tag{2.32}$$

where

$$\begin{aligned} a(\mathbf{E},\mathbf{w}) = \frac{1}{k_0\eta_0}\int_\Omega (\nabla\times\mathbf{E}.\hat{\mu}_r^{-1}\nabla\times\mathbf{w} - k_0^2\mathbf{E}\cdot\hat{\varepsilon}_r\mathbf{w})dV \\ + \jmath\sum_{q=1}^{\mathrm{N}}\sum_{l=1}^{\infty} V_l^{(q)}(\mathbf{E})\,V_l^{(q)}(\mathbf{w}) + \frac{\jmath}{\eta_0}\int_{\Gamma_o}\mathbf{E}_t\cdot\mathbf{w}_t dS \end{aligned} \tag{2.33}$$

and

$$b(\mathbf{w}) = 2\jmath V_k^{(p)}(\mathbf{w}) + \jmath\int_{\Gamma_o}\left(\mathbf{H}^i + \frac{1}{\eta_0}\mathbf{E}^i\times\mathbf{a}_\mathrm{n}\right)\times\mathbf{w}\cdot\mathbf{a}_\mathrm{n}\, dS \tag{2.34}$$

Assuming that the domain is discretized into n_t^f tetrahedral elements, within each element e the electric field vector can be expanded as

$$\mathbf{E}^e = \sum_{i=1}^{n_e}\mathbf{W}_i^e E_i^e \tag{2.35}$$

where n_e is the number of basis functions defined on the element, $\mathbf{W}_i \in \boldsymbol{V}^h$ is the ith basis function and E_i is the coefficient of the ith basis function, numbered globally from 1 to n. Superscript e indicates that the quantities are numbered locally, within element e. Choosing $\mathbf{w}$ to be each basis function $\mathbf{W}_i$ in turn, (2.32) yields the system of equations

$$Ax = b \tag{2.36}$$

where A is an $n \times n$ complex-symmetric matrix and x is a column vector of the unknown values E_i.

2.5.2 Finite Element T − Ω formulation for the Quasi-static Magnetic Field

A typical quasi-static magnetic field problem is depicted in Figure 2.2. It consists of an eddy current region Ω_c with nonzero conductivity and a surrounding non-conducting region Ω_0. The entire problem domain is the union of Ω_c and Ω_0, i.e., $\Omega = \Omega_c \cup \Omega_0$. The red arrow in the figure indicates a prescribed net current flow in a coil, which is part of Ω_c.

Over the surfaces of Ω, different boundary conditions of practical importance can be applied to the normal component of flux density or tangential component of magnetic field intensity. Specifically, $\mathbf{B}_n = 0$ and $\mathbf{H}_t = 0$ are applied on virtual boundaries and symmetry planes, respectively. Over the boundary of Ω_c the interface condition between the conducting and non-conducting regions is imposed.

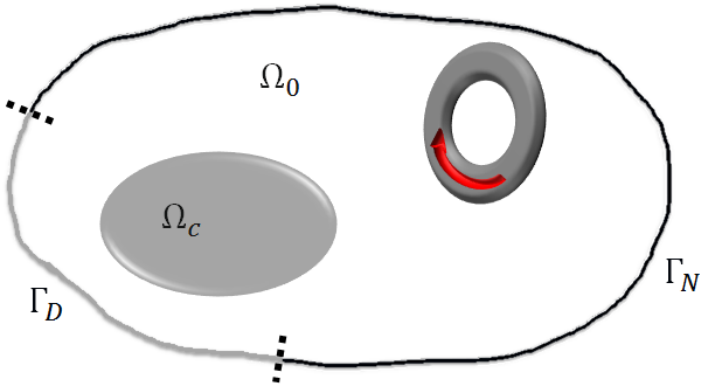


Figure 2.2: Illustration of a typical quasi-static magnetic field problem.

For the conducting region, Maxwell's equations can be written as:

$$\nabla \times \mathbf{E} = -j\omega\hat{\mu}\mathbf{H} \tag{2.37}$$

$$\nabla \times \mathbf{H} = j\omega\hat{\varepsilon}\mathbf{E} + \hat{\sigma}\mathbf{E} \tag{2.38}$$

For the quasi-static problems in which $\omega\hat{\varepsilon} \ll \hat{\sigma}$, equation (2.37) reduces to

$$\nabla \times \mathbf{H} = \hat{\sigma}\mathbf{E} \tag{2.39}$$

From that we get **E** as follows:

$$\mathbf{E} = \hat{\sigma}^{-1}\nabla \times \mathbf{H} \tag{2.40}$$

The substitution for **E** in (2.37) from (2.40) leads to

$$\nabla \times \hat{\sigma}^{-1}\nabla \times \mathbf{H} = -j\omega\hat{\mu}\mathbf{H} \tag{2.41}$$

Therefore, the phasor magnetic field, **H**, solves the following quasi-static equations in conducting and non-conducting regions, respectively:

$$\nabla \times \sigma^{-1}\nabla \times \mathbf{H} + j\omega\hat{\mu}\mathbf{H} = 0 \tag{2.42}$$

$$\nabla \cdot \hat{\mu}\mathbf{H} = 0 \tag{2.43}$$

Various formulations of eddy current problems in terms of scalar or vector potentials have been proposed [37] [38] [39] [40]. Edge elements can be used to implement the $\mathbf{T} - \Omega$ method [40]. A substantial reduction in computational effort can be achieved by using a magnetic scalar potential, ϕ, in the region free of eddy currents. In conducting regions, **H** is represented as $\mathbf{T} - \nabla\phi$, where **T** is an unknown vector potential. First-order edge elements, also called Whitney elements, have one degree of freedom associated with each edge of the tetrahedron. For gauging the decomposition, a tree from the graph of all the edges in the conductors is extracted. **T** is represented in each element, e, by the Whitney functions corresponding to the element edges that are part of the *cotree*:

$$\mathbf{T}^e = \sum_{i\ cotree} \mathbf{W}_i^e T_i^e \tag{2.44}$$

Two different means of exciting the field are used in this work. In one case, ϕ is set to a nonzero value on some part of the boundary of Ω_0. The other method is applied when there is a prescribed net current circulating in a coil, as shown in Figure

2.2. The coil itself is part of Ω_c. To make sure that the path integral of $\mathbf{H}$ is correct around a loop in the air that encloses the coil current, a "cut" surface is introduced in Ω_0, spanning the coil. The potential ϕ is made to increase by the appropriate amount in crossing the cut surface. This jump in potential drives the field in the problem.

Using scalar and vector basis functions from an FE space $\boldsymbol{V}^h$ and following the standard Galerkin method, the differential equations governing $\mathbf{H}$, are reduced to an algebraic form

$$Ax = b \tag{2.45}$$

2.6 Higher Order Elements, Accuracy and Efficiency

In the FEM, first-order elements are widely used. However, to improve the accuracy and efficiency, a higher order approximation of the field may be desired. This can be achieved by employing higher order basis functions. The idea of *p*-version finite element methods is to use a fixed triangulation and obtain better accuracy by increasing the polynomial order of the basis. Compared to the more common first-order elements, higher convergence rates with less numerical dispersion can be obtained using higher order degrees of freedom [3]. The idea has become more and more popular during recent years and numerous bases for the curl -conforming spaces have been presented in the literature [41] [42] [3]. Bases can be classified into two families, *interpolatory* and *hierarchical.*

Using interpolatory basis functions [41], the order of the bases is uniform within the computational domain and the unknown associated with each basis function is typically the field at a point associated with the basis function.

On the other hand, hierarchical basis functions [2] [43] [44][45] [3] allow the use of different orders within the same computational domain. This property can be utilized to adaptively increase the order of the basis functions only in the regions where they result in greatest improvement in the accuracy. Hierarchical bases are popular for other reasons too, such as efficient error estimation and the simplicity of creating multilevel preconditioners to accelerate iterative algorithms [44] [47] [3].

2.6.1 Hierarchical Elements

The basis functions employed in this thesis are the ones proposed in [3] for tetrahedral elements. The FE space is constructed to provide a good representation of the field and its curl.

If p denotes the complete polynomial order of elements, the optimum choice of modeling is to use incomplete orders by removing *gradient* degrees of freedom and keeping the field and its curl complete to order $p-1$. In [3], the basis is provided by taking the gradient of conforming scalar functions and then extending the basis to the full polynomial space. The basis set is divided into gradients of scalar functions (shown here by symbol $\mathcal{G}$) and vector-valued rotational functions, with non-zero curl (shown here by symbol $\mathcal{W}$). The representation of basis functions in terms of barycentric co-ordinates ($\xi_1, \xi_2, \xi_3, \xi_4$) is given in Table 2.1 and Table 2.2.

The advantage of the basis functions in Table 2.2 is that the inexact Helmholtz decomposition is already fulfilled for the higher order vector basis functions [46]. However, the Whitney space $\widetilde{\boldsymbol{V}}_1$ is not decomposed: it contains within it the gradient space of order zero.

The basis function space for the element of order p used in this thesis is defined by

$$\boldsymbol{V}^p = \widetilde{\boldsymbol{V}}_1 \oplus \cdots \oplus \widetilde{\boldsymbol{V}}_p, \qquad p = 1,2,3 \ldots \tag{2.46}$$

Table 2.1: Scalar basis functions

p	Space	Basis	Associated with
1	$\mathcal{V}_1^i$	ξ_i	node i
2	$\mathcal{V}_2^{ij}$	$\xi_i\xi_j$	edge ij
3	$\mathcal{V}_3^{ij}$	$\xi_i\xi_j(\xi_i-\xi_j)$	edge ij
	$\mathcal{V}_3^{ijk}$	$\xi_i\mathcal{V}_2^{ik}$	face ijk
4	$\mathcal{V}_4^{ij}$	$\xi_i\xi_j(\xi_i^2-3\xi_i\xi_j+\xi_j^2)$	face ijk
	$\mathcal{V}_4^{ijk}$	$\xi_i\mathcal{V}_3^{jk}$	face ijk
	$\mathcal{V}_4^{jki}$	$\xi_j\mathcal{V}_3^{ki}$	face ijk
	$\mathcal{V}_4^{ijkl}$	$\xi_i\mathcal{V}_3^{jkl}$	volume $ijkl$

Table 2.2: Vector basis functions

$\widetilde{\boldsymbol{V}}_p$	Space	Basis	Associated with
$\widetilde{\boldsymbol{V}}_1$	$\mathcal{W}_1^{ijk}$	$\xi_i\nabla\xi_j-\xi_j\nabla\xi_i$	edge ij
$\widetilde{\boldsymbol{V}}_2$	$\mathcal{G}_1^{ij}$	$\nabla(\mathcal{V}_2^{ij})$	edge ij
	$\mathcal{W}_2^{ijk}$	$3\mathcal{V}_2^{jk}\nabla\xi_i-\nabla(\mathcal{V}_2^{jk}\xi_i)$	face ijk
	$\mathcal{W}_2^{jki}$	$3\mathcal{V}_2^{ki}\nabla\xi_j-\nabla(\mathcal{V}_2^{ki}\xi_j)$	face jki
$\widetilde{\boldsymbol{V}}_3$	$\mathcal{G}_2^{ij}$	$\nabla(\mathcal{V}_3^{ij})$	edge ij
	$\mathcal{G}_2^{ijk}$	$\nabla(\mathcal{V}_3^{ijk})$	face ijk
	$\mathcal{W}_2^{ijk}$	$4\mathcal{V}_3^{jk}\nabla\xi_i-\nabla(\mathcal{V}_3^{jk}\xi_i)$	face ijk
	$\mathcal{W}_3^{jki}$	$4\mathcal{V}_3^{ki}\nabla\xi_j-\nabla(\mathcal{V}_3^{ki}\xi_j)$	face ijk
	$\mathcal{W}_3^{kij}$	$4\mathcal{V}_3^{ij}\nabla\xi_k-\nabla(\mathcal{V}_3^{ij}\xi_k)$	face ijk
	$\mathcal{W}_3^{ijkl}$	$4\mathcal{V}_3^{jkl}\nabla\xi_i-\nabla(\mathcal{V}_3^{jkl}\xi_i)$	volume $ijkl$
	$\mathcal{W}_3^{jkli}$	$4\mathcal{V}_3^{kli}\nabla\xi_j-\nabla(\mathcal{V}_3^{kli}\xi_j)$	volume $ijkl$
	$\mathcal{W}_3^{klij}$	$4\mathcal{V}_3^{lij}\nabla\xi_k-\nabla(\mathcal{V}_3^{lij}\xi_k)$	volume $ijkl$
$\widetilde{\boldsymbol{V}}_4$	$\mathcal{G}_3^{ij}$	$\nabla(\mathcal{V}_4^{ij})$	edge ij
	$\mathcal{G}_3^{ijk}$	$\nabla(\mathcal{V}_4^{ijk})$	face ijk
	$\mathcal{G}_3^{jki}$	$\nabla(\mathcal{V}_4^{jki})$	face ijk
	$\mathcal{W}_4^{ijk,1}$	$5\mathcal{V}_4^{jk}\nabla\xi_i-\nabla(\mathcal{V}_4^{jk}\xi_i)$	face ijk
	$\mathcal{W}_4^{jki,1}$	$5\mathcal{V}_4^{ki}\nabla\xi_j-\nabla(\mathcal{V}_4^{ki}\xi_j)$	face ijk
	$\mathcal{W}_4^{kij,1}$	$5\mathcal{V}_4^{ij}\nabla\xi_k-\nabla(\mathcal{V}_4^{ij}\xi_k)$	face ijk
	$\mathcal{W}_4^{ijk,2}$	$(6\xi_i-\xi_j-\xi_k)\mathcal{V}_3^{jk}\nabla\xi_i$ $+(6\xi_j-\xi_k-\xi_i)\mathcal{V}_3^{ki}\nabla\xi_j$ $+(6\xi_k-\xi_i-\xi_j)\mathcal{V}_3^{ij}\nabla\xi_k$	face ijk
	$\mathcal{G}_3^{ijkl}$	$\nabla(\mathcal{V}_4^{ijkl})$	volume $ijkl$
	$\mathcal{W}_4^{ijkl}$	$5\mathcal{V}_4^{jkl}\nabla\xi_i-\nabla(\mathcal{V}_4^{jkl}\xi_i)$	volume $ijkl$
	$\mathcal{W}_4^{jkli}$	$5\mathcal{V}_4^{kli}\nabla\xi_j-\nabla(\mathcal{V}_4^{kli}\xi_j)$	volume $ijkl$
	$\mathcal{W}_4^{klij}$	$5\mathcal{V}_4^{lij}\nabla\xi_k-\nabla(\mathcal{V}_4^{lij}\xi_k)$	volume $ijkl$
	$\mathcal{W}_4^{lijk}$	$5\mathcal{V}_4^{ijk}\nabla\xi_l-\nabla(\mathcal{V}_4^{ijk}\xi_l)$	volume $ijkl$
	$\mathcal{W}_4^{iljk}$	$5\mathcal{V}_4^{jkl}\nabla\xi_i-\nabla(\mathcal{V}_4^{jkl}\xi_i)$	volume $ijkl$
	$\mathcal{W}_4^{jikl}$	$5\mathcal{V}_4^{jik}\nabla\xi_j-\nabla(\mathcal{V}_4^{jik}\xi_j)$	volume $ijkl$
	$\mathcal{W}_4^{kjli}$	$5\mathcal{V}_4^{kjl}\nabla\xi_k-\nabla(\mathcal{V}_4^{kjl}\xi_k)$	volume $ijkl$
	$\mathcal{W}_4^{lkij}$	$5\mathcal{V}_4^{lki}\nabla\xi_l-\nabla(\mathcal{V}_4^{lki}\xi_l)$	volume $ijkl$

2.6.2 Adaptivity

The usual process of finite element analysis starts from the generation of a mesh and element orders. Some experience is required to determine the appropriate mesh and orders to achieve the required accuracy. Another approach is costly generation of a second solution on a finer mesh and then comparison of the two solutions.

Adaptive procedures on the other hand try to automatically refine a mesh or adjust the orders in an optimal fashion to achieve a solution having a certain accuracy. In adaptive finite element methods, the computation typically begins with solving the problem at a low level of representation. The error of this solution is then evaluated and if it fails to satisfy a prescribed value, adjustments are made. The general goal is to obtain the desired accuracy with minimal computational effort. Adaptive finite element methods have been studied for many years [48] [49] [50] [51] and common strategies can be classified in to two main categories:

- Local refinement of the mesh by splitting the elements, know as *h-refinement.*
- Locally adding to the polynomial degree of the basis functions, know as *p-refinement.*

The *h-refinement* approach is a popular way of increasing the convergence rate, particularly when singularities are present [49] [3].With *p-refinement* strategy, the mesh is not changed but the order of the finite element basis is increased over the computation domain. As with *h-refinement*, continuity of the field at element boundaries must be ensured. *p-refinement* can be powerful with exponential convergence rates when solutions are smooth [52] [3].The approach is most useful with hierarchical bases, since they permit mixing the element orders freely within the mesh, without violating continuity requirements of the field. When increasing the polynomial degree of the basis, portions of the stiffness and mass matrices and also the right hand side vector will remain unchanged. The refinement strategies may be also applied in combination, resulting in *hp-refinement*, where both the element size h and the order of the method p are varied. Studies on *hp-refinement* can be found in [53] [54] [55].

In order to guide the adaption process, *a posteriori* error estimation is usually employed, i.e., estimation of the error in each element *after* the solution has been

obtained. In this thesis, the following error estimator is used. Suppose that the weak form of the problem to be solved is:

$$a(\mathbf{E}, \boldsymbol{w}) = b(\boldsymbol{w}) \tag{2.47}$$

The error estimate for element e is an estimate of the magnitude of $|b(\mathbf{E}_e) - b(\mathbf{E})|$, where $\mathbf{E}$ is the existing FE solution and $\mathbf{E}_e$ is the new field solution when the order of element e is increased by 1. The field $\mathbf{E}_e$ is a linear combination of basis functions, and for the *p*-type hierarchal elements, we can divide it into two parts, depending on whether those basis functions were originally present or only added when we increased the order of the element, as follows:

$$\mathbf{E}_e = \mathbf{E} + \Delta\mathbf{E} \tag{2.48}$$

When $\mathbf{E}_e$ is not approximate**,** the following quantities are all exactly the same

$$b(\mathbf{E}_e) - b(\mathbf{E}) \tag{2.49}$$

$$a(\mathbf{E}_e, \mathbf{E}_e) - a(\mathbf{E}, \mathbf{E}) \tag{2.50}$$

$$a(\Delta\mathbf{E}, \Delta\mathbf{E}) \tag{2.51}$$

$$b(\Delta\mathbf{E}) - a(\Delta\mathbf{E}, \mathbf{E}) \tag{2.52}$$

However, they will be slightly different when $\mathbf{E}_e$ is approximated, as it must be because it is too costly to find exactly. The advantage of (2.52) is that it is insensitive to errors in $\Delta\mathbf{E}$ that lie in the original FE space (in which $\mathbf{E}$ was found). Therefore, when $\mathbf{E}_e$ is approximated, we don't have to worry about the "lower order" part of it. An efficient way of approximating $\mathbf{E}_e$ is given in [50].

In this framework, it is possible to define an error indicator that focuses on a parameter of interest, so that the error in the parameter is reduced more rapidly as the accuracy of the electric or magnetic field is improved. For example, in analyzing microwave components, minimizing the error related to S parameters is a common approach. For each element, the error indicator estimates the change in the value of an S parameter that would be caused by increasing the order of basis functions over the element. For the discrete problem of (2.32), driven by the excitation of port p rather than an incoming plane wave, the change in S_{pp} can be represented in the form [50]

$$\Delta S^e = \left| \frac{1}{2} b(\Delta\mathbf{E}) - \frac{1}{2} a(\mathbf{E}, \Delta\mathbf{E}) \right| \tag{2.53}$$

where $\Delta\mathbf{E}$ is the field change after increasing the order of element e, as explained in (2.48). The individual expressions (2.53) can be computed by considering one element at a time and solving relatively inexpensive local problems. The indicator then allows us to detect elements in regions where the largest errors in S_{pp} come from. At each adaptive step, the order for a fixed fraction of the elements, those with the highest errors, are increased. The adaptive procedure can be terminated after the desired accuracy has been achieved.

CHAPTER 3

Multilevel and Algebraic Multigrid Preconditioning

In this chapter, the multilevel and algebraic multigrid method are presented as an efficient preconditioner for solving higher order FEM linear systems with Krylov solvers. The focus is on solving matrix equations arising from the *p*-type hierarchical finite element method used adaptively and a multilevel preconditioner for this is presented in Section 3.1. In Section 3.2, the lowest level correction step in the multilevel method is discussed in detail and the requirements for development of AMG solutions is considered. This is followed by the topic of Auxiliary Space Preconditioning (ASP) and damped operator preconditioning techniques in Section 3.3 and 3.4 respectively. Several classes of novel ASP/AMG preconditioners for the lowest order elements, for the wave equation and the T- Ω Method, are presented in Section 3.5 and 3.6. We then review in Section 3.7 a standard nodal AMG method. Finally, the complete multilevel/ASP/AMG preconditioning cycle is presented in Section 3.8.

3.1 Multilevel Methods for *p*-type FEM Systems

The concept of designing algebraic multilevel preconditioners for *p*-type FEM systems is similar to the multigrid approaches for nested grids [56] [57]. In traditional multigrid methods, the different levels are associated with different element sizes and the basis functions are the same at each level. On the other hand, in the *p*-type multilevel algorithm [60] [47] [58] [59] [60] [61], a single grid with multiple levels of basis functions are considered. Each space of basis functions is then treated as a level. They are nested since they satisfy the following property

$$\boldsymbol{V}^1 \subset \boldsymbol{V}^2 \cdots \boldsymbol{V}^{p-1} \subset \boldsymbol{V}^p \tag{3.1}$$

In this thesis, the finite elements used are the tetrahedral, hierarchical, incomplete-order, vector elements described in 2.6.1. The order 1 elements are the well-known Whitney elements [35], with 6 unknowns, associated with 6 basis functions, which will be referred to as "first order". The element of order 2 contains these 6 basis functions, plus another 14, which are called "second order". The elements of orders 3 and 4 add 25 "third order" and 39 "fourth order" basis functions, respectively. Since the elements are hierarchical, different orders can be mixed together in the same mesh, as will typically happen after the first iteration of *p*-adaption.

Following discretization of the curl-curl PDE using hierarchical elements and assuming that elements of different orders are present (with maximum order p), the resulting linear systems (2.35) and (2.45) can be restated as:

$$\mathcal{A}^p x^p = b^p \tag{3.2}$$

The unknowns are numbered so that those associated with first order basis functions come first, then those of second order, and so on. (Note that, for example, a first order unknown may belong only to elements of order higher than 1). Numbering unknowns in such a way allows us to partition the system matrix $\mathcal{A}^p$, the unknown vector and the right-hand side vector in the following block structured form:

$$\mathcal{A}^p = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1p} \\ A_{21} & A_{22} & & \\ \vdots & & \ddots & \vdots \\ A_{p1} & & \cdots & A_{pp} \end{bmatrix}, \; x^p = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}, \; b^p = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{bmatrix} \tag{3.3}$$

In the following sections of this chapter, multilevel solution of (3.3) is considered. Here we note that for the problems that require a very large mesh to model geometrical complexity, the bulk of the elements in the above partitioning may remain at low order and higher order unknowns may be introduced (adaptively) only in a few parts of the mesh. In general, higher order parts give denser matrices and their conditioning depends on how the element bases are constructed. The number of unknowns per tetrahedron and an estimate of the typical number of nonzero entries per row for a regular rectangular grid of p^{th} order elements are given in Table 3.1 [62]. As observed, the higher order systems are relatively larger and denser. The sparsity pattern of a typical hierarchical FEM matrix with 3 levels (uniform order) is shown in figure 3.1 and provides a good indication of the size of the blocks in $\mathcal{A}^p$.

Table 3.1: Number of unknowns and nonzero entries per row for hierarchical FEM.

Order (p)	Number of unknowns per tetrahedron	Number of nonzero entries per row
1	6	26
2	20	50
3	45	90
4	84	149

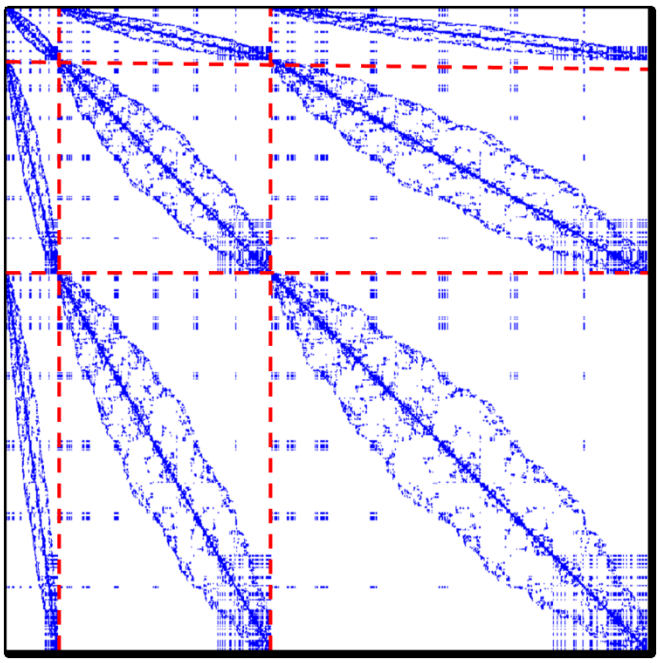


Figure 3.1: The sparsity pattern of $\mathcal{A}^p$ $(p = 3)$, a typical hierarchical FEM matrix for uniform order elements.

3.1.1 Abstract Schwarz Theory and Two-level Scheme

Equation (3.2) can also be written in the following block form

$$\begin{bmatrix} \mathcal{A}^{p-1} & \mathcal{B}_{p-1}^T \\ \mathcal{B}_{p-1} & A_{pp} \end{bmatrix} \begin{bmatrix} x^{p-1} \\ x_p \end{bmatrix} = \begin{bmatrix} b^{p-1} \\ b_p \end{bmatrix} \tag{3.4}$$

in which

$$\mathcal{B}_{p-1} = \begin{bmatrix} A_{p1} & A_{p2} & \cdots & A_{p(p-1)} \end{bmatrix} \tag{3.5}$$

$$\mathcal{B}_{p-1}^T = \begin{bmatrix} A_{1p}^T & A_{2p}^T & \dots & A_{(p-1)p}^T \end{bmatrix}^T \tag{3.6}$$

As shown by (3.4), $\mathcal{A}^{p-1}$ appears as the square, upper left, submatrix of $\mathcal{A}^p$ with lower right block A_{pp}. The same hierarchical inclusion is followed for the sequence of matrices $\mathcal{A}^l$, $l = 2,3 \cdots p-1$. Let n^{p-1} and $n_p = n^p - n^{p-1}$ be the dimensions of b^{p-1} and b_p, respectively. The rewritten FEM matrix in (3.4) can then be factorized:

$$\begin{bmatrix} \mathcal{A}^{p-1} & \mathcal{B}_{p-1}^T \\ \mathcal{B}_{p-1} & A_{pp} \end{bmatrix} = \begin{bmatrix} I^{p-1} & \mathcal{B}_{p-1}^T A_{pp}^{-1} \\ 0 & I_p \end{bmatrix} \begin{bmatrix} \mathcal{A}^{p-1} - \mathcal{B}_{p-1}^T A_{pp}^{-1} \mathcal{B}_{p-1} & 0 \\ 0 & A_{pp} \end{bmatrix} \begin{bmatrix} I^{p-1} & 0 \\ A_{pp}^{-1} \mathcal{B}_{p-1} & I_p \end{bmatrix} \tag{3.7}$$

in which I^{p-1} and I_p are identy matrices of size $n^{p-1} \times n^{p-1}$ and $n_p \times n_p$, respectively. According to the multiplicative Schwarz theory and based on the above factorization, a two-level preconditioning matrix of $\mathcal{A}^p$ is:

$$\mathcal{M}^p = \begin{bmatrix} I^{p-1} & \mathcal{B}_{p-1}^T A_{pp}^{-1} \\ 0 & I_p \end{bmatrix} \begin{bmatrix} \mathcal{A}^{p-1} & 0 \\ 0 & A_{pp} \end{bmatrix} \begin{bmatrix} I^{p-1} & 0 \\ A_{pp}^{-1} \mathcal{B}_{p-1} & I_p \end{bmatrix} \tag{3.8}$$

The term $\mathcal{A}^{p-1} - \mathcal{B}_{p-1}^T A_{pp}^{-1} \mathcal{B}_{p-1}$ is called the Schur component of the block A_{pp}. The above preconditioner is obtained by dropping the coupling term $\mathcal{B}_{p-1}^T A_{pp}^{-1} \mathcal{B}_{p-1}$ from the Schur component. In practice, the hierarchical basis functions are partially orthogonalized and the coupling of the different order basis functions is weak [47] [62]. The detailed theory of Schwarz method is provided in [63].

An advantage of the above matrix factorization is that it provides a good understanding of the approximation made by $\mathcal{M}^p$. It is also seen from (3.8) that to apply the preconditioner to the vector b^p, namely $x = (\mathcal{M}^p)^{-1} b^p$, the inverses of $\mathcal{A}^{p-1}$ and A_{pp} are required. However, the overall cost for a preconditioner will be kept reasonable only if the number of operations involved remains a small multiple of

the number of non-zero elements in $\mathcal{A}^{p-1}$ and A_{pp}. This means that practical two-level schemes have to rely on an approximate inversion of these matrices, shown below as $R(\mathcal{A}^{p-1})$ and $R\left(A_{pp}\right)$. Usually the approximation consists of a few iterations of a standard relaxation method, like Jacobi or Gauss-Seidel, so that rapidly varying components of error are effectively reduced. The Gauss-Seidel has long been the method of choice within multigrid schemes, because of its effectiveness and numerically attractive performance on problems of practical interest.

In addition, from a practical point of view, the action of applying $(\mathcal{M}^p)^{-1}$ should be defined implicitly by an algorithm, rather than explicitly by building an approximation to $(\mathcal{M}^p)^{-1}$. It can be shown that $(\mathcal{M}^p)^{-1}b^p$ can be approximated from the following algorithm:

Algorithm 3.1: The two-level scheme $x \leftarrow (\mathcal{M}^p)^{-1}b^p$		
1.	Approximately solve $A_{p,p}x_p = b_p$	$x_p \leftarrow R\left(A_{pp}\right)b_p$; $x \leftarrow x_p$
2.	Residual update:	$b^p \leftarrow b^p - \mathcal{A}^p x$
3.	Coarsening:	$b^{p-1} \leftarrow$ "coarse" part of b^p
4.	Approximately solve $\mathcal{A}^{p-1}x^{p-1} = b^{p-1}$	$x^{p-1} \simeq R(\mathcal{A}^{p-1})b^{p-1}$;
5.	Prolongating:	$x \leftarrow x +$prolongation of x^{p-1}
6.	Residual update:	$b^p \leftarrow b^p - \mathcal{A}^p x$
7.	Approximately solve $A_{p,p}x_p = b_p$	$x_p \leftarrow R\left(A_{pp}\right)b_p$; $x \leftarrow x + x_p$

In line 3, coarsening means that the representation of the field is coarsened by dropping the entries corresponding to order p. Prolongating (line 5) indicates adding back these entries and setting their values to zero. The coarsening and prolongation operations are the main components of the multilevel approach that make the connections between the levels.

3.1.2 From Two Levels to Multilevel

The above construction procedure for a two-level hierarchical matrix can be extended to all the levels of the multilevel matrix $\mathcal{A}^p$. The multilevel approach is in fact based on the recursive use of the two-level scheme.

It is known that classical iterative methods like Jacobi and Gauss-Seidel tend to eliminate the high-frequency components of the residual error in (3.4) more rapidly than the low-frequency components. The multilevel method is based on this property, that is, the high-frequency residual components can be eliminated from the higher

order unknowns by applying small numbers of relaxation iterations. This step is called smoothing and provides a cost-effective approximation to the higher order DOFs in (3.4).

The remaining residual components after applying the smoothing step are projected onto a coarser level, in which they now have high frequency part to be eliminated by the action of smoother again. A series of steps can be employed based on recursive use of the two-level scheme, i.e., the coarser problem is solved approximately with another application of the two-level scheme on that level, and so on, until the lowest level A_{11} is reached, where a more accurate solution is generally required.

This process computes an improvement in the solution vector and reduces the residual accordingly. The process is equivalent to the descending half of a cycle for transferring the residual to coarser levels. The correction at the final step is usually called the *coarse grid correction*. When the matrix problem is small enough, the coarse grid correction can be handled exactly by a direct method. The ascending half is then continued by applying relaxations again and the result is called a V-cycle *p*-type multilevel Schwarz (*p*MUS) algorithem. Figure 3.2 graphically shows the cycle for a 4 level *p*MUS. A solid black circle around a matrix means that it is used in an application of a relaxation method, such as one step of backward Gauss-Seidel; a solid gray circle means that it is used in an application of a relaxation method that matches the one in the descending half, yet may be different from it, to preserve the symmetry of the overall preconditioning matrix. The box means an exact solution using this matrix.

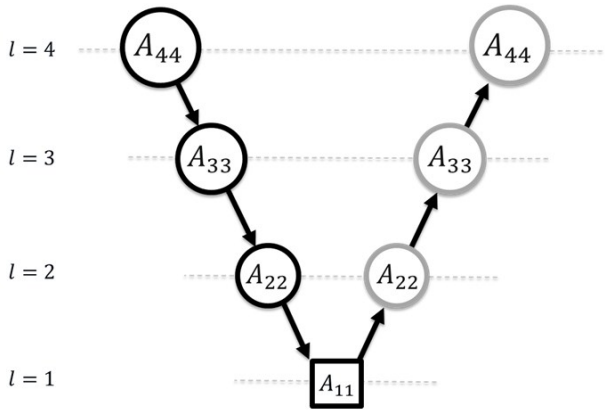


Figure 3.2: The V-Cycle *p*-type Multilevel Schwarz Preconditioner.

We define a pair of trivial prolongation and coarsening mappings, P^l and $(P^l)^T$:

$$P^l = \begin{bmatrix} I^{l-1} \\ 0 \end{bmatrix}_{n^l \times n^{l-1}}, \ (P^l)^T = [I^{l-1} \quad 0]_{n^{l-1} \times n^l} \tag{3.9}$$

where I^{l-1} is the $n^{l-1} \times n^{l-1}$ identity matrix. $(P^l)^{\mathrm{T}}$ "coarsens" the representation of the field by dropping the entries corresponding to order l and P^l "prolongates" the representation by adding back these entries and setting their values to zero. In comparison with multigrid methods, the prolongation and projection here are very simple and independent of domain geometry. Also notice that, by using (3.9), it is easy to verify that the prolongation and projection satisfy the relation

$$\mathcal{A}^{l-1} = (P^l)^{\mathrm{T}} \mathcal{A}^l P^l \ \text{ for } \ l = 2, \cdots, p. \tag{3.10}$$

Then the V-cycle method for finding $x \cong (\mathcal{A}^l)^{-1} r^l$ can be described recursively as follows:

Step 1 (*Pre-Smoothing*): The smoothing operation is applied to the system of equations $A_{l,l} \Delta x_l = r_l$ for the highest order unknowns to obtain an approximate solution Δx_l, In this step, the high-frequency components in the solution error are eliminated. The vector Δx_l is placed in the highest order part of x, with the remaining entries being set to zero.

Step 2: The residual vector corresponding to the approximate solution, is calculated $r^l \leftarrow r^l - \mathcal{A}^l x$.

Step 3 (Restriction): The residual vector is projected onto a coarser level using the restriction matrix: $r^{l-1} \leftarrow (P^l)^T r^l$.

Step 4: The residual equation $\mathcal{A}^{l-1} \Delta x^{l-1} = r^{l-1}$ is solved approximately to obtain the solution vector Δx^{l-1}, corresponding to the representation of the problem at the coarser level. This step involves solving a smaller problem and takes a shorter time, because there is a smaller number of unknowns in x^{l-1} comparing to x^l.

Step 5 (Prolongation): The solution of the coarser problem is transferred back to the fine level using the prolongation matrix $P^l \Delta x^{l-1}$ and the solution obtained in Step 1 is corrected using $x \leftarrow x + P^l \Delta x^{l-1}$.

Step 6: The residual vector corresponding to the recent approximate solution is updated: $r^l \leftarrow r^l - \mathcal{A}^l x$.

Step 7: (*Post-Smoothing*): The smoothing operation is applied to the equation $A_{l,l}\Delta x_l = b_l$ again to correct the values of higher order unknows in vector x. This procedure is called post-smoothing.

We can now state the standard *p*-type multilevel V-cycle algorithm as follows:

Algorithm 3.2: Standard *p*-type Multilevel Schwarz: $x \leftarrow p\text{MUS}(\mathcal{A}^l, r^l)$

1.	If $l = 1$:	
2.	Directly Solve order 1:	$x \leftarrow (\mathcal{A}^1)^{-1} r^1$
3.	Else if $l > 2$:	
4.	Pre-Smoothing:	$\Delta x_l \leftarrow R_b(A_{ll}) r^l;\ x \leftarrow \Delta x_l$
5.	Residual update:	$r^l \leftarrow r^l - \mathcal{A}^l x$
6.	Coarsen:	$r^{l-1} \leftarrow (P^l)^T r^l$
7.	Solve coarse:	$\Delta x \leftarrow p\text{MUS}(\mathcal{A}^{l-1}, r^{l-1})$
8.	Prolongate:	$x \leftarrow x + P^l \Delta x$
9.	Residual update:	$r^l \leftarrow r^l - \mathcal{A}^l x$
10.	Post-Smoothing:	$\Delta x_l \leftarrow R_f(A_{ll}) r_l;\ x \leftarrow x + \Delta x_l$

The iteration matrices $R_f(A_{ll})$ and $R_b(A_{ll})$ used in the algorithm are the Gauss-Seidel matrices given in (1.8) and (1.9). They act as pre and post smoothers (Step 1 and 7, respectively) for the diagonal blocks $A_{l,l}$. For the standard multilevel V-cycle method, if the smoother used is symmetric, then the cycle is also a symmetric preconditioning method. More generally, if two different smoothers are used, one in the descending path, the other in the ascending path, and they are the transpose of each other, then the standard multilevel V-cycle method is still symmetric. The symmetric smoother assumption in the above discussion is important. Without this assumption, the standard multilevel V-cycle scheme may not be applicable for preconditioning the Krylov methods that work based on symmetric Lanczos orthogonalization (discussed in Chapter 4).

3.2 Lowest level Correction

For the multilevel Schwarz method to have satisfactory convergence, the solution at the lowest level has to be more accurate than at the other levels [62][60]. The lowest level correction in Algorithm 3.2 involves solving the following system

$$\mathcal{A}^1 x = r^1 \tag{3.11}$$

In this section, some of the difficulties in solving the above equation for the wave problem are explained. To discuss the challenges, we first need to be more specific

about the structure of matrix $\mathcal{A}^1 = A_{11}$. The finite element method discretization of (2.32) at the Whitney level ($\boldsymbol{V}^h = \boldsymbol{V}^1$) leads to the following form for $\mathcal{A}^1$:

$$\mathcal{A}^1 = \mathcal{S}^1 - k_0^2 \mathcal{M}^1 + \mathcal{C}^1 \tag{3.12}$$

in which $\mathcal{S}^1$ is the stiffness matrix, $\mathcal{M}^1$ is the mass matrix and $\mathcal{C}^1$ represents boundary (surface mass) terms. The elements of the matrices $\mathcal{S}^1$, $\mathcal{M}^1$and $\mathcal{C}^1$are given by

$$[\mathcal{S}^1]_{ij} = \frac{1}{k_0 \eta_0} \int_{\Omega} \left(\nabla \times \mathbf{W}_i . \hat{\mu}_r^{-1} \nabla \times \mathbf{W}_j \right) dV \tag{3.13}$$

$$[\mathcal{M}^1]_{ij} = \frac{k_0}{\eta_0} \int_{\Omega} \left(\mathbf{W}_i \cdot \hat{\varepsilon}_r \mathbf{W}_j \right) dV \tag{3.14}$$

$$[\mathcal{C}^1]_{ij} = \jmath \sum_{q=1}^{\mathrm{N}} \sum_{l=1}^{\infty} V_l^{(q)}(\mathbf{W}_i)\, V_l^{(q)}\left(\mathbf{W}_j\right) + \frac{\jmath}{\eta_0} \int_{\Gamma_o} \mathbf{E}_{it} \cdot \mathbf{W}_{jt} dS \tag{3.15}$$

Assuming that the relative permeability tensor is real (passive materials and no magnetic loss), the stiffness matrix $\mathcal{S}^1$ is symmetric positive semidefinite and its nullspace is spanned by a set of linearly independent vectors corresponding to irrotational fields. The boundary matrix $\mathcal{C}^1$ is symmetric and complex. The mass matrix $\mathcal{M}^1$ is symmetric and will be complex for electrically lossy material (complex $\hat{\varepsilon}_r$), with a positive definite real part and a small imaginary part (for small losses). Therefore the negative $-k_0^2$ factor in (3.12) characterizes $\mathcal{A}^1$ as an indefinite system, meaning that the eigenvalues reside on both right and left sides of the complex plane. Moreover, the condition number of the system typically increases with frequency and grows as the mesh is refined [64].

In order to better characterize the spectral properties of matrix $\mathcal{A}^1$, we consider the following related eigenvalue problem, which governs the natural resonances of the closed system (with the boundary terms omitted):

$$(\mathcal{S}^1 - \mu_i^2 \mathcal{M}^1) v_i = 0 \tag{3.16}$$

Assume that the eigenvalue-eigenvector pair (μ_i, v_i) satisfies this generalized eigenvalue problem. The eigenvalues of $\mathcal{A}^1$ with negative real part ("negative" eigenvalues) can be classified into two different sets corresponding to physical and nonphysical modes of (3.16) [13][14]. Nonphysical modes are related to the gradient space of $\boldsymbol{V}^1$ [65] [66] and are usually denoted by the term "Type A" eigenvalues in the

literature [13] [14]. Existence of Type A eigenvalues results in serious convergence issues for Krylov iterative methods, as investigated in pervious works and also shown by the numerical experiments in Chapter 5.

The other negative eigenvalues of $\mathcal{A}^1$ are associated with physical modes of (3.16) and are therefore of "Type B". However, they are at frequencies μ_i less than the operating frequency, k_0. Therefore, when the operating frequency is close to a resonance of the structure, the finite element matrix equation (3.11) can become highly ill-conditioned.

For the Whitney space $\boldsymbol{V}^1$, the gradient and rotational spaces are not separated and nonphysical negative eigenvalues of the system cannot be easily preconditioned into positive eigenvalues [62]. That is the main reason for the slow convergence exhibited by conventional iterative schemes applied to the lowest order block, $\mathcal{A}^1$, with standard preconditioners like incomplete Cholesky (IC) and Gauss-Seidel [65].

In the conventional *p*MUS algorithm, $\mathcal{A}^1$ is decomposed exactly to avoid this difficulty. However, many modern problems involve very complex geometries, making the number of elements after meshing extremely large. For realistic three dimensional problems, a large number of Whitney degrees of freedom can arise after discretization and the lowest level linear system is itself often very large. Therefore, the decomposition of $\mathcal{A}^1$, if not impossible to apply, will become the bottleneck. Consequently, the efficient, robust (e.g., with respect to the properties of the PDE) and fast treatment of the lowest level correction remains a difficulty for the *p*MUS algorithm.

As possible replacements for the direct treatment, multigrid methods are among the most efficient approaches for approximate inversion of large linear systems. They follow the same idea as *p*MUS: damping errors by utilizing multiple resolutions in a recursive scheme. High-energy and oscillatory components are effectively reduced through a simple smoothing procedure, while the low-energy components are tackled using a lower resolution version of the problem (coarse grid).

For large scale problems, the geometric multigrid (GMG) methods can be used for solving (3.11) if we have a nested sequence of meshes [56] [57] [67]. By defining appropriate transfer operators, the FE spaces are linked together. Recursive application of a two grid process can be applied by performing pre-smoothing steps

on the fine system, and then restricting the residual on the coarse mesh. On the coarsest level the coarse system is solved (exactly) and the defect is prolongated on the fine level. Finally, post-smoothing steps are performed.

When nested meshes are not available, algebraic multigrid (AMG) methods can be used instead. Eliminating the decomposition of $\mathcal{A}^1$ by the application of AMG is investigated in this thesis.

3.2.1 Algebraic Multigrid for the $\mathcal{H}(\mathbf{curl})$ Problem

In contrast to GMG, where a grid hierarchy is required explicitly, algebraic multigrid (AMG) is able to define the prolongation operators (and construct the corresponding matrix) in a pure algebraic way. AMG operates more on the level of the matrix than of the underlying nested FE meshes. The crucial point in the construction of the algebraic methods is the numerical effort of the coarsening process and the construction of appropriate transfer operators. The challenge is to construct an AMG method with a good convergence rate but rather low costs in the setup and the application.

Classical algebraic multigrid methods have had a lot of success on Poisson problems, obtained from scalar, nodal, basis functions. Nodal AMG methods, also known as scalar AMG, simply apply a relaxation scheme to a hierarchy of algebraically constructed coarse-grid problems. Efficient implementations of AMG have been under intensive research in the last decade and several well-established software libraries have been developed [17].

However, a unique difficulty in solving (3.11) with AMG arises from the nature of the curl-curl operator. Traditional AMG methods are adapted only to the Laplacian operator on $\mathcal{H}^1(\Omega)$ and deal badly with the curl-curl operator [23] [75]. The first serious and practical AMG approach for problems in $\boldsymbol{\mathcal{H}}(\mathrm{curl})$ was made in 2002 by Reitzinger and Schoberl [23]. The principal idea of their method is to maintain the representation of a discrete gradient and its relationship to the discrete curl-curl operator on coarse meshes. This can be achieved by creating two multigrid hierarchies for the nodes and edges of the mesh and aggregating the edges via nodal aggregation, in such a way that necessary conditions on the prolongation operators are imposed. After constructing the AMG hierarchy, each coarser version of $\mathcal{A}^1$ has a nullspace

spanned by the gradient of a set of linearly independent scalars over the coarse grid. The proposed method is easy to compute and has low multigrid operator complexity [68], but the prolongator has poor approximation properties and the convergence rates suffer [25] [68]. An improved version of [23] can be found in [69] [70] [25], where a better prolongation was proposed, with mesh-independent convergence rates. These method however require two coarsening procedures and hybrid smoothers at every level to handle nonelliptic discretization problems and efficient multigrid methods developed for scalar AMGs are not often directly applicable.

3.3 Auxiliary Space Preconditioning

Recently, Hiptmair and Xu proposed an innovative approach for solving $\boldsymbol{\mathcal{H}}(\mathrm{curl})$ systems, known as HX auxiliary space preconditioning [72] [73]. The approach relies on a decomposition of Whitney vector fields and employs the framework of the auxiliary space method. In contrast to applying coarsening to the edge unknowns, it uses "edge to node" projection and solves approximate problems in auxiliary spaces defined on the nodes of the original mesh. This property significantly simplifies the computations and, more importantly, allows us to makes use of the proven ability of scalar AMG for nodal FEM problems.

The idea of auxiliary space preconditioning (ASP) is to invert approximately the curl-curl operator by transferring the problem into subspaces in which scalar AMG can be applied. Although this method was proposed for the positive-semidefinite case, which occurs in time-domain, quasi-static problems, extensive numerical experiments reported in this thesis demonstrate that it is also efficient and robust for the frequency-domain problems, both quasi-static and full-wave.

3.3.1 Whitney Space Decomposition

The present section summarizes the main ingredients used to derive the auxiliary space $\boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$ preconditioner (see [71] [72] [74] [75] for details).

a) Regular Decomposition

Let $\boldsymbol{V}^1 \subset \boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$ be the space spanned by the lowest order edge element basis functions on a tetrahedral mesh $\mathcal{T}^h$ (the Whitney space). There is an associated space $N \subset \mathcal{H}^1(\Omega)$ of piecewise linear, scalar functions on the same mesh and it is known that ∇N is the irrotational subspace of $\boldsymbol{V}^1$ [75] [78].

It is also known that for every $\boldsymbol{v} \in \boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$, there exists a member $\boldsymbol{u}$ in the space $\boldsymbol{\mathcal{H}}^1(\Omega) = [\mathcal{H}^1(\Omega)]^3$ with the same curl, i.e. $\nabla \times \boldsymbol{u} = \nabla \times \boldsymbol{v}$ [75]. This suggests that functions in $\boldsymbol{V}^1$ can be represented approximately using functions in N and $\boldsymbol{\mathcal{H}}^1(\Omega)$. With the purpose of defining an auxiliary space preconditioner, we first need to define the Nedelec interpolation operator Π in order to map vectors of $\boldsymbol{\mathcal{H}}^1(\Omega)$ into $\boldsymbol{V}^1$ as follows:

$$\Pi \boldsymbol{u} \triangleq \sum_i \left(\int_{\text{edge } i} \boldsymbol{u} \cdot \mathrm{d}\mathbf{l} \right) \mathbf{e}_i \tag{3.17}$$

where $\mathbf{e}_i$ is the Whitney basis function defined in Table 2.2 and $\mathrm{d}\mathbf{l}$ is a vector tangent to edge i.

Suppose now that $\boldsymbol{v} = \mathbf{v} \in \boldsymbol{V}^1$. Then it follows that there exists $\boldsymbol{u} \in \boldsymbol{\mathcal{H}}^1(\Omega)$ such that $\nabla \times \boldsymbol{u} = \nabla \times \mathbf{v} \in \nabla \times \boldsymbol{V}^1$. For such a $\boldsymbol{u}$, it can be shown that $\nabla \times \boldsymbol{u} = \nabla \times \Pi \boldsymbol{u}$. It follows that:

$$\nabla \times \mathbf{v} = \nabla \times \boldsymbol{u} = \nabla \times \Pi \boldsymbol{u} \tag{3.18}$$

Therefore,

$$\nabla \times (\mathbf{v} - \Pi \boldsymbol{u}) = 0 \tag{3.19}$$

Since $\mathbf{v} - \Pi \boldsymbol{u} \in \boldsymbol{V}^1$ and has a zero curl, it can be represented by $\nabla \Phi$ for some $\Phi \subset N$. The following Whitney space decomposition, known as a *regular decomposition* [20], can be achieved:

$$\mathbf{v} = \Pi \boldsymbol{u} + \nabla \Phi \tag{3.20}$$

Assuming that c stands for a constant independent of the mesh size and in accordance to the norms defined in Section 2.4, the following inequality holds [75]:

$$\|\Pi \boldsymbol{u}\|_0 + \|\nabla \Phi\|_0 \leq c \|\mathbf{v}\|_0 \tag{3.21}$$

b) Hiptmair-Xu Decomposition

The regular decomposition in (3.20) is in fact semi-discrete, since it involves the term $\boldsymbol{u} \in \boldsymbol{\mathcal{H}}^1(\Omega)$ that, in general, is not a finite element function. The challenge is to convert (3.19) into a purely discrete decomposition with some stability condition similar to (3.21). There are several ways to further approximate the non-discrete component [75]. In the Hiptmair-Xu (HX) decomposition, it is achieved by incorporating another high frequency, "small" component.

For the HX decomposition, we introduce the space of vector "nodal" functions, $N^3 \subset \boldsymbol{\mathcal{H}}^1(\Omega)$. These are vector functions that, unlike Whitney basis functions, are both normally and tangentially continuous from one element to the next and are fully first-order in each element. For any $\boldsymbol{u} \in \boldsymbol{\mathcal{H}}^1(\Omega)$ there exists $\mathbf{u} \in N^3$ that satisfies [75]:

$$h^{-1}\|\boldsymbol{u} - \mathbf{u}\|_0 + \|\mathbf{u}\|_1 \leq c\|\boldsymbol{u}\|_1 \tag{3.22}$$

Defining the small contribution $\mathbf{v}_s = \Pi(\boldsymbol{u} - \mathbf{u}) \in \boldsymbol{V}^1$, we have $\Pi\boldsymbol{u} = \mathbf{v}_s + \Pi\mathbf{u}$ to be substituted in (3.20). Therefore, the space $\boldsymbol{V}^1$, the scalar space and the vector nodal space are linked by the following result: for any $\mathbf{v} \in \boldsymbol{V}^1$ there exists $\mathbf{u} \in N^3$ and $\Phi \in N$ such that:

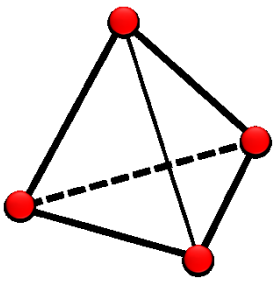
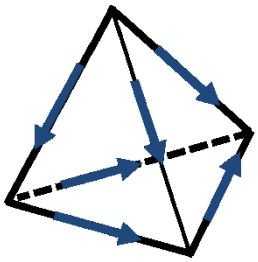
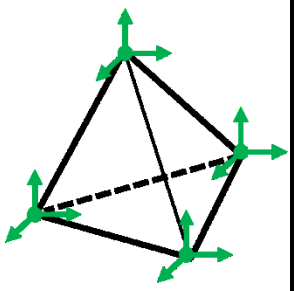
$$\mathbf{v} = \mathbf{v}_s + \Pi\mathbf{u} + \boldsymbol{\nabla}\Phi \tag{3.23}$$

and the following stability estimate holds (the proof can be found in [75]):

$$h^{-1}\|\mathbf{v}_s\|_0 + \|\mathbf{u}\|_1 + \|\boldsymbol{\nabla}\Phi\|_0 \leq c\|\mathbf{v}\|_{\mathrm{curl}} \tag{3.24}$$

Table 3.2 summarizes the spaces involved in the HX decomposition and gives a graphical representation of the basis functions for the FE spaces.

Table 3.2: Spaces involved in the HX decomposition, along with corresponding members and basis functions illustration.

Hilbert Space	$\mathcal{H}^1(\Omega)$	$\boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$	$\boldsymbol{\mathcal{H}}^1(\Omega) = [\mathcal{H}^1(\Omega)]^3$
Members	$g \in \mathcal{H}^1(\Omega)$	$\boldsymbol{v} \in \boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$	$\boldsymbol{u} \in \boldsymbol{\mathcal{H}}^1(\Omega)$
FE space	1st order Scalar Space	Whitney Space	1st order Vector Space
Relations	$N \subset \mathcal{H}^1(\Omega)$	$\nabla N \subset \boldsymbol{V}^1 \subset \boldsymbol{\mathcal{H}}(\mathrm{curl}, \Omega)$	$N^3 \subset \boldsymbol{\mathcal{H}}^1(\Omega)$
Members	$\Phi \in N$	$\mathbf{v} \in \boldsymbol{V}^1$	$\mathbf{u} \in N^3$
Basis illustration			

3.3.2 Method of Subspace Correction

Based on the HX decomposition, we are able to represent the space $\boldsymbol{V}^1$ as the summation of subspaces. This splitting concept also suggests that it might be possible to solve the problem approximately by solving related problems on the auxiliary spaces N and N^3. Therefore, the original problem in (3.11) can be split into sub-problems if we can find appropriate mapping operators between the spaces. The sub-problems are relatively easier to solve, since they represent smaller problems in nodal spaces. Providing that required projection operators to the subspaces are available, different components of error can be captured separately by these subspaces. Solution components can then be combined in various versions that will be described later.

To realize the preconditioners for the finite element matrix, the decomposition discussed in the previous section should be adapted to a matrix setting too. We use the usual nodal, interpolatory basis in N and N^3, as well as the Whitney edge basis in $\boldsymbol{V}^1$. Then functions $\Phi \in N$ and $\mathbf{v} \in \boldsymbol{V}^1$ can be represented by column vectors of node and edge values, respectively. But first the mapping operators are introduced.

3.3.3 Mapping Operators

The matrix representation of the mapping operator from N to $\boldsymbol{V}^1$ is commonly called the discrete gradient matrix, denoted by sparse matrix G. It is simply the node to edge mapping matrix with entries of -1 and 1 per row. This matrix is readily

available since it describes the edges of the mesh in terms of its vertices. If e is an edge from node i to j, then the only two non-zero entries of G in the row corresponding to e are [13] [75]

$$[G]_{ei} = -[G]_{ej} = 1 \tag{3.25}$$

To preserve symmetry, we use the transpose, G^T, to map backwards from $\boldsymbol{V}^1$ to N.

For N^3 we use the N basis for each Cartesian component of the vector and represent $\mathbf{u} \in N^3$ by a column vector in which nodal values of the x, y and z components occupy 3 successive blocks. Then the operator Π becomes a sparse matrix Π which also has a block form:

$$\Pi = [\Pi_x \quad \Pi_y \quad \Pi_z] \tag{3.26}$$

Each block has the same dimension and sparsity pattern as G, i.e., two nonzero entries per row [75]. If e is an edge connecting nodes i and j, with coordinates (x_i, y_i, z_i) and (x_j, y_j, z_j), then

$$[\Pi_x]_{ei} = [\Pi_x]_{ej} = \frac{1}{2}(x_i - x_j) \tag{3.27}$$

As observed, the two nonzero values for row e of Π_x, are identical. It can be shown that their value is also equal to the eth row of the product $\frac{1}{2}Gx_c$, where $x_c = [x_1, x_2, x_3 \ldots]^T$ is a vector containing the x coordinates of the nodes.

Similarly for Π_z and Π_z:

$$[\Pi_y]_{ei} = [\Pi_y]_{ej} = \frac{1}{2}(y_i - y_j) \tag{3.28}$$

and

$$[\Pi_z]_{ei} = [\Pi_z]_{ej} = \frac{1}{2}(z_i - z_j) \tag{3.29}$$

We use Π^T to map backwards, from $\boldsymbol{V}^1$ to N^3.

3.3.4 Discretized Operators for the Auxiliary Spaces

Along with these matrices, we also need discretized operators representing the original bilinear form in each auxiliary space. Corresponding to the bilinear form (2.32), we introduce the following weak form for the problems defined on auxiliary spaces N and N^3:

$$\text{Find } \mathbf{E} \in \nabla N \subset \boldsymbol{V}^1\colon\ a(\mathbf{E},\boldsymbol{\upsilon}) = b(\boldsymbol{\upsilon}) \qquad \forall \boldsymbol{\upsilon} \in \nabla N \tag{3.30}$$

$$\text{Find } \mathbf{E} \in \Pi N^3 \subset \boldsymbol{V}^1\colon\ a_\gamma(\mathbf{E},\boldsymbol{\omega}) = b(\boldsymbol{\omega}) \qquad \forall \boldsymbol{\omega} \in \Pi N^3 \tag{3.31}$$

In (3.31), $a_\gamma(\cdot,\cdot)$ stands for a modified version of (2.32), in which the wavenumber k_0^2 is replaced by $(1-\jmath\gamma)k_0^2$ in the volume integral only. Setting $\gamma = 0$ results in exactly the original weak form. The reason for this complexification is explained later, in Section 3.5.

The above problems are equivalent to

$$\text{Find } \widetilde{\Phi} \in N\colon\ a(\nabla\widetilde{\Phi},\nabla\widetilde{\psi}) = b(\nabla\widetilde{\psi}) \qquad \forall\widetilde{\psi} \in N \tag{3.32}$$

$$\text{Find } \tilde{\mathbf{E}} \in N^3\colon\ a_\gamma(\Pi\tilde{\mathbf{E}},\Pi\tilde{\boldsymbol{\omega}}) = b(\Pi\tilde{\boldsymbol{\omega}}) \qquad \forall\tilde{\boldsymbol{\omega}} \in N^3 \tag{3.33}$$

We can now rephrase the problem of solving Maxwell's equations as the following:

$$\text{Find } \widetilde{\Phi} \in N\colon\ a_n(\widetilde{\Phi},\widetilde{\psi}) = b_n(\widetilde{\psi}) \qquad \forall\widetilde{\psi} \in N \tag{3.34}$$

$$\text{Find } \tilde{\mathbf{E}} \in N^3\colon\ a_\Pi(\tilde{\mathbf{E}},\tilde{\boldsymbol{\omega}}) = b_\Pi(\tilde{\boldsymbol{\omega}}) \qquad \forall\tilde{\boldsymbol{\omega}} \in N^3 \tag{3.35}$$

in which

$$a_n(\widetilde{\Phi},\widetilde{\psi}) \triangleq a(\nabla\widetilde{\Phi},\nabla\widetilde{\psi}) \tag{3.36}$$

$$b_n(\widetilde{\psi}) \triangleq b(\nabla\widetilde{\psi}) \tag{3.37}$$

and

$$a_\Pi(\tilde{\mathbf{E}},\tilde{\mathbf{w}}) \triangleq a_\gamma(\Pi\tilde{\mathbf{E}},\Pi\tilde{\mathbf{w}}) \tag{3.38}$$

$$b_\Pi(\tilde{\mathbf{w}}) \triangleq b(\Pi\tilde{\mathbf{w}}) \tag{3.39}$$

Following the standard procedure, a set of linearly independent, first-order, nodal basis functions $\{\xi_1,\xi_2,\dots,\xi_n\}$ and $\{\boldsymbol{\omega}_1,\boldsymbol{\omega}_2,\dots,\boldsymbol{\omega}_{3n}\}$ can be introduced for expansion of $\widetilde{\Phi}$ and $\tilde{\mathbf{E}}$. The above problems may be reduced to the following algebraic matrix equations:

$$\mathcal{A}_n^0\Phi_n = b_n^0 \tag{3.40}$$

$$\mathcal{A}_\Pi^0\tilde{x} = b_\Pi^0 \tag{3.41}$$

In which

$$[\mathcal{A}_n^0]_{ij} = a_{\mathrm{n}}(\xi_i, \xi_j);\ [b_n^0]_i = b_n(\xi_i) \tag{3.42}$$

$$[\mathcal{A}_\Pi^0]_{ij} = a_\Pi(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j);\ [b_\Pi^0]_i = b_\Pi(\boldsymbol{\omega}_i) \tag{3.43}$$

Each of the problems (3.42) and (3.43) are in fact equivalent to (3.11), but solved in the auxiliary spaces. It can be shown that the square matrices $\mathcal{A}_n^0$ and $\mathcal{A}_\Pi^0$ for the auxiliary problems can also be computed by the Galerkin product:

$$\mathcal{A}_n^0 = G^T \mathcal{A}^1 G \tag{3.44}$$

$$\mathcal{A}_\Pi^0 = \Pi^T \mathcal{A}_\gamma^1 \Pi \tag{3.45}$$

where $\mathcal{A}_\gamma^1$ is the system matrix that is obtained instead of $\mathcal{A}^1$ when $a_\gamma(\cdot, \cdot)$ is used instead of $a(\cdot, \cdot)$. The numerical cost for building each of these is roughly equivalent to four matrix-vector products of $\mathcal{A}^1$ [79].

3.4 Damped Operator Preconditioning

It was mentioned in Section 3.2 that the spectral properties of the matrix associated with the discretization of the vector wave operator at level 1 makes extremely difficult the convergence of iterative solvers applied to the related linear system. As explained by equations (3.13) to (3.15), the discretization of (2.32) at the Whitney level leads to the following form for $\mathcal{A}^1$:

$$\mathcal{A}^1 = \mathcal{S}^1 - k_0^2 \mathcal{M}^1 + \mathcal{C}^1 \tag{3.46}$$

Referring to the discussion in Section 3.2, the negative $-k_0^2$ factor in relation (3.46) makes $\mathcal{A}^1$ indefinite. As the wavenumber increases beyond each physical resonance, more negative eigenvalues appear and the indefiniteness of system causes slower convergence of iterative solvers.

Although the linear algebra theory for definite linear systems is well-developed, it is not the case for indefinite linear systems. In the last few decades, much effort has been directed towards the construction and implementation of robust and efficient preconditioners for indefinite linear systems. Particularly in the last few years, physics-based preconditioners have attracted the attention of many researchers [80] [81] [82] [83] [84]. The main idea is to include information related to the physical properties of the problem and also the structure of the operators, when

trying to build a preconditioner. Among the main contributions, the Shifted Laplace Preconditioner (SLP) [80] has received much attention.

Essentially, the idea in SLP is to build the preconditioner based on a complex version of the original operator. In fact, positive definiteness of the overall system is reinforced through the introduction of a fictitious dissipative term into the operator from which the preconditioner is derived. The approach has been applied successfully by Erlangga *et al.* to the wave-like problems described by the scalar Helmholtz operator [80]. To the best of our knowledge, it has not previously been applied to the curl-curl operator.

Following the idea of SLP, it would be beneficial to use a preconditioner based on a damped operator version of (2.32) by introducing a complex factor in the wavenumber. The preconditioner matrix is then obtained from discretization of (2.32), in which the complex modification $(1 - i\gamma)k_0^2$ is applied to k_0^2. The boundary conditions are left unchanged, i.e., the preconditioner is built from:

$$\mathcal{A}_\gamma^1 = \mathcal{S}^1 - (1 - j\gamma)k_0^2\mathcal{M}^1 + \mathcal{C}^1 \tag{3.47}$$

Since the convergence of Krylov methods is closely related to the spectrum of the iteration matrix, the favourable spectrum of the preconditioned matrix $\left(\mathcal{A}_\gamma^1\right)^{-1}\mathcal{A}^1$ will give some insights into the improved convergence rate. Numerical evidence shows that, unlike the spectrum of the unpreconditioned system, the eigenvalues of $\left(\mathcal{A}_\gamma^1\right)^{-1}\mathcal{A}^1$ are mapped to a roughly circular pattern in the right-hand half of the complex plane. In fact, similar to the positive effects of spectral changes for the Helmholtz equation preconditioned by the Shifted Laplace operator [81], when all the eigenvalues of the $\left(\mathcal{A}_\gamma^1\right)^{-1}\mathcal{A}^1$ are located in the right half plane, better convergence of an iterative solver can be ensured. In addition, the eigenvalues are mostly shifted away from the origin and clustered around 1. This suggests that $\mathcal{A}_\gamma^1$ leads to a better preconditioner than $\mathcal{A}^1$. Adding an artificial dissipative term also implies that computing the approximate inverse of $\mathcal{A}_\gamma^1$ can be performed more efficiently using multigrid methods [80].

According to this analysis, one strategy is to build the complete auxiliary space preconditioner from $\mathcal{A}_\gamma^1$ instead of $\mathcal{A}^1$. That would mean using $\mathcal{A}_\gamma^1$ for the smoothing step and for building $\mathcal{A}_n^0$ and $\mathcal{A}_\Pi^0$ in (3.44) and (3.45). However, experiments show

that it is most effective for damping the rotational type errors and using $\mathcal{A}_\gamma^1$ for smoothing and for building $\mathcal{A}_n^0$ actually worsens the performance of the preconditioner. Therefore, it is used only to build $\mathcal{A}_\Pi^0$ in (3.45).

The convergence of an iterative Krylov subspace solver applied to the linear system with SLP and AMG will be independent of the mesh refinement and dependent (but in a weaker way than without the preconditioner) on the wavenumber k_0. The convergence is also clearly affected by the choice of the parameter γ. Following numerical studies, we propose the choice $\gamma = 1.5$, which leads to an efficient and robust preconditioning method.

What is important for the convergence of Krylov subspace methods is that, though the spectrum of the preconditioned system is nicely clustered, some eigenvalues still lie close to zero. With an increase in the wavenumber, the number of eigenvalues around zero appears to increase also.

3.5 ASP Preconditioners for the Wave Equation

We can now present efficient methods for approximately solving $\mathcal{A}^1 x = r^1$ using ASP. Depending on different ways of connecting the 3 FE spaces involved, the following symmetric preconditioners are considered at the lowest level:

3.5.1. Additive Preconditioner

In order to approximate $(\mathcal{A}^1)^{-1} r^1$, we can first use an SSOR preconditioner to damp small components in (3.23) like $\mathbf{v}_s$. This takes the form $R_{bf}(\mathcal{A}^1) r^1$ where $R_{bf}(\mathcal{A}^1) = R_f(\mathcal{A}^1)\mathcal{D}^1 R_b(\mathcal{A}^1)$ (see equation (1.13).

After the smoothing step, we can transfer the residual into each auxiliary space and solve the residual equation in each space. Exact inversion of $\mathcal{A}_n^0$ and $\mathcal{A}_\Pi^0$ is expensive, but fortunately scalar AMG (which will be defined in Section 3.7) is effective in approximately inverting $\mathcal{A}_n^0$ and each of the three diagonal blocks of $\mathcal{A}_\Pi^0$:

$$\mathcal{A}_x^0 \triangleq \Pi_x^T \mathcal{A}_\gamma^1 \Pi_x \tag{3.48}$$

$$\mathcal{A}_y^0 \triangleq \Pi_y^T \mathcal{A}_\gamma^1 \Pi_y \tag{3.49}$$

$$\mathcal{A}_z^0 \triangleq \Pi_z^T \mathcal{A}_\gamma^1 \Pi_z \tag{3.50}$$

We call the approximate inverses of these four matrices $\mathcal{B}_n^0$, $\mathcal{B}_x^0$, $\mathcal{B}_y^0$ and $\mathcal{B}_z^0$, respectively. Now we can approximately compute $(\mathcal{A}^1)^{-1}r^1$. The main body of the resulting preconditioner consists of three components:

1. Smoothing in the space $\boldsymbol{V}^1$ using sweeps of SSOR, expressed by $R_f(\mathcal{A}^1)\mathcal{D}^1R_b(\mathcal{A}^1)r^1$.
2. Approximate solution in the kernel of $\boldsymbol{V}^1$ (with a nodal AMG solver), expressed by $G\mathcal{B}_n^0G^Tr^1$.
3. Approximate solution in the vector nodal counterpart of $\boldsymbol{V}^1$ (with nodal AMG solver), expressed by $\Pi_i\mathcal{B}_i^0\Pi_i^Tr^1$, $i = x, y, z$.

Based on the framework of auxiliary space correction, solutions from each problem are assembled to give a correction in the space $\boldsymbol{V}^1$. In the simplest method, the four corrections for a given residual can be added, resulting in the following additive preconditioner with operations that can be performed in parallel to each other:

$$(\mathcal{A}^1)^{-1}r^1 \cong R_f(\mathcal{A}^1)DR_b(\mathcal{A}^1)r^1 + G\mathcal{B}_n^0G^Tr^1 + \sum_{i=x,y,z}\Pi_i\mathcal{B}_i^0\Pi_i^Tr^1 \qquad (3.51)$$

The additive preconditioner (3.51) is coded in the following algorithm. The approximate inverse of $\mathcal{A}^1$ defined by this algorithm is denoted $\mathcal{B}_a^1$.

Algorithm 3.3: Additive Preconditioner $x \leftarrow \mathcal{B}_a^1r^1$

1.	SSOR:	$\Delta x \leftarrow R_f(\mathcal{A}^1)DR_b(\mathcal{A}^1)r^1$
2.	Auxiliary spaces:	$\Delta x \leftarrow \Delta x + G\mathcal{B}_n^0G^Tr^1 + \sum_{i=x,y,z}\Pi_i\mathcal{B}_i^0\Pi_i^Tr^1$
3.	Solution update:	$x \leftarrow x + \Delta x$

3.5.2. Multiplicative V-cycle Preconditioner

Another way to assemble the corrections is a multiplicative V-cycle that splits the SSOR correction in Algorithm 3.3 into backward and forward Gauss-Seidel iterations in the downward and upward sides of a V-cycle. This version performs better than the additive approach in general:

Algorithm 3.4: Multiplicative V-cycle ASP Preconditioner $x \leftarrow \mathcal{B}_v^1 r^1$

1.	Backward GS:	$\Delta x \leftarrow R_b(\mathcal{A}^1) r^1$
2.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow \Delta x$
3.	Auxiliary spaces:	$\Delta x \leftarrow G\mathcal{B}_n^0 G^T r^1 + \sum_{i=x,y,z} \Pi_i \mathcal{B}_i^0 \Pi_i^T r^1$
4.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow x + \Delta x$
5.	Forward GS:	$\Delta x \leftarrow R_f(\mathcal{A}^1) r^1$
6.	Solution update:	$x \leftarrow x + \Delta x$

The approximate inverse of $\mathcal{A}^1$ defined by Algorithm 3.4 is denoted $\mathcal{B}_v^1$. In an alternative approach, one may consider a multiplicative version also for the 4 auxiliary space corrections by repeating steps 2, 4 and 5 for each nodal correction. However, considering the size of problems and the relatively higher cost of Gauss-Seidel iterations and residual updates for edge DOFs, parallel computation of nodal corrections would be numerically more attractive.

3.5.3. Multiplicative W-cycle Preconditioner

Since the work done at lines 2-6 of the previous algorithm is relatively inexpensive (comparing to higher order smoothing steps in *p*MUS) and yet important to the accuracy of coarse level correction, it is also worth considering the following W-cycle alternative. In practice, the best performing preconditioner arises from a multiplicative W-cycle. The approximate inverse of $\mathcal{A}^1$ defined by the following algorithm is denoted $\mathcal{B}_w^1$.

Algorithm 3.5: Multiplicative W-cycle ASP Preconditioner $x \leftarrow \mathcal{B}_w^1 r^1$

1.	Backward GS:	$\Delta x \leftarrow R_b(\mathcal{A}^1) r^1$
2.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow \Delta x$
3.	Auxiliary spaces:	$\Delta x \leftarrow G\mathcal{B}_n^0 G^T r^1 + \sum_{i=x,y,z} \Pi_i \mathcal{B}_i^0 \Pi_i^T r^1$
4.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow x + \Delta x$
5.	Backward GS:	$\Delta x \leftarrow R_b(\mathcal{A}^1) r$
6.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow \Delta x$
7.	Auxiliary spaces:	$\Delta x \leftarrow G\mathcal{B}_n^0 G^T r^1 + \sum_{i=x,y,z} \Pi_i \mathcal{B}_i^0 \Pi_i^T r^1$
8.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow x + \Delta x$
9.	Forward GS:	$\Delta x \leftarrow R_f(\mathcal{A}^1) r^1$
10.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow x + \Delta x$
11.	Auxiliary spaces:	$\Delta x \leftarrow G\mathcal{B}_n^0 G^T r^1 + \sum_{i=x,y,z} \Pi_i \mathcal{B}_i^0 \Pi_i^T r^1$
12.	Residual update:	$r^1 \leftarrow r^1 - \mathcal{A}^1 \Delta x \; ; \quad x \leftarrow x + \Delta x$
13.	Forward GS:	$\Delta x \leftarrow R_f(\mathcal{A}^1) r^1$
14.	Solution update:	$x \leftarrow x + \Delta x$

3.6 ASP Preconditioners for the T- Ω Method

The matrix equation of the $\mathbf{T}-\Omega$ formulation (Section 2.5.2) at the lowest level, $\mathcal{A}^1 x = r^1$, is further partitioned into rows and columns corresponding to **T** and $\nabla\Omega$ basis functions:

$$\begin{bmatrix} A_{TT} & A_{T\Omega} \\ A_{\Omega T} & A_{\Omega\Omega} \end{bmatrix} \begin{Bmatrix} x_T \\ x_\Omega \end{Bmatrix} = \begin{Bmatrix} r_T \\ r_\Omega \end{Bmatrix} \tag{3.52}$$

Following the idea behind ASP, the problem is transferred to a number of auxiliary function spaces. For the unknown **T,** represented by Whitney edge elements over the edges of the mesh, the same auxiliary spaces N and N^3 are involved. Equivalent problems, similar to (3.40) and (3.41) are constructed for the **T** part of the field. However, since this exists only inside the conducting regions and vanishes tangentially on the surface, the corresponding auxiliary spaces are defined by just the nodes interior to Ω_c.

We define matrices Π_n and $\Pi = [\Pi_x \quad \Pi_y \quad \Pi_z]$ that map column vectors representing functions in N and N^3, respectively, to column vectors of the same size as r^1 that represent $\mathbf{T}-\Omega$ fields of order 1. Specifically, they map to the **T** partition of r^1, representing functions in the Whitney space inside Ω_c. In fact, since there are no edge functions for the tree edges in Ω_c, they map just to *cotree* edges, but the matrix entries are the same as in (3.25) and (3.27) to (3.29).

The $\nabla\Omega$ part of the field also has to be accomodated and for this we use the auxiliary space that is spanned by the piecewise linear scalar functions of the whole mesh (not just the mesh in Ω_c). The matrix for this space is Π_Ω. Like Π_n, it is a discrete representation of the gradient operator.

The matrices $\mathcal{A}_a^0$, $a = x, y, z, n, \Omega$, which are the auxiliary space counterparts of $\mathcal{A}^1$, are given by

$$\mathcal{A}_a^0 = \Pi_a^T \mathcal{A}^1 \Pi_a \tag{3.53}$$

Approximate inverses of these five matrices are also denoted $\mathcal{B}_a^0$, $a = x, y, z, n, \Omega$. We can now construct the ASP algorithms based on V-cycle and W-cycle approximation for solving $\mathcal{A}^1 x = r^1$. The effective multiplicative preconditioners are very similar to Algorithm 3.4 and Algorithm 3.5, but with 5 auxiliary space

corrections instead of 4. For example, in the multiplicative V-cycle preconditioner, line 3 of Algorithm 3.4 is modified as follows:

$$\Delta x \leftarrow G\mathcal{B}_n^0 G^T r^1 + \sum_{i=x,y,z} \Pi_i \mathcal{B}_i^0 \Pi_i^T r^1 + \Pi_\Omega \mathcal{B}_\Omega^0 \Pi_\Omega^T r^1 \tag{3.54}$$

and similarly for the W-cycle approximation in lines 3, 7 and 11.

3.7 Standard AMG Methods for Poisson Problems: Nodal AMG

Multigrid methods are well known to be cost effective for elliptic problems. AMG is seen by many as one of the most promising methods for solving the large-scale problems that arise in this context [74] [75] [77]. Here we present the basic principles and techniques for solving nodal problems using AMG. Consider a problem of the form

$$\mathcal{A}_i^0 x = r^0, \qquad i = n, x, y, z, \Omega \tag{3.55}$$

The approximate inverses of these four nodal matrices that were denoted by $\mathcal{B}_i^0$ $(i = n, x, y, z, \Omega)$ in the additive and multiplicative algorithms are never formed explicitly. Instead, the product $\mathcal{B}_i^0 r^0$ is found by scalar AMG:

$$\left(\mathcal{A}_i^0\right)^{-1} r^0 \cong \mathcal{B}_i^0 r^0 = scalarAMG\left(i, r^0, 0\right) \tag{3.56}$$

The third argument, 0, is the AMG level number (see below). The central idea in scalar AMG is that error components not eliminated by relaxation must be removed by coarse-grid correction. Simple relaxations reduce high frequency error components efficiently, but they are very slow at removing low energy components. However, the error that remains after relaxation can be approximated more accurately on a coarser grid. This is done by solving the residual equation on a coarser grid, then interpolating the error back to the fine grid and using it to correct the fine-grid approximation. The coarse-grid problem itself can be solved by a recursive application of this method. One iteration of this process, proceeding through all levels, results in the multigrid cycle.

In AMG the main task is to determine a coarsening process that approximates the error that a chosen relaxation cannot reduce. The grid that appropriately represents coarse degrees of freedom for the original problem is called a “virtual” finite element mesh. In order to construct and connect these virtual grids together, several

components are required. Using superscripts to indicate the level number, so that $\mathcal{T}^0$ correspond to the real tetrahedral mesh, the components that AMG needs are as follows:

1) (virtual) Grids: $\mathcal{T}^{-q} \subset \cdots \mathcal{T}^{-2} \subset \mathcal{T}^{-1} \subset \mathcal{T}^0$.
2) A relaxation scheme for each level, like Gauss-Seidel iteration.
3) Grid transfer (prolongation) operators: P_i^{-q+1},… , P_i^{-1} , P_i^0, which are sparse, rectangular matrices for relating the nodes of a coarse grid to the nodes of a finer grid.
4) Grid operators: $\mathcal{A}_i^{-q}$, … $\mathcal{A}_i^{-2}$, $\mathcal{A}_i^{-1}$, $\mathcal{A}_i^0$, which are a series of progressively smaller, square, sparse matrices. These matrices are defined using the prolongation operators:

$$\mathcal{A}_i^{-l-1} = \left(P_i^l\right)^T \mathcal{A}_i^{-l} P_i^l \ , l = 0,1, \dots, q-1$$

Once these components are defined, the recursively defined cycle of scalar AMG is as follows:

Algorithm 3.6: scalar AMG $x \leftarrow scalarAMG(i,r,l) = \mathcal{B}_i^l r \cong \left(\mathcal{A}_i^l\right)^{-1} r$

1.	If $l = -q$ (the coarsest level considered):	
2.	Solve exactly:	$x \leftarrow \left(\mathcal{A}_i^{-q}\right)^{-1} r$
3.	Else:	
4.	Backward GS:	$\Delta x \leftarrow R_b\left(\mathcal{A}_i^l\right) r$
5.	Residual update:	$r \leftarrow r - \mathcal{A}_i^l \Delta x$; $x \leftarrow \Delta x$
6.	Coarsen:	$r^c \leftarrow \left(P_i^l\right)^T r$
7.	Coarse solution:	$\Delta x^c \leftarrow scalarAMG(i, r^c, l-1)$
8.	Prolongate:	$\Delta x \leftarrow P_i^l \Delta x^c$
9.	Residual update:	$r \leftarrow r - \mathcal{A}_i^l \Delta x$; $x \leftarrow x + \Delta x$
10.	Forward GS:	$\Delta x \leftarrow R_f\left(A_i^l\right) r$
11.	Solution update:	$x \leftarrow x + \Delta x$

The preparation of the components of AMG is done in a separate setup phase. In the setup phase, before the start of the Krylov iteration, the matrices $\mathcal{A}_i^0$ are formed explicitly, and then the AMG matrices P_i^l and $\mathcal{A}_i^{-l}$. These quantities are used repeatedly throughout the iteration.

Algorithm 3.7: AMG setup Phase

1.	Set $k = 0$
2.	Partition $\mathcal{T}^k$ into disjoint sets C^k and F^k.
3.	Set $\mathcal{T}^{k-1} = C^k$
4.	Define prolongation matrices P_i^k

5. Set $\mathcal{A}_i^{-k-1} = \left(P_i^k\right)^T \mathcal{A}_i^{-k} P_i^k$
6. If $\mathcal{T}^{k-1}$ is small enough for factorizing $\mathcal{A}_i^{-k-1}$, set $q = -k+1$ and stop. Otherwise, set $k = k-1$ and go to step 2.

The details of steps 2 and 4 are given in [17].

Adding more levels reduces the memory required for the coarsest level exact factorization, but sacrifices the performance if coarsening is too great. Generally, adding more levels should be continued provided that the virtual AMG meshes can adequately represent field variations on the scale of the wavelength (for wave problems). We can estimate the "edge length" on a virtual mesh as $h_{avg}\sqrt[3]{n_n^f/n_n^c}$, where h_{avg} is the average edge length in the original mesh and n_n^f/n_n^c is the ratio of the number of nodes in the original mesh to the number of nodes in the virtual mesh. A value of $0.1\lambda_0$ for the above measure has shown satisfactory performance in our numerical results (see Chapter 5).

3.8 Multilevel and Algebraic Multigrid Cycles: *pMUSASP*

In this section, I introduce the Multilevel/ASP/AMG version of the *p*MUS method explained in Algorithm 3.2 for solving (3.2). The approach is called $pMUSASP$ and employs the AMG treatment at the lowest level instead of the direct solve. The function $pMUSASP(\mathcal{A}^l, r^l)$ returns an approximation to $(\mathcal{A}^l)^{-1}r^l$, where r^l is a vector of the appropriate length. It involves primarily the application of forward and backward Gauss-Seidel (GS). The algorithm $pMUSASP(\mathcal{A}^l, r^l)$ is given recursively as follows:

Algorithm 3.8: Multilevel/Algebraic Multigrid *p*MUS: $x \leftarrow pMUSASP(\mathcal{A}^l, r^l)$

1. If $l = 1$:
2. Solve order 1: $x \leftarrow ASP(r^1)$
3. Else if $l > 2$:
4. Backward GS: $\Delta x_l \leftarrow R_b(A_{ll})r_l$; $x \leftarrow \Delta x_l$
5. Residual update: $r^l \leftarrow r^l - \mathcal{A}^l x$
6. Coarsen: $r^{l-1} \leftarrow (P^l)^{\mathrm{T}} r^l$,
7. Solve coarse: $\Delta x \leftarrow pMUSASP(r^{l-1}, l-1)$
8. Prolongate: $\Delta x \leftarrow P^l \Delta x^{l-1}$; $x \leftarrow x + \Delta x$
9. Residual update: $r^l \leftarrow r^l - \mathcal{A}^l x$
10. Forward GS: $\Delta x_l \leftarrow R_f(A_{ll})r_l$; $x \leftarrow x + \Delta x_l$

The vectors Δx_l and r_l are blocks of Δx and r^l, respectively, corresponding to the partitions in (3.2). The operation $x \leftarrow ASP(r^1)$ is one of the algebraic approximations described by Algorithms 3.3 to 3.5, i.e. $x \leftarrow \mathcal{B}_a^1 r^1$, $x \leftarrow \mathcal{B}_v^1 r^1$ or $x \leftarrow \mathcal{B}_w^1 r^1$.

The call $pMUSASP(\mathcal{A}^p, r^p)$ with $ASP(r^1) = \mathcal{B}_v^1 r^1$ (Algorithm 3.4), then, initiates the V-cycle shown in Figure 3.3, during which GS is applied to the sequence of matrices shown – backward GS on the downward portion, and forward GS on the upward portion. This completes the V-cycle approximation of $(\mathcal{A}^p)^{-1} r^p$. The output is used in every preconditioning step of a Krylov solver that solves $\mathcal{A}^p x^p = b^p$.

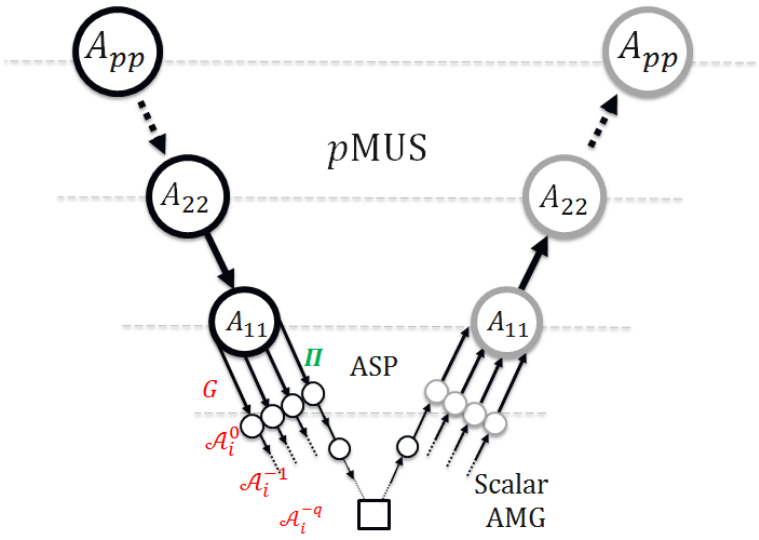


Figure 3.3: The V-cycle version of $pMUSASP(r, p)$. Dashed arrows imply a series of steps with decreasing (downward) or increasing (upward) matrix superscripts.

Since the most computationally intensive part of the preconditioner is pMUS, it has been found beneficial to execute the ASP/AMG part more than once in each cycle, because this reduces the number of Krylov iterations at relatively small additional cost. This gives the extended W-cycle shown in Figure 3.4, in which Algorithm 3.5 for $ASP(r^1) = \mathcal{B}_w^1 r^1$ is used. Note that the backward and forward GS has been arranged to preserve the symmetry of the overall preconditioner. Other symmetric arrangements are also possible.

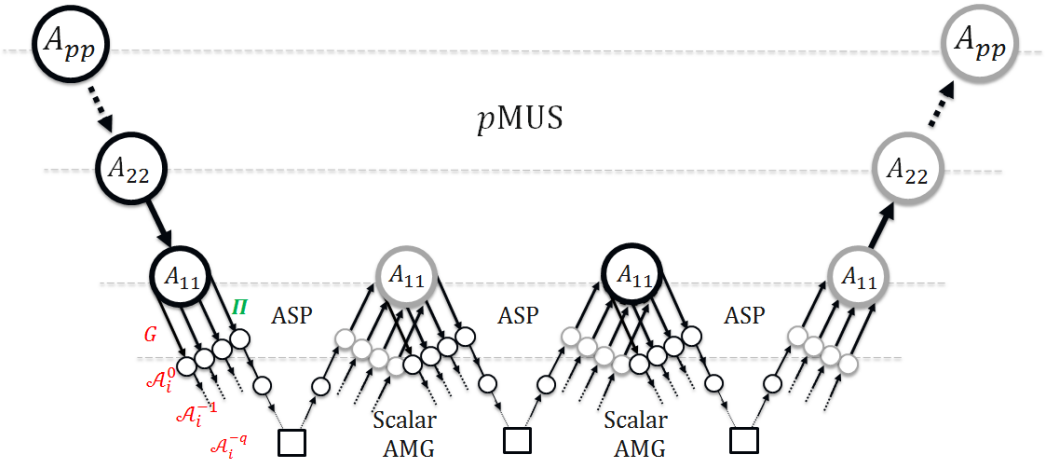


Figure 3.4: The W-cycle version of $pMUSASP(r,p)$.

The Multilevel/ASP/AMG version of pMUS for the $\mathbf{T}-\Omega$ problem is very similar to Figure 3.3 and Figure 3.4. The only difference is that the ASP/AMG parts involve parallel approximation of five nodal systems instead of four.

CHAPTER 4

Krylov Methods and Deflation

In this chapter, Krylov subspace methods for solving the $n \times n$ complex-symmetric linear system of the form

$$Ax = b \tag{4.1}$$

are presented. Matrix A and vector b correspond to the adaptive systems (2.36) or (2.45) with the block structure pattern defined in (3.3) . Krylov subspace methods are extensively used for the iterative solution of large linear systems of equations and in particular those arising from discretizations of Maxwell's equations. We begin our discussion in Section 4.1 by introducing the Krylov subspace approach and discussing methods based on Lanczos and Arnoldi algorithms. In Section 4.2, preconditioned Krylov methods for complex symmetric matrices are presented by extending the standard conjugate gradient approach. Next, in Section 4.3 the method of Generalized Minimal Residual is explained. Section 4.4 introduces the concept of deflation and how it can be used in designing more robust Krylov subspace iterative methods.

4.1 Krylov Subspace Methods

The Krylov subspace iteration methods are based on the construction of a Krylov subspace, i.e.

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, A^2r_0, \cdots, A^{m-1}r_0\} \tag{4.2}$$

where $r_0 = b - Ax_0$ is the residual corresponding to the initial guess vector x_0. The dimension of $\mathcal{K}_m$ is equal to m and increases by one at each step of the approximation process.

The idea of designing iterative methods based on the Krylov subspace can be outlined as follows. For an initial solution x_0, approximation x_m to the solution of (4.1) is computed in the mth iteration as

$$x_m \in x_0 + \mathcal{K}_m(A, r_0), \qquad m > 1 \tag{4.3}$$

For this purpose, all the Krylov subspace methods construct the basis vectors $v_1, v_2, \dots, v_m$ where

$$V_m = [v_1, v_2, \dots, v_m] \in \mathcal{K}_m(A, r_0) \tag{4.4}$$

With the residual $r_m = b - Ax_m$, the expression for the solution and residual at the mth step can be written as:

$$x_m = x_0 + V_m z_m \tag{4.5}$$

$$r_m = r_0 - AV_m z_m \tag{4.6}$$

where $z_m \in \mathbb{C}^n$. From (4.5) and (4.6) we observe that Krylov subspace methods rely on constructing the basis of V_m and the vector z_m. The differences among the various algorithms are the ways of computing these vectors. If the components can be selected properly, the approximate solution x_m can be very close to the exact solution for even a small number of iterations ($m \ll n$).

In order to determine the V_m subspace, many algorithms have been proposed. In general we identify two methods that can be used for constructing the basis: the Symmetric Lanczos method and the Arnoldi method. In the following sections, we present some Krylov subspace algorithms based on these methods that are used in numerical simulations in this thesis. In particular, we discuss the Conjugate Gradient (CG) and its extension to the complex systems, Minimal Residual (MINRES) and Generalized Minimal Residual (GMRES) algorithms.

We end this section by describing some of the notation. For complex vector $x \in \mathbb{C}^n$, the complex-conjugate transpose is denoted by x^H. By $\langle x, y\rangle = x^H y$ we denote an inner product between the vectors $x, y \in \mathbb{C}^n$. In this case, the induced vector norm is $\|x\| = \sqrt{\langle x, x\rangle} \geq 0$. We also use a different bilinear functional denoted by $(x, y) = x^T y$ which is used to derive a CG-like method for complex symmetric matrices. When $(x, y) = 0$, the complex vectors x and y are called conjugate orthogonal. The unit vector $e_m \in \mathbb{C}^m$ is defined by $e_m = [0, \ldots, 0, 1]^T$. For a vector z that is partitioned as (3.3), we refer to the first partition as $z_{[1]}$.

4.2 The Symmetric Lanczos Process

Here we assume the matrix A to be Hermitian. Given A and b, the symmetric Lanczos process [15] computes a sequence of Lanczos vectors v_i and scalars γ_i and η_i for $i = 1, 2, \cdots$, starting from $\eta_1 = \|r_0\|$ and $v_1 = r_0/\eta_1$. Defining $v_0 \triangleq 0$, the process satisfies the following three-term recurrence:

$$\eta_{i+1} v_{i+1} = A v_i - \gamma_i v_i - \eta_i v_{i-1}, \quad i = 1, 2, \cdots \tag{4.7}$$

in which

$$\gamma_i = v_i^H A v_i, \quad i = 1, 2, \cdots \tag{4.8}$$

and $\eta_{i+1} \geq 0$ chosen such that

$$\|v_{i+1}\| = 1, \quad i = 1, 2, \cdots \tag{4.9}$$

In exact arithmetic, the vectors v_i are orthogonal (i.e. $\langle v_i, v_j\rangle = 0$ for $i \neq j$) and the process stops when $\eta_{m+1} = 0$ for some $m \leq n$. In matrix form, for $i = 1, 2, \cdots, m$, (4.7) is:

$$A V_m = V_m T_m + \eta_{m+1} v_{m+1} e_m^H \tag{4.10}$$

where $V_m = [v_1, v_2, \ldots, v_m]$ and T_m is the following tridiagonal matrix:

$$T_m = \begin{bmatrix} \gamma_1 & \eta_1 & & & \\ \eta_1 & \gamma_2 & \eta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \eta_{m-1} & \gamma_{m-1} & \eta_m \\ & & & \eta_m & \gamma_m \end{bmatrix} \tag{4.11}$$

An important property of the Lanczos vectors in V_m is that they lie in the Krylov subspace $\mathcal{K}_m(A, r_0)$ defined by (4.4). Hence, at iteration m, we look for an approximate solution $x_m = x_0 + V_m z_m$. The associated residual vector is

$$r_m = b - Ax_m = \eta_1 v_1 - AV_m z_m \tag{4.12}$$

By choosing z_m in a way to make r_m small, we can arrive at different iterative methods for solving the linear system. Since V_m is theoretically orthonormal, we can find z_m by solving

$$T_m z_m = \eta_1 e_1 \tag{4.13}$$

Two particular ways of solving (4.11) lead to CG [85] and MINRES [85], which are established methods for symmetric systems. They both exploit the three-term recurrent scheme for delivering an efficient Krylov subspace method for symmetric systems, but each method has a different minimization property that suggests a particular factorization of T_m.

4.2.1 Conjugate Gradients for Complex Symmetric Systems

The Conjugate Gradient (CG) method was introduced for solving $Ax = b$ when A is Hermitian Positive Definite (HPD). CG iterations are characterized by minimizing $\|x - x_m\|_A$, where

$$\|u\|_A = \sqrt{\langle Au, u \rangle} \tag{4.14}$$

The algorithm can be derived directly from its equivalent Lanczos formulation that computes the solution of the tridiagonal system $T_m z_m = \eta_1 e_1$ progressively by using Gaussian elimination without pivoting [15]. The LU factorization of matrix T_m can be written in the form

$$T_m = L_m U_m = \begin{bmatrix} 1 & & & & \\ \lambda_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \lambda_{m-1} & 1 & \\ & & & \lambda_m & 1 \end{bmatrix} \begin{bmatrix} \delta_1 & \eta_1 & & & \\ & \delta_2 & \eta_2 & & \\ & & \ddots & \ddots & \\ & & & \delta_{m-1} & \eta_m \\ & & & & \delta_m \end{bmatrix} \tag{4.15}$$

and the approximate solution at mth step is :

$$x_m = x_0 + V_m U_m^{-1} L_m^{-1}(\eta_1 e_1) \tag{4.16}$$

Letting

$$P_m = V_m U_m^{-1} \tag{4.17}$$

and

$$y_m = L_m^{-1}(\eta_1 e_1) \tag{4.18}$$

then (4.16) is

$$x_m = x_0 + P_m y_m \tag{4.19}$$

However, computation of (4.19) can be done progressively, since the last column of the matrix P_m, shown by p_m, and last entry of vector y_m, shown by ς_m, can be obtained from the previous values following simple updates:

$$p_m = \delta_m^{-1}(v_m - \eta_m p_{m-1}) \tag{4.20}$$

$$\varsigma_m = -\lambda_m \varsigma_{m-1} \tag{4.21}$$

As a result, x_m can be updated at each step as

$$x_m = x_{m-1} + \varsigma_m p_m \tag{4.22}$$

and the residual vector of the approximate solution x_m is such that

$$r_m = -\eta_{m+1} e_m^H z_m v_{m+1} \tag{4.23}$$

The above relations give us an algorithm which is usually called the direct version of the Lanczos algorithm [15]. The residual vectors for this algorithm are orthogonal to each other due to equation (4.23) and the vectors p_m are also A-orthogonal, meaning that $P_m^H A P_m$ is a diagonal matrix [15]. As a consequence, an efficient version of the conjugate gradient algorithm (in terms of required storage) can be derived by imposing these orthogonalities, in which based on (4.22) the new approximation vector x_{i+1} at every iteration is expressed as

$$x_{i+1} = x_{i+1} + \alpha_i p_i \tag{4.24}$$

where p_i is the search direction and

$$p_{i+1} = r_{i+1} + \beta_i p_i \tag{4.25}$$

The residual vectors then satisfy the recurrence

$$r_{i+1} = r_i - \alpha_i A p_i \tag{4.26}$$

For all r_i to be orthogonal, the necessary condition is that $\langle r_{i+1}, r_i \rangle = 0$. Thus

$$\langle r_i - \alpha_i A p_i, r_i \rangle = 0 \tag{4.27}$$

Since the next search direction p_{i+1} is a linear combination of r_{i+1} and p_i, we find that

$$\alpha_i = \frac{\langle r_i, r_i \rangle}{\langle A p_i, p_i \rangle} \tag{4.28}$$

Also because $\langle A p_{i+1}, p_i \rangle = 0$, we have

$$\beta_i = \frac{\langle r_{i+1}, r_{i+1} \rangle}{\langle r_i, r_i \rangle} \tag{4.29}$$

Equations (4.24) to (4.29) completely describe all the necessary calculations at every iteration of the CG algorithm. The implementation requires only one product Ap_i for some vector p_i and a small amount of computations for computing the vectors x_{i+1}, r_{i+1} and p_{i+1}. If A is HPD, the norm of the residual vector $\|r_{i+1}\|$ also reduces monotonically at each iteration, which makes CG highly efficient and preferable. However, for non-Hermitian matrices, like the global FE matrix of the vector wave equation, the algorithm may not converge. This is because the orthogonality condition (4.27) cannot be satisfied in general.

One simple remedy for complex-symmetric systems ($A = A^T$) is to replace the orthogonality condition by the so-called conjugate orthogonality condition based on the following inner product

. $$(x, y) = x^T y \tag{4.30}$$

In this case, the vectors $r_0, r_2, \dots, r_{m-1}$ also form a basis for $\mathcal{K}_m(A, r_0)$. Furthermore, the vectors r_i are conjugate orthogonal:

. $$\left(r_i, r_j\right) = 0 \text{ if } i \neq j \tag{4.31}$$

Therefore, solution x_{i+1} can be constructed by requiring the residual r_{i+1} to be conjugate orthogonal to $\mathcal{K}_m(A, r_0)$. This results in an algorithm which is similar to CG, but with the Hermitian inner product being replaced by (4.30). The algorithm, called conjugate orthogonal-conjugate gradient (COCG) is as follows:

Algorithm 4.1: Conjugate Orthogonal Conjugate Gradient (COCG)

1. Set the initial guess: x_0. Compute $r_0 = b - Ax_0$.
2. Set $p_0 = r_0$
3. **for** $i = 0,1, \dots m$
4. $\alpha_i = (r_i, r_i)/(Ap_i, p_i)$
5. $x_{i+1} = x_i + \alpha_i p_i$
6. $r_{i+1} = r_i - \alpha_i A p_i$ (Stop if accurate enough)
7. $\beta_i = (r_{i+1}, r_{i+1})/(r_i, r_i)$
8. $p_{i+1} = \beta_i p_i + z_{i+1}$
9. **End**

Algorithm 4.1 also preserves the nice property that it requires only one matrix-vector multiplication and a few vector updates per each iteration. However, it is no longer optimal in a sense that the error at every step is not minimal. This means that the error is not monotonically decreasing. The condition $A = A^T$ is also restrictive in that it only allows incorporation of a symmetric preconditioner.

4.2.2 Preconditioned COCG

The convergence of Krylov methods like COCG might be very slow for the type of equations defined in (4.1), which are generally ill-conditioned systems. A Krylov subspace method is usually combined with a preconditioner to reduce the total number of iterations. Iterative solutions can be accelerated if well-chosen preconditioners are available. Our objective in Chapter 3 was to find appropriate (symmetric) preconditioners which can easily be used with Krylov subspace methods and also reduce the total number of iterations.

Suppose that M represents a preconditioner whose inverse is easy to compute (like the multilevel/algebraic multigrid in Algorithm 3.8). By employing the preconditioner, instead of solving the original linear system (4.1), we solve a better conditioned system

$$M^{-1}Ax = M^{-1}b \tag{4.32}$$

Therefore, the preconditioned Krylov subspace method is defined by constructing iteration x_m such that

$$x_m \in x_0 + \mathcal{K}_m(M^{-1}A, r_0), \qquad m > 0 \tag{4.33}$$

One important aspect of incorporating a symmetric preconditioner is that a preconditioned version of COCG can be derived in a straightforward way. Supposing that M has Cholesky factorization LL^T and by redefining the variables

$$Lr_i \to r_i,\ L^{-T}p_i \to p_i,\ L^{-T}x_i \to x_i \tag{4.34}$$

we can directly apply COCG to (4.32), yielding the following algorithm:

Algorithm 4.2: Preconditioned Conjugate orthogonal Conjugate Gradient (PCOCG)

1. Set the initial guess: x_0. Compute $r_0 = b - Ax_0$
2. $z_0 = M^{-1}r_0$, $p_0 = z_0$
3. **for** $i = 0,1,\ldots m$
4. $\quad \alpha_i = (r_i, z_i)/(Ap_i, p_i)$
5. $\quad x_{i+1} = x_i + \alpha_i p_i$
6. $\quad r_{i+1} = r_i - \alpha_i Ap_i$ (Stop if accurate enough)
7. $\quad z_{i+1} = M^{-1}r_i$
8. $\quad \beta_i = (r_{i+1}, z_{i+1})/(r_i, z_i)$
9. $\quad p_{i+1} = \beta_i p_i + z_{i+1}$
10. **End**

In fact, the iterates produced by the above algorithm and Algorithm 4.1 applied to the system $L^{-1}AL^{-T}x = L^{-1}b$ are identical, provided that the same initial guess is used.

If A is very far from the nearest positive definite matrix, Algorithm 4.2 may converge slowly. When A is ill-conditioned, the Cholesky factorization of T_m might fail or be numerically unstable. A better approach is to use a more numerically stable factorization for solving (4.13).

4.2.2 Conjugate Orthogonal Minimal Residual, CO-MINRES

The CG method and the minimal residual method (MINRES) are both Krylov subspace techniques for the iterative solution of Hermitian linear equations. While CG is commonly used when the matrix A is positive definite, MINRES is generally reserved for indefinite systems. MINRES [11] [85] is characterized by the following minimization:

$$x_m = V_m z_m \text{ where } z_m \text{ minimizes } \|b - AV_m z_m\| \tag{4.35}$$

Thus, MINRES minimizes $\|r_m\|$ within the mth Krylov subspace. Since this minimization is well-defined regardless of the definiteness of A, MINRES is

applicable to both positive definite and indefinite Hermitian systems. It applies the process of QR factorization ($T_m = Q_m R_m$) during iterations to find z_m. A more in-depth discussion of its properties is given in [11].

In the present case, it is modified to handle complex symmetric A, rather than Hermitian A. Direct implication of the conjugate orthogonality concept to the standard MINRES method results in an algorithm, called Conjugate Orthogonal Minimal Residual (CO-MINRES). Similar to the modification considered for derivation of COCG, standard MINRES can also be adapted by replacing every instance of the Hermitian operator by a simple transpose [10]. The preconditioned CO-MINRES algorithm based on the MATLAB implementation [87] is given as follows.

Algorithm 4.3: Precondtioned Conjugate Orthogonal MINRES

1. Set the initial guess: x_0, Compute $r_0 = b - Ax_0$.
2. $y_0 = M^{-1} r_0$
3. $\eta_0 = \sqrt{(r_0, y_0)}$, $\tau_1 = \eta_0$, $\tau_0 = 0$, $d = 0$, $e = 0$, $\phi_1 = \tau_1$, $c = -1$, $s = 0$
4. Set: $\omega = 0$, $\omega_2 = 0$, $u_1 = r_0$, $u_2 = r_0$
5. **for** $i = 0,1, \ldots m-1$
6. $\quad v_{i+1} = y_i / \eta_i$
7. $\quad y_{i+1} = A v_{i+1}$
8. $\quad$ If $i \geq 1$, $y_{i+1} = y_{i+1} - (\eta_i / \tau_0) u_1$
9. $\quad \gamma_{i+1} = (v_{i+1}, y_{i+1})$
10. $\quad y_{i+1} = y_{i+1} - (\gamma_{i+1} / \eta_i) r_2$
11. $\quad u_1 = u_2$, $u_2 = v_{i+1}$
12. $\quad y_{i+1} = M^{-1} u_2$
13. $\quad \eta_{i+1} = \sqrt{(u_2, y_{i+1})}$
14. $\quad \tau_0 = \eta_i, e_0 = e$, $\delta = cd + s\gamma_{i+1}$, $g = sd - c\gamma_{i+1}$, $e = s\eta_{i+1}$, $d = -c\eta_{i+1}$
15. $\quad \chi = \sqrt{g^2 + \eta_{i+1}^2}$, $c = g/\chi$, $s = \eta_{i+1}/\chi$, $\phi = c\phi_1$, $\phi_1 = s\phi_1$
16. $\quad \omega_1 = \omega_2$, $\omega_2 = \omega$
17. $\quad \omega = (v - e_0 \omega_1 - \delta \omega_2)\, \chi^{-1}$
18. $\quad x_{i+1} = x_i + \phi \omega$
19. $\quad r_{i+1} = r_i - (\phi / \chi) u_2$ (Stop if residual norm is small enough)
20. **End**

In general, Algorithm 4.3 is more expensive than Algorithm 4.2. However, MINRES converges faster and also provides more information: the key components of the symmetric Lanczos process, which are the orthogonal vectors v_{i+1} as well as scalars γ_{i+1} and η_{i+1} defined in Section 4.2, can be extracted easily from lines 6, 9

and 13 of the above algorithm. These provide information about the coefficient matrix and can be used in modified Krylov methods (explained in Section 4.4).

4.3 Generalized Minimal Residual, GMRES

The Generalized Minimal Residual (GMRES) method is an iterative method for general non-symmetric matrices (and preconditioners), which minimizes the residual norm over the Krylov subspace. The method is based on the Arnoldi approach [15] for full orthogonalization and is given as follows:

Algorithm 4.4: Generalized Minimal Residual (GMRES)

1. Set the initial guess: x_0, Compute $r_0 = b - Ax_0$
2. Set $\beta = \|r_0\|_2$ and $v_1 = r_0/\beta$
3. **for** $i = 1,2, \dots m$
4. $\quad \omega_i = Av_i$
5. $\quad$ **for** $j = 1,2, \dots m$
6. $\quad\quad h_{j,i} = \langle \omega_i, v_j \rangle$, $\omega_i = \omega_i - h_{ji}v_j$
7. $\quad$ **End**
8. $\quad h_{i+1,i} = \|\omega_i\|_2$
9. $\quad v_{i+1} = \omega_i / h_{i+1,i}$
10. **End**
11. Compute z_m, the minimizer of $\|\beta e_1 - H_m z_m\|_2$ and $x_m = x_0 + V_m z_m$
12. **End**

Lines 1 to 10 of the above algorithm are Arnoldi iterations for creating a fully orthogonal subspace, in which every new vector v_{i+1} is orthogonalized to the all previous vectors in V_i. The approximate solution in GMRES is then obtained by computing the vector z_m by a sophisticated version of the QR decomposition of the matrix H_m, such that the 2 norm of the residual $r_m = b - Ax_m$ is minimized over $\mathcal{K}_m$. See [85] for a preconditioned version of GMRES.

Although the GMRES method has a good convergence performance, it requires storing the entire sequence of vectors v_i and the memory requirement increases as the number of iterations increases. If the iteration number m is large, the GMRES algorithm becomes impractical because of memory and computational requirements. To remedy this problem, the algorithm can be restarted. The restarted GMRES follows the idea of the original GMRES, except that after some steps, the algorithm is repeated by a new initial guess.

4.4 Krylov Deflation

Incorporation of a good preconditioner is one of the most widely used approaches to accelerate the speed of convergence for Krylov methods. By applying the preconditioner, the spectrum of A is modified with the general goal of obtaining a matrix that is close to the identity matrix. In this section, we consider an approach for convergence acceleration that is different from preconditioning, called *deflation*. Here the search space of the Krylov subspace method is enlarged by a suitably chosen subspace that contains useful information about the system that is not adequately addressed by the action of the preconditioner.

The need for deflation comes from the fact that the convergence of any Krylov subspace method is affected by the spectrum of the matrix A, especially the largest and smallest eigenvalues. It is clear that CG methods converge more slowly for a matrix that has small eigenvalues, even after applying a preconditioner.

The Multilevel/ASP/AMG preconditioning cycles described in the previous chapter try to capture error components related to the highest and lowest eigenvalues successively, by construction of a multilevel hierarchy. This hierarchy was defined by transfer operators between the higher order levels and continued with virtual grid transfer operators below the Whitney level. The cycle aims to transfer error parts invariant to the relaxation to the next lowest (coarser) level. Errors that remain after the smoother has been applied and that must be reduced at the next level are called "*algebraically smooth*". Algebraically smooth components belong to the space of eigenvectors of A that have small eigenvalues and for rapid convergence, accurate representation of them on the coarser level is needed. The nature of the algebraically smooth errors and the near null space of A affect the performance of the preconditioned Krylov solver and degrade the convergence.

A frequently effective remedy for slow convergence is to directly deflate the troublesome eigenvalues (eigenvectors) from the problem. This can be done with construction of an auxiliary approximation subspace that is specifically designed to target the problematic modes. Numerous techniques can be used to remove the problematic eigenvectors. Here, the deflated Krylov approach is used. The goal of deflated Krylov is to remove from the system the eigenspace corresponding to troublesome, algebraically smooth, eigenvectors. As will be shown, it permits the

incorporation of existing knowledge about the space of troublesome eigenvectors available from a previous calculation, or a nearby problem.

Deflation is often used for solving one particular matrix with multiple right-hand sides vectors. The idea then is to use information from the solution of the first right-hand side to assist subsequent right-hand sides. Here, the deflation idea is tailored specifically to solving a sequence of problems, $Ax = b$, which are generated during the *p*-adaption process. In the next section, a framework on how COCG and CO-MINRES can be improved based on previously-obtained information is described. After that, the case of solving progressively enlarged system of equations during *p*-adaption is considered.

4.4.1 A Framework for Deflated Krylov Methods

In this section, we describe a general framework for deflation of iterative methods based on the Lanczos process, which simultaneously covers implementations of several robust Krylov subspace methods. Krylov deflation is in fact a generalization of the aforementioned basic techniques, that in some instances greatly extends their efficiency and robustness. It appeared for the first time in the paper of Nicolaides [88] and a comparable approach was then proposed in [4] and [5]. These techniques glean "important" information from past problems to enrich the subspace.

Let $M = LL^T$ be the Cholesky factorization of the applied preconditioner in line 7 of preconditioned COCG or line 12 of preconditioned CO-MINRES. Here, we assume that the columns of a matrix W are the "deflation vectors" that approximately span the space of slow converging modes of the preconditioned system $\tilde{A}\tilde{x} = \tilde{b}$, where $\tilde{A} = L^{-1}AL^{-T}$, $\tilde{b} = L^{-1}b$ and $\tilde{x} = L^T x$. The deflated Lanczos algorithm [88] starts with an initial vector $v_1 = r_0/\|r_0\|$ orthogonal to W and builds a sequence of vectors $v_{i+1}, i = 1,2,..m-1$ such that [4]:

$$v_{i+1} \perp \text{span}\{W, v_1, v_2, \dots, v_i\}\ ;\ \|v_{i+1}\| = 1 \tag{4.36}$$

The subspace $V_m' = [v_1, v_2, \dots, v_m]$ can be obtained by applying the standard Lanczos procedure to the auxiliary matrix

$$\hat{A} = \tilde{A} - \tilde{A}W\tilde{U}W^T\tilde{A} \tag{4.37}$$

where $\widetilde{U} = \left(W^T \tilde{A} W\right)^{-1}$. If the matrix $W^T \tilde{A} W$ is singular, its pseudo-inverse is considered. To satisfy $v_1 \perp W$, the residual vector $r_0 = \tilde{b} - \tilde{A}\tilde{x}_0$ should be associated with a special initial guess

$$\tilde{x}_0 = \tilde{x}_{-1} + W\widetilde{U}W^T\left(\tilde{b} - \tilde{A}\tilde{x}_{-1}\right) \tag{4.38}$$

for some arbitrary $\tilde{x}_{-1}$. The resulting Krylov subspace at the mth step is denoted by $\mathcal{K}_m(\tilde{A}, W, v_1)$ and the algorithm seeks an approximate solution $x_i \in x_0 + \mathcal{K}_m(\tilde{A}, W, v_1)$ by requiring that the residual be orthogonal to $\hat{A}\mathcal{K}_m(\tilde{A}, W, v_1)$. Therefore, the change to the standard symmetric Lanczos algorithm is minimal and can be summarized as follow:

- Based on a given W, pre-compute $\widetilde{U} = \left(W^T \tilde{A} W\right)^{-1}$ and the initial approximation vector using (4.38).
- Construct a symmetric Lanczos space based on the coefficient matrix $\hat{A}$ instead of $\tilde{A}$ by modifying all the matrix-vector products as follows:

$$\hat{A}v = \tilde{A}v - \tilde{A}W\widetilde{U}W^T\tilde{A}v \tag{4.39}$$

As with the standard Lanczos algorithm, the scalar computed during the process can be stored in a tridiagonal matrix, T_m, and the resulting vectors satisfy $\hat{A}V_m \simeq V_m T_m$. The difference here is that since the solution is approximated in the space generated by W and V_m', the inclusion is beneficial in accelerating the convergence, even when W contains only poor approximations of slow-converging eigenvectors.

4.4.2 Preconditioned Deflated COCG

Putting relations (4.36) to (4.39) together and following the similar approach for derivation of standard COCG algorithm, the deflated version of COCG that satisfies all the recurrence relations can be obtained as follows [4]. The main difference in iterations of preconditioned deflated COCG comparing to COCG is that, the regular update of the descent direction vector $p_{i+1} = \beta_i p_i + z_{i+1}$ is changed to $p_{i+1} = \beta_i p_i + z_{i+1} - WUW^T A z_{i+1}$, where z_{i+1} is the preconditioned residual vector. The algorithm is as follows:

Algorithm 4.5: Preconditioned Deflated COCG

1. Set the initial guess: x_{-1}. Compute $r_{-1} = b - Ax_{-1}$
2. Define deflation subspace $W = [w_1, w_2, \dots, w_k]$ and also pre-compute $U = (W^T AW)^{-1}$
3. Set starting vector: $x_0 = x_{-1} + WUW^T r_{-1}$. Compute $r_0 = b - Ax_0$
4. $z_0 = M^{-1} r_0$, $p_0 = z_0 - WUW^T Az_0$
5. **for** $i = 0,1, \dots m$
6. $\alpha_i = (r_i, z_i)/(Ap_i, p_i)$
7. $x_{i+1} = x_i + \alpha_i p_i$
8. $r_{i+1} = r_i - \alpha_i Ap_i$ (Stop if accurate enough)
9. $z_{i+1} = M^{-1} r_{i+1}$
10. $\beta_i = (r_{i+1}, z_{i+1})/(r_i, z_i)$
11. $p_{i+1} = \beta_i p_i + z_{i+1} - WUW^T Az_{i+1}$
12. **End**

In Deflated COCG, we must store W and U in addition to the usual vectors of COCG, which means an additional storage. Deflated-COCG also requires extra multiplications for pre-computing U and calculation of $WUW^T Az_{i+1}$ at each step. However, the cost is much less than that of the preconditioning and the gain in decreasing the number of iterations makes it well worthwhile in general.

4.4.3 Preconditioned Deflated CO-MINRES

The CO-MINRES algorithm can also be modified by inclusion of the W subspace. The preconditioned deflated version of this Krylov method is given next, in which modifications to Algorithm 4.3 occur on lines 2, 3 and 20.

Algorithm 4.6: Preconditioend Deflated CO-MINRES

1. Set the initial guess: x_{-1}. Compute $r_{-1} = b - Ax_{-1}$
2. Define deflation subspace $W = [w_1, w_2, \dots, w_k]$ and also pre-compute $U = (W^T AW)^{-1}$
3. Set starting vector: $x_0 = x_{-1} + WUW^T r_{-1}$. Compute $r_0 = b - Ax_0$
4. $y_0 = M^{-1} r_0$
5. $\eta_0 = \sqrt{(r_0, y_0)}$, $\tau_1 = \eta_0$, $\tau_0 = 0$, $d = 0$, $e = 0$, $\phi_1 = \tau_1$, $c = -1$, $s = 0$
6. Set: $\omega = 0$, $\omega_2 = 0$, $u_1 = r_0$, $u_2 = r_0$
7. **for** $i = 0,1, \dots m - 1$
8. $v_{i+1} = y_i/\eta_i$
9. $y_{i+1} = Av_{i+1}$
10. If $i \geq 1$, $y_{i+1} = y_{i+1} - (\eta_i/\tau_0)u_1$
11. $\gamma_{i+1} = (v_{i+1}, y_{i+1})$

12. $y_{i+1} = y_{i+1} - (\gamma_{i+1}/\eta_i) r_2$
13. $u_1 = u_2, u_2 = v_{i+1}$
14. $y_{i+1} = M^{-1} u_2$
15. $\eta_{i+1} = \sqrt{(u_2, y_{i+1})}$
16. $\tau_0 = \eta_i, e_0 = e, \delta = cd + s\gamma_{i+1}, g = sd - c\gamma_{i+1}, e = s\eta_{i+1}, d = -c\eta_{i+1}$
17. $\chi = \sqrt{g^2 + \eta_{i+1}^2}, c = g/\chi, s = \eta_{i+1}/\chi, \phi = c\phi_1, \phi_1 = s\phi_1$
18. $\omega_1 = \omega_2, \omega_2 = \omega$
19. $\omega = (v - e_0\omega_1 - \delta\omega_2)\, \chi^{-1}$
20. $x_{i+1} = x_i + \phi\omega - \phi WUW^T A\omega$
21. $r_{i+1} = r_i - (\phi/\chi) u_2$ (Stop if residual norm is small enough)
22. **End**

4.5 Eigenvector Estimation

As observed in the algorithms for the deflated Krylov solver, the number of desired eigenvectors k must be chosen, along with the eigenvalues to be targeted. In particular, small eigenvalues related to the near-null space are the most important ones for accelerating the convergence. There are several distinct ways to find approximate eigenvectors that are inadequately clustered by the preconditioner and thus, contribute to poor convergence [4] [5] [89] [90]. One approach is to find pairs (θ_i, w_i) such that the vector $(\tilde{A} - \theta_i I) w_i$ is minimized. Different projection techniques can be applied for the minimization [90]. This approach however requires additional costs in advance for identifying the eigenvalues of $\tilde{A}$ that are problematic for convergence.

A better approach is to get the Ritz vectors by solving a previous and slightly different system [89]. The hierarchical nature of the sequence of linear systems $Ax = b$ generated by p-adaption suggests that there are similarities between the systems. For any higher order system, a considerable portion of the matrix and right-hand side vector are exactly the same as they are for the previous system. In fact, by increasing the orders from step $s - 1$ to s, we can assume that the bulk of the solution vector also remains roughly as it was at step $s - 1$ and mainly higher frequency variations are represented by the newly added unknowns.

From the above reasoning, we assume that deflation information about the adaptive system can be obtained during the solution of the problem at the first adaptive step, when all elements are at their lowest order ($s = p = 1$), and not from a prior knowledge about the problem. For this first step, the solution is obtained by a

CO-MINRES Krylov subspace solver because this preserves better the orthogonality of the Ritz vectors. Determination of approximate eigenvalues to be applied to subsequent adaptive steps is achieved by the following harmonic projection method [4][89].

Given subspace V_m and tridiagonal matrix T_m from the CO-MINRES iterations, the method computes k eigenpairs (θ_i, y_i) by solving the generalized eigenproblem

$$G_m y = \theta F_m y \tag{4.40}$$

where

$$G_m = T_m^T T_m \tag{4.41}$$

$$F_m = T_m^T (V_m^T V_m) \tag{4.42}$$

Defining $Y_k = [y_1, y_2, .., y_k]$, the problematic modes are expressed in the form $W_k = V_m Y_k$. Equation (4.40) can be solved at a low cost by any technique for dense generalized eigenvalue problems. For the results in this thesis, the QZ algorithm in Matlab [87] is used.

4.6 Deflated Krylov for p-Adaption

Consider now the matrix equation $Ax = b$ at the s^{th} adaptive step, $s = 1, 2, ...$, with n^s unknowns. Each matrix A is of the form (3.3), with the same upper left block A_{11}. For $s > 1$, the equation can be solved by deflated COCG with deflation vectors W^s obtained from W_k as follows:

$$W^s = \begin{bmatrix} W_k \\ 0 \end{bmatrix}_{n^s \times k} \tag{4.43}$$

In summary, the procedure for solving p-type hierarchical systems using a deflated preconditioned Krylov solver is as follows:

- *First-order solution:* Set W^1=0 and apply m iterations of CO-MINRES, preconditioned with an appropriate preconditioner M (like the ASP/AMG algorithm described in Chapter 3), to solve $A_{11}x = b$. This computes the matrix V_m that has the Lanczos vectors, and the tridiagonal matrix T_m.
- Select k, the number of eigenvectors to be used.

- *Eigenvector computation:* Find G_m and F_m using (4.41) and (4.42). Solve (4.40) for k eigenvalues and let Y_k be the matrix containing the corresponding vectors. Set $W_k = V_m Y_k$ and compute $U = (W_k^T A_{11} W_k)^{-1}$.

- *Adaption iterations (Algorithm 4.7):* for $s = 2, 3, 4, \ldots$

  Initialize $x_{-1} = 0$ and then transfer into it values from previous solution, with appropriate indexing.

  Set W^s according to (4.43).

  Solve $Ax = b$ by preconditioned deflated COCG with W^s and from initial guess $x_0 = x_{-1} + W^s U (W^s)^T (b - Ax_{-1})$.

From (4.43) we observe that the computational cost of deflation at every iteration is limited to finding just the first n_1 values of Az_i and then multiplying by $W^s U (W^s)^T$. Algorithm 4.7 presents the efficient way of solving adaptive iteration for $s > 1$ in which line 7 to 9 are the required modifications related to deflation for every iteration.

Algorithm 4.7: Preconditioned Deflated COCG for p-adaption system $Ax = b$

1. Initialize x_{-1} from previous adaption step, Compute $r_{-1} = b - Ax_{-1}$
2. Define deflation subspace $W_k = [w_1, w_2, \ldots, w_k]$ and also pre-compute $U = (W_k^T A_{11} W_k)^{-1}$
3. Set $x_0 = x_{-1}$and $\beta_0 = 0$, modify:$x_{0[1]} = x_{0[1]} + W_k U W_k^T r_{-1[1]}$. Compute $r_0 = b - Ax_0$,
4. **for** $i = 0,1, \ldots m$
5. $\quad z_i = M^{-1} r_i$
6. $\quad$ If $i \geq 1$ $\beta_i = (r_{i+1}, z_{i+1})/(r_i, z_i)$
7. $\quad$ **for** $k = 1, \ldots p$
8. $\quad\quad z_{i[1]} = z_{i[1]} - W_k U W_k^T A_{1k} z_{i[1]}$
9. $\quad$ **End**
10. $\quad p_i = z_i + \beta_i p_i$
11. $\quad \alpha_i = (r_i, z_i)/(Ap_i, p_i)$
12. $\quad x_{i+1} = x_i + \alpha_i p_i$
13. $\quad r_{i+1} = r_i - \alpha_i A p_i$ (Stop if accurate enough)
14. **End**

CHAPTER 5

Numerical Studies

This chapter demonstrates the performance of the Multilevel/Algebraic Multigrid approach through some numerical examples of practical interest. We also study the performance of Krylov subspace solvers equipped with deflation acceleration. The reference methods for comparison of the current approach are first introduced in Section 5.1. Next, in Section 5.2 numerical results for the electric field wave problems are presented. The results for problems involving the quasi-static magnetic field are given in Section 5.3.

5.1 The Methods Tested

In order to demonstrate the performance of the proposed approach, several numerical experiments are considered. In each experiment, we adaptively increase the order of the elements and solve the matrix problem at each adaptive step by either COCG, CO-MINRES or GMRES, preconditioned with variants of the ASP preconditioner described in Chapter 3. Depending on the method used for approximating the solution to the lowest-level system, $\mathcal{A}^1 x = r^1$, the preconditioners tested are:

- *p*MUS[ASP(V)] and *p*MUS[ASP(W)]: The proposed new methods, i.e., *p*MUS (Algorithm 3.8) and ASP correction with V-Cycle AMG (Algorithm 3.4) or the W-cycle AMG (Algorithm 3.5) [97] [98] [99].
- *p*MUS[A-V]: The lowest-level system is solved by one backward GS step followed by one forward GS step, both applied to the vector-scalar potential (A-V) version of the Whitney system, in compact form [13] [92] [97]. This amounts to a SSOR preconditioner applied to the A-V version of the entire matrix. This conventional method can be regarded simply as a variant of the V-cycle of Algorithm 3.8 combined with a version of Algorithm 3.4 in which the projection onto N^3 (line 3) is omitted and the approximate inverse $\mathcal{B}_n^0$ is just another application of the SSOR preconditioner.
- *p*MUS[SSOR]: Conventional *p*MUS with SSOR preconditioning at the lowest level. This is equivalent to Algorithm 3.8 combined with a version of Algorithm 3.4 in which the projections onto both N^3 and N are omitted (line 2 to line 4). It is also equivalent to direct application of the SSOR preconditioner (1.13) to the entire matrix.
- *p*MUS[ILU]: Sparse incomplete factorization of $\mathcal{A}^1$ [15]. See Section 1.3.3.
- *p*MUS[LU]: Complete LU factorization of $\mathcal{A}^1$ using UMFPACK [7], which is the conventional *p*MUS approach (Algorithm 3.2).

The Krylov iterations are terminated when the infinity norm of the residual is reduced by a factor of ε_K. For the examples presented below, the value for this parameter is set to 10^{-9}, unless otherwise stated. For all calculations, double precision arithmetic is used. Simulations are done using Matlab [87] and performed on a PC with a 64-bit, 4-core, Intel 3.07 GHz processor and 24 GB of RAM. The geometries are modeled and meshed with ElecNet [91].

5.2 Numerical Results for Wave Problems

For this section, three different kinds of test problem are used. The first involves closed domains excited with an incoming waveguide mode in order to find an S parameter. The second kind concerns plane-wave scattering by metallic structures in free space. Finally, free space scattering form complex objects with composite materials are presented.

5.2.1 Waveguide Cavity Filter: *Illustration of the spectrum of the first order system and sensitivity to mesh refinement*

The waveguide cavity filter studied in [3] and [94] is analyzed. The dimension of the filter shown in Figure 5.1 is $0.76\lambda_0 \times 0.23\lambda_0 \times 2\lambda_0$, where λ_0 is the free space wavelength at 5.86 GHz. At the excitation port, the dominant TE_{10} waveguide mode is incident. Outgoing waveguide modes are absorbed at the other port. The structure is discretized with tetrahedral elements and simulations are performed with varying element sizes, resulting in discretizations with 18,236 elements in the coarsest mesh to 853,868 elements in the finest mesh.

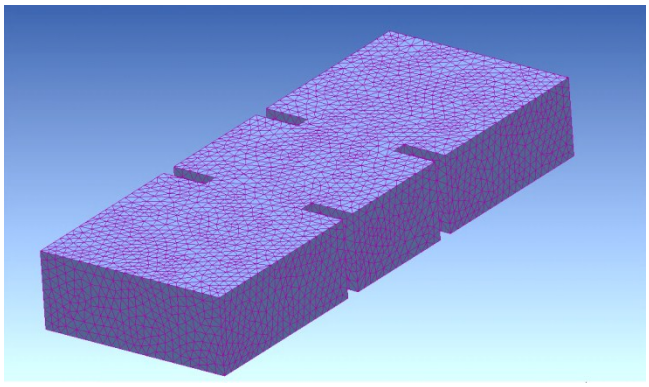


Figure 5.1: Waveguide cavity filter.

Initially, we consider the effects of the spectral properties of the indefinite curl-curl operator at the lowest level on the convergence behavior of the COCG solver. Figure 5.2(a) shows the spectrum of the SSOR-preconditioned matrix $R_f(\mathcal{A}^1)\mathcal{D}^1 R_b(\mathcal{A}^1)\mathcal{A}^1$ at 7.60 GHz, with a matrix dimension 24,285. It can be seen that the range of eigenvalues covers both the positive and negative sections of the real axis. As explained in Section 3.2, in addition to the Type A [93][14] [93] negative eigenvalues related to the null space of the curl operator and associated with unphysical modes, there are Type B negative eigenvalues related to physical resonances. More of the Type B eigenvalues will become negative as k_0 increases. This can lead to a very large condition number for the discrete problem and poor convergence of iterative solvers. With this preconditioner COCG converged in 497 iterations.

In Figure 5.2(b), the spectrum of the SSOR-preconditioned A-V system is shown. From the displaced eigenvalue distribution, it is apparent that many of the negative eigenvalues, presumably Type A, can be removed by incorporation of the mapping to

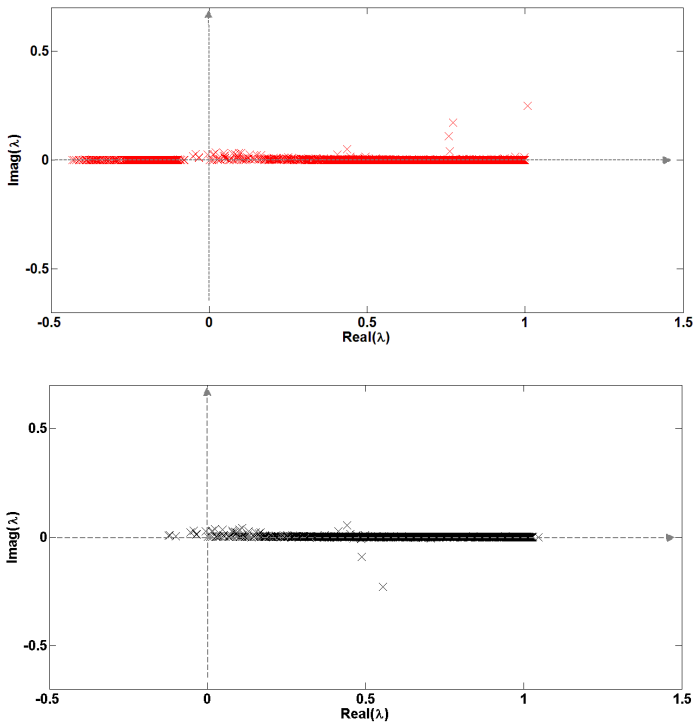


Figure 5.2: Eigenvalue spectrum for preconditioned matrix of cavity filter, (a) SSOR, (b) A-V SSOR.

the scalar space, N. For this better conditioned system, the COCG solver converged in 173 iterations. However, we see that there are still eigenvalues close to zero, both positive and negative, and this leads to the degradation of the Krylov solver performance.

Finally, the results of employing the additive ASP preconditioner $\mathcal{B}_a^1$ (Algorithm 3.3 with 1 level AMG approximation, $q = 0$) are illustrated in Figure 5.3(a) and (b), for the undamped ($\gamma = 0$) and damped ($\gamma = 1.5$) matrix (3.45), respectively. The first point to notice is the strong clustering around 1 in Figure 5.3(b). Even better, by introducing the complex shift factor, the negative imaginary half plane of the spectrum is mapped to a roughly circular pattern in the right half plane. The COCG iteration counts for the undamped and damped systems are 97 and 43, respectively. We see that the preconditioned system with more eigenvalues shifted away from the origin and located in the right-hand half of the complex plane can be solved much more efficiently with the Krylov solver.

Table 5.1 compares the corresponding iterations and CPU times required for COCG preconditioned with SSOR, A-V and ASP methods with varying mesh size. For ASP, the value of γ is taken as 1.5 and the AMG levels are increased from 1 to 3 through the refinement, to avoid the direct solution of a matrix with dimension larger

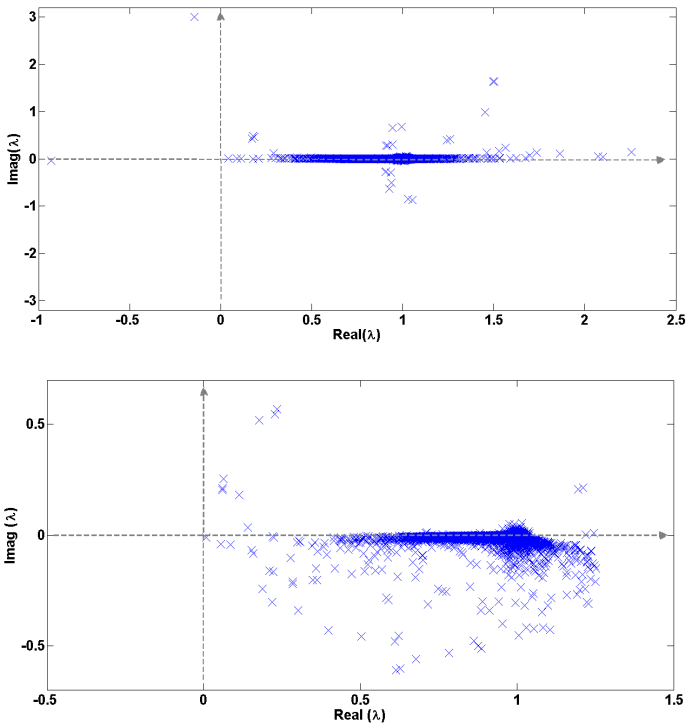


Figure 5.3: Eigenvalue spectrum for preconditioned system of cavity filter, (a) ASP with $\gamma = 0$, (b) ASP with $\gamma = 1.5$.

than 8,000. The convergence of ASP in additive and multiplicative versions is superior to the other methods, and the performance is least sensitive to the mesh refinement parameter h. In fact, the ASP-preconditioned systems converge in a number of iterations that is independent of the mesh size. Among the different versions of the ASP preconditioner, the W-cycle gives the lowest number of iterations, with roughly same CPU time as the V-cycle, and in the following tests just the results with $\mathcal{B}_w^1$ are given. In Figure 5.4, it is shown how the iteration counts and CPU time of the preconditioned COCG increase as the total number of unknowns and

Table 5.1: Comparison of number of iterations and CPU time (s) for the waveguide cavity filter at 5.86 GHz.

Number of unknowns (n)		24,285	57,487	121,203	220,753	502,073	1,043,207
Number of non-zeros (m)		263,840	750,549	1,598,916	4,034,374	9,935,252	27,914,831
h_{max}/λ		0.192	0.113	0.069	0.049	0.043	0.031
SSOR	Iteration	467	827	1256	1267	1977	2202
	CPU Time	5.58	29.96	98.15	263.95	1109.4	3,609.8
A-V SSOR	Iteration	128	168	242	296	376	497
	CPU Time	2.11	7.98	24.70	81.20	270.04	1029.31
		1 Level AMG		2 Level AMG		3 Level AMG	
ASP-$\mathcal{B}_a^1$	Setup Time	0.37	1.41	1.86	4.25	9.75	25.55
	Iteration	**62**	**67**	**63**	**65**	**69**	**69**
	CPU Time	1.63	5.67	9.49	24.51	67.45	192.16
ASP-$\mathcal{B}_v^1$	Iteration	**46**	**47**	**46**	**50**	**49**	**48**
	CPU Time	1.20	4.11	8.32	22.33	56.25	155.92
ASP-$\mathcal{B}_w^1$	Iteration	**26**	**26**	**23**	**25**	**24**	**24**
	CPU Time	1.64	4.32	9.12	24.79	62.39	177.81

nonzeros entries of $\mathcal{A}^1$ is varied by increasing the number of tetrahedra. From the ASP curve, the computational complexity is $O(m^{1.03})$, where m is the number of nonzeros in $\mathcal{A}^1$. By contrast, the complexity of A-V is $O(m^{1.33})$. The computed scattering parameter S_{11} versus the frequency of excitation is shown in Figure 5.5 and demonstrates very good agreement with reported results in [94].

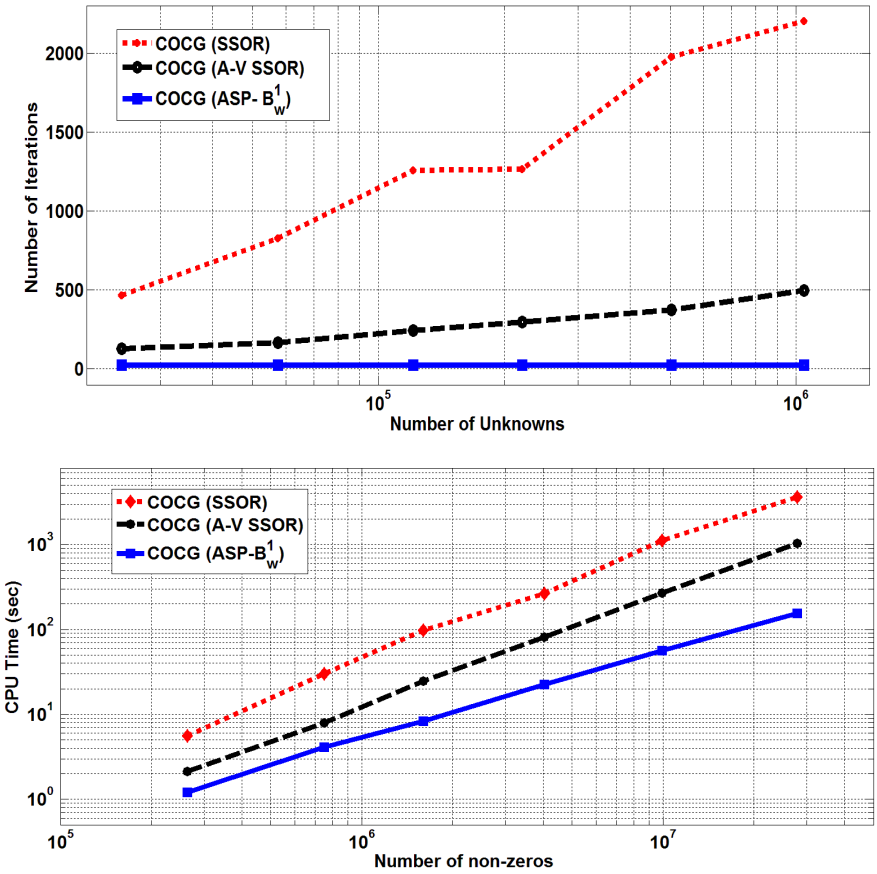


Figure 5.4: Variation of number of iterations and CPU time for waveguide cavity filter.

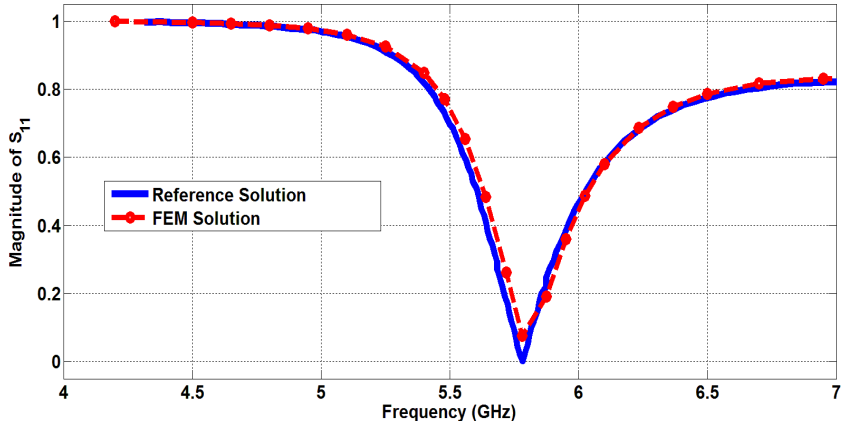


Figure 5.5: Absolute value of S_{11} for waveguide cavity filter. Reference results are taken from [94].

5.2.2 Scattering from Conducting Sphere*: Sensitivity to frequency*

The next test case concerns plane-wave scattering by a metallic PEC sphere with a radius of 1 m, with the radius of the spherical ABC boundary set to 4 m. The number of unknowns in the finest grid is 1,024,065, and using a three-level AMG (i.e., $q = 2$), the size of direct solution on the coarsest grid reduces to 12,508. The wavenumber varies from 1 to 5 rads/m in three steps and at the highest frequency the average edge length in the mesh is $h_{avg} = \lambda_0/8$;

We consider the application of two Krylov subspace iterative methods, COCG and the GMRES for solving the problem at the lowest level. The number of iterations and CPU time required by each of these solvers are given in Table 5.2, where A-V SSOR and multiplicative W-cycle ASP preconditioners are employed for convergence acceleration. It can be seen that for both solvers the new approach achieves the shortest solution times and requires significantly fewer iterations than A-V SSOR. An important observation for the PEC sphere scattering problem is that the number of iterations and the required CPU time increase with the excitation frequency, showing the ill-conditioned nature of the system at higher frequencies. For both problems, the use of $\gamma = 1.5$ rather than $\gamma = 0$ greatly reduces the computation time, and for the sphere it can be seen that the reduction increases with frequency.

The computed co-polarized total electric field $|\vec{E}_z|$ along the x axis at $k_0 = 3\ \text{rads/m}$ is compared to the analytical Mie solution in Figure 5.6. Once again, a good agreement is observed.

Table 5.2: Comparison of different Krylov solvers for scattering analysis of Metallic PEC sphere.

Solver	COCG					
Preconditioner	A-V SSOR		ASP $-\ \mathcal{B}_w^1(\gamma = 0)$		ASP $-\ \mathcal{B}_w^1\ (\gamma = 1.5)$	
Wavenumber	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)
$k_0^2 = 1$	680	00:08:34	96	00:05:14	**40**	**00:02:21**
$k_0^2 = 9$	1991	00:24:44	552	00:25:23	**71**	**00:03:54**
$k_0^2 = 25$	2483	00:30:50	824	00:30:31	**127**	**00:06:45**
Solver	GMRES					
Preconditioner	A-V SSOR		ASP $-\ \mathcal{B}_w^1\ (\gamma = 0)$		ASP $-\ \mathcal{B}_w^1\ (\gamma = 1.5)$	
Wavenumber	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)
$k_0^2 = 1$	1534	00:58:38	69	00:06:19	**37**	**00:03:02**
$k_0^2 = 9$	2083	01:31:40	443	00:41:50	**118**	**00:05:22**
$k_0^2 = 25$	2403	02:00:33	702	01:01:21	**175**	**00:09:48**

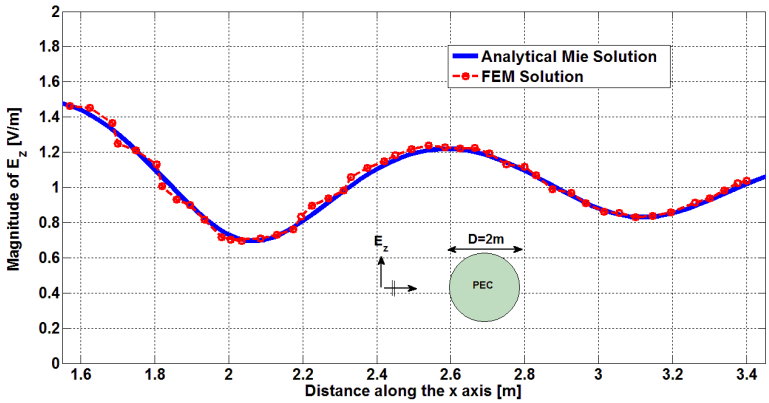


Figure 5.6: Magnitude of co-polarized electric field along the propagation direction.

5.2.3 Metallic Frequency Selective Surface (FSS): *Sensitivity to the damping parameter* $\boldsymbol{\gamma}$

This test problem concerns scattering from a $5.33\lambda_0 \times 5.33\lambda_0 \times 0.33\lambda_0$ metallic-grid frequency selective surface (FSS) consisting of rectangular perforations of varying size on a metallic screen (Figure 5.7), as described in [95]. The geometry is illuminated with a 10 GHz plane wave. The incident wave is along the z axis. The outer truncation ABC boundary is a rectangular box, which is the same shape as the target. The distance between the ABC boundary and the target is $0.67\lambda_0$. Because of the fine details of this non-periodic scatterer, a large number of elements is needed. The edge lengths in the mesh range from $\lambda_0/9400$ to $\lambda_0/11$, with $h_{avg} = \lambda_0/13$, and there are 2,416,388 unknowns. Once again, a three-level multilevel scheme is used to reduce to 30,932 the size of the matrix to which direct solution is applied.

We consider the effects of the damping parameter γ on the performance of COCG Krylov subspace method applied to $\mathcal{A}^1 x = b^1$. The number of iterations and also CPU

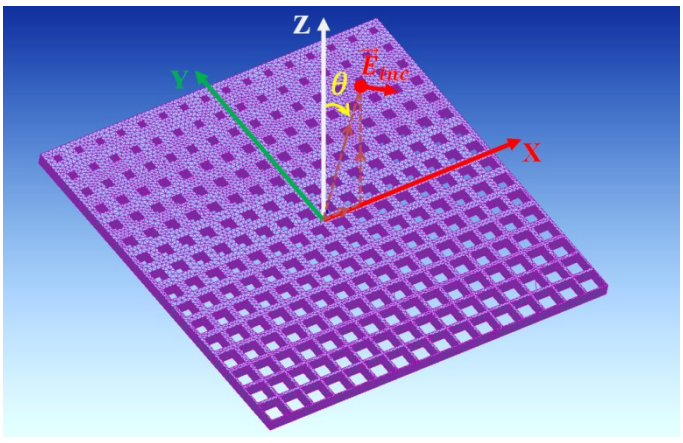


Figure 5.7. Metallic frequency selective surface with rectangular perforations.

Table 5.3: Comparison of effects of damping parameter γ on the convergence of Krylov solver for metallic FSS problem.

Solver	COCG	
$\varepsilon_K = 10^{-9}$ Preconditioner	Iter.	CPU Time (hh:mm:ss)
A-V SSOR	2384	01:01:22
ASP $-\mathcal{B}_w^1$ $(\gamma = 0.00)$	696	01:08:06
ASP $-\mathcal{B}_w^1$ $(\gamma = 0.50)$	466	00:52:20
ASP $-\mathcal{B}_w^1$ $(\gamma = 1.00)$	249	00:27:41
ASP $-\mathcal{B}_w^1$ $(\gamma = 1.25)$	201	00:21:17
ASP $-\mathcal{B}_w^1$ $(\gamma = 1.50)$	173	00:20:48
ASP $-\mathcal{B}_w^1$ $(\gamma = 1.75)$	178	00:20:51
ASP $-\mathcal{B}_w^1$ $(\gamma = 2.00)$	194	00:21:36
ASP $-\mathcal{B}_w^1$ $(\gamma = 3.00)$	217	00:24:15
ASP $-\mathcal{B}_w^1$ $(\gamma = 5.00)$	278	00:29:50

time for different damping parameters are given in Table 5.3, where multiplicative W-cycle ASP preconditioners and A-V SSOR are employed for convergence acceleration. It can be seen that the use of $\gamma = 1.5$ for shifted Laplacian preconditioner achieves the shortest solution times and requires significantly fewer iterations than A-V SSOR. Figure 5.8 demonstrates the convergence behavior of the preconditioned system. Clearly, the iterative solvers preconditioned with ASP exhibit a robust and fast convergence for the first order problem. The computed electric field over the xy plane, 0.5 cm from the scatterer, is shown in Figure 5.9.

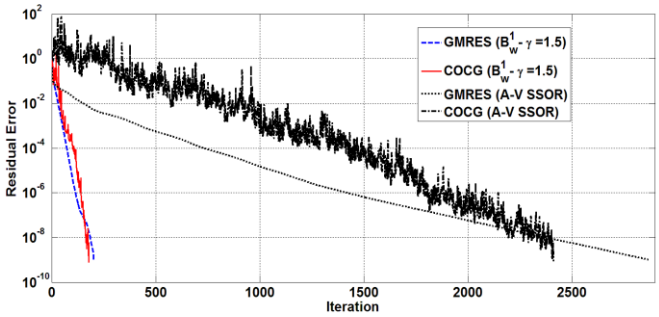


Figure 5.8: Convergence history of preconditioned Krylov solver for FSS scattering analysis.

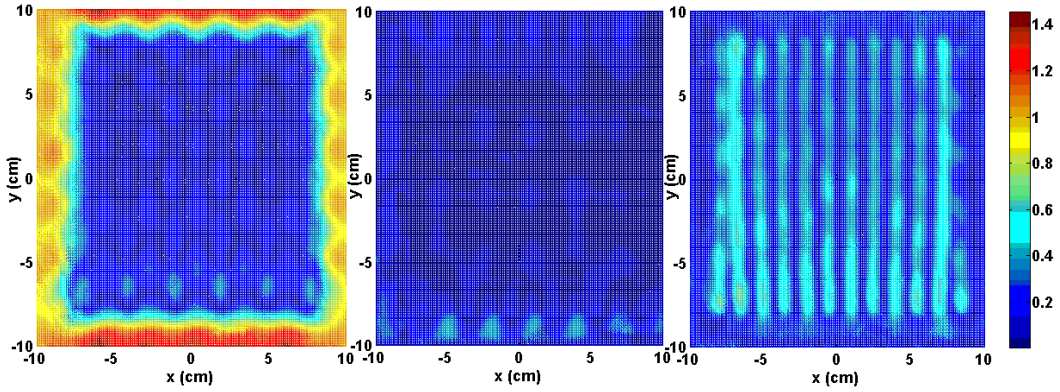


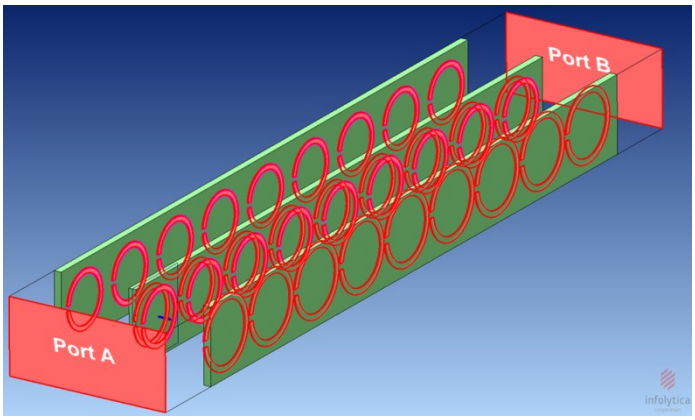
Figure 5.9: Scattering from metallic FSS; magnitude of the electric field. (a) Ex, (b) Ey, (c) Ez.

5.2.4 SRR Loaded rectangular Waveguide: *p-adaption and deflation*

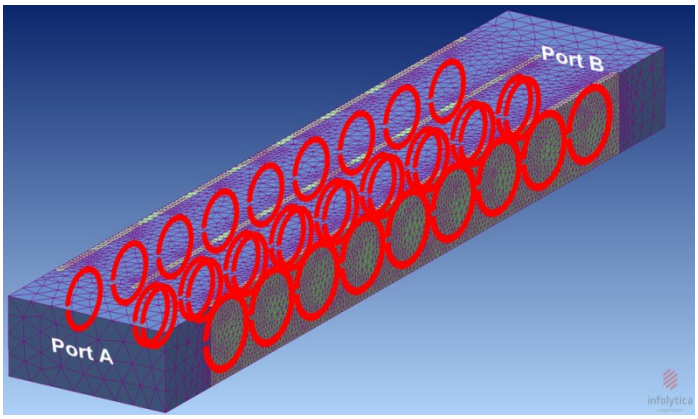
The next example is a rectangular metallic waveguide loaded with split ring resonators (SRRs). This configuration has attracted extensive attention [100][101], since it provides the realization of negative index metamaterials and novel methods to miniaturize waveguide-based devices. The geometry is directly taken from [100], proposed for the creation of a stopband above the ordinary waveguide cutoff frequency. The geometry, showing the excitation port A and absorption port B, is presented in Figure 5.10.

SRR elements can be found in [100], noting that printed elements on the middle and left lateral wall slabs are a factor of 0.8 smaller than the elements on the right

(a)



(b)



(c)

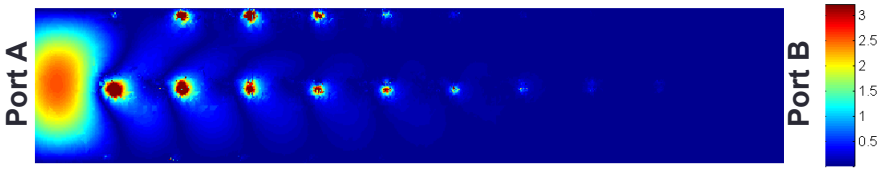


Figure 5.10: (a) The geometry of SRR loaded waveguide filter along with the excitation and absorption ports. (b) The mesh discretization. (c) Intensity distribution of electric field strength within the loaded waveguide.

Table 5.4: Solution Details For the SRR loaded waveguide problem.

Adaption Step s	$s=1$			$s=2$		
DOFs	1,050,016			2,353,671		
Preconditioner (+ Deflation)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)
***p*MUS[ASP]+Def[50%]**	**947**	**00:36:25**	**955**	**916**	**01:43:21**	**8,803**
*p*MUS[A-V]	4,056	01:25:46	834	6,201	03:35:04	1,724
Computed S_{11}	$0.08472 - j0.946204$			$0.241874 - j0.939322$		
Adaption Step s	$s=3$			$s=4$		
DOFs	3,967,045			5,994,104		
Preconditioner (+ Deflation)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)
***p*MUS[ASP]+Def[50%]**	**1,551**	**03:57:10**	**10,416**	**2,084**	**08:11:28**	**13,023**
*p*MUS[A-V]	9,033	10:35:38	3,176	10,115	22:04:29	5,942
Computed S_{11}	$0.305198 - j0.952192$			$0.298847 - j0.954262$		

side. The entire composite domain is discretized with 883,375 tetrahedra elements, resulting in edge lengths $h_{min} \simeq \lambda_0 /949$, and $h_{max} \simeq \lambda_0 /5$. Adaption was used to compute the reflection coefficient at port A, S_{11}, and the results are given in Table 5.4.

The convergence of deflated Krylov preconditioned with *p*MUS and 3 level AMG is clearly superior to *p*MUS[A-V] in terms of iterations and CPU time. The importance of deflating the Krylov solver from hampering eigenvectors and its effects on the cumulative CPU time is shown in Figure 5.11. While the required time to accurately solve the problem in 4 adaption steps is around 38 hours with *p*MUS[A-V], *p*MUS[ASP] reduces that to 23 hours and with deflation it is less than 14.5 hours. The magnitude of the electric field intensity at the mid-plane passing through the geometry is plotted in Figure 5.10 (c). It is apparent that energy propagation toward the output port is suppressed by the ring resonator elements.

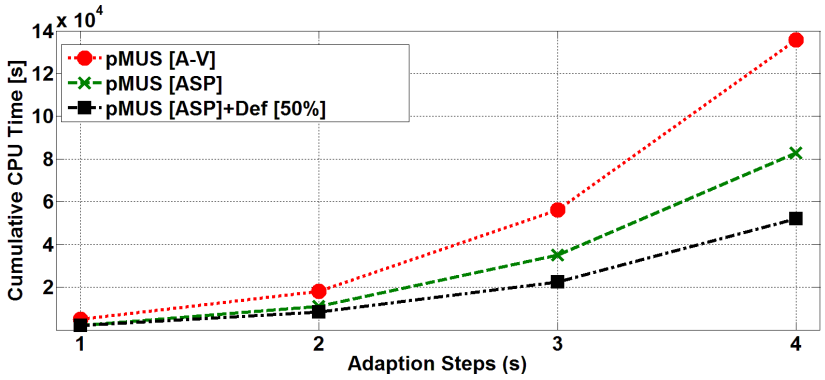


Figure 5.11: Cumulative CPU time versus adaption steps for the SRR loaded waveguide filter.

5.2.5 Conducting Sphere Surrounded by 60-Node Buckyball: *Sensitivity to the size of the deflation subspace*

This test case concerns free space scattering from a conducting sphere surrounded by a polyhedral, 60-node "buckyball", acting as a metallic frame to reduce the backscattered wave from the sphere at 46 MHz [102]. The problem is shown in Figure 5.12. The diameter of the internal solid sphere is 6 m and the external polyhedral conducting frame has diameter of 10 m with an edge width and thickness of 0.19723 m and 0.05061 m, respectively. An ABC is applied over a spherical surface 5 m away from the buckyball. The mesh obtained after discretization consists of 122,101 nodes and 676,802 elements with minimum and maximum edge lengths of $0.0019\lambda_0$ and $0.1344\lambda_0$, respectively. In this computation, the number of unknowns increases from 810,595 to 5,710,953 in 4 adaptive steps.

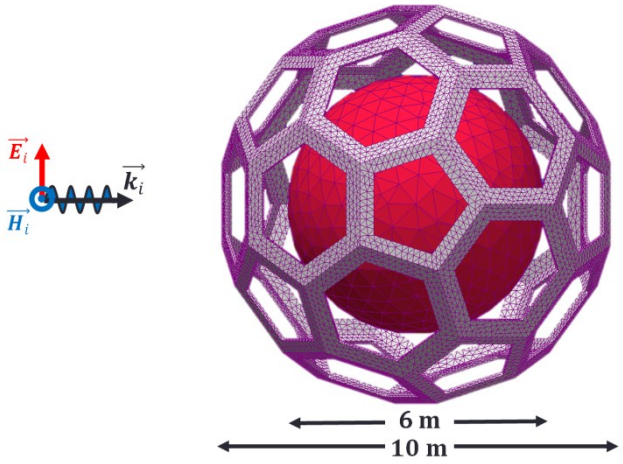


Figure 5.12: Geometry of 6 m metal sphere surrounded with a 10m spherical polyhedral frame.

The detailed computational information is listed in Table 5.5. For *p*MUS[ASP], 3 AMG levels are used, with 106,199, 26,430 and 9,662 DOFs, respectively.

Table 5.5: Solution Details For the buckyball scattering problem.

Adaption Step s	$s=1$			$s=2$		
DOFs	810,595			2,319,586		
Preconditioner (+ Deflation)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)
***p*MUS[ASP]+Def[50%]**	**108**	**00:03:58**	**928**	**124**	**00:08:39**	**2,635**
*p*MUS[A-V]	2,058	00:21:10	686	2,173	01:19:52	1,767
Adaption Step s	$s=3$			$s=4$		
DOFs	3,716,547			5,710,953		
Preconditioner (+ Deflation)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)	Iter.	CPU Time (hh:mm:ss)	Memory (MB)
***p*MUS[ASP]+Def[50%]**	**130**	**00:12:50**	**3,621**	**157**	**00:25:30**	**5,895**
*p*MUS[A-V]	2,138	02:18:15	2,673	2,188	04:19:25	4,908

As indicated in the table, the run time required by *p*MUS[ASP] with 57 deflation vectors (50%) is much less than with *p*MUS[A-V] at each adaption step.

Next we consider the effect of varying the dimension, k, of the deflation subspace from 0 (no deflation) to $k = 108$ (100% deflation). Figures 5.13 (a-c) compare the number of iterations, computation time and memory usage for several choices of k. Increasing k leads to fewer iterations, but at the cost of more memory usage. We also see that increasing k beyond about 50% does not bring much further reduction in computation time. Towards the end of the adaption, 50% deflation leads to a 43% reduction in cumulative computation time and a 46% reduction in Krylov iterations.

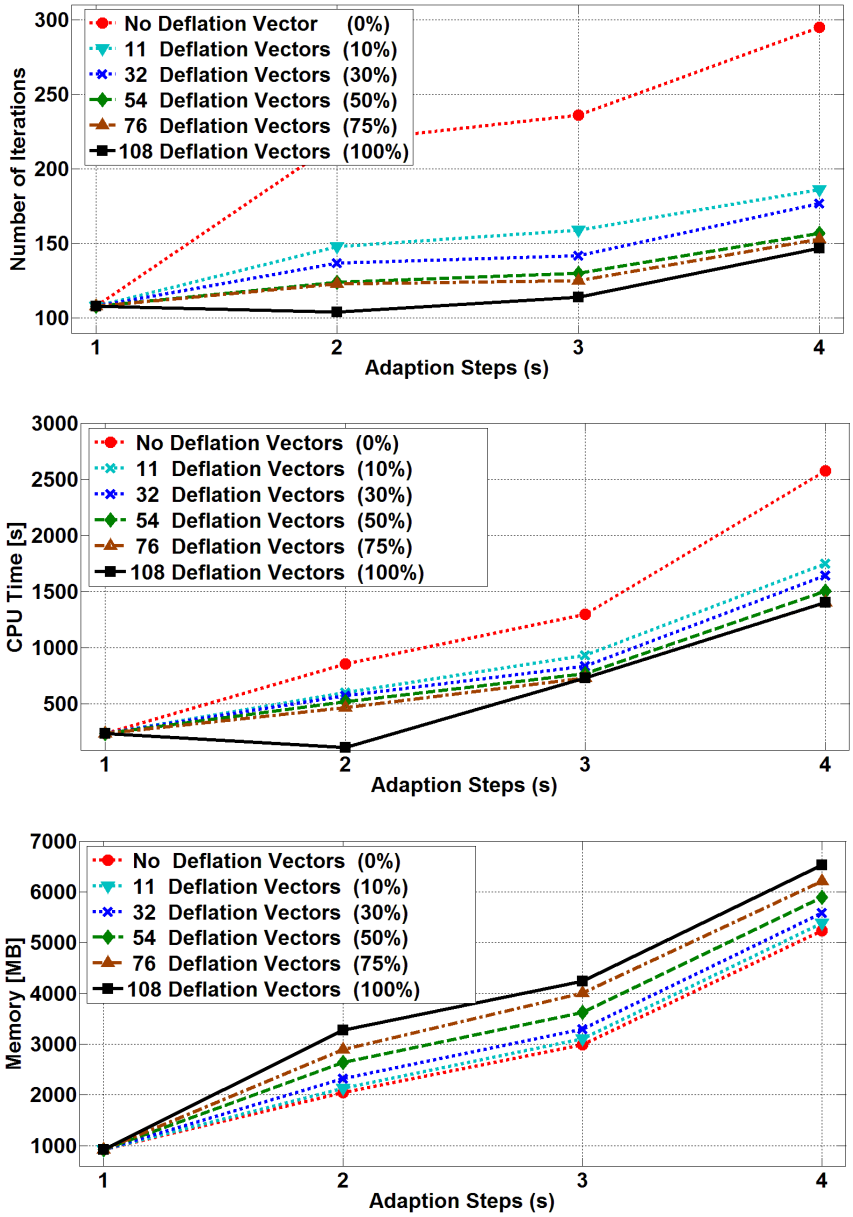


Figure 5.13: Comparison of effects of deflation on: (a) Krylov iterations, (b) CPU time and (c) memory usage, for the buckyball problem.

5.2.3 Jerusalem Cross Screens FSS: *Illustration of matrix dimensions in multilevel hierarchy*

For the final experiment related to wave problems, results obtained from the scattering analysis of a finite, noncommensurate, frequency selective surface (FSS) are presented. The device, shown in Figure 5.14, consists of two dissimilar Jerusalem cross screens (dimensions taken from [103]) printed on opposite sides of a $7.469\lambda_0 \times 7.469\lambda_0 \times 0.204\lambda_0$ dielectric slab, where λ_0 is the free space wavelength at 3 GHz. The dielectric relative permittivity is $\epsilon_r = 2.1 - j1.4$. Such composite and multi-layer structures with different periodicities are difficult to handle rigorously using periodic boundary conditions [103] [104] and analysis of the entire system is usually needed.

For the computational domain truncation, the geometry is enclosed in a box at least $0.76\lambda_0$ away from any point on the FSS and the ABC is applied. The structure is illuminated normally ($-z$ direction) with an x-polarized plane wave. The mesh consists of 1,914,954 tetrahedra. As observed in Figure 5.14, the mesh is highly nonuniform, with an average element size of $h_{avg} \simeq \lambda_0/17$ and sizes ranging from $\lambda_0/645$ around the metallic regions to $\lambda_0/5$ near the truncation boundary.

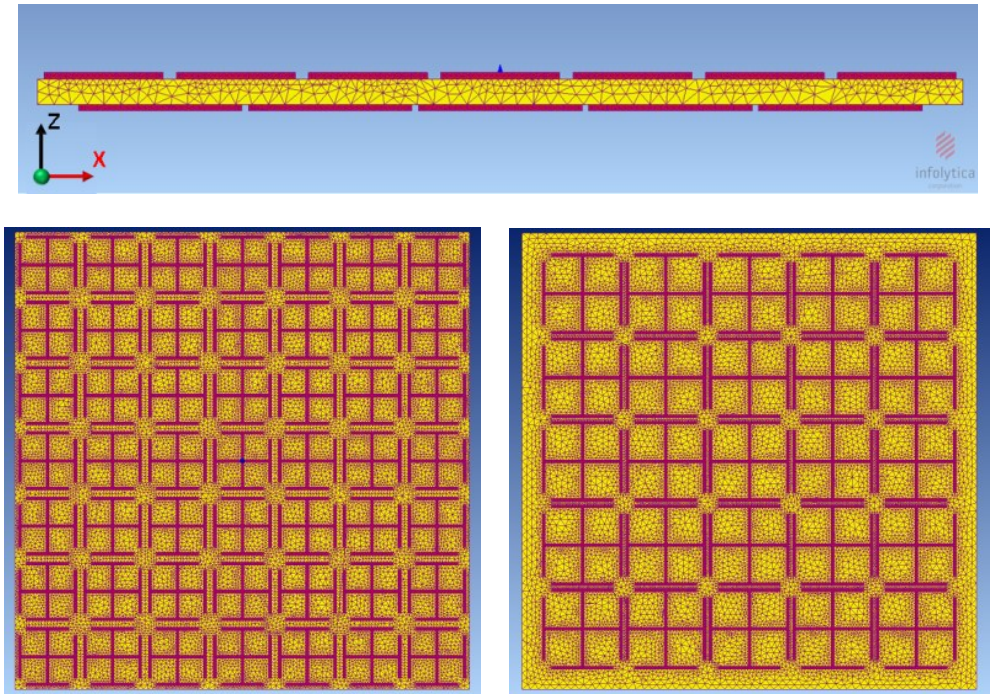


Figure 5.14: Geometry of a Jerusalem cross noncommenserate FSS
(a) Side view. (b) Top view showing the 7×7 upper screen.
(c) Bottom view showing the 5×5 lower screen.

Table 5.6: Solution details for the noncommenserate FSS scattering problem.

s	DOFs	Nonzeros	Iter.	CPU Time (hh:mm:ss)	Memory (MB)
1	2,261,319	36,824,307	533	00:54:28	2,314
2	5,482,526	188,871,828	626	02:07:01	11,241
3	8,909,294	370,550,000	683	03:18:42	14,347
4	12,264,629	626,152,411	594	03:45:56	17,424

A summary of computational statistics for 4 steps of p-adaption, using pMUS[ASP]+Def[50%], is given in Table 5.6 ($\varepsilon_K = 10^{-6}$). In this case pMUS[A-V] failed at each adaptive step to reduce the residual by a factor of 10^{-2} after 1500 iterations. The column labeled "Nonzeros" gives the number of nonzero entries in the matrix. Note that the matrices become denser as the adaption proceeds, which is characteristic of p-adaption. Here, as in the previous examples, the number of iterations is relatively stable, despite the growth in the size of the matrix. At step 4, a complex valued system with almost 12.2 million unknowns and 0.62 billion nonzeros is solved on a PC in reasonable time.

Table 5.7 gives, for the matrix A^4 (4^{th} adaptive step), the matrices $\mathcal{A}^l$ used in its preconditioner. Since there are some 4^{th} order elements present at this step, the largest matrix is $\mathcal{A}^4$ (which is just A^4 itself). Four levels of AMG are used (i.e., $q = 3$), so the smallest matrix is $\mathcal{A}^{-3}$.

The residual history in solving the A^4 system is shown in Figure 5.15. Removing the accelerations provided by deflation results in an increase of 42% and 19% in Krylov iterations and run time respectively, as well as 35% reduction in memory usage. Figure 5.16 shows the computed electric field over two cross-sections, both passing through the center of the structure.

Table 5.7: Matrix Hierarchy Details For the noncommenserate FSS scattering problem at the 4th adaption step.

Matrix	DOFs	Nonzeros
$\mathcal{A}^4$	12,264,629	626,152,411
$\mathcal{A}^3$	12,051,413	525,271,225
$\mathcal{A}^2$	10,794,234	416,843,750
$\mathcal{A}^1$	2,261,319	36,824,307
$\mathcal{A}^0$	259,089	3,487,890
$\mathcal{A}^{-1}$	90,448	1,236,772
$\mathcal{A}^{-2}$	34,162	480,396
$\mathcal{A}^{-3}$	20,174	287,358

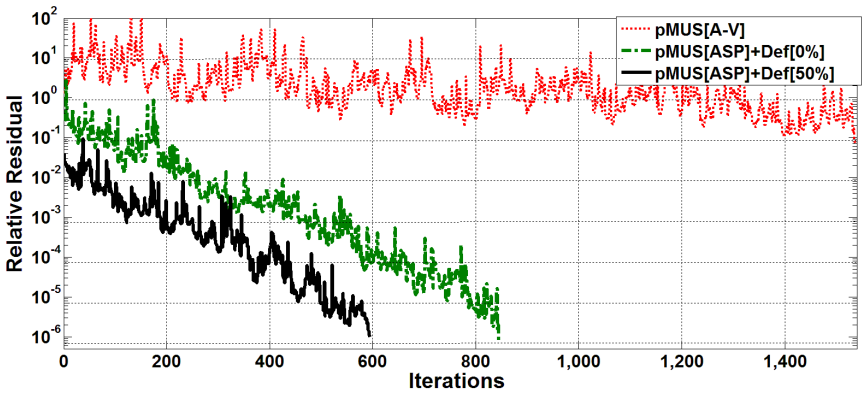


Figure 5.15: Residual history of (deflated) preconditioned COCG method for the noncommenserate FSS scattering at 4th adaption step.

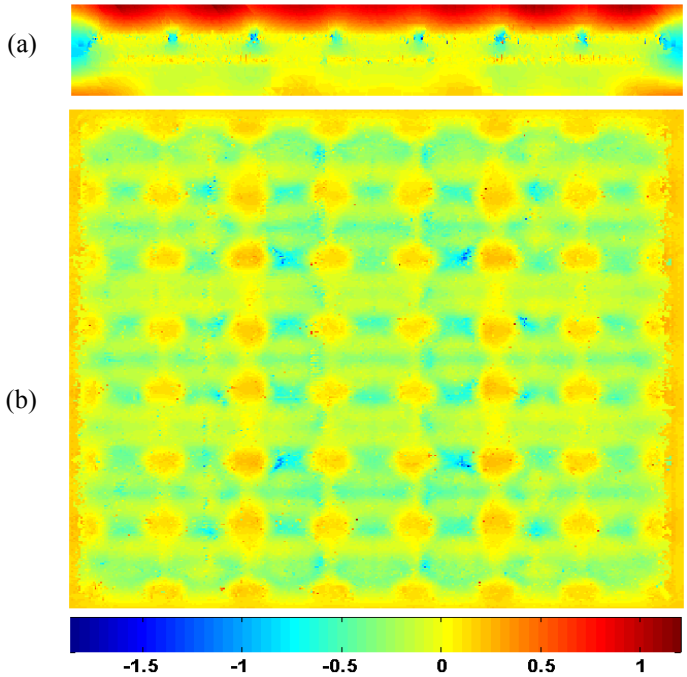


Figure 5.16: Visualization of electric field for the noncommenserate FSS (a) E_x over plane $y = 0$. (b) E_x over plane $z = 0$.

5.3 Numerical Results for the Quasi-static Magnetic Field Problems

In this section, numerical results related to quasi-static magnetic field problems are presented. The performance of the current method (pMUS and ASP approach introduced in Section 3.6) is compared to a standard preconditioner for three examples.

5.3.3 Shielding Inside a Metallic Cube*: V cycle vs. W cycle*

For the first example, a hollow, conducting cube placed in a uniform field is considered. The geometry and the mesh are shown in Figure 5.17. The permeability and the conductivity of the conductor are $\mu = \mu_0$ and $\sigma = 1.45 \times 10^6$ S/m respectively. The skin depth at the excitation frequency, 100 Hz, is $\delta = 42.8$ mm. The length of the cube is 7 m and the wall thickness is 2δ. The computational domain, Ω, is a cube of side 20 m. The scalar potential is constrained on two opposite surfaces of Ω to produce a magnetic field, $\mathbf{H}_u$, that would be uniform in the absence of the conducting cube. The domain is discretized with 3,750,322 tetrahedra, resulting in 663,471 scalar and 2,212,397 vector unknowns at the lowest order. Three steps of *p*-adaption are applied. The matrix problems are solved by the preconditioned COCG method [86].

Table 5.8 presents the number of unknowns, the number of COCG iterations and the corresponding CPU times for solving the matrix problem with the accuracy of $\varepsilon_K = 10^{-6}$ at each adaptive step. For the ASP method, scalar AMG employs a hierarchy of three levels with 517,421, 167,589 and 55,560 unknowns. The comparison shows the superior iteration count and run time of ASP at each adaptive step. The experiments also indicate that a significant improvement can be achieved for higher order systems if a better algebraic solver, like ASP(W-Cycle), is used at the first order. The cumulative CPU time for solving the problem with three *p*-adaptive steps using *p*MUS[ASP(W-cycle)] is 3.4 times less than that required by SSOR and for step 3 the CPU time is 4 times less.

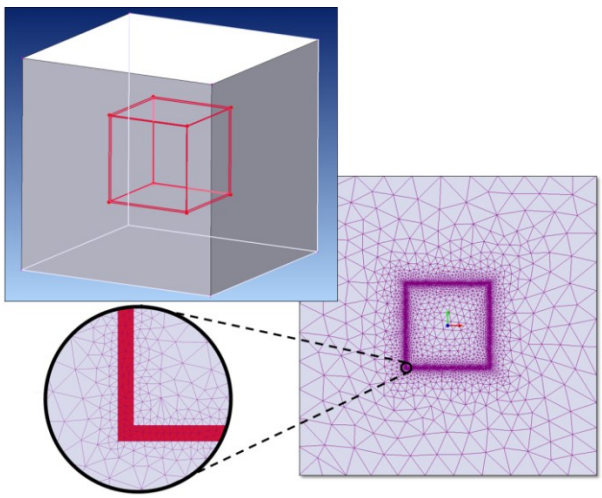


Figure 5.17: Illustration of geometry and discretization for the conducting cube problem.

Table 5.8: Solution Details For the conducting Cube Problem.

	Step 1		Step 2		Step 3	
Unknowns	2,875,868		8,578,954		13,345,009	
Nonzeros	66,707,982		350,799,470		615,703,547	
Preconditioner	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)
*p*MUS[SSOR]	608	00:23:49	652	01:41:55	654	03:03:14
*p*MUS[ILU(0)]	741	00:28:13	685	01:57:35	628	02:51:45
***p*MUS[ASP(V-Cycle)]**	**150**	**00:12:28**	**205**	**00:39:09**	**206**	**01:08:05**
***p*MUS[ASP(W-Cycle)]**	**60**	**00:13:23**	**88**	**00:28:19**	**94**	**00:45:55**

The convergence history of COCG at step 3 is shown in Figure 5.18. It is observed that while the convergence rates for SSOR and *p*MUS[ILU(0)] are nearly the same, *p*MUS[ASP(W-cycle)] is able to reduce the number of iterations by a factor of almost 7. Results for the computed $|\mathbf{H}|$ over a cross-section and field penetration inside the conducting cube are shown in Figure 5.19.

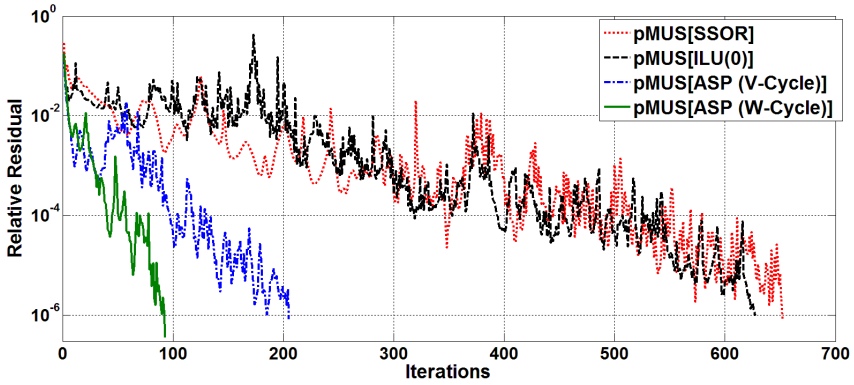


Figure 5.18: Convergence history of COCG with different preconditioners for solving the conducting cube problem at p-adaptive step 3.

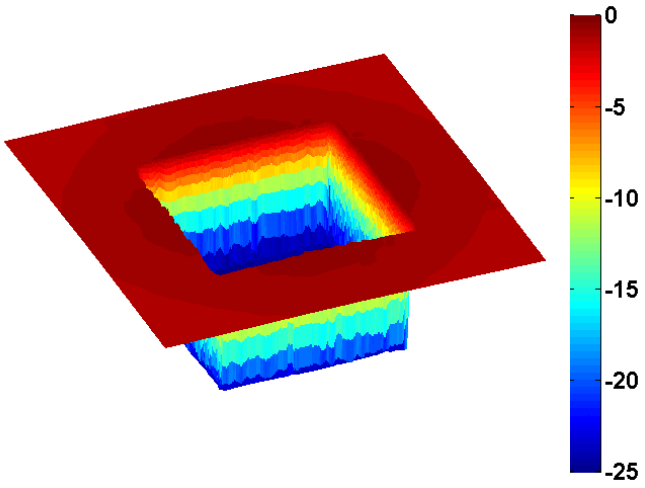


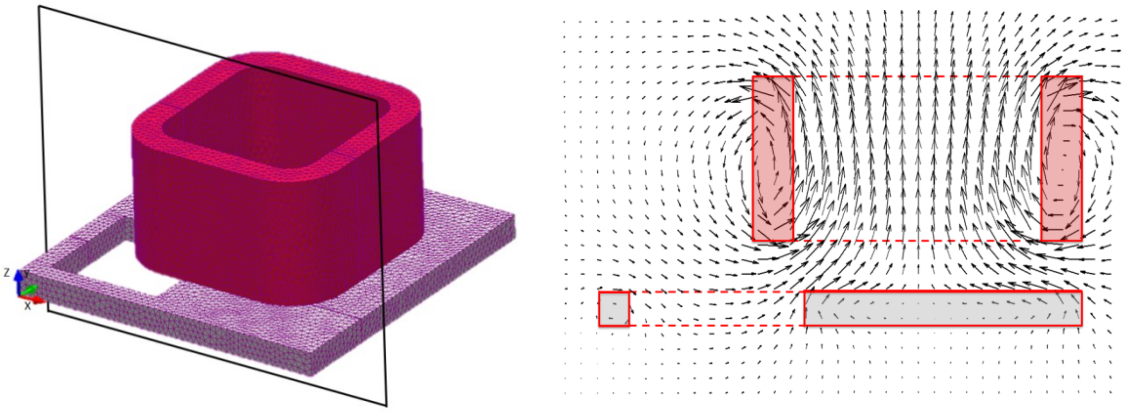
Figure 5.19: Variation of the normalized magnetic field strength $20\log(|\mathbf{H}|/|\mathbf{H}_u|)$ over a cross section passing through the cube center.

5.3.1 Benchmark TEAM Problem No. 7: *Scalability by p-refinement*

For the second example, the benchmark TEAM problem no. 7 is analyzed [105][106]. It consists of a conducting aluminum plate with a hole, above which a racetrack coil with a time-harmonic driving current is placed. The aluminum plate has a conductivity of $\sigma = 3.526 \times 10^7$ S/m and the sinusoidal driving current of the coil is 2742 AT. The frequency is 200 Hz. In order to employ the $\mathbf{T} - \Omega$ formulation, the hole is treated as a conductor with very low conductivity ($\sigma = 0.001$ S/m).

The computational domain is restricted to a cube of 1 m edge length with the boundary condition $H_{normal} = 0$ over its surfaces. For the discretization, 1,074,357 tetrahedra are used and the resulting mesh consists of 186,922 nodes and 1,267,642 edges. The geometry of the problem (without the surrounding box) and its discretization in the conducting region can be seen in Figure 5.20 (a). Note that the mesh is well refined on the surface of the conductor regions.

The problem is analyzed *p*-adaptively in 6 steps with $\varepsilon_K = 10^{-9}$. In Table 5.9, the performance of the COCG method for different adaption steps and different preconditioning algorithms are shown. For the ASP method, 2 level AMG is employed for approximating the vector nodal systems and the number of unknowns at the lowest level is 36,794. From the experiment, it can be seen that using ASP for the lowest order correction results in a greatly reduced number of iterations.



(a) (b)

Figure 5.20: (a) Mesh discretization for the TEAM 7 benchmark problem, (b) Magnetic field (Re(**H**)) at the cut plane $y = 72mm$.

Table 5.9: Solution details for the TEAM problem no. 7

Preconditioner		*p*MUS[SSOR]		*p*MUS[ILU(0)]		*p*MUS [ASP(W-Cycle)]		Speed up (SSOR/ASP)
Step	DOFs (N)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	
1	623,613	622	00:05:34	480	00:03:42	**63**	**00:01:48**	3.1
2	1,526,157	494	00:13:24	494	00:13:50	**64**	**00:03:01**	4.3
3	3,050,178	498	00:33:59	495	00:36:30	**75**	**00:06:52**	4.8
4	4,592,119	496	01:04:32	475	00:59:38	**87**	**00:11:54**	5.4
5	5,624,662	463	01:07:24	468	01:18:15	**70**	**00:12:58**	5.6
6	7,014,699	454	01:31:59	487	01:26:34	**71**	**00:16:39**	5.8

More significant than the number of iterations is the CPU time. The reduction in iterations results in fewer time-consuming smoothing steps for the higher order blocks of the matrix. Figure 5.21 compares the time in minutes to solve the eddy current problem at each adaption step. Here we see that *p*MUS[ASP] works very well and is always faster than SSOR and *p*MUS[ILU(0)]. The cumulative CPU time for all 6 steps using ASP is more than 5 times less than when using SSOR. For the last 4 points of the ASP curve, the CPU time grows roughly as $n^{1.1}$. The computed magnetic field at step 6 over the specified cut plane is displayed in Figure 5.20 (b). The problem has measured results as well [106]. Figure 5.22 compares the numerical values of the vertical magnetic flux density $B_z = \mu_0 H_z$ with measured values along a pre-defined line of the TEAM 7 benchmark problem.

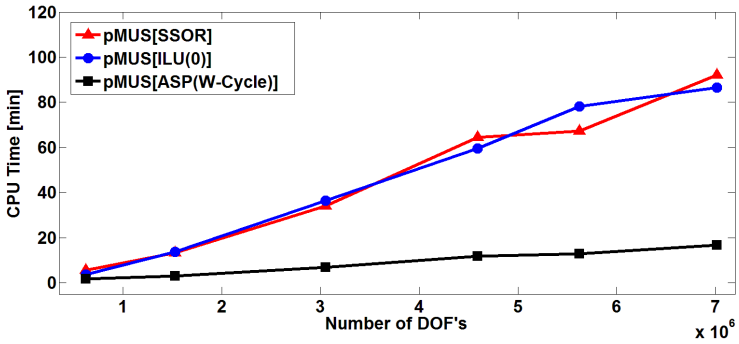


Figure 5.21: CPU time versus number of unknowns for different adaption steps of TEAM 7 problem.

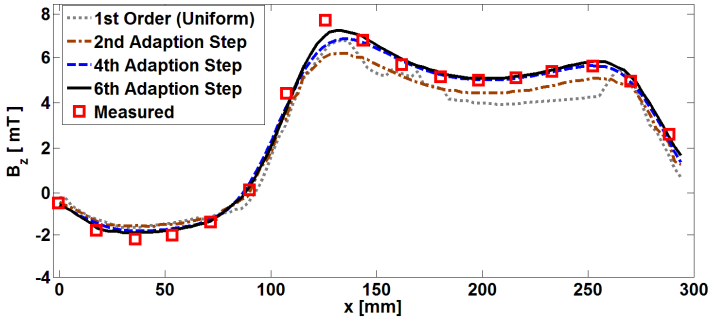


Figure 5.22: Computed B_z along a pre-defined line of the TEAM 7 problem ($y = 72$ mm, $z = 34$ mm, and $0 < x < 288$ mm) compared with measured values.

5.3.2 Benchmark TEAM problem No. 21b

For the last test case, eddy current analysis of TEAM workshop problem 21-b [107], shown in Figure 5.23, is carried out. The numbers of tetrahedral elements, scalar and vector DOFs are 1,045,257, 181,576 and 479,380 respectively. In Table 5.10, the computation time and iteration counts of *p*MUS[ASP(W-Cycle)] with 2 level AMG are compared to those obtained with the standard SSOR preconditioner, for COCG with convergence tolerance $\varepsilon_K = 10^{-9}$. Although the number of iterations increases in each step, a good speed up is obtained by using *p*MUS[ASP].

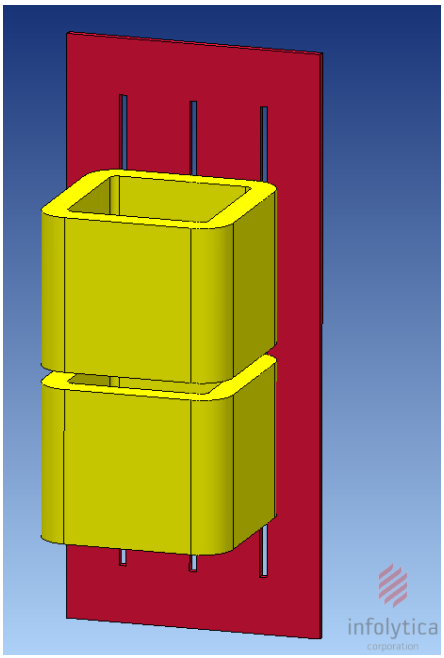


Figure 5.23: TEAM workshop problem 21-b.

Table 5.10: Solution Details For the TEAM problem no. 21-b

Preconditioner			SSOR		*p*MUS [ASP(W-Cycle)]		Speed up
Step	DOFs (n)	Nnz (m)	Iter.	CPU Time (hh:mm:ss)	Iter.	CPU Time (hh:mm:ss)	
1	660,956	15,276,046	461	00:03:38	**34**	**00:01:31**	2.4
2	1,604,034	63,375,974	562	0:15:13	**61**	**00:03:51**	3.9
3	3,177,209	220,372,167	558	00:40:57	**92**	**0:10:53**	3.8
4	5,368,881	533,765,377	559	01:27:27	**107**	**00:22:21**	3.9

CHAPTER 6

Conclusions

6.1 Summary

The finite element discretization of the vector curl-curl equation leads to a sparse and complex-symmetric linear system of equations. In high frequency simulation of the electric field inside complex structures the linear system is generally large, highly indefinite and extremely ill-conditioned. These properties make solution of the linear system a challenging step, especially when direct methods are out of reach. Application of Krylov iterative methods preconditioned with standard preconditioners, like SSOR or incomplete LU factorization, does not result in a satisfactory convergence. Specialized preconditioning techniques are required.

In this thesis, an effective iterative method to solve the discrete curl-curl equation with *p*-adaptive finite element analysis has been proposed. The method is a preconditioned Krylov subspace iteration, with a *p*-type Multilevel and Algebraic Multigrid cycle serving as the preconditioner for the block structured system. For the lowest order system at the bottom of the V-cycle, performance of different correction approaches has been tested, which are denoted by *p*MUS[method] in this work. Effects of different lowest level correction on the convergence rate are shown and

incorporation of a successful ASP/AMG algorithm has been carried out as a replacement for the commonly used direct solve. The success of the algebraic solver is in fact based on proper choice of multigrid components with respect to the characteristics of the underlying PDEs.

The construction of coarsening operators in nodal AMG is based on robust nodal aggregation algorithms [17] which allows for fast, automated coarse space construction, without the need to build a hierarchy of nested finite-element meshes. The coarse grid operators are also created in an algebraic fashion using the Galerkin coarse grid approximation which allows the coarse sub-problems to be constructed quickly without discretization. This results in an overall fast setup phase with reduced time compared to the iteration phase. The performance of the algebraic solver also depends on the accuracy of nodal AMGs for discrete second-order elliptic equations and improvements in this area can provide further robustness of the proposed preconditioning approach.

Extensive numerical experiments have been performed and results are presented for both wave and quasi-static problems. The test cases used are complicated problems to solve and the promising numerical results have shown the robustness and efficiency of the new preconditioner. The well-known A-V SSOR preconditioning approach is used as an efficient reference method. Krylov methods like COCG, CO-MINRES and GMRES preconditioned with the current approach were always considerably faster than A-V for the test problems considered. The performance of Krylov algorithms is even further enhanced during the *p*-adaption analysis by construction of a Deflation sub-space and removing the contribution of smallest eigenvalues from the preconditioned system.

6.2 Contributions

The main contributions of this thesis are summarized as below:

- **Application of multilevel Schwarz preconditioning to *adaptive* FEM systems:** This work extends the solution idea of *p*-type multilevel Schwarz (*p*MUS) for uniform order systems [60] [47] to the linear systems that arise in *p*-adaption. By a

suitable numbering of the high order DOFs, a block structured matrix is obtained which can be approximately inverted by a V-cycle, as with conventional *p*MUS.

- **Development of an ASP/AMG correction for the wave equation:**
  I applied the Auxiliary Space Preconditioning approach for Whitney elements, which was only tested before on quasi-static problems, to wave-scattering problems. Design of several preconditioning algorithms has been carried out and development of a fully algebraic correction based on symmetric V-cycle and W-cycle multigrid approximations has been achieved. The results shows that a ASP/AMG correction combined with a Krylov iteration is an effective technique for wave problems that are analyzed with large numbers of Whitney finite elements. As the mesh density increases, the number of Krylov iterations is constant and the CPU time grows linearly with the number of matrix nonzeros.

- **Development of an ASP/AMG correction for the T- Ω method:**
  An efficient algebraic multigrid algorithm for the solution of quasi-static problems by the $\mathbf{T}-\Omega$ formulation is developed. Similar algebraic methods have been applied previously to the matrices arising from purely vector formulations, using Whitney edge elements [23] [74] [75]. However, the method in this thesis is different in that it works with the $\mathbf{T}-\Omega$ method, which involves both scalar and vector unknowns. For the assembled matrix equation at level one, a partitioning is considered into vector unknowns (defined over the *cotree* edges of conductors) and scalar unknowns (defined over the whole mesh). Corresponding auxiliary space problems are then constructed for the separate treatment of the error related to the vector and scalar DOFs. The first order residual in this case is transferred into five auxiliary space problems: four equivalent nodal problems inside the conducting regions and one nodal problem for the whole mesh. Corrections are then connected in symmetric V and W cycles to result in an efficient preconditioner for the Krylov methods.

- **Incorporation of the ASP/AMG method for the first order problem into the *p*MUS algorithm:**
  Complete matrix factorization of the lowest level matrix in *p*MUS is not applicable to large scale discrete PDEs. Considering the promising performance of the proposed ASP/AMG approaches for Whitney elements, I combined

corrections of high-order, hierarchical DOFs in *p*MUS with the new ASP corrections at level one into a symmetric preconditioner, which allows *p*MUS to be applied even when the matrix at lowest order is too big for direct solution. The algebraic lowest level correction is especially useful for problems with fine geometric details, requiring a very large mesh in which the bulk of the elements remain at low order. *p*MUS with ASP/AMG treatment has shown to be always faster than *p*MUS[A-V] and in some cases 10 times faster.

- **Application of "shifted Laplace" preconditioning to the curl-curl wave equation:**

  For wave problems, the ASP/AMG algorithm is tailored to use a modified bilinear form. This is similar to the "shifted Laplace" technique that was developed for the scalar Helmholtz equation. Rather than being applied to the original bilinear form, corrections of the vector nodal part in the algorithm are applied to a shifted version, which can be thought of as the original operator with a complex perturbation. This is shown to provide a satisfactory multigrid inverse approximation and a significant convergence acceleration for the Krylov subspace methods, because of the improved spectral properties of the preconditioned linear system.

- **Development of a deflated Krylov scheme for faster solution of adaptive FEM systems:**

  For problems with strong ill-conditioning, some eigenvalues may lie very close to zero and by proper treatment of these small eigenvalues further acceleration of Krylov methods can be achieved. The so-called deflation technique is used in this work to project out smallest eigenvalues from the preconditioned system, such that their contribution is removed from the residual. Rather than using expensive eigenvector computation, the construction of the deflation sub-space is, for the first time, based on an estimation of eigenvectors from information obtained when solving the first problem in a *p*-adaptive sequence. Numerical experiments show that the deflated Krylov solver provides additional reduction in run time time (20% to 40%), at the expense of a greater memory requirement (10% to 35%).

6.3 Recommendations

Suggestions for future research include:

- **Parallelization of solver components:**

  Some parallel implementation of the proposed method would be possible for very large scale applications. In particular, the correction steps involved in handling each of the four or five nodal sub-problems in ASP (from the finest grid to the coarsest grid) are separate and can be executed in parallel. The setup phase and some steps of the deflated Krylov solvers can also be implemented on a parallel machine.

- **Combination with multilevel-based incomplete LU methods**

  For the current approach, adding more levels in the nodal AMG algorithm can be continued as long as the virtual AMG meshes can adequately represent field variations on the scale of the wavelength (for wave problems). If this limit is exceeded, the preconditioner quality suffers and the Krylov solver takes longer to converge, or may even fail to converge. Therefore, for problems which are electrically very large, the coarsest nodal problem may still be too large to factorize. In this case, one alternative approach for continuing the same algebraic concept at the bottom of the algorithm is to use a recently proposed inverse-based multilevel incomplete factorization [108]. In contrast to conventional *ILU* methods, the fully algebraic and symmetric multilevel inverse-based *ILU* preconditioner may exhibit good efficiency (especially in terms of fill-in) for solving the coarsest nodal systems by employing a graph partition technique and reordering into a hierarchical multilevel structure.

- **Combination with domain decomposition as a local solver**

  For solving very large and multiscale electromagnetic scattering problems, the finite element based nonoverlapping domain decomposition methods (DDM) have been exploited recently that enable each subdomain to be discretized and solved independently on one processing unit. Of particular interest would be the combination of the proposed approach with currently developed DDM algorithms. One may employ the *p*MUS/ASP algorithm as an effective local solver in every subdomain. This would allow larger subdomains, which would result in better performance.

- **Non-symmetric multigrid components**

  In this work, I have focused on the complex-symmetric property of the systems in order to choose multigrid components. Aside from requiring less storage, the symmetric property of the linear system allows us to use efficient Krylov subspace algorithms resulting from three term recurrences and based on the specialized conjugate orthogonality condition. All the preconditioning cycles in this work are substantially designed to exploit the symmetry in order to maximize the performance of solvers. However, by using some components that do not use symmetry, we might be able to develop better multigrid algorithms. For instance, recently some strategies have been proposed to use a non-symmetric GMRES smoother as an alternative [109] [110]. Outer iterations related to the Krylov methods can also be modified by using novel approaches, such as generalized conjugate residual method [111].

Bibliography

[1] J. F. Lee, D. K. Sun and Z. Cendes, "Full-wave analysis of dielectric waveguides using tangential vector finite elements," *IEEE Transactions on Microwave Theory and Techniques*, vol. 39, no. 8, pp. 1262-1271, 1991.

[2] J. P. Webb, "Hierarchical vector basis functions of arbitrary order for triangular and tetrahedral finite elements," *IEEE Transactions on Antennas and Propagation*, vol. 47, no. 8, pp. 1244–1253, 1999.

[3] P. Ingelstrom, "A new set of H(curl)-conforming hierarchical basis functions for tetrahedral meshes," *IEEE Transactions on Microwave Theory and Techniques*, vol. 54, pp. 106-114, Jan 2006.

[4] Y. Saad, M. Yeung, J. Erhel and F. Guyomarcha, "Deflated Version of the Conjugate Gradient Algorithm," *SIAM Journal on Scientific Computing.* vol. 21, no. 5, pp. 1909–1926, Apr 2000.

[5] A. M. Abdel-Rehim, R. B. Morgan, D. A. Nicely and W. Wilcox, "Deflated and Restarted Symmetric Lanczos Methods for Eigenvalues and Linear Equations With Multiple Right-Hand Sides," *SIAM Journal on Scientific Computing*, vol. 32, no. 1, pp. 129-149, 2010.

[6] T. A. Davis, "Direct Methods for Sparse Linear Systems," SIAM, Philadelphia, Part of the SIAM Book Series on the Fundamentals of Algorithms, Sep. 2006

[7] T. A. Davis, "Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method," *ACM Transactions on Mathematical Software*, vol. 30, no. 2, pp. 196-199, June 2004.

[8] T. A. Davis, "UMFPACK User Guide," Jun, 2012.

[9] P. Tsuji, E. Bjorn, and Y. Lexing, "A sweeping preconditioner for time-harmonic Maxwell's equations with finite elements," *Journal of Computational Physics,* vol. 231, no. 9, pp. 3770-3783, 2012.

[10] R. Freund, "Conjugate Gradient-Type Methods for Linear Systems with Complex Symmetric Coefficient Matrices," *SIAM Journal on Scientific and Statistical Computing*, 1992.

[11] C. C. Paige, and M. A. Saunders, "Solution of sparse indefinite systems of linear equations," *SIAM Journal on Numerical Analysis*, vol. 12, no 4, pp. 617-629, Sep 1975.

[12] Y. Saad, and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 3, pp. 856-869, Jul 1986.

[13] R. Dyczij-Edlinger, G. Peng and J. F. Lee, "A Fast Vector-Potential Method Using Tangentially Continuous Vector Finite Elements," *IEEE Transactions on Microwave Theory and Techniques*, vol. 46, pp. 86-868, Jun 1998.

[14] R. Dyczij-Edlinger, G. Peng and J.-F. Lee, "Stability conditions for using TVFEMs to solve Maxwell equations in the frequency domain," *Int. J. Numer. Modeling*, vol. 13, pp. 245–260, 2000.

[15] Y. Saad, "*Iterative Methods for Sparse Linear Systems"*, PWS Publishing Company, 1996, Chapter 10 - Preconditioning Techniques.

[16] M. Bollhofer and Y. Saad, "Multilevel preconditioners constructed from inverse-based ILUs," *SIAM J. Sci. Comput*., vol. 27, no. 5, pp.1627–1650, 2006.

[17] Y. Notay, "An Aggregation-Based Algebraic Multigrid Method," *Electronic Transactions on Numerical Analysis,* vol. 37, pp. 123-146, 2010.

[18] R. Hiptmair, "Multigrid method for Maxwell's equations," *SIAM Journal on Numerical Analysis*, vol. 36, pp. 204-225, Dec 1998.

[19] J. Gopalakrishnan, J. E. Pasciak, and L. F. Demkowicz, "Analysis of a multigrid algorithm for time harmonic Maxwell equations," *SIAM Journal on Numerical Analysis*, vol. 42, pp. 90-108, 2004.

[20] K. Watanabe, H. Igarashi, and T. Honma, "Comparison of Geometric and Algebraic Multigrid Methods in Edge-Based Finite-Element Analysis", *IEEE Transactions On Magnetics*, vol. 41, no. 5, May 2005.

[21] D. Lahaye, H. De Gersem, S. Vandewalle and K. Hameyer "Algebraic multigrid for complex symmetric systems," *IEEE Transactions on Magnetics*, vol. 36, no. 4, pp. 1535-1538, Jul 2000.

[22] K. Stuben, "A review of algebraic multigrid," *Journal of Computational and Applied Mathematics*, vol. 128, pp. 281-309, Mar 2001.

[23] S. Reitzinger and J. Schoberl, "An algebraic multigrid method for finite element discretizations with edge elements," *Numerical Linear Algebra with Applications*, vol. 9, pp. 223-238, Apr-May 2002.

[24] M. Kaltenbacher and S. Reitzinger, "Nonlinear three-dimensional magnetic field computations using lagrange finite-element functions and algebraic multigrid," *IEEE Transactions on Magnetics*, vol. 38, pp. 1489-1496, May 2002.

[25] J. J. Hu, R. S. Tuminaro, P. B. Bochev, , C. J. Garasi, and A. C. Robinson, "Toward an h-independent algebraic multigrid method for Maxwell's equations," *SIAM Journal on Scientific Computing*, vol. 27, no. 5, pp. 1669-1688, 2006.

[26] T. Mifune, S. Isozaki, T. Iwashita, and M. Shimasaki, "Algebraic multigrid preconditioning for 3-D magnetic finite-element analyses using nodal elements and edge elements," *IEEE Transactions on Magnetics*, vol. 42, pp. 635-638, Apr 2006.

[27] A. Quarteroni, A. Valli, "Domain Decomposition Methods for Partial Differential Equations", Oxford University Press, New York, 1999.

[28] M. Gander, F. Magoule`s, F. Nataf, "Optimized Schwarz methods without overlap for the Helmholtz equation," *SIAM Journal on Scientific Computing*. vol. 24, no. 1, pp. 38–60, 2002.

[29] S. C. Lee, M. N. Vouvakis, and J. F. Lee, "A non-overlapping domain decomposition method with non-matching grids for modeling large finite antenna arrays," *Journal of Computational Physics*, vol. 203. no. 1, pp. 1-21, 2005.

[30] K. Zhao,V. Rawat, S. C. Lee and J. F. Lee, "A Domain Decomposition Method with nonconformal Meshes for Finite Periodic and Semi-Periodic Structures ," *IEEE Transactions on Antennas and Propagation*, vol. 55, pp. 2559-2570, Sep. 2007.

[31] V. Rawat and J. F. Lee, "Non-overlapping domain decomposition method with second order transmission condition for the time-harmonic Maxwell's equations," *SIAM Journal on Scientific Computing*, vol.32, pp. 3584–3603, 2010.

[32] Z. Peng and J. F. Lee, "Non-conformal domain decomposition method with second order transmission conditions for time-harmonic electromagnetics," *SIAM Journal on Scientific Computing*, vol. 229, pp. 5615–5629, 2010.

[33] Y. Shao, Z. Peng, and J. F. Lee, "Full-Wave Real-Life 3-D Package Signal Integrity Analysis Using Nonconformal Domain Decomposition Method," *IEEE Transactions on Microwave Theory and Techniques*, vol. 59, pp. 230-241, Feb 2011.

[34] Z. Peng, and J. F. Lee, "A Scalable Nonoverlapping and Nonconformal Domain Decomposition Method for Solving Time-Harmonic Maxwell Equations

in $\mathbf{R}^3$," *SIAM Journal on Scientific Computing*, vol. 34, no. 3, pp. 1266-1295, 2012.

[35] J. Jin, *The finite element method in electromagnetics*, 2nd edition, John Wiley, 2002.

[36] M. M. Gavrilovic, and J. P. Webb, "A port element for p-adaptive S-parameter calculation," *IEEE Transactions on Magnetics*, vol. 35, no. 3, pp. 1530-1533, 1999.

[37] O. Bíró and K. Preis, "On the use of the magnetic vector potential in the finite-element analysis of three-dimensional eddy currents," *IEEE Transactions on Magnetics*, vol. 24, no. 4, pp. 3145-3159, Jul 1989.

[38] O. Bíró, "Edge element formulations of eddy current problems," *Computer methods in applied mechanics and engineering*, vol. 169, no. 3, pp. 391-405. 1999.

[39] J. P. Webb and B. Forghani, "A scalar-vector method for 3D eddy current problems using edge elements," *IEEE Transactions on Magnetics*, vol. 26, no. 5. pp. 2367-2369. Sep 1990.

[40] J. P. Webb and B. Forghani, "The low-frequency performance of $H-\phi$ and $T-\Omega$ methods using edge elements for 3D eddy current problems," *IEEE Transactions on Magnetics*, vol. 29, no. 6, pp. 2461-2463, Nov 1993.

[41] R. D Graglia, D. R.Wilton and A. F. Peterson, "Higher order interpolatory vector bases for computational electromagnetics," *IEEE Transaction on Antennas and Propagation,* vol. 45, no. 3, Mar 1997.

[42] R. D. Graglia, A. F Peterson and F. P. Andriulli, "Curl-conforming hierarchical vector bases for triangles and tetrahedral," *IEEE Transactions on Antennas and Propagation*, vol. 59, no. 3, pp. 950-959. Dec 2011.

[43] Z. Ren, and N. Ida, "High-order differential form-based elements for the computation of electromagnetic field," *IEEE Transactions on Magnetics*, vol. 36, no. 4, pp. 1472–1478, 2000.

[44] D. K. Sun, J. F. Lee and Z. Cendes "Construction of nearly orthogonal Nedelec bases for rapid convergence with multilevel preconditioned solvers," *SIAM Journal on Scientific Computing*, vol. 23, no. 4, pp. 1053–1076, 2001

[45] J. Schoberl, and S. Zaglmayr "High-order Nedelec elements with local complete sequence properties", *COMPEL*, vol. 24, no. 2, pp. 374–384, 2005.

[46] S. C Lee, J. F Lee and R. Lee, "Hierarchical vector finite elements for analyzing waveguide structures," *IEEE Transactions on Microwave Theory and Techniques*, vol. 51, no. 8, pp. 1897-1905, Aug 2003.

[47] P. Ingelstrom, V. Hill, and R. Dyczij-Edlinger, "Comparison of hierarchical basis functions for efficient multilevel solvers," *IET Science Measurement & Technology*, vol. 1, pp. 48-52, Jan 2007.

[48] J. P. Webb, "P-adaptive methods for electromagnetic wave problems using hierarchal tetrahedral edge elements," *Electromagnetics*, vol. 22, no. 5, pp. 443-451, 2002.

[49] P. Ingelström A. Bondeson, "Goal-oriented error estimation and h-adaptivity for Maxwell's equations," *Computer methods in applied mechanics and engineering*, vol. 192, no. 22, pp. 2597-2616, 2003.

[50] D. Nair and J. P. Webb, "P-Adaptive Computation of the Scattering Parameters of 3-D Microwave Devices," *IEEE Transactions on Magnetics*, vol. 40, no. 2, Mar 2004.

[51] J. Q. Chen, Z. M. Chen, T. Cui, and L. B. Zhang, "An Adaptive Finite Element Method for the Eddy Current Model with Circuit/Field Couplings," *SIAM Journal on Scientific Computing*, vol. 32, pp. 1020-1042, 2010.

[52] L. S. Andersen and J. L. Volakis, "Adaptive multiresolution antenna modeling using hierarchical mixed-order tangential vector finite elements," *IEEE Transactions on Antennas and Propagation*, vol. 49, pp. 211-222, Feb 2001.

[53] W. Rachowicz and L. Demkowicz, "An *hp*-adaptive finite element method for electromagnetics—Part II: A 3D implementation," *International Journal for Numerical Methods in Engineering,* vol. 53, no. 1, pp. 147–180, 2002.

[54] P. D. Ledger, K. Morgan, J. Peraire, and N. P. Weatherill, "The development of an *hp*-adaptive finite element procedure for electromagnetic scattering problems," *Finite Elements Analysis Design*, vol. 39, pp. 751–764, 2003.

[55] M. M. Botha and J. M. Jin, "Adaptive finite element-boundary integral analysis for electromagnetic fields in 3-D," *IEEE Transactions on Antennas and Propagation*, vol. 53, pp. 1710-1720, May 2005.

[56] Y. Zhu, , A. C. Cangellaris, "Nested multigrid vector and scalar potential finite element method for fast computation of two-dimensional electromagnetic scattering," *IEEE Transactions on Antennas and Propagation,* vol. 50. No.12, pp. 1850-1858, Dec 2002.

[57] Y. Zhu, and A. C. Cangellaris, "Nested multigrid vector and scalar potential finite element method for three-dimensional time-harmonic electromagnetic analysis", *Radio Science*, vol. 37, no 3, 2002.

[58] G. H. Peng, D. Dyczij-Edlinger, and J. F. Lee, "Hierarchical methods for solving matrix equations from TVFEMs for microwave components," *IEEE Transactions on Magnetics*, vol. 35, pp. 1474-1477, May 1999.

[59] Y. Zhu and A. C. Cangellaris, "Hierarchical multilevel potential preconditioner for fast finite-element analysis of microwave devices," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, pp. 1984-1989, Aug 2002.

[60] J. F. Lee and D. K. Sun, "*p*-Type multiplicative Schwarz (*p*MUS) method with vector finite elements for modeling three-dimensional waveguide discontinuities," *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, pp. 864-870, Mar 2004.

[61] Y. J. Sheng, R. S. Chen, and X. W. Ping, "An efficient p-version multigrid solver for fast hierarchical vector finite element analysis," *Finite Elements in Analysis and Design*, vol. 44, pp. 732-737, Aug 2008.

[62] D. K. Sun, J. F. Lee and Z. Cendes, "Construction of nearly orthogonal Nedelec bases for rapid convergence with multilevel preconditioned solvers," *SIAM Journal on Scientific Computing*, vol. 23, no.4, pp. 1053-1076, 2001.

[63] J. H. Bramble, J. E. Pasciak, J.Wang, and J. Xu, "Convergence estimates for product iterative methods with applications to domain decompositions and multigrid," *Math. Comput*., vol. 57, pp. 1–21, 1991.

[64] H. Igarashi, "On the property of the curl-curl matrix in finite element analysis with edge elements," *IEEE Transactions on Magnetics*, vol. 37, no.5, pp. 3129-3132, Sep. 2001.

[65] R. Dyczij-Edlinger, , O. Biro, "A joint vector and scalar potential formulation for driven high frequency problems using hybrid edge and nodal finite elements," IEEE Transactions on Microwave Theory and Techniques, vol. 44, no. 1, pp. 15-23, 1996.

[66] R. Dyczij- Edlinger, G. Peng, and J. F. Lee, "Efficient finite element solvers for the Maxwell equations in the frequency domain" *Computer methods in applied mechanics and engineering*, vol. 169, no.3, pp. 297-309. 1999.

[67] C. Chen and O. Bíró "Geometric Multigrid With Plane Smoothing for Thin Elements in 3-D Magnetic Fields Calculation," *IEEE Transactions On Magnetics*, vol. 48, no. 2, Feb 2012.

[68] 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*, vol. 25, no.2, pp. 623-642, 2003.

[69] 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*, vol. 25, pp. 623-642, 2003.

[70] R. Perrussel, L. Nicolas, F. Musy, L. Krahenbuhl, M. Schatzman, and C. Poignard, "Algebraic multilevel methods for edge elements," *IEEE Transactions on Magnetics*, vol. 42, pp. 619-622, Apr 2006.

[71] J. Xu, "The auxiliary space method and optimal multigrid preconditioning techniques for unstructured grids," *Computing*, vol. 56, pp. 215-235, 1996.

[72] R. Hiptmair, G. Widmer, and J. Zou, "Auxiliary space preconditioning in $H_0(\mathrm{curl};\Omega)$," *Numerische Mathematik*, vol. 103, pp. 435-459, May 2006.

[73] R. Hiptmair and J. C. Xu, "Nodal auxiliary space preconditioning in H(curl) and H(div) spaces," *SIAM Journal on Numerical Analysis*, vol. 45, pp. 2483-2509, 2007.

[74] R. Hiptmair and J. C. Xu, "Auxiliary space preconditioning for edge elements," *IEEE Transactions on Magnetics*, vol. 44, pp. 938-941, Jun 2008.

[75] T. V. Kolev and P. S. Vassilevski, "Parallel Auxiliary Space AMG for H(Curl) Problems," *Journal of Computational Mathematics*, vol. 27, pp. 604-623, Sep 2009.

[76] J. Xu, L. Chen, and R. H. Nochetto, "Optimal multilevel methods for H(grad), H(curl) and H(div) systems on graded and unstructured grids", *Multiscale, nonlinear and adaptive approximation*, R. DeVore and A. Kunoth, Eds., ed: Springer Berlin Heidelberg, 2009, pp. 599-659.

[77] J. Xu, Y. Zhu, "Robust preconditioner for H(curl) interface problems", in *Domain Decomposition Methods in Science and Engineering XIX*, Lecture Notes in Computational Science and Engineering 78, Springer-Verlag, 2011.

[78] A. Bossavit and I. Mayergoyz, "Edge-Elements for scattering problems," *IEEE Transactions on Magnetics*, vol. 25, pp. 2816-2821, Jul 1989.

[79] R. Perrussel, L. Nicolas, F. Musy, "An efficient preconditioner for linear systems issued from the finite-element method for scattering problems," *IEEE Transactions on Magnetics*, vo. 40, no.2, pp. 1080-1083, Mar 2004.

[80] Y. Erlangga, C. Oosterlee, and C. Vuik, "A novel multigrid based preconditioner for heterogeneous Helmholtz problems," *SIAM Journal on Scientific Computing*, vol. 27, pp. 1471-1492, 2006.

[81] M. B. van Gijzen, Y. A. Erlangga, , C. Vuik, "Spectral analysis of the discrete Helmholtz operator preconditioned with a shifted Laplacian," *SIAM Journal on Scientific Computing*, vol. 29, no.5, pp. 1942-1958, 2007.

[82] Y. A. Erlangga, "Advances in iterative methods and preconditioners for the Helmholtz equation," *Archives of Computational Methods in Engineering*, vol. 15, pp. 37-66, Mar 2008.

[83] E Björn and L. Ying, "Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers." *Multiscale Modeling and Simulation,* vol. 9, no. 2 pp.686-710, 2011.

[84] P. Tsuji, B. Engquist and L. Ying, "A sweeping preconditioner for time-harmonic Maxwell's equations with finite elements," *Journal of Computational Physics*, vol. 231, no. 9, pp. 3770-3783, 2012.

[85] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, H. Van der Vorst, "Templates for the solution of linear systems: building blocks for iterative methods," no. 43, *Society for Industrial and Applied Mathematics* 1987.

[86] H. A. van der Vorst, J. B. M. Melissen, "A Petrov–Galerkin type method for solving $Ax = b$, where A is symmetric complex", *IEEE Transactions on Magnetics*, vol. 26, no. 2, pp. 706-708, Mar 1990.

[87] *Matlab 7.10, the MathWorks, Inc., Natick, MA, 2000*. http://www.mathwork.com.

[88] R. A. Nicolaides, "Deflation of Conjugate Gradients with Applications to Boundary Value Problems", *SIAM J. Numer. Anal.*, 24 (1987), pp. 355–365.

[89] C. Andrew and Y. Saad. "Deflated and augmented Krylov subspace techniques," *Numerical linear algebra with applications*, vol. 4 no. 1, pp, 43-66, 1997.

[90] M. H.Carpenter, C. Vuik, P. Lucas, M. B. Van Gijzen and H. Bijl, "A general algorithm for reusing Krylov subspace information. I." *Unsteady Navier-Stokes. NASA/TM*, 2010.

[91] ElecNet 7.3, 2012, Infolytica Corporation, Montreal, Canada, http://www.infolytica.com.

[92] B. Weiß, O. Bíró, P. Caldera, K. Hollaus, G. Paoli, and K. Preis, "A Multigrid Solver for Time Harmonic Three-Dimensional Electromagnetic Wave Problems", *IEEE Transactions on Magnetics*, vol. 42, pp. 639-642, Apr 2006.

[93] Y. Zhu and A. C. Cangellaris, "Nested Multigrid Vector and Scalar Potential Finite Element Method for Fast Computation of Two-Dimensional Electromagnetic Scattering", *IEEE Transactions on Antennas and Propagation*, vol. 50, no. 12, Dec 2002.

[94] P. Ingelström and A. Bondeson, "Goal-oriented error estimation and h-adaptivity for Maxwell's equations," *Comput. Methods Appl. Mech. Eng.*, vol. 192, pp. 2597–2616, 2003.

[95] E. B. Tchikaya, F. Khalil, F. A. Tahir, and H. Aubert, "Multi-scale approach for the electromagnetic simulation of finite size and thick frequency selective surfaces", *Progress In Electromagnetics Research*, vol. 17, pp. 43-57, 2011.

[96] I. Gomez-Revuelto, L. E. Garcia-Castillo, and M. Salazar-Palma, "Goal-Oriented Self-Adaptive *hp*-Strategies for Finite Element Analysis of Electromagnetic Scattering and Radiation Problems," *Progress in Electromagnetics Research-Pier*, vol. 125, pp. 459-482, 2012.

[97] A. Aghabarati, J. P. Webb, "An algebraic multigrid method for the finite element analysis of large scattering problems", *IEEE Trans. Antennas and Propagation*, vol. 61, no.2, February 2013.

[98] A. Aghabarati, J. P. Webb, "Multilevel Methods for *p*-adaptive Finite Element Analysis of Electromagnetic Scattering", *IEEE Transactions on Antennas and Propagation*, Accepted for publication on Jul 30, 2013.

[99] A. Aghabarati, J. P. Webb, "Multilevel Preconditioning for Time-harmonic Eddy Current Problems Solved with Hierarchical Finite Elements", *4 pages journal submitted to the IEEE Transactions on Magnetics,* Jun 14, 2013.

[100] E. R. Iglesias, Ó. Q. Teruel, and M. M. Kehn, "Multiband SRR Loaded Rectangular Waveguide", *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 5, pp. 1570-1574, May 2009.

[101] F. Y. Meng, Q. Wu, D. Erni and L.W. Li, "Controllable Metamaterial-Loaded Waveguides Supporting Backward and Forward Waves", *IEEE Transactions on Antennas and Propagatio*n, vol. 59, no. 9, Sep. 2011.

[102] P. A. Bernhardt, "Radar Backscatter from Conducting Polyhedral Spheres", *IEEE Antennas and Propagation Magazine*, vol. 52, No.5, Oct 2010.

[103] Y. E. Erdemli, K. Sertel, R. A. Gilbert, D. E. Wright, and J. L. Volakis, "Frequency-Selective Surfaces to Enhance Performance of Broad-Band Reconfigurable Arrays", *IEEE Transactions on Antennas and Propagation*, vol. 50, no. 12, Dec 2002.

[104] R. Mittra, "A Look at Some Challenging Problems in Computational Electromagnetics", *IEEE Transactions on Antennas and Propagation*, vol. 46, no. 5, pp. 18-32, Oct 2004.

[105] Fujiwara and T. Nakata, "Results for benchmark problem 7," *COMPEL*, vol. 9, no. 3, pp. 137–154, 1990.

[106] G. Kovács, and M. Kuczmann, "Solution of the TEAM workshop problem No. 7 by the Finite Element Method," Przeglad Elektrotechniczny, vol 3, no 3, pp 99-102, 2011.

[107] C. Zhiguang, N. Takahashi and B. Forghani, "TEAM Problem 21 Family (V. 2009)," *approved by the International Compumag Society at Compumag*, 2009.

[108] M. Bollhöfer, , M. J. Grote, and O. Schenk, "Algebraic multilevel preconditioner for the Helmholtz equation in heterogeneous media," *SIAM Journal on Scientific Computing*, vol. 31, no. 5, pp. 3781-3805, 2009.

[109] H.C. Elman, O.G. Ernst, and D.P. O'Leary, "A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations," *SIAM Journal on Scientific Computing*, vol. 23, pp. 1291–1315, 2001.

[110] H. Chen, , H. Wu, and X. Xu, "Multilevel Preconditioner with Stable Coarse Grid Corrections for the Helmholtz Equation," *arXiv preprint*, pp. 1304.6825, 2013.

[111] M. L. Parks, , E. De Sturler, G. Mackey, D. D. Johnson and S. Maiti, "Recycling Krylov Subspaces for Sequences of Linear Systems," *SIAM Journal on Scientific Computing*, vol.28, no.5, pp. 1651-1674, 2006.