LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-TH-205466

# A Novel High Order Time Domain Vector Finite Element Method for the Simulation of Electromagnetic Devices

Robert N. Rieben

July 23, 2004

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

**A Novel High Order Time Domain Vector Finite Element Method for the Simulation of Electromagnetic Devices**

by

ROBERT N. RIEBEN

B.S. (University of California at Riverside) 1999
M.S. (University of California at Davis) 2001

DISSERTATION
Submitted in partial satisfaction of the requirements for the degree of
DOCTOR OF PHILOSOPHY

in

Applied Science

in the

OFFICE OF GRADUATE STUDIES
of the
UNIVERSITY OF CALIFORNIA
DAVIS

Approved:

Chair

Committee in Charge

2004

**A Novel High Order Time Domain Vector Finite Element Method for the Simulation of Electromagnetic Devices**



by

Robert N. Rieben

Robert N. Rieben

July 2004

Applied Science

A Novel High Order Time Domain Vector Finite Element Method for the Simulation of Electromagnetic Devices

## Abstract

The goal of this dissertation is twofold. The first part concerns the development of a numerical method for solving Maxwell's equations on unstructured hexahedral grids that employs both high order spatial and high order temporal discretizations. The second part involves the use of this method as a computational tool to perform high fidelity simulations of various electromagnetic devices such as optical transmission lines and photonic crystal structures to yield a level of accuracy that has previously been computationally cost prohibitive. This work is based on the initial research of Daniel White who developed a provably stable, charge and energy conserving method for solving Maxwell's equations in the time domain that is second order accurate in both space and time. The research presented here has involved the generalization of this procedure to higher order methods. High order methods are capable of yielding far more accurate numerical results for certain problems when compared to corresponding *h*-refined first order methods , and often times at a significant reduction in total computational cost. The first half of this dissertation presents the method as well as the necessary mathematics required for its derivation. The second half addresses the implementation of the method in a parallel computational environment, its validation using benchmark problems, and finally its use in large scale numerical simulations of electromagnetic transmission devices.

Professor Garry H. Rodrigue
Dissertation Committee Chair

This work is dedicated to my mom. Without her years of personal sacrifi ce and selfless nature, none of this would have been possible. I am forever grateful.

# Contents

# List of Figures

# List of Tables

## Acknowledgements

I would like to acknowledge my dissertation committee for their wisdom, guidance, and insight. I owe a tremendous debt of gratitude to my academic adviser Garry Rodrigue for inspiring me to enter the field of computational science, for teaching me the fine points of numerical methods with exceptional clarity, for guiding me through my research and keeping me focused on challenging and relevant problems and for advising on me on my long term scientific career goals. Thank you Garry for being my mentor these past four years. The results of this dissertation would not have been possible without the outstanding guidance of my technical adviser Daniel White. Thank you Dan for imparting your astounding knowledge and skill in my training, for constantly being available, for taking the time to show me the right way to do things and most importantly, for holding me to such a high standard. You have truly been an inspirational and trusted counselor to me these past years. Finally, I wish to thank Richard Freeman for his superb teaching skills, for serving on my examination committee (and asking me such tough questions!), for his leadership as the chairman of our department and for his singular wit.

I would also like to acknowledge my coworkers whose help and input is greatly appreciated. Thanks to Joe Koning for his reliable assistance with the EMSolve project and for imparting his physics knowledge and computer skills. Thanks to Paul Castillo for his mathematical advice and rigor, and for his significant contribution to the FEMSTER project. Thanks to Mark Stowell for his consistent and extraordinary support on the EMSolve project. Much thanks to the DAS administrative staff, Jane Keene, Estelle Miller, Dee Kindelt and Donna Clifford, for keeping me informed, up to date and on task. Thanks to the SEGRF program for providing me with financial support during my graduate research years.

Many thanks are due to my family for their support and encouragement. Thanks to my mom for raising me as a single parent and making every personal sacrifice to ensure that I could receive a good education. Thanks to my sister Liz and her husband John for being incredibly reassuring and invigorating during my years up here. Thanks to my nephew Kevin and my nieces Kathleen and Cecilia, you are the smartest, most creative and kind hearted people I know and I am certain you will all achieve great success as you progress through the years.

My years here in Livermore would have been unbearable were it not for the faithful support, relief and all around good times provided by my dear friends. To my best friends Terry and Kevin, our years together as rock-and-roll day-dreamers are among my fondest. Thanks for the countless weekends of pure joy where for a brief few moments, we knew we truly had it all – the memories I have of these times are hiding between the lines of this lengthy document, God bless the Mint! Thanks to Aaron and Patty, Tuesday nights at the Ale House will forever remain a time honored tradition for me. Thanks to Jason for being my roommate all these years – I think our quirky natures complimented each other in an equally quirky way. Thanks to Bahrad and Dave for some very wacky and Canadian inspired antics. Thanks to Emily and Jaime for letting me crash on your couch and keeping me entertained in the city. Thanks to Jae, Jamin, Kierstyn, Brandy, Meagan, Ben, Pete, Kristi, Dez and Seth for being such good friends.

# Chapter 1

# Introduction

## 1.1 Computational and Physical Motivation

In his 1873 publication *Treatise on Electricity and Magnetism* [1], James Clerk Maxwell combined the experimental results of Coulomb, Faraday, Oersted and Ampere, concerning the forces exerted between electric charges and electric currents, into a complete mathematical formulation of what is now known as the electromagnetic field. The hallmark of this work are the elegant and ubiquitous Maxwell equations, a set of partial differential equations (PDEs) that completely model all of classical electromagnetic phenomenon. To this day, these equations remain highly relevant. The ability to generate and control the propagation of electromagnetic energy has formed the basis of both the electronic and information technical revolutions of the past several decades. Solutions to Maxwell's equations have proved invaluable to electrical engineers at the forefront of research and development in communication technology. In many cases, a single electronic device can exhibit a very complicated structure involving a number of conductors and (possibly anisotropic and/or inhomogeneous) dielectric materials of arbitrary shapes. Because of this complexity, such devices are becoming increasingly expensive to fabricate; making a trial and error design process to yield optimal performance a cost prohibitive strategy. Highly accurate characterization methods based on solutions to Maxwell's equations are therefore extremely desirable for the development of such structures. This process is integral

to a primary tenet of computational physics and engineering: design and development in a computational environment to determine optimal physical parameters, followed by fabrication of the device according to the results.

Like most systems of PDEs that model physical phenomenon, Maxwell's equations rarely lend themselves to closed form analytic solutions. The cases in which this is true are limited to physical systems represented by trivial geometries (e.g. cubic or spherical domains) or systems in which a great degree of symmetry is present. Solutions to the time dependent Maxwell equations in non-trivial geometries (often coupled with non-trivial boundary and initial value conditions) require an approximation scheme. The various types of schemes span the entire spectrum of applied mathematics; however, they can typically be cast into two distinct categories: analytical approximate solutions and numerical approximate solutions. An example of an analytical approximation is a multi-pole expansion of the time-dependent electric potential of an arbitrary charge distribution [2]. The accuracy of such a scheme is determined by the number of non-vanishing moments that are computed in the expansion and by certain limiting cases such as the requirement that the time dependent positions of the charges are *slowly varying* [3]. Numerical approximations typically involve the direct replacement of the continuum fields and their spatial and temporal derivatives (or integrals) with discrete numerical analogues, thus converting the continuum equation into a finite number of algebraic equations which can be solved either directly or iteratively using a computer. This dissertation is concerned with a method of the latter type.

Numerical methods for solving Maxwell's equations have gained in popularity in recent years; a direct result of the rapid advances made in high performance computing and the need to model devices with non-trivial geometries. In general, these methods can be classified according to their type of numerical formulation: methods based on PDE formulations, such as the finite difference time domain (FDTD) and finite element methods (FEM), or methods based on integral formulations such as finite volume (or boundary element) methods. A further classification can be made depending on whether the method is formulated in the frequency or time domain. Each has its computational advantages and disadvantages as well as a respective area of computational electromagnetics (CEM) in which it excels. Examples of various numerical

approximation methods and their classifications are shown in Figure 1.1. This dissertation proposes a high order time domain vector finite element method for the direct numerical solution of Maxwell's equations on unstructured hexahedral grids. We focus our attention on the high order discretization of this problem in both space and time, and treat each case separately. As such, this method is also suitable for frequency domain (or time harmonic) problems; however, this aspect will not be fully explored in this dissertation. We will instead concentrate on direct time domain analysis. The computational advantages, disadvantages and range of physical problems for which the method is best suited will be discussed in detail.

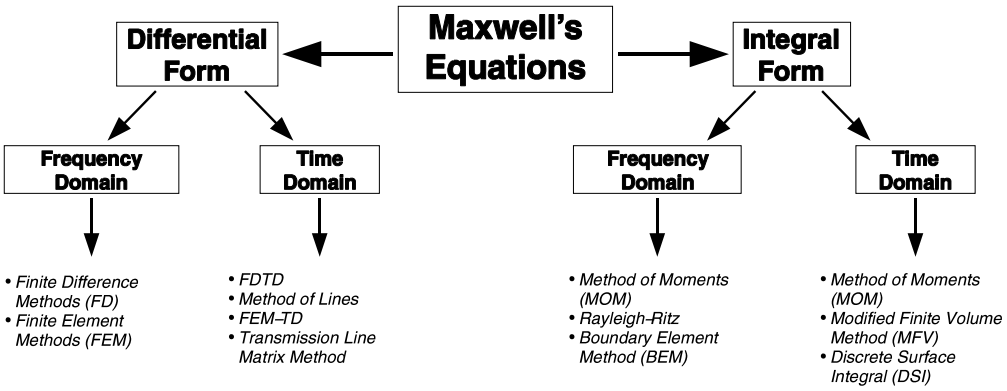


Figure 1.1: Classification of some numerical methods for solving Maxwell's equations.

Most numerical methods in use today are grid based, meaning that the unknown field variables are represented at discrete portions of space (and time) defined by a collection of points or cells. Oftentimes the numerical method used will be dictated by the type of grid or vice versa. As such, it is worthwhile to point out some salient features of various grid types. Figure 1.2 gives examples in order of increasing complexity of common grid types encountered in numerical modeling. Complex physical objects will require correspondingly complex grids to accurately represent their geometry. Likewise, complex grids require more advanced numerical approximation methods. The simplest case is the orthogonal, structured, conforming (or Cartesian) grid. It is very straightforward to implement a finite difference scheme on grids of this nature; however, the range of physical problems which can be accurately modeled is severely limited by the simple nature of the

grid. Non-orthogonal grids can alleviate this restriction by allowing for piece-wise linear approximations to curved boundaries. However, to accurately model the widest class of geometries, an unstructured grid is required. Finite difference methods break down on grids of this type because there is no one-to-one mapping from points in an unstructured grid to a Cartesian grid. Therefore, cell based methods such as the FEM must be used. The method proposed in this dissertation is valid on grids of this type. Non-conforming grids are a special case typically encountered in situations where a particular region is refined. They are discernible by their so called "hanging-nodes." Special constraints must be applied between elements that share a "hanging-node" in order for a method to work on a non-conforming grid. Due to their complexity and limited use, non-conforming grids will not be addressed any further.

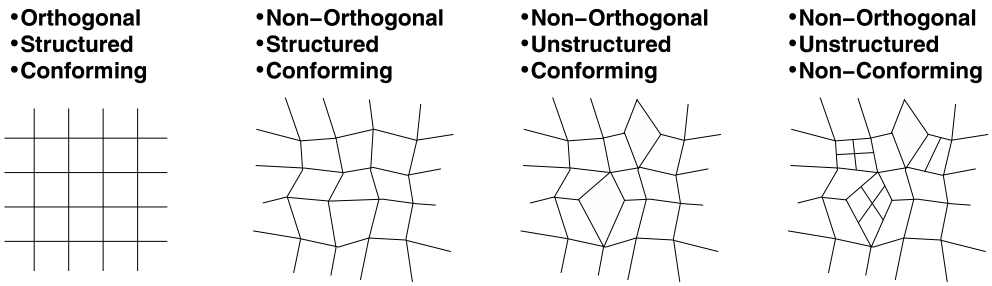


Figure 1.2: Comparison of grid types of increasing complexity.

## 1.2 Historical Context of the Method

Direct numerical solutions of Maxwell's Equations have been in existence for several decades; here we provide a brief historical development of such methods and the eventual progression to the current state of the art, including the proposed method. This overview is by no means exhaustive; for more historical details the reader is referred to [4], [5], [6], [7] and [8].

### 1.2.1 Earliest Methods – Finite Difference and Method of Moments

Grid based numerical methods for electromagnetics originated in 1966 when Yee proposed the FDTD method [9]; a technique that remains very popular to this day. In the original formulation, the electric field is discretized over a point grid that is offset both spatially and temporally from a "dual-grid" over which the magnetic field is discretized. The curl operator is approximated with a second order central difference formula. The system is integrated in time via an explicit, second order "leap frog" update method where field values at a current time step are calculated in terms of the field values at previous time steps. In an alternative formulation, the curl-curl operator can be discretized on a single electric field grid yielding a numerical analogue to the second order vector wave equation. Both methods are conditionally stable, charge and divergence preserving, non-dissipative and consistent, leading to second order convergence as the grid is refined in space and time [10]. The effects of numerical dispersion and grid anisotropy are kept under control by requiring a sufficient amount of grid points per wavelength [11]. To account for open-region problems (e.g. the Sommerfeld radiation boundary condition), several techniques have been developed (many of which can be adapted for other methods) such as the absorbing boundary condition (ABC) and the perfectly matched layer (PML) [12], [13], [14]. The combination of these properties along with the method's efficiency, elegance and ease of implementation have established the FDTD as the benchmark method in CEM to which all other methods are compared.

Despite these benefits, the method is certainly not without it limitations, and it is precisely these limitations which have lead to the development of new, more advanced methods. The shortcomings of the FDTD method are mainly twofold. The first is the restriction of the method to Cartesian grids (or those which can be mapped to Cartesian grids). Objects with curved boundaries must be gridded in a "stair-step" manner and it has been shown that such approximations can give very poor results [15], [16], [17]. The second limitation of the FDTD method is numerical dispersion. Numerical dispersion is the nonphysical dependence of computed wave propagation velocity on frequency; resulting in a cumulative growth in global phase error for time-dependent problems (also called the pollution effect [18]). The consequences of this purely numerical phenomenon are present in any grid based method; the goal therefore is to reduce its effect

as much as possible. This is typically accomplished by adding grid points to a mesh to more fully resolve the spatial and temporal nature of an electromagnetic wave. However, for certain "electrically large" problems in which several wavelengths span the computational domain or for certain broad-band applications, it can become prohibitively expensive to achieve a prescribed tolerance for numerical dispersion error using standard grid refinement. A more efficient way to reduce numerical dispersion is to employ a higher order method. High-order spatial discretizations can yield extremely accurate and efficient results for certain problems with smoothly curved boundaries, and they can drastically reduce the effects of numerical dispersion [19], [20], [21], [22]. Extensions of the FDTD method to higher order versions have been published, such as the fourth order accurate methods of [23], [24], [25] and [26], however these methods were developed for 2D orthogonal grids and often have difficulties maintaining accuracy at material interfaces and PEC boundaries due to the high order finite difference stencils.

Emerging at roughly the same time as the FDTD method is the so called method of moments (MOM); originally proposed for CEM applications in 1967 by Harrington [27], [28]. A precursor to the FEM, this method is suitable for either differential or integral equations [29]. Historically, it achieved its popularity in the discretization of the electric field integral equation (EFIE) for open-region scattering problems. Unlike differential equation-based methods, the integral equation formulations solve for the field sources (charges and currents). The sources can be line (e.g. wire antennas), surface (e.g. conducting plates) or volume (e.g. dielectric bodies) based. Beginning with the boundary conditions for the open region problem, the fields are expressed in terms of the sources through certain integrals (usually with the aid of potentials) which involve Green's functions. This formulation explicitly accounts for the open-region boundary condition; therefore no artificial or absorbing boundary condition is required (as is the case with FDTD). The sources appear as unknowns under the appropriate integrals and are discretized by a basis function expansion. The most common formulation is for 2D surface scattering in which the sources are expanded over the RWG basis functions originally proposed in 1982 by [30]. These functions are defined for triangular patches and are free of fictitious line or point charges which had plagued previous implementations. For scattering problems which can be formulated as surface integral equations, this method can yield extremely efficient results. However,

there are some serious drawbacks to the MOM. The range of problems which can be expressed as a surface integral equation is limited. In particular MOM based techniques for the EFIE can become impractical when dealing with structures with highly inhomogeneous or nonlinear media. Also, the resulting linear systems which are generated by the discretization process are quite dense (completely full for 3D problems) and have complex (imaginary) values, resulting in poor performance for iterative solvers. Most importantly, *a priori* knowledge of the Green's function is required; a case which is seldom known for complex electromagnetic configurations. Despite these limitations, the MOM remains very popular and is implemented in several large scale codes including the well known Numerical Electromagnetics Code (NEC) originally developed at the Lawrence Livermore National Laboratory in 1981 [31] and currently in widespread commercial use.

### 1.2.2 Finite Volume Methods

In order to circumvent the limitations of structured and/or orthogonal grids, there have been several attempts to generalize the FDTD method to unstructured grids [32]. In 1990 Madsen proposed the modified finite volume (MFV) technique, a direct generalization of the FDTD to arbitrary convex polygons [33], [34]. Like the FDTD, the method is formulated over a dual volumetric grid for spatial discretizations of the electric and magnetic field intensities. Discrete versions of the integral from of the Ampere and Faraday laws are formed, then computed using low-order numerical integration techniques. The time discretization is accomplished using the non-dissipative second order accurate leap frog scheme. This method will reduce to the standard FDTD when used on orthogonal grids and maintains the divergence-free property of the fields to within machine precision. However, a serious defect of the MFV method is that it exhibits late time growth of the solution amplitude (often called a "late-time instability") for problems on non-orthogonal grids, irrespective of the time step used. As such, Madsen proposed another method in 1995 called the discrete surface integral (DSI) method [35]. The DSI algorithm consists of a surface integral approximation and likewise preserves the divergence-free property and reduces to the FDTD on orthogonal grids. However, the late time-instability is delayed but not completely eliminated. The instability is caused by the non-symmetric discretization of the curl-curl operator [5]. The use of dissipative time integration schemes counteracts this

non-physical solution growth, but this results in a violation of numerical charge and energy conservation. In addition, neither method generalizes to higher order versions.

Developing alongside the previous methods are a different class of FVTD techniques derived from approaches used commonly in the field of computational fluid dynamics (CFD) [36], [37], [38]. In these methods the Ampere and Faraday laws are cast in so-called conservative form, resulting in a PDE that resembles the Euler equation. Typically these methods are implemented on a non-orthogonal structured, hexahedral grid where flux matching techniques are used to relate the fields across cell boundaries. The standard CFD methods of time integration, such as Lax-Wendroff or Jameson-style Runge Kutta, are used to solve the resulting equations. It has been shown that these methods are stable and consistent, thus very high levels of accuracy can be obtained as the grid is refined. However, these methods rely on dissipative time integration to maintain numerical stability, thus they do not conserve energy. Additionally, the method does not reduce to FDTD on orthogonal grids and the divergence-free properties of the fields are maintained only to the level of the method's truncation error resulting in divergent magnetic fields (i.e. magnetic monopoles); a direct violation of Maxwell's equations. This fact can be disconcerting for a computational physicist; however, from a purely numerical standpoint, such violations do not seem to affect the accuracy of certain calculations such as the computation of the radar cross section (RCS) of a dielectric body. Thus these methods remain popular.

### 1.2.3 Early Finite Element Methods

The finite element method (FEM) has been in use in the engineering community since the late 1950's. It was originally developed to solve structural and aeronautical mechanics problems on arbitrary geometries, and is therefore well suited for unstructured grids. It was later adapted for CEM applications by Silvester [39] in 1969. In a standard FEM procedure, the original boundary value PDE is cast in a variational formulation in terms of an energy-related functional. The computational domain is then subdivided into discrete elements such as triangles and quadrilaterals (for 2D problems) or tetrahedra and hexahedra (for 3D problems). A "trial" solution is then defined over each element in terms of a basis function expansion. The resulting variational functional is minimized with respect to the unknown basis function coefficients through

a process called the Galerkin method. This yields a set of algebraic equations which are then solved to obtain the basis function coefficients and therefore a piecewise approximate solution to the field problem. Any derivatives or integrals involved in the computation are computed exactly or to some prescribed error tolerance. In a classical FEM formulation, the basis functions are first order interpolating polynomials and the coefficients of the expansion are the approximate values of the field unknowns at the vertices (or nodes) of the element.

The classic FEM using nodal (or scalar valued) basis functions has been quite successful in solving static electromagnetic problems where the continuous electrostatic potential can be employed [40], [41], [42]. However this approach has been unsuccessful for directly solving the full vector form of Maxwell's equations for either the electric or magnetic fields. There are two problems with nodal basis functions in this context: 1) they force continuity of the fields across material interfaces, even when there is supposed to be a field discontinuity, and 2) they permit "spurious modes", or non-physical solutions, which do not disappear as the grid is refined, resulting in a non-converging method and late time instabilities [43], [44], [45], [46]. While the subject of spurious modes has been extensively investigated in the context of frequency domain electromagnetics [47], [48], [49], [50] [51], it is also a problem with recently developed time-domain methods [52], [53], [54]. Procedures have been proposed to separate or distinguish these spurious modes from the desired physical solutions, typically involving the application of constraints after the FEM solution has been computed; however these techniques have never been successful at fully removing their presence for all CEM problems [55], [56], [57]. The ultimate source of these problems lies in the improper spatial discretization of the curl operator in Maxwell's equations and is not a problem with the FEM intrinsically, but rather with the choice of basis functions used in the formulation of the method.

### 1.2.4 Advanced Finite Element Methods

In 1980 Nédelec published his seminal paper in which he proposed a set of conforming finite elements for the spaces $H(Curl)$ and $H(Div)$ for use in the direct computation of the vector valued electric and magnetic fields of Maxwell's equations [58]; a generalization of a previous method developed for 2D

problems by Raviart and Thomas [59]. These new vector valued finite elements were capable of correctly discretizing the curl and divergence operator, thus eliminating spurious modes. However, because of the highly mathematical nature of this paper, direct implementations of these results were slow to materialize, and often times plagued with errors. Nevertheless, over the course of two decades this work spawned a deluge of publications proposing new vector valued basis functions for the direct solution of Maxwell's equations. Due to the sheer volume of these publications and the abstract nature of the original paper, much confusion permeated the CEM community. A myriad of names began popping up such as Nédeléc elements, Whitney elements [60], covariant projection elements [61], [62], [63], edge elements [64], [65], [66] , face elements [67], tangential elements [68], curl-conforming elements [69], divergence-conforming elements, vector finite elements [70], differential form-based elements [71], and mixed-order elements [72]. At first glance, it is difficult to discern that all of these methods are in fact variations of the same fundamental result of Nédelec's original publication. Nonetheless, all of these works have the common desire to discretize the full vector form of Maxwell's equations (in either the frequency or time domain) over non-trivial geometries while eliminating the presence of spurious non-physical solutions.

Another key factor in the surge of research into advanced finite element methods for CEM was the promise of higher order extensions of the method. The results of Nédelec's paper placed no restrictions on the order of spatial accuracy that could be obtained with the newly proposed elements; however no explicit high order basis functions were provided. Several concrete implementations of this idea were subsequently published [73], [74], [75], [76], [77]. As investigation in the field reached a steady state, a few publications stand out. In [78], formulae for arbitrary order interpolatory bases on various element topologies are developed. These basis functions are explicit and easily computed, but are based on a fixed set of uniformly spaced interpolation points. It is not clear that they are optimal for high-order interpolation or how they affect the conditioning of element mass and stiffness matrices. In addition, since these basis functions are purely interpolatory, they are not scale invariant and the construction process requires the elimination of redundant basis functions in a post-processing step. In [79], hierarchical vector bases for tetrahedrons are presented up to degree 3. The tetrahedral bases presented in [80] and [81] are generalized for arbitrary polynomial degree.

Hierarchical bases enable the implementation of *p*-refinement methods, where elements in a mesh can have different degrees of approximation; a task which is prohibitively complicated to perform using standard interpolatory bases. This can prove to be quite useful as sections of a computational domain can be selectively or adaptively *p*-refined in order to achieve a greater error tolerance without the cost of refining the entire domain.

While each of these publications present high order vector bases for discretizing the electric and magnetic fields in space, they are primarily concerned with frequency domain calculations and do not discuss direct time domain discretization. In addition, a full finite element solution scheme for Maxwell's equations (in either the frequency or time domain) requires many more mathematical / numerical components in order to be useful for realistic applications. These include the ability to construct local mass and stiffness matrices which account for possibly tensor valued and / or spatially dependent constitutive relations, the ability to construct rectangular matrices which represent discrete versions of the curl and divergence operators for mixed finite element methods and the ability to construct local matrices which account for radiation boundary conditions such as the ABC matrix. Even more important is the need for a global matrix assembly process which "glues together" the local matrices into a global system matrix in a manner that ensures correct element to element connectivity. Once the global matrix is constructed, it is necessary to apply possibly time and space dependent boundary conditions (such as voltage and current sources) in a consistent manner that is valid for arbitrary order basis functions. A rigorous validation procedure requires a normed error analysis for problems which have a known analytic solution. These tasks require a mathematical object known in the finite element community as a "projection operation" (or degrees of freedom). An explicit formulation of this is required to account for arbitrary sources and boundary conditions in any high order finite element scheme. Finally, once the spatial discretization has been achieved by constructing the global matrices, a stable and consistent time integration method must be emplyed to propagate the solutions in time.

Of the few high order vector finite element methods that are implemented in the time domain, none employ high order temporal discretization methods. Standard second order accurate schemes such as the leap frog method are still used [82]. Given the substantial computational effort to assemble highly accurate spatial

discretizations, it becomes necessary to use higher order time integration methods as well in order to avoid "throwing away" some of the achieved accuracy. While it is certain that high order time integration schemes can yield more accuracy, it is not clear that it will it be worth the resulting computational cost. In addition, the leap frog method has the desired benefit of being energy conserving. A goal of this dissertation is to obtain high order methods which are also energy conserving.

### 1.2.5 Mimetic Discretizations

In hindsight of the years of research into CEM problems it became clear to many that despite its simplicity and restrictions, the original Yee scheme of 1966 possessed several highly desirable properties, namely: conservation of charge and energy, divergence-free fields, numerical stability and no spurious modes. This realization has spawned a new philosophy for numerical methods, known today as mimetic (or compatible) discretization methods [83], [84], [85]. The guiding principle is that given a set of equations which model a particular physical system, the discretization process should accurately reproduce all of the salient continuum features of the physical system. For spatial discretizations of Maxwell's equations this implies that numerical versions of the curl and divergence operator should reproduce the mathematical properties of their continuum counterparts (e.g. the divergence of the curl of a vector field is always zero). The same can be said of temporal discretizations of Maxwell's equations. For physical systems in which total electromagnetic energy is conserved; subsequent integration methods should conserve the numerical value of the electromagnetic field energy (i.e. they should be non-dissipative). It turns out that the Yee FDTD scheme is fully mimetic. A more contemporary method is that of White's 1997 doctoral dissertation, in which a fully mimetic, second order accurate, time domain vector finite element method for solving Maxwell's equations on unstructured grids was presented [5].

Recently, Hiptmair, motivated by the exterior algebra of differential forms, presented a unified framework for the construction of conforming finite element spaces. Remarkably, the standard scalar valued (or Lagrangian) finite element space and the newly developed $H(Curl)$ and $H(Div)$ conforming finite element spaces can be derived within this framework; along with precise definitions of their degrees of freedom

and interpolation operators [86]. The equations of electromagnetics can be simply and elegantly cast in the language of differential forms [87], [88], [89], [90], [91], [71]. In this approach the scalar electrostatic potential is a 0-form, the electric and magnetic fields are 1-forms, the electric and magnetic fluxes are 2-forms,and the scalar charge density is a 3-form. The basic operators are the exterior (or wedge) product, the exterior derivative, and the Hodge star. Precise rules (i.e. a calculus) prescribe how these forms and operators can be combined. In this modern geometrical approach to electromagnetics the fundamental conservation laws are not obscured by the details of coordinate system dependent notation. Hiptmair's framework presents the notion of a "discrete" differential form for approximating differential fields. In our terminology, a discrete differential $l$-form is a finite element basis function expansion used to discretize an $l$-form field. By working within the discrete differential forms framework, we are gauranteed that our resulting spatial discretization schemes are fully mimetic.

## 1.3 Benefits of the Proposed Method

The high order time domain vector finite element method described in this dissertation is unique in several ways. Here we present the key advantages of the proposed method.

- **Arbitrary Order Accuracy in Space**

High order spatial discretization was accomplished by developing a set of general polynomial vector basis functions of *arbitrary* degree. These vector basis functions satisfy the properties of the recently proposed differential forms based approach for constructing 1-form (also known as curl-conforming, $H(Curl)$ or "edge") bases and 2-form (also known as divergence-conforming, $H(Div)$ or "face") bases. For the Galerkin procedure applied to either the frequency domain or time dependent Maxwell equations, there are significant advantages to both 1-form and 2-form finite element basis functions [92]; including the proper modeling of the jump discontinuity of field intensities and flux densities across material interfaces, the elimination of spurious modes in eigenvalue computations and the conservation of charge in time-dependent simulations [92]. High-order spatial discretizations can yield extremely accurate and efficient results for certain problems with

smoothly curved boundaries, and they can drastically reduce the effects of numerical dispersion [20] [21]. High-order polynomial basis functions can be classified as either interpolatory or hierarchical, both types will be discussed and developed. In addition, a global assembly process based on symmetry group operations is presented which ensures the correct element to element connectivity for arbitrarily oriented elements in an unstructured mesh.

- **Up to 4th Order Accurate in Time**

High order temporal discretization was achieved by adapting a general symplectic integration algorithm (commonly used in the fields of astrophysics and molecular dynamics) for the case of the coupled first order Maxwell equations. Symplectic methods have the benefit of conserving total electromagnetic field energy and are therefore preferred over dissipative methods (such as Runge-Kutta) in applications that require high-accuracy and energy conservation over long periods of time integration. We show that in the context of symplectic methods, several popular schemes can be elegantly cast in a single algorithm which can easily be extended to higher order versions. We present numerical evidence which demonstrates the superior performance of high order time integration methods, especially when used in conjunction with high order spatial discretizations.

- **Metric Free Discrete Differential Forms Framework**

We derive the spatial discretization process using the language of discrete differential forms. This in turn yields a cohesive, metric-free framework where we can define and develop all of the necessary operators for constructing arbitrary order spatial discretizations of the various fields from Maxwell's equations. We follow the work of Ciarlet [93] and adhere to the strict mathematical definition of a finite element as a set of three distinct objects $(\Sigma, \mathcal{P}, \mathcal{A})$ such that:

- $\Sigma$ is the polyhedral domain over which the element is defined
- $\mathcal{P}$ is a finite dimensional polynomial space from which basis functions are constructed
- $\mathcal{A}$ is a set of linear functionals (*Degrees of Freedom*) dual to $\mathcal{P}$

Separating the element $\Sigma$ from the basis allows for curvilinear elements of arbitrary geometry order. Precise definitions for $\mathcal{A}$ are needed to define a projection operation. This is necessary for applying boundary conditions and performing normed error analysis. To construct the local element matrices, we also present explicit bilinear forms. These bilinear forms allow for the construction of local mass and stiffness matrices with spatially dependent tensor valued material property functions. In addition, the use of special rectangular bilinear forms allows for mixed finite element methods to solve coupled PDE systems. The bilinear forms are implemented using arbitrary order integration rules, allowing for exact integration of the bilinear form in many cases. Finally, the discrete differential forms framework lends itself well for object oriented implementation. We present the C++ class library FEMSTER, which combines all of the tools necessary for constructing single element spatial discretization matrices of arbitrary order.

- **Fully Mimetic in Space and Time**

The use of discrete differential form basis functions provides divergence free and curl free fields and thus conservation of charge, as well as the correct continuity of fields across material interfaces and the elimination of spurious modes. These properties are crucial for the elimination of late time instabilities caused by improper spatial discretization as investigated by [94], [95] and [96]. Conservation of energy is maintained by using symplectic time integrators. All of these properties are preserved automatically, i.e. they are a direct consequence of the discretization method and no post-processing is required.

## 1.4 Dissertation Synopsis

The structure of the dissertation is as follows. Chapter 2 gives a brief overview of the physics involved in electromagnetics; namely the time dependent Maxwell equations and the material dependent constitutive relations. We will overview the fundamental properties of electromagnetic fields and waves, their interactions with each other and matter, there properties at material interfaces, conservation of charge and energy, and wave propagation. We will present the formal set of PDEs which this dissertation is concerned with. This section will be presented using the standard notation of vector calculus. In Chapter 3 we provide

a concise introduction to the calculus of differential forms and explain its use and relevance to CEM. From this point on, all mathematical descriptions of Maxwell's equations and its components will be written in the language of differential forms. Chapter 4 gives an exhaustive overview of the spatial discretization process and all of the components required to ultimately produce matrix versions of the spatial differential operators of Maxwell's equations (e.g. discrete versions of the curl, divergence and curl-curl operators). In Chapter 5 we give an overview of the FEMSTER software, an object oriented class library of discrete differential forms. Chapter 6 explains the symplectic time integration process used to numerically integrate the time dependent algebraic equations which are produced from the spatial discretization process. In Chapter 7 we validate the method by performing tests on simple problems with known analytic solutions and verify the error convergence properties. Having validated the method, we now proceed to large scale simulations of actual devices in Chapter 8. In Chapter 9 we conclude the dissertation and discuss the future application of the method.

# Chapter 2

# Electromagnetics and Maxwell's Equations

## 2.1 Electromagnetic Fields

Electromagnetic phenomena are fully characterized by four vector fields: the electric and magnetic field intensities, **E** and **H**, and the electric and magnetic flux densities (often called displacement fields), **D** and **B**. These fields are generated by sources and are sustained in a specific medium or material. Maxwell's equations define the precise spatial and temporal relationships between these fields, their sources and the medium in which they exist. In rational MKS units, they are traditionally written as

$$\nabla \cdot \mathbf{D} = \rho \tag{2.1}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{2.2}$$

$$\nabla \times \mathbf{H} = \frac{\partial}{\partial t}\mathbf{D} + \sigma\mathbf{E} + \mathbf{J} \tag{2.3}$$

$$\nabla \times \mathbf{E} = -\frac{\partial}{\partial t}\mathbf{B} \tag{2.4}$$

Expressed in this manner, the left hand side of each equation defines one of the four vector fields in terms of its sources appearing on the right hand side. For example, the source for the electric displacement field **D** is a free charge density denoted by $\rho$, while the source for the electric field intensity **E** is a time varying magnetic flux density **B**. The term **J** is a free current density source while the term $\sigma$ represents the electric conductivity of the medium in which the fields exist. The differential operators $\nabla\times$ and $\nabla\cdot$ (*curl* and *div* respectively) impose restrictions on the spatial nature of each field in terms of its sources. For example, equation (2.2) states that the magnetic flux density **B** must be divergence free, implying that there are no monopole sources for this particular field.

The relationships between the displacement fields, **D** and **B**, and the electric and magnetic field intensities, **E** and **H**, are typically defined in terms of media-dependent constitutive relations which have the general form

$$\mathbf{D} = \mathbf{D}(\mathbf{E},\mathbf{H})$$

$$\mathbf{B} = \mathbf{B}(\mathbf{E},\mathbf{H})$$

The detailed structure of the constitutive relations is often determined by experiment, or in some special cases it may be derived analytically from atomic and solid state theory. These relations determine the response of the medium to the incident electric and magnetic field intensities. This response can be non-linear for certain materials or for strong enough incident field intensities. Examples of this are soliton wave propagation in optical fibers and second harmonic generation in non-linear crystals. In this dissertation, we restrict the constitutive relations to be linear in nature, and of the particular form

$$\mathbf{D} = \varepsilon\mathbf{E} \tag{2.5}$$

$$\mathbf{B} = \mu\mathbf{H} \tag{2.6}$$

where $\varepsilon$ is known as the dielectric permittivity and $\mu$ is the magnetic permeability. The values of $\varepsilon$, $\mu$ and $\sigma$ define the physical properties of the medium in which the electromagnetic fields exist. The parameter $\varepsilon$ represents the dielectric properties of a material and will determine its response to an electric field. Typically this response is isotropic; however there are certain materials which have dispersion characteristics that are

dependent on the direction of the incident field. In such cases, the relation between the electric displacement field **D** and the incident electric field intensity **E** of (2.5) will be an-isotropic in nature. The magnetic permeability, $\mu$, represents the degree to which a material can concentrate magnetic field lines. The more concentrated the field lines, the greater the magnetic permeability. The parameter $\sigma$ represents the electric conductivity and is a measure of how well a material accommodates the transport of electric charge. This in turn determines how "lossy" the material is, i.e. how much electromagnetic energy is dissipated from the fields to Joule heating over time. Typically, these parameters are simple scalar valued constants which can span a wide spectrum of values. In such cases, these values represent rather simple materials such as linear, isotropic and homogeneous (LIH) media. However, in more interesting materials, the values of these parameters will be more complicated. In general, they can be arbitrary tensor functions of both time and space. For example, a photonic crystal structure is defined by a periodic array of dielectric materials which may be anisotropic. In this case, the value of $\varepsilon$ will be a symmetric positive definite tensor that is a step function of space. In this dissertation, we restrict the values of $\varepsilon$, $\mu$ and $\sigma$ to be independent of time for the reasons mentioned in Section 1.3 of Chapter 1. However, they are free to be spatially dependent and tensor valued.

## 2.2 Boundary Conditions at a Surface of Discontinuity

As mentioned in Chapter 1, the behavior of the vector fields at the interface between two materials is important. Consider an arbitrary vector field **F** and two regions, each with different material properties, which meet at a common interface. We want to determine how the components of the vector field behave across this interface. Suppose we perform a volume integral spanning both regions using the cylindrical domain depicted in Figure 2.1. Integrating the divergence of the field **F** and using the Gauss divergence theorem gives

$$\int_v (\nabla \cdot \mathbf{F})\, dv \;=\; \oint_s \mathbf{F} \cdot d\mathbf{s} \;=\; \hat{\mathbf{n}} \cdot (\mathbf{F}_2 - \mathbf{F}_1)\, \Delta a + W$$

where the term $W$ represents the surface integral over the cylindrical wall and the terms $\mathbf{F}_1$ and $\mathbf{F}_2$ represent the field in regions 1 and 2 respectively. The vector $\hat{\mathbf{n}}$ points from region 1 to region 2. Now if we let the

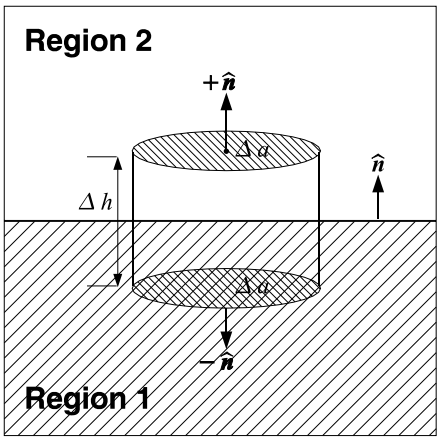


Figure 2.1: Depiction of volume integral for deriving normal continuity of a flux density.

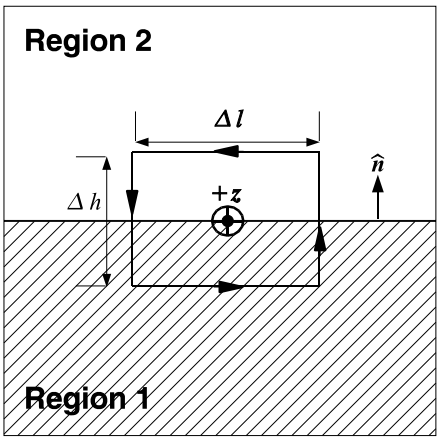


Figure 2.2: Depiction of surface integral for deriving tangential continuity of a field intensity.

distance separating the two end caps, $\Delta h$, go to zero, the surface integral term $W$ will vanish and we are left with the result

$$\hat{\mathbf{n}} \cdot (\mathbf{F}_2 - \mathbf{F}_1) \;=\; \lim_{\Delta h \to 0} \Delta h (\nabla \cdot \mathbf{F}) \tag{2.7}$$

The right hand side of this result will be zero provided that the divergence of the field **F** is "well defined" (a term we will clarify in Chapter 3). If this is the case, then (2.7) implies that at the interface between regions 1 and 2, the *normal* component of **F** will be continuous, allowing for the possible discontinuity of the tangential components. Now suppose we perform a surface integral spanning the same regions using the rectangular domain depicted in Figure 2.2. This time we integrate the curl of the field **F** and use the Stokes theorem to obtain a directed path integral

$$\int_s (\nabla \times \mathbf{F}) \cdot d\mathbf{s} \;=\; \oint_l \mathbf{F} \cdot d\mathbf{l}$$

where **l** is the directed path representing the boundary of the rectangular surface as shown in Figure 2.2. Again, we let the value $\Delta h$ go to zero, and after some calculations we are left with the result

$$\hat{\mathbf{n}} \times (\mathbf{F}_2 - \mathbf{F}_1) \;=\; \lim_{\Delta h \to 0} \Delta h (\nabla \times \mathbf{F}) \tag{2.8}$$

The right hand side of this result will be zero provided that the curl of the field **F** is well defined. If this is the case, then (2.8) implies that at the interface between regions 1 and 2, the *tangential* components of **F** will be

continuous, allowing for the possible discontinuity of the normal component.

From this analysis we have determined that for the case of three dimensional space, there are two types of vector field continuity at a material interface. This continuity is determined by the nature of the integral under which the field appears (or the differential operator which acts on the field). Using the standard notation of vector calculus, it is not clear which type of continuity an arbitrary field may posses; a field is either vector valued or not. In Chapter 3 we will introduce the calculus of differential forms, making the distinction between these two types clear by providing a more general insight into the characterization of differential fields. For the case of Maxwell's equations, it turns out that the field intensities, **E** and **H**, have tangential continuity while the electric and magnetic flux densities (displacement fields), **D** and **B**, have normal continuity.

## 2.3 Electromagnetic Wave Propagation

Solutions to Maxwell's equations can take various forms, including very simple text book cases such as electrostatic and magnetostatic radiation patterns, standing waves in a resonant cavity, plane and spherical waves, etc .... From an engineering perspective however, the interesting solutions involve propagating electromagnetic (EM) waves of a more complex nature. Propagating EM waves are incredibly important because they allow for the long distance transmission of electromagnetic power and energy; and more importantly, the transmission of information which can be encoded in the EM wave. For example, radio antennas broadcast audio signals which are encoded in an EM carrier wave by either amplitude ($\sim$ 1 MHz carrier) or frequency ($\sim$ 100 MHz carrier) modulation; while guiding devices such as optical fibers or coaxial waveguides are designed for the directed flow of power, energy and information.

We can fully characterize propagating EM waves by solving a wave equation, which can be derived from the original Maxwell equations. The fundamental wave equation of this dissertation is derived from (2.1) – (2.4) and the constitutive relations of (2.5) and (2.6). Since this dissertation is ultimately concerned with the modeling of electromagnetic communication devices, we impose the physical restriction that there are no

free charges in the problem domain; and therefore, require only the presence of current (or voltage) sources. The coupled wave equation is written as a three dimensional, vector valued PDE in time and space. This results in an initial boundary value problem (IBVP); as such, both boundary conditions and initial conditions must be specified to fully define the problem. This gives the following PDE

$$\begin{aligned}
\varepsilon\frac{\partial}{\partial t}\mathbf{E} &= \nabla\times(\mu^{-1}\mathbf{B}) - \sigma\mathbf{E} - \mathbf{J} && \text{in } \Omega \\
\frac{\partial}{\partial t}\mathbf{B} &= -\nabla\times\mathbf{E} && \text{in } \Omega \\
\nabla\cdot(\varepsilon\mathbf{E}) &= 0 && \text{in } \Omega \\
\nabla\cdot\mathbf{B} &= 0 && \text{in } \Omega \\
\hat{\mathbf{n}}\times\mathbf{E} &= \mathbf{E}_{bc} && \text{on } \partial\Omega \\
\mathbf{E}(t) &= \mathbf{E}_{ic} && \text{at } t = t_0 \\
\mathbf{B}(t) &= \mathbf{B}_{ic} && \text{at } t = t_0
\end{aligned} \tag{2.9}$$

where $\Omega$ is a three dimensional domain in which the wave exists, $\partial\Omega$ is the two dimensional boundary of the domain and $\hat{\mathbf{n}}$ is the outwardly directed unit normal of this boundary. The value $\mathbf{E}_{bc}$ represents an arbitrary boundary condition imposed on the electric field intensity and can range anywhere from the simple case of a perfect electric conductor (PEC) to more complicated voltage sources or absorbing boundary conditions (ABC). The values $\mathbf{E}_{ic}$ and $\mathbf{B}_{ic}$ represent the initial conditions of the problem, and $t_0$ is the initial time. The PDE of (2.9) is a coupled equation with two field unknowns: the electric field intensity $\mathbf{E}$ and the magnetic flux density $\mathbf{B}$. It involves only first order differential operators in time and space. Note that because we have restricted the constitutive relations to be independent of time, we are allowed to separate these values from the time derivative operation.

The wave equation of (2.9) contains all of the necessary components for modeling a vast majority of modern communication devices. The source for the propagating EM wave will be a time and possibly space dependent current source (or voltage source). The geometry of the problem $\Omega$ can be divided into several different materials, each with their own unique material property functions, separated by the appropriate boundary conditions. For example, to model a radiating EM device such as a radio antenna (Figure 2.3), the source would be modeled by a modulated oscillating current. The computational region $\Omega$ would be

divided into two different materials, the dielectric antenna and the air surrounding it. An absorbing boundary condition would be placed on the surface of $\Omega$ to account for the essential infinite boundary of the continuum problem. For guiding structures such as optical fibers (Figure 2.4), the wave will again be generated by some time dependent current source. In this case, the geometry of the problem $\Omega$ would be divided into at least two material regions, the core and the cladding; each with their own dielectric material property function. A PEC boundary condition could be applied to the outer surface of the wall for the case of a lossless fiber. A more realistic simulation would add a third material region, the "jacket" that is placed around real world optical fibers. An imperfect conducting boundary condition could be applied between the cladding and jacket and conductivity terms for each region could be added to account for a "lossy" fiber.

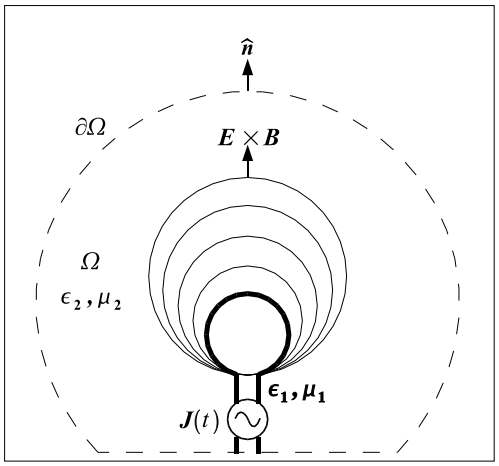


Figure 2.3: Example of a typical radiating wave problem.

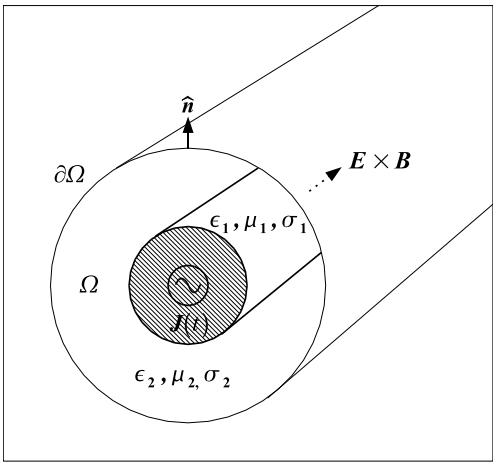


Figure 2.4: Example of a typical guided wave problem.

EM waves have a characteristic frequency (or wavelength) that is determined by the nature of the sources and the geometry of the problem. These frequencies span the electromagnetic spectrum, including the familiar regions of microwave, radio and optical frequencies [97]. Often times, the physical size of a problem will be determined by the desired propagating frequency of the EM wave. For example, single-mode optical fibers with the capability of transmitting infrared laser light (1,300 to 1,550 nanometer wavelengths) require small cores about 9 micrometers in diameter. Some optical fibers can be made from plastic with a large core about 1 millimeter in diameter, and can transmit visible red light (650 nanometer wavelength) from

light emitting diodes (LEDs). We can classify EM wave problems by the measure $\Omega/\lambda$, the ratio of the characteristic size of the geometry to the characteristic wavelength. Cases where this measure is significantly greater than unity ($\Omega/\lambda > 10^3$) are considered high frequency problems while cases where this measure is is very small ($\Omega/\lambda < 10^{-3}$) are considered quasi-static problems. Asymptotic approximation techniques can be exploited to solve either of these types of problems. However, for problems that lie in between these extremities, more advanced approximation techniques are required. Problems in which $\Omega/\lambda$ is large, but not large enough to be considered high frequency, are called "electrically large," since several wavelengths of the EM wave span the domain.

Conservation of charge in EM wave propagation is dictated by the so called continuity equation. For our particular concerns we have restricted the existence of free charges in the solution domain; the continuity equation therefore takes the form

$$\nabla \cdot \mathbf{J} = -\frac{\partial}{\partial t}\rho = 0 \tag{2.10}$$

If the current sources are divergence free, then the time dependent fields, $\varepsilon\mathbf{E}$ and $\mathbf{B}$, from (2.9) will also be divergence free for all time, provided that the the initial condition data is itself divergence free.

Electromganetic waves also conserve energy and this is often referred to as Poynting's theorem. It can be expressed by the following integral relation

$$\oint_{\partial\Omega} (\mu^{-1}\mathbf{E} \times \mathbf{B}) \cdot \hat{\mathbf{n}} = -\int_{\Omega} (\mathbf{J} \cdot \mathbf{E} + \sigma\mathbf{E} \cdot \mathbf{E} + \varepsilon\mathbf{E} \cdot \frac{\partial}{\partial t}\mathbf{E} + \mu^{-1}\mathbf{B} \cdot \frac{\partial}{\partial t}\mathbf{B}) \tag{2.11}$$

The left hand side represents the net power flow leaving the volume $\Omega$ through the surface $\partial\Omega$. The term under this surface integral is referred to as the Poynting vector and represents the power flux through a differential surface patch. The Poynting vector defines the direction of power flow and is typically alligned normal to the propagating wave fronts, defined as surfaces of constant phase. The right hand side of this equation represents the sources of this power loss contained in the volume $\Omega$. The first term represents the power supplied to the volume by the current sources. The second term represents the power absorbed by the medium, i.e. the rate of conversion of electromagnetic energy into thermal energy. The third and fourth terms represent the time rate of change of energy stored directly in the electric and magnetic fields respectively.

# Chapter 3

# Mathematical Preliminaries

## 3.1 Differential Forms for Electromagnetics

The calculus of differential forms provides a cohesive and intuitive framework for computational electromagnetics. Within this context, we can classify the various fields from Maxwell's equations in a more elucidating manner than standard vector calculus. As an introduction, Table 3.1 lists various physical quantities in electromagnetics (previously discussed in Chapter 2) and their associated differential form. Loosely speaking, a differential $l$-form is a quantity appearing under an $l$-dimensional integral.

| **Physical Quantity** | **Units** | **Vector/Scalar** | **Differential Form** |
|---|---|---|---|
| Scalar Potential | $V/m^0$ | $\phi$ | 0-form |
| Electric Field Intensity | $V/m^1$ | $\mathbf{E}$ | 1-form |
| Magnetic Field Intensity | $A/m^1$ | $\mathbf{H}$ | 1-form |
| Electric Flux Density | $C/m^2$ | $\mathbf{D}$ | 2-form |
| Magnetic Flux Density | $W/m^2$ | $\mathbf{B}$ | 2-form |
| Electric Charge Density | $C/m^3$ | $\rho$ | 3-form |

Table 3.1: Physical quantities and their associated differential forms.

The three main operators in the calculus of differential forms are the *exterior product*, the *exterior derivative*, and the *Hodge star*. In this section we review the key properties of these operators to aid in the

development of finite element solutions to Maxwell's equations. Therefore, we restrict the discussion of differential forms to the the three dimensional space $R^3$. Let $\{x^1, x^2, x^3\}$ denote the standard basis for $R^3$. Now consider the vector space

$$\Delta V = span(dx^1,\ dx^2,\ dx^3) \tag{3.1}$$

The $dx^i$ are called differentials and the space $\Delta V$ is called the cotangent space. For any point $r \in R^3$, a differential form of degree 1 (or 1-form) is a mapping from $R^3$ to $\Delta V$ given by

$$\mathbf{f}^1(r) = \beta_1(r)\ dx^1 + \beta_2(r)\ dx^2 + \beta_3(r)\ dx^3$$

where the $\beta_i(r)$ define the components of the form associated with each of the differentials. Because a differential form is an integrable quantity, the $\beta_i(r)$ are required to be square-integrable functions of the variable $r$ such that $\beta_i(r) \in L^2$ (see Table 3.2 for a formal definition of the space $L^2$). Higher degree cotangent spaces, denoted by $\Delta^l V$, are defined recursively by the *exterior product* (or wedge product). Specifically, the exterior product is a mapping of the form

$$\Delta^l V \wedge \Delta^m V \mapsto \Delta^{l+m} V \tag{3.2}$$

with the following properties

$$f \wedge g = -g \wedge f \tag{3.3}$$

$$(af + bg) \wedge h = a(f \wedge h) + b(g \wedge h) \tag{3.4}$$

where $a$ and $b$ are scalars and $f, g$ and $h$ are cotangent vectors. Since $f \wedge f = 0$, it follows that

$$\begin{aligned}
\Delta^1 V &= span(dx^1,\ dx^2,\ dx^3) \\
\Delta^2 V &= span(dx^2 \wedge dx^3,\ dx^3 \wedge dx^1,\ dx^1 \wedge dx^2) \\
\Delta^3 V &= span(dx^1 \wedge dx^2 \wedge dx^3)
\end{aligned}$$

For completeness we adopt the convention $\Delta^0 V = R^3$.

A differential form of degree $l$ (or $l$-form) is a mapping from $R^3$ to $\Delta^l V$. For $R^3$ there are 4 distinct

differential forms. We have

$$\text{0-form: } f^0(r) = \beta(r) \tag{3.5}$$

$$\text{1-form: } \mathbf{f}^1(r) = \beta_1(r)\,dx^1 + \beta_2(r)\,dx^2 + \beta_3(r)\,dx^3 \tag{3.6}$$

$$\text{2-form: } \mathbf{f}^2(r) = \beta_1(r)\,dx^2 \wedge dx^3 + \beta_2(r)\,dx^3 \wedge dx^1 + \beta_3(r)\,dx^1 \wedge dx^2 \tag{3.7}$$

$$\text{3-form: } f^3(r) = \beta(r)\,dx^1 \wedge dx^2 \wedge dx^3 \tag{3.8}$$

In each of the four cases, the $\beta_i(r)$ are required to be square-integrable functions. Thus for the case of $R^3$, we have two scalar valued forms, (3.5) and (3.8), as well as two vector valued forms, (3.6) and (3.7). Note that the 1-forms of (3.6) can be viewed as a vector function expanded over a *contra-variant* basis while the 2-forms of (3.7) can be viewed as a vector function expanded over a *co-variant basis*. The notion of a contra-variant basis and a co-variant basis will be further explained in Chapter 4 when we discuss coordinate system transformations.

The exterior product extends in a natural way to the space of differential forms of degree $l$, denoted as $\Psi^l$. For any $l,m \in \{0,1,2,3\}$ with $l+m \leq 3$, the exterior product of an $l$-form $f^l$ and an $m$-form $g^m$ is an $(l+m)$-form

$$f^l(r) \wedge g^m(r) = h^{l+m}(r)$$

For example,

$$(A\,dx^1 + B\,dx^2 + C\,dx^3) \wedge (P\,dx^1 + Q\,dx^2 + R\,dx^3) =$$

$$(BR - CQ)\,dx^2 \wedge dx^3 + (CP - AR)\,dx^3 \wedge dx^1 + (AQ - BP)\,dx^1 \wedge dx^2$$

and

$$(A\,dx^1 + B\,dx^2 + C\,dx^3) \wedge (P\,dx^2 \wedge dx^3 + Q\,dx^3 \wedge dx^1 + R\,dx^1 \wedge dx^2) =$$

$$(AP + BQ + CR)\,dx^1 \wedge dx^2 \wedge dx^3$$

Note that the above equations are reminiscent of the vector cross product and dot product, respectively.

The 1-forms, 2-forms, and 3-forms are sometimes referred to as *work form*, *flux form*, and *density form*, respectively. This is because an $l$-form, in simple terms, is the quantity that appears underneath an

$l$-dimensional integral. The line integral of a 1-form along a path measures the work done by a vector field in moving a particle along the path, the surface integral of a 2-form measures the net flux of a vector field through the surface, and the volume integral of a 3-form measures the net amount of material (i.e. mass, charge, etc ...) in the volume. For completeness, the 0-form is associated with a point (or delta function) integral, i.e. evaluation of the form at a point. Neglecting the conductivity term for now, Maxwell's equations from (2.1) – (2.4) can be expressed in integral form as

$$\begin{aligned}
\oint_{\partial\Omega_3} \mathbf{D} &= \int_{\Omega_3} \rho \\
\oint_{\partial\Omega_3} \mathbf{B} &= 0 \\
\oint_{\partial\Omega_2} \mathbf{H} &= \frac{\partial}{\partial t}\int_{\Omega_2} \mathbf{D} + \int_{\Omega_2} \mathbf{J} \\
\oint_{\partial\Omega_2} \mathbf{E} &= -\frac{\partial}{\partial t}\int_{\Omega_2} \mathbf{B}
\end{aligned}$$

where **E** and **H** are the 1-form electric and magnetic field intensities, **D** and **B** are the 2-form electric and magnetic flux densities, **J** is the 2-form electric current flux density, and $\rho$ is the 3-form electric charge density. In these equations $\Omega_3$ and $\Omega_2$ denote arbitrary volumes and surfaces, respectively, and $\partial\Omega_3$ and $\partial\Omega_2$ are their respective boundaries.

Given a 0-form $f^0$, the differential of $f^0$ is a 1-form given by

$$df^0 = \frac{\partial f^0}{\partial x^1}dx^1 + \frac{\partial f^0}{\partial x^2}dx^2 + \frac{\partial f^0}{\partial x^3}dx^3$$

with the requirement that each of the components $\frac{\partial f^0}{\partial x^i}$ are in the space $L^2$ (i.e. are square-integrable). This operation, referred to as the *exterior derivative*, is denoted simply by $d$, such that $d : \Psi^0 \to \Psi^1$. In general, the exterior derivative $d$ is a mapping $d : \Psi^l \to \Psi^{l+1}$, for $l = 0, 1, 2$, such that

$$d(f^l \wedge g^m) = df^l \wedge g^m + (-1)^{lm} f^l \wedge dg^m \tag{3.9}$$

$$d(df^l) = 0 \tag{3.10}$$

An explicit formula for the exterior derivative of a 1-form can be computed by re-writing a 1-form $\mathbf{f}^1$ as

$$\mathbf{f}^1 = A \wedge dx^1 + B \wedge dx^2 + C \wedge dx^3$$

where the components $A$, $B$, and $C$ are 0-forms. These three components along with the three independent variables yield a total of nine possible combinations. Applying the chain rule formula (3.9) yields

$$d(A\wedge dx^1+B\wedge dx^2+C\wedge dx^3)=$$
$$(\frac{\partial C}{\partial x^2}-\frac{\partial B}{\partial x^3})(dx^2\wedge dx^3)+(\frac{\partial A}{\partial x^3}-\frac{\partial C}{\partial x^1})(dx^3\wedge dx^1)+(\frac{\partial B}{\partial x^1}-\frac{\partial A}{\partial x^2})(dx^1\wedge dx^2)$$

The components of the resulting 2-form are required to be square-integrable such that

$$\begin{aligned}(\frac{\partial C}{\partial x^2}-\frac{\partial B}{\partial x^3}) &\in L^2\\ (\frac{\partial A}{\partial x^3}-\frac{\partial C}{\partial x^1}) &\in L^2\\ (\frac{\partial B}{\partial x^1}-\frac{\partial A}{\partial x^2}) &\in L^2\end{aligned}$$

Note that by application of the exterior derivative, the terms $\frac{\partial A}{\partial x^1}, \frac{\partial B}{\partial x^2}$ and $\frac{\partial C}{\partial x^3}$ are not required to be square integrable. Likewise, for the exterior derivative of a 2-form $\mathbf{f}^2$ we have

$$d\left(A\wedge(dx^2\wedge dx^3)+B\wedge(dx^3\wedge dx^1)+C\wedge(dx^1\wedge dx^2)\right)=(\frac{\partial A}{\partial x^1}+\frac{\partial B}{\partial x^2}+\frac{\partial C}{\partial x^3})(dx^1\wedge dx^2\wedge dx^3)$$

with the requirement that

$$(\frac{\partial A}{\partial x^1}+\frac{\partial B}{\partial x^2}+\frac{\partial C}{\partial x^3})\in L^2$$

Note that, conversely to the case of $1-$forms, the terms $\frac{\partial A}{\partial x^2}, \frac{\partial A}{\partial x^3}, \frac{\partial B}{\partial x^1}, \frac{\partial B}{\partial x^3}, \frac{\partial C}{\partial x^1}$ and $\frac{\partial C}{\partial x^2}$ are not required to be square integrable. Note also that the formula for the components of the exterior derivative of a 0-form, 1-form, and 2-form are the same as those for the standard vector calculus operations gradient, curl and divergence, respectively.

For the special case of $R^3$, each of the four spaces of differential forms $\Psi^l$ are isometric to the standard function (or Hilbert) spaces from vector calculus, namely: $H(Grad)$, $H(Curl)$, $H(Div)$ and $L^2$. These function spaces are summarized in Table 3.2. For example, the space $H(Curl)$ consists of all vector valued functions such that: 1) the components of the function are square-integrable and 2) the components of the curl of the function are also square-integrable. This is in fact the formal definition for the term "well-defined" that was used in deriving the continuity of vector fields in Chapter 2. With these insights, the

distinction between the 1-form field intensities and the 2-form flux densities and their continuity at a material interface are now clear. In addition, there are two types of scalar fields: 1) those which are fully continuous across material interfaces such as the electric potential and 2) those which are fully discontinuous across material interfaces such as electric charge densities.

| **Form Space** | **Isometric Function Space** | **Formal Definition** | **Interface Continuity** |
|---|---|---|---|
| $\Psi^0$ | $H(Grad)$ | $\{u : u \in L^2;\ \nabla u \in \left(L^2\right)^3\}$ | Total |
| $\Psi^1$ | $H(Curl)$ | $\{\mathbf{u} : \mathbf{u} \in \left(L^2\right)^3;\ \nabla \times \mathbf{u} \in \left(L^2\right)^3\}$ | Tangential |
| $\Psi^2$ | $H(Div)$ | $\{\mathbf{u} : \mathbf{u} \in \left(L^2\right)^3;\ \nabla \cdot \mathbf{u} \in L^2\}$ | Normal |
| $\Psi^3$ | $L^2$ | $\{u;\ \int_{-\infty}^{\infty} u^* u\ < \infty\}$ | None |

Table 3.2: Isometric function spaces for each of the differential forms

An *exact sequence* is a sequence of maps between a sequence of spaces of the form [98]

$$\alpha_i : A_i \longrightarrow A_{i+1}$$

which satisfies

$$im(\alpha_i) = ker(\alpha_{i+1})$$

where *im* denotes the image (or range space) and *ker* the group kernel (or null space). The isometric function spaces (and by correspondence, the spaces of differential forms) are linked to each other via the exterior derivative and form the following exact sequence

$$H(Grad) \xrightarrow{d} H(Curl) \xrightarrow{d} H(Div) \xrightarrow{d} L^2 \xrightarrow{d} 0 \tag{3.11}$$

In this case, the exterior derivative is a surjective mapping (i.e. it is onto, but not one-to-one). This sequence has several far reaching consequences for computational electromagnetics. For example, the sequence above implies that the range space of the curl operator is the function space $H(Div)$ while the null space is the set of all functions which can be written as the gradient of scalar functions, i.e. the set of all *irrotational* functions. Likewise, the range space of the divergence operator is the function space $L^2$ while the null space is the set of all functions which can be written as the curl of vector functions, i.e. the set of all *solenoidal functions*. These are direct consequences of the properties of the exterior derivative given in (3.9) and (3.10).

The generalized Stokes' theorem for differential forms is given by

$$\int_{\Omega} d f^l = \oint_{\partial\Omega} f^l \tag{3.12}$$

where $f^l$ is an $l$-form, $l = 0, 1$ or 2, and $\Omega$ is an $l+1$ dimensional manifold, and $\partial\Omega$ is the manifold boundary. This compact expression unifies several key integration theorems of vector calculus, namely

| | | |
|---|---|---|
| $l = 0$ | Fundamental Theorem of Calculus | $\int_a^b du = u(b) - u(a)$ |
| $l = 1$ | Stokes Theorem | $\int_s (\nabla \times \mathbf{u}) \cdot d\mathbf{s} = \oint_l \mathbf{u} \cdot d\mathbf{l}$ |
| $l = 2$ | Gauss Divergence Theorem | $\int_v (\nabla \cdot \mathbf{u})\, dv = \oint_s \mathbf{u} \cdot d\mathbf{s}$ |

Using the generalized Stokes' theorem of (3.12), the integral form of Maxwell's equations can be written as

$$\begin{aligned}
\oint_{\Omega_3} d\mathbf{D} &= \int_{\Omega_3} \rho \\
\oint_{\Omega_3} d\mathbf{B} &= 0 \\
\oint_{\Omega_2} d\mathbf{H} &= \frac{\partial}{\partial t} \int_{\Omega_2} \mathbf{D} + \int_{\Omega_2} \mathbf{J} \\
\oint_{\Omega_2} d\mathbf{E} &= -\frac{\partial}{\partial t} \int_{\Omega_2} \mathbf{B}
\end{aligned}$$

As these equations are valid for any region of integration, the integrals can be removed to yield Maxwell's equations in differential form

$$d\mathbf{D} = \rho \tag{3.13}$$

$$d\mathbf{B} = 0 \tag{3.14}$$

$$d\mathbf{H} = \frac{\partial}{\partial t}\mathbf{D} + \mathbf{J} \tag{3.15}$$

$$d\mathbf{E} = -\frac{\partial}{\partial t}\mathbf{B} \tag{3.16}$$

It is important to point out that the time derivative does not effect the degree of a form. In Figure 3.1 we show the time-dependent Maxwell equations in schematic form, where $\phi$ and $\mathbf{A}$ denote the scalar and vector potentials respectively and converging arrows denote summation. The left portion of Figure 3.1 encompasses Faraday's law (3.16), Gauss's law for the magnetic field (3.14), and the fact that the electric field $\mathbf{E}$ can be written in terms of potentials as $\mathbf{E} = d\phi - \frac{\partial}{\partial t}\mathbf{A}$. The right portion of Figure 3.1 encompasses

Ampere's law (3.15), Gauss' law for the electric field (3.13), and the continuity equation $d\mathbf{J} - \frac{\partial}{\partial t}\rho = 0$. The two portions of the diagram are connected via the constitutive relations of (3.17) and (3.18).

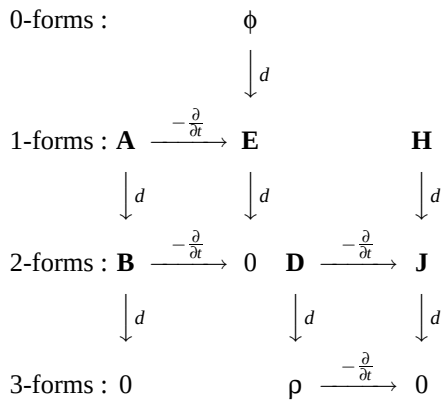


Figure 3.1: Schematic representation of Maxwell's equations in differential form.

In order to discuss material interfaces and electromagnetic energy, the *Hodge star* operator is required. For the particular case of three dimensional space, the Hodge star operator is a non-singular (i.e. an invertible) linear function that maps $l$-forms to $(3-l)$-forms and is denoted by $\star : \Psi^l \rightarrow \Psi^{3-l}$. Because the Hodge function is invertible, it satisfies the following for 0-forms and 3-forms

$$\star 1 = dx^1 \wedge dx^2 \wedge dx^3$$

1-forms and 2-forms are related by

$$\star dx^1 = dx^2 \wedge dx^3, \;\; \star dx^2 = dx^3 \wedge dx^1, \;\; \star dx^3 = dx^1 \wedge dx^2$$

In electromagnetics the Hodge star operator is associated with the constitutive relations

$$\mathbf{D} = \star_\varepsilon \mathbf{E} \tag{3.17}$$

$$\mathbf{B} = \star_\mu \mathbf{H} \tag{3.18}$$

where the electric and magnetic material properties are now explicitly defined in terms of a specific Hodge star function which converts the 1-form field intensities to 2-form flux densities. For the special case of linear isotropic homogeneous (LIH) materials, the function $\star_\varepsilon$ is the permittivity of free space constant with units

of $farad/m$ while the the function $\star_\mu$ is the permeability of free space constant with units of $henry/m$. In general, these specific Hodge operators are tensor functions (e.g. the tensor valued dielectric constant for anisotropic media).

Using the four spaces of differential forms and the Hodge star operator, we can construct a more revealing sequence known as the DeRham diagram

$$\begin{array}{ccccccccccc} & & \Psi^0 & \xrightarrow{d} & \Psi^1 & \xrightarrow{d} & \Psi^2 & \xrightarrow{d} & \Psi^3 & \xrightarrow{d} & 0 \\ & & \downarrow\star & \hookleftarrow & \downarrow\star & \hookleftarrow & \downarrow\star & \hookleftarrow & \downarrow\star & & \\ 0 & \xleftarrow{d} & \Psi^3 & \xleftarrow{d} & \Psi^2 & \xleftarrow{d} & \Psi^1 & \xleftarrow{d} & \Psi^0 & & \end{array} \tag{3.19}$$

The top sequence of (3.19) is the exact sequence of (3.11) written using the four spaces of differential forms. On the bottom is the same sequence written in reverse order. The two sequences are linked via the Hodge star operator. The $\hookleftarrow$ symbols represent a three-step sequence which defines the standard second order spatial differential operators of vector calculus that we are familiar with. For example, the first $\hookleftarrow$ symbol defines the grad-div operator commonly found in scalar wave equations (or Poisson's equation), which is a mapping from scalar valued 0-forms to scalar valued 3-forms such that: $\Psi^0 \xrightarrow{d} \Psi^1 \xrightarrow{\star} \Psi^2 \xrightarrow{d} \Psi^3$. The second $\hookleftarrow$ symbol defines the curl-curl operator found in the vector Helmholtz equation while the third symbol defines the div-grad operator commonly found in acoustic wave equations.

## 3.2 Variational Formulation

We are now ready to derive a variational formulation of PDE (2.9). For further reference on the fundamental theorem of variational calculus and variational formulations, see Appendix A. Consider the coupled Ampere and Faraday equations of (2.9) re-written in the language of differential forms

$$\star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} = d(\star_\mu \mathbf{B}) - \star_\sigma \mathbf{E} - \mathbf{J} \tag{3.20}$$

$$\star_\mu \frac{\partial}{\partial t}\mathbf{B} = -\star_\mu d\mathbf{E} \tag{3.21}$$

where each of the material property functions are now represented by a specific Hodge function (e.g. $\star_\mu = \mu^{-1}$). Note that every term in (3.20) is a 2-form (the time derivative does not affect the degree of a differential

form) while every term in (3.21) is a 1-form (this is accomplished by introducing the Hodge function $\star_\mu$ to both sides of the equation). We can now construct a variational formulation of this equation. We begin by computing the wedge product of each term in (3.20) with a 1-form test function $\mathbf{E}'$ and each term in (3.21) with a 2-form test function $\mathbf{B}'$ to yield 3-forms. The resulting 3-form equations can then be integrated over a volume to yield

$$\begin{aligned}\int_\Omega \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E}\wedge\mathbf{E}' &= \int_\Omega d(\star_\mu \mathbf{B})\wedge\mathbf{E}' - \int_\Omega \star_\sigma \mathbf{E}\wedge\mathbf{E}' - \int_\Omega \mathbf{J}\wedge\mathbf{E}' \\ \int_\Omega \star_\mu \frac{\partial}{\partial t}\mathbf{B}\wedge\mathbf{B}' &= -\int_\Omega \star_\mu d\mathbf{E}\wedge\mathbf{B}'\end{aligned}$$

Using the chain rule formula (3.9) to perform integration by parts and the generalized Stokes Theorem (3.12) yields the following linear functionals

$$\int_\Omega \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E}\wedge\mathbf{E}' = \int_\Omega \star_\mu \mathbf{B}\wedge d\mathbf{E}' - \int_\Omega \star_\sigma \mathbf{E}\wedge\mathbf{E}' - \int_\Omega \mathbf{J}\wedge\mathbf{E}' - \oint_{\partial\Omega} \star_\mu \mathbf{B}\wedge\mathbf{E}' \tag{3.22}$$

$$\int_\Omega \star_\mu \frac{\partial}{\partial t}\mathbf{B}\wedge\mathbf{B}' = -\int_\Omega \star_\mu d\mathbf{E}\wedge\mathbf{B}' \tag{3.23}$$

The linear functionals of (3.22) and (3.23) are one possible starting point for a finite element solution of Maxwell's equations. In Chapter 4 we will present explicit bilinear forms and basis functions which can be used to yield a discrete version (i.e. a system of linear equations) of the above variational forms. This process makes use of the Galerkin procedure, in which the field unknowns, $\mathbf{E}$ and $\mathbf{B}$, and the test functions, $\mathbf{E}'$ and $\mathbf{B}'$, are expanded over the same discrete finite element space. For further reference on the Galerkin procedure, see Appendix A.

Note that by using (3.9) and (3.12) it has been assumed that the field variables have a certain amount of smoothness, namely that

$$\mathbf{E} \in \{\Psi^1 : \int_\Omega \star\mathbf{E}\wedge\mathbf{E} + \int_\Omega \star d\mathbf{E}\wedge d\mathbf{E} < \infty\} \tag{3.24}$$

$$\mathbf{B} \in \{\Psi^2 : \int_\Omega \star\mathbf{B}\wedge\mathbf{B} + \int_\Omega \star d\mathbf{B}\wedge d\mathbf{B} < \infty\} \tag{3.25}$$

The constraints of (3.24) and (3.25) have a very relevant physical interpretation for the case of Maxwell's equations. The first term in the constraints above states that the electric and magnetic field energies must remain finite, while the second term in the constraints is a direct consequence of Ampere's law (3.15) and

Faraday's law (3.16), implying that the time varying 2-form electric and magnetic flux densities must be finite as well. There are further implications from the constraints of (3.24) and(3.25). Consider the situation of a material interface with a surface separating the two materials with differing dielectric constants. In the absence of a surface charge density, Gauss' law (3.13) implies that the normal component of **D** is continuous, therefore the normal component of **E** is discontinuous. Conversely, Faraday's Law (3.16) implies that the tangential component of the electric field **E** is continuous, therefore the tangential component of **D** is discontinuous.

## 3.3 Discrete Differential Forms

Having established the properties of differential forms in the context of electromagnetics, we are now ready to discuss the notion of a *discrete* differential form to use in constructing finite element solutions to Maxwell's equations. Given some differential field $f^l \in \Psi^l$, we define a discrete differential form to be a basis function expansion of $f^l$ using a finite subset of the space $\Psi^l$. Specifically, we denote this finite subspace as $\mathcal{P}^l$ such that $\mathcal{P}^l \subset \Psi^l$. The basis function expansion is then of the form

$$f^l \approx \Pi(f^l) \equiv \sum_{i}^{Dim(\mathcal{P}^l)} \mathcal{A}_i(f^l)\, W_i \tag{3.26}$$

where $W_i$ denote the discrete differential form basis functions such that $W_i \in \mathcal{P}^l$, these will be discussed in great detail in Chapter 4. For the basis function expansion of (4.1) we have introduced the projection operator denoted as $\Pi$, and the degrees of freedom denoted as $\mathcal{A}$ . The degrees of freedom $\mathcal{A}$ are a set of linear functionals that map a function onto the set of real valued scalars, i.e. $\mathcal{A}_i : f^l \mapsto \Re$. These scalar values represent the components of the differential field $f^l$ along each of the basis functions $W_i$. The operation $\Pi(f^l)$ represents the projection of the function $f^l$ onto the finite space $\mathcal{P}^l$. The precise form of the linear functionals comprising the set $\mathcal{A}$ will be discussed in great detail in Chapter 4.

The discrete differential forms as defined by (4.1) are required to reproduce the exact sequence

property of (3.11) on a discrete level. This implies that

$$\begin{array}{ccccccccc} \Psi^0 & \xrightarrow{d} & \Psi^1 & \xrightarrow{d} & \Psi^2 & \xrightarrow{d} & \Psi^3 & \xrightarrow{d} & 0 \\ \downarrow \Pi & & \downarrow \Pi & & \downarrow \Pi & & \downarrow \Pi & & \\ \mathcal{P}^0 & \xrightarrow{d} & \mathcal{P}^1 & \xrightarrow{d} & \mathcal{P}^2 & \xrightarrow{d} & \mathcal{P}^3 & \xrightarrow{d} & 0 \end{array} \tag{3.27}$$

In other words, the basis functions need to span a polynomial space which is a proper subset of its continuum counterpart. In addition, it is also very beneficial (though not required) for the discrete differential form basis functions (and subsequent degrees of freedom and bilinear forms) to scale in the same way as their continuum counterparts. In other words, we would like our discrete differential $l$-form basis functions to have units of $m^{-l}$, where $m$ is an arbitrary metric of distance.

Generally speaking, a grid-based numerical solution of a PDE involves the replacement of continuum operators (such as the exterior derivative and the time derivative) with finite dimensional matrices and the continuum fields (such as the electric field intensity) with finite dimensional vectors consisting of approximate versions of the field evaluated at discrete points in space and time. As such, the properties of the discrete matrices and vectors should faithfully reproduce the corresponding properties of their continuum counterparts. This is the notion of a mimetic discretization method. By requiring that (3.27) hold true for all of the discrete differential forms, we are guaranteed that our resulting approximation method will be mimetic in space. For example, failure to reproduce the sequence of (3.27) on a discrete level can lead to so called "spurious modes" in any eigenvalue computations and will yield methods which do *not* conserve charge. Spurious modes and violation of numerical charge conservation are in fact the direct result of discrete operators (i.e. matrices) which have incorrect range and null spaces.

## 3.4 Polynomials of a Single Variable

In this dissertation, we use polynomial basis functions to discretize the space of differential forms. Here we introduce some notation and give concrete examples of different polynomials of a single variable. These polynomials will become the building blocks for the discrete differential form basis functions we will

eventually construct in Chapter 4. Throughout this section, all polynomials will be presented over the one dimensional reference segment of $[0,1]$; however, it is simple enough to transform all results to an arbitrary segment $[a,b]$ by means of a linear transformation of variables.

The Lagrange interpolatory polynomial of degree $p$ is defined by a distinct set of $p+1$ real valued interpolation points denoted by the symbol $X$, such that $X = \{X_0, X_1, \ldots, X_p\}$. The polynomial is constructed in such a way that it has a value of unity at interpolation point $i$ and a value of zero at every other interpolation point. The precise definition for the Lagrange interpolatory polynomial of degree $p$ is given by

$$L_i^p(x;X) = \prod_{\substack{j=0 \\ j\neq i}}^{p} \frac{(x - X_j)}{(X_i - X_j)} \tag{3.28}$$

The set of $p+1$ interpolation points, $X$, can at this point be arbitrary; however, it is important to note that properties of any finite element basis constructed from interpolatory polynomials can be directly affected by the choice of this set. In section Chapter 4 we will show that the choice of interpolation points directly affects the conditioning of element mass matrices [99]. Traditionally, the set $X$ consists of $p+1$ interpolation points uniformly distributed over the unit interval. This set is simple to generate but turns out to yield the worst case scenario of exponential growth of condition number as a function of $p$. The best case scenario is logarithmic growth and can be achieved by distributing the $p+1$ interpolation points in a *non-uniform* fashion over the unit interval. The growth of condition numbers is related to an object from approximation theory known as the Lebesgue constant [100], [101], [102], which is defined as

$$\Lambda(X) = \max_{x\in[0,1]} \sum_{i=0}^{p} |L_i^p(x,X)| \tag{3.29}$$

The Lebesgue constant depends only on the choice of interpolation points $X$ and will grow as the value of $p$ is increased. For a given value of $p$, there exists an optimal set of interpolation points which will yield a minimum Lebesgue constant; however there is no analytic formula for generating these points in an efficient manner. As such, we follow the results of [100] and note that there is a *near* optimal set of points that can be computed quite easily. This set of points, referred to as the Extended Chebyshev set, is generated by computing the zeros of the Chebyshev Polynomial of the first kind, then applying a linear transformation to map the results over the domain $[0,1]$. Let $\bar{X}^p$ denote the Extended Chebyshev set of $p+1$ interpolation

points over the domain $[0,1]$ defined as

$$\bar{X}^p = \{\frac{-\cos[(2i+1)\pi/(2p+2)]}{2\cos[\pi/(2p+2)]} + \frac{1}{2};\ i = 0,1,\ldots,p\} \tag{3.30}$$

To make the notation more compact we will occasionally omit the interpolation points $X$ from the Lagrange interpolatory polynomials. In this case, the set will be implied to be arbitrary or it will be defined in advance.

For hierarchical basis functions, we require orthogonal polynomials. We use a variation of the Legendre polynomials defined and normalized over the reference segment $[0,1]$, which we will denote as $\bar{l}^p(x)$, where $p$ is the degree of the polynomial. They are written as

$$\bar{l}^p(x) = \sqrt{2p+1}\, l^p(2x-1) \tag{3.31}$$

where $l^p(x)$ is the standard definition of the Legendre polynomial of degree $p$ defined over the segment $[-1,1]$ with respect to the weighting function $w(x) = 1$ found in most references. Figure 3.2 and Figure 3.3 give some visual examples of a set of interpolatory polynomials (using the Extended Chebyshev interpolation points) and a set of orthonormal polynomials, each over the reference segment $[0,1]$.

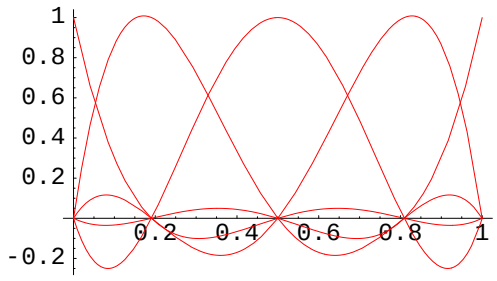


Figure 3.2: Interpolatory polynomials $L_i^3(x;X)$, for $i = 0,1,2,3$.

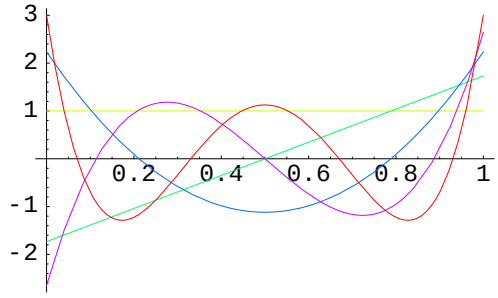


Figure 3.3: Orthonormal polynomials $\bar{l}^i(x)$, for $i = 0,1,2,3$.

# Chapter 4

# High Order Spatial Discretization

## 4.1 Spatially Discretized PDE

Let $\Sigma_h$ be a piecewise discretization of the physical domain $\Omega$ of (2.9) using a mesh of hexahedral elements of characteristic volume $\Delta h$. The field variables **E** and **B** are then approximated over each element $\Sigma \in \Sigma_h$ by basis function expansions of the form

$$\begin{aligned} \mathbf{E}(\mathbf{r},t) &\approx \sum_i e_i(t)\,\mathbf{w}_i(\mathbf{r}), \quad \mathbf{w}_i \in W_h \subset \Psi^1 && (4.1)\\ \mathbf{B}(\mathbf{r},t) &\approx \sum_i b_i(t)\,\mathbf{f}_i(\mathbf{r}), \quad \mathbf{f}_i \in F_h \subset \Psi^2 && (4.2)\\ \text{for} &\qquad \mathbf{r} \in \Sigma, \quad t_0 \le t \le t_{fin} \end{aligned}$$

where $e_i(t)$ are the time dependent 1-form degrees of freedom, $b_i(t)$ are the time dependent 2-form degrees of freedom, $\mathbf{w}_i(\mathbf{r})$ are the spatially dependent 1-form polynomial basis functions and $\mathbf{f}_i(\mathbf{r})$ are the spatially dependent 2-form polynomial basis functions. Applying Galerkin's method to the variational formulations of (3.22) and (3.23) yields the following linear system of first order ordinary differential equations (ODEs)

$$\begin{aligned} M_\varepsilon \frac{\partial}{\partial t} e &= K^T M_\mu\, b - M_\sigma\, e - M_\varepsilon\, j && (4.3)\\ \frac{\partial}{\partial t} b &= -K\, e \end{aligned}$$

where $e$ and $b$ represent the discrete differential 1-form and 2-form electric and magnetic fields respectively, $K$ is a rectangular matrix representing the discrete curl operator, $M_\varepsilon$ is a symmetric positive definite (SPD) 1-form mass matrix computed using the material property function $\varepsilon$ to represent the dielectric properties, $M_\sigma$ is the SPD 1-form mass matrix computed using the material property function $\sigma$ to represent the electric conductivity, $M_\mu$ is the SPD 2-form mass matrix computed using the material property function $\mu$ to represent the magnetic permeability and $j$ is the discrete 2-form time dependent current source.

In this chapter we present all of the tools needed to construct the linear system of ODEs from (4.3). These tools include

- Interpolatory and hierarchical basis functions
- Local to global element mappings
- Local to global basis function mappings
- Explicit degrees of freedom for the projection operation
- Standard bilinear forms for local mass and stiffness matrices
- Mixed bilinear forms for coupled equations
- Surface bilinear forms for absorbing boundary conditions
- Global assembly routine for local matrices

We will discuss in great detail the mathematics and implementation of these tools. Together, these tools comprise a complete finite element method which can be used to yield linear systems representing discrete versions of the differential operators and fields from Maxwell's equations while accounting for various types of boundary conditions and source types. As mentioned in Chapter 1, we follow the work of Ciarlet [93] and define a finite element as a set of three distinct objects $(\Sigma, \mathcal{P}, \mathcal{A})$ such that:

- $\Sigma$ is the polyhedral domain over which the element is defined
- $\mathcal{P}$ is a finite dimensional polynomial space from which basis functions are constructed
- $\mathcal{A}$ is a set of linear functionals (*Degrees of Freedom*) dual to $\mathcal{P}$

Finite element basis functions are *not* uniquely specified until all three components of $(\Sigma, \mathcal{P}, \mathcal{A})$ are defined. There are several criterion for choosing one particular basis over another. These include, but are not limited to, ease of implementation, conditioning of mass and stiffness matrices, orthogonality of basis functions, and the ability to throw away high order terms in a basis to allow conformity with lower order elements. These points will be addressed as the individual bases are presented.

## 4.2 $\Sigma$ - Element Topology and Geometry

The first step in our $(\Sigma, \mathcal{P}, \mathcal{A})$ construction process is the element $\Sigma$ itself. At this point it is very important to make the distinction between the element $\Sigma$ and a basis function (which is a member of $\mathcal{P}$). These two objects are mutually distinct and as such can be treated independently of each other. In our framework, the element $\Sigma$ handles all of the information regarding the topology and geometry (i.e. local to global mappings) of a finite element procedure. This separation provides a very computationally efficient approach for constructing bases on unstructured grids. Unlike the approach presented in [78], [79] and [80]; the bases presented here are first defined and constructed on a standard reference element and later transformed as necessary to physical mesh elements via a set of well defined transformation rules based on the properties of differential forms. There are several reasons to do this. The first is the implementation of bilinear forms (discussed in detail in Section 4.6). All relevant matrices in a finite element computation (e.g. mass and stiffness matrices) require the use of a bilinear form which involves integration over the element $\Sigma$ . In general, elements from an unstructured mesh will have non-trivial geometries, and as such, integration over the actual elements can be cumbersome and computationally expensive. However, integration over a standard reference element can be done quite easily and since the bases are polynomial in nature, integration can quite often be done *exactly* using a quadrature rule of the appropriate order. In addition, given the appropriate transformation rules the bases need only be evaluated on the reference element then transformed accordingly. This gives rise to a very computationally efficient algorithm for computing finite element approximations. For a given element topology and basis order,the basis functions only need to be computed *once*. Then, for every element

of the same topology in the mesh, the results from the reference element can simply be mapped according to the transformation rules. This can significantly reduce computational time and storage requirements for a typical finite element computation.

We begin this section with the definition for the standard reference hexahedral domain which will be used throughout this chapter. We then give an introduction to generalized curvilinear coordinate systems which will introduce the local to global mapping from reference coordinate systems to those of physical mesh elements as well as provide understanding for the later development of the transformation rules for vector functions presented in Section 4.4.

### 4.2.1 Reference Element

All hexahedral elements (including curved elements) in a physical mesh are topologically equivalent to a reference hexahedral element. In order to make integration over the reference element as simple as possible, we adopt a standard Cartesian coordinate system with an origin at the point $(0,0,0)$ as our reference coordinate system. Throughout the remainder of this chapter, all objects explicitly defined with respect to this reference coordinate system will be accented with a *hat* symbol. Let $\hat{\Omega}$ denote the unit hexahedron such that

$$\hat{\Sigma} = \{(\hat{r}_1, \hat{r}_2, \hat{r}_3);\ 0 \leq (\hat{r}_1, \hat{r}_2, \hat{r}_3) \leq 1\} \tag{4.4}$$

The topology (or connectivity) of a hexahedron is determined by 8 vertices, which we will label with the generic IDs $\{a,b,c,d,e,f,g\}$. The connectivity standard for the reference hexahedron that we have adopted is depicted in Figure 4.1. Given this definition, we can also define local edge and face connectivity standards; these are listed in Table 4.1 and Table 4.2. The local face orientations are defined by applying the right-hand rule to the list of vertices. Now that we have defined the reference coordinate system and its connectivity, we need a procedure for transforming points defined in this system to those defined in a more general system, like the kind used to define a physical mesh element. We refer to this procedure as the *local to global mapping*, where local is used to denote the point in the reference Cartesian system and global is used to denote the same point with respect to the coordinate system of a physical mesh element.

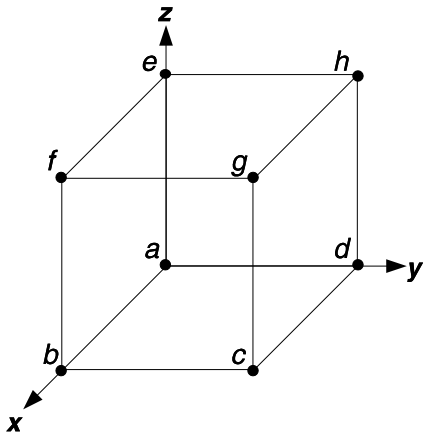


Figure 4.1: Topology standard for the reference element.

### 4.2.2 Generalized Curvilinear Coordinate Systems

Now let $\hat{\mathbf{r}} = (\hat{r}_1, \hat{r}_2, \hat{r}_3)$ denote the coordinates of an arbitrary point in the reference Cartesian system, and let $\mathbf{r} = (r_1, r_2, r_3)$ denote the coordinates of the same point with respect to a new coordinate system. The two systems are related through the local to global mapping

$$\mathbf{r} = \Phi(\hat{\mathbf{r}}) \tag{4.5}$$

The order of this mapping need not be the same as the order of the vector basis. We assume the mapping to be single valued and to have continuous derivatives so the correspondence between $\mathbf{r}$ and $\hat{\mathbf{r}}$ is unique (this implies that the element is non-twisted). This mapping, sometimes referred to as the iso-parametric mapping, is usually constructed by means of a basis function expansion using what are commonly referred to as shape functions. The shape functions are used to interpolate inside the domain of a global mesh element given the global coordinates of its vertices. In the language of differential forms, these shape functions are in fact 0-form scalar functions belonging to the $H(Grad)$ piecewise continuous function space.

For hexahedral elements, these shape functions can be constructed in a very efficient manner using

| Edge ID | Vertices | Local Orientation |
|---|---|---|
| 1 | $\{a,e\}$ | $+z$ |
| 2 | $\{d,h\}$ | $+z$ |
| 3 | $\{b,f\}$ | $+z$ |
| 4 | $\{c,g\}$ | $+z$ |
| 5 | $\{a,d\}$ | $+y$ |
| 6 | $\{e,h\}$ | $+y$ |
| 7 | $\{b,c\}$ | $+y$ |
| 8 | $\{f,g\}$ | $+y$ |
| 9 | $\{a,b\}$ | $+x$ |
| 10 | $\{d,c\}$ | $+x$ |
| 11 | $\{e,f\}$ | $+x$ |
| 12 | $\{h,g\}$ | $+x$ |

Table 4.1: Local edge connectivity for the reference element.

| Face ID | Vertices | Local Orientation |
|---|---|---|
| 1 | $\{a,b,c,d\}$ | $+z$ |
| 2 | $\{e,f,g,h\}$ | $+z$ |
| 3 | $\{a,b,f,e\}$ | $-y$ |
| 4 | $\{d,c,g,h\}$ | $-y$ |
| 5 | $\{a,d,h,e\}$ | $+x$ |
| 6 | $\{b,c,g,f\}$ | $+x$ |

Table 4.2: Local face connectivity for the reference element.

a tensor product of the Lagrange interpolatory polynomials. A mapping of order $s$ can be constructed as

$$\Phi(\hat{\mathbf{r}}) = \sum_{i=1}^{(s+1)^3} n_i N_i(\hat{\mathbf{r}}) \tag{4.6}$$

$$\{N(\hat{\mathbf{r}})\} = \{L_i^s(\hat{r}_1)L_j^s(\hat{r}_2)L_k^s(\hat{r}_3); \quad i,j,k = 0,\ldots,s\} \tag{4.7}$$

where $n_i$ are the global coordinates of the $(s+1)^3$ vertices used to define the global hexahedron and the set of interpolation points $X$ (omitted for clarity) are uniformly distributed. The geometry order $s$ is independent of the basis order and it determines the degree of distortion of the global mesh element. For example a mapping of order $s = 1$ implies a coordinate transformation to a linearly distorted element defined by the global coordinates of 8 vertices, while a mapping of order $s = 2$ implies a transformation to a quadratically distorted element defined by the global coordinates of 27 vertices.

Now recall from vector calculus that for a general three dimensional curvilinear coordinate sys-

tem defined by three independent variables, $(r_1, r_2, r_3)$, we have in general three distinct coordinate surfaces defined by the relations $r_1 = c_1, r_2 = c_2, r_3 = c_3$, where $c_1, c_2, c_3$ are constants [103]. These three surfaces intersect at curves known as coordinate curves. If the coordinate surfaces intersect at right angles, the curvilinear coordinate system is called orthogonal. This is the case with the standard Cartesian coordinate system which we have chosen to define our reference elements in. However, in a general coordinate system (such as the system used to define a global mesh element), this will no longer be necessarily true. An important consequence of non-orthogonal curvilinear coordinate systems lies in the definition of basis vectors. As we know, any vector in a three dimensional coordinate system can be represented by a linear combination of three basis vectors (not necessarily normalized). However, in general, we can define two distinct sets of basis vectors for a given curvilinear coordinate system; basis vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ defined to be *tangent* to coordinate curves and basis vectors $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3$ defined to be *normal* to coordinate surfaces. These two sets are identical if and only if the curvilinear coordinate system is orthogonal (as is the case with the Cartesian system).

In order to maintain coordinate independence, all properties of the basis vectors defined in one coordinate system must be preserved under a transformation to a new coordinate system. This implies that basis vectors $\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2, \hat{\mathbf{v}}_3$ defined to be tangent to the coordinate curves of the reference coordinate system must remain tangent to the coordinate curves of the new coordinate system. Also, basis vectors $\hat{\mathbf{V}}_1, \hat{\mathbf{V}}_2, \hat{\mathbf{V}}_3$ defined to be normal to the coordinate surfaces of the reference coordinate system must remain normal to the coordinate surfaces of the new coordinate system. We define the Jacobian of the mapping (4.5) as

$$J_{i,j} = \frac{\partial r_j}{\partial \hat{r}_i} \tag{4.8}$$

Using this definition, basis vectors defined in the reference coordinate system will transform to another coordinate system under the mapping (4.5) by the following rules

$$\mathbf{v}_i = J^T \hat{\mathbf{v}}_i \tag{4.9}$$

$$\mathbf{V}_i = |J| J^{-1} \hat{\mathbf{V}}_i \tag{4.10}$$

These are the so called *contravariant* and *covariant* transformations respectively. Note that the definition presented here for a covariant vector transformation is scaled by the determinant of the Jacobian matrix; the

reason for this will be explained in Section 4.4. Figures 4.2 – 4.4 provide visual representations of these two transformations by plotting the three basis vectors at 9 distinct points in their respective domains. Note how the contravariantly transformed basis vectors remain *tangent* to the coordinate curves of the new curvilinear coordinate system while the covariantly transformed basis vectors remain *normal* to the coordinate surfaces. Also note how the basis vectors are no longer constant over the transformed coordinate systems; this is because the mapping used to define the transformation is not constant.

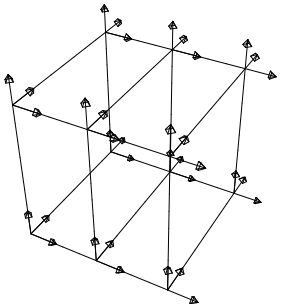

Figure 4.2: Basis vectors on the reference element.

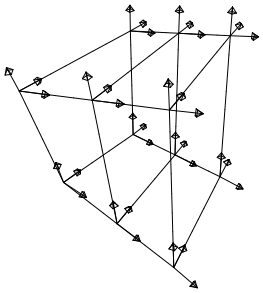

Figure 4.3: Contravariantly transformed basis vectors on a distorted element.

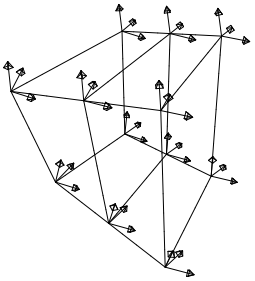

Figure 4.4: Covariantly transformed basis vectors on a distorted element.

## 4.3 $\mathcal{P}$ - Polynomial Spaces

In this section we define the polynomial spaces, $\mathcal{P}$ , which will be used to construct 1-form and 2-form vector bases on hexahedrons. These spaces were originally proposed and are well documented in [58]. The polynomial space can be thought of as a template for basis function construction. By itself, it does not provide a concrete method for formulating a set of basis functions; but rather it provides constraints on how these basis functions can be made. The polynomial space is defined to properly reproduce the features of a particular $l$-form field in a discrete sense. For example, the polynomial space for a discrete 1-form must be a proper subspace of $H(Curl)$ ; meaning that it must possess a well defined curl. Similarly, the polynomial space for a discrete 2-form must be a proper subspace of $H(Div)$ ; meaning that it must possess a well defined divergence. In order to formulate a concrete basis, the polynomial space must be used in conjunction with

the degrees of freedom, $\mathcal{A}$ . This process will be described in detail in Section 4.5.

Let $Q_{p_1,p_2,\ldots,p_n}$ denote a polynomial of $n$ variables $(x_1,x_2,\ldots,x_n)$ whose maximum degree is $p_1$ in $x_1$, $p_2$ in $x_2$, ..., $p_n$ in $x_n$. For example, the polynomial $(2x+2)(3y^2+y)(z^2+4)$ would belong to the set of three dimensional polynomials denoted $Q_{1,2,2}$. Also, let $\mathcal{P}^{l,p}(\Sigma)$ denote the polynomial space for a discrete $l$-form function of order $p$ defined over the finite element domain $\Sigma$ . Using this notation, the 1-form polynomial space on the unit hexahedron and its dimension (i.e. the number of basis functions needed to fully define the space) are given by [58]

$$\begin{aligned} \mathcal{P}^{1,p}(\hat{\Sigma}) &= \{\mathbf{u}; u_x \in Q_{p-1,p,p}, u_y \in Q_{p,p-1,p}, u_z \in Q_{p,p,p-1}\} \\ dim(\mathcal{P}^{1,p}(\hat{\Sigma})) &= 3p(p+1)^2 \end{aligned} \tag{4.11}$$

The 2-form polynomial space on the unit hexahedron and its dimension are given by

$$\begin{aligned} \mathcal{P}^{2,p}(\hat{\Sigma}) &= \{\mathbf{u}; u_x \in Q_{p,p-1,p-1}, u_y \in Q_{p-1,p,p-1}, u_z \in Q_{p-1,p-1,p}\} \\ dim(\mathcal{P}^{2,p}(\hat{\Sigma})) &= 3p^2(p+1) \end{aligned} \tag{4.12}$$

As pointed out by [104], these spaces have the desirable property that $d\mathcal{P}^{1,p} \in \mathcal{P}^{2,p}$, i.e. the curl of 1-forms is contained in the 2-form space. This is necessary for satifying the discrete exact sequence property of (3.27) which in turn is necessary for the conservation of numerical charge as shown in Chapter 6.

## 4.4 Basis Functions

We now present explicit formulae for the construction of interpolatory and hierarchical basis functions. The basis functions presented are valid only on the reference element; so we also present the appropriate transformation rules which map 1-form and 2-form vector functions from the reference element to physical mesh elements. For clarity, we will denote the three independent variables in the reference system using the standard Cartesian notation of $(\hat{x},\hat{y},\hat{z})$. In addition, we will denote the contra-variant basis vectors as $\hat{\mathbf{x}},\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$, while the covariant basis vectors will be denoted as $\hat{\mathbf{X}},\hat{\mathbf{Y}}$ and $\hat{\mathbf{Z}}$. This is a trivial distinction on the reference element, for as mentioned in section 4.2.2 these basis vectors are identical, have unit magnitude, and are con-

stant over the domain of the reference hexahedron. However, as we have seen, this is no longer necessarily true for basis vectors defined on a general curvilinear coordinate system (like the coordinate system used to define a global mesh element). For this reason we make the distinction clear.

In order to ensure the proper conformity across element to element interfaces, it is crucial that the basis functions (and consequently the degrees of freedom) be associated with the various sub-simplices of the element (e.g. nodes, edges, faces, etc ...). This property is referred to as locality [86] and will be explained in greater detail in section 4.5.

### 4.4.1 1-form Basis Functions

Let $\hat{W}$ denote a 1-form basis on the reference element, with individual basis functions denoted as $\hat{\mathbf{w}}_i$ such that $\hat{\mathbf{w}}_i \in \hat{W}$. In order to satisfy the locality property, we can break this set of basis functions into three mutually disjoint subsets such that

$$\hat{W} = \hat{W}_e \cup \hat{W}_f \cup \hat{W}_v \tag{4.13}$$

where the subscripts $e, f$ and $v$ denote the edges, faces and volume of the reference element respectively. The dimensions of these subsets will in general be *number of sub-simplices per element* times *number of degrees of freedom per sub-simplex*. For 1-forms, locality implies that the edge basis functions should have non-vanishing tangential components along one and only one edge. The face basis functions will have non-vanishing tangential components along one and only one face with no tangential components along any edges. Finally, the volume basis functions will have no tangential components along either edges or faces.

#### Interpolatory 1-form Basis Functions

Interpolatory basis functions will be denoted by an $I$ superscript. The set of interpolation points, $X$, has been omitted for clarity and can thus be arbitrary; however we will show later on that use of the Extended Chebyshev set, $\bar{X}^p$, will yield better conditioned mass and stiffness matrices. Due to the nature of interpolatory polynomials (see Figure 3.2), each of the 1-form interpolatory basis functions are members

of the full polynomial space, $\mathcal{P}^{1,p}(\hat{\Sigma})$, and therefore, are of maximum polynomial degree. The interpolatory edge basis functions of polynomial degree $p$ are given by

$$\hat{W}_e^I = \begin{cases} L_i^p(\hat{y})L_j^p(\hat{z})L_k^{p-1}(\hat{x})\,\hat{\mathbf{x}} \\ L_i^p(\hat{x})L_j^p(\hat{z})L_k^{p-1}(\hat{y})\,\hat{\mathbf{y}} \\ L_i^p(\hat{x})L_j^p(\hat{y})L_k^{p-1}(\hat{z})\,\hat{\mathbf{z}} \end{cases} \quad i,j = 0,p;\ \ k = 0,\ldots,p-1 \tag{4.14}$$

This set of functions is grouped into three sub-sets, one for each contravariant basis vector. The indices $i$ and $j$ loop over the 4 edges that are tangent to these basis vectors. The index $k$ loops over the $p$ basis functions per edge for a total of $12p$. The interpolatory face basis functions of polynomial degree $p$ are given by

$$\hat{W}_f^I = \begin{cases} L_i^p(\hat{x})L_j^p(\hat{z})L_k^{p-1}(\hat{y})\,\hat{\mathbf{y}} \\ L_i^p(\hat{x})L_j^p(\hat{y})L_k^{p-1}(\hat{z})\,\hat{\mathbf{z}} \\ L_i^p(\hat{y})L_j^p(\hat{z})L_k^{p-1}(\hat{x})\,\hat{\mathbf{x}} \\ L_i^p(\hat{y})L_j^p(\hat{x})L_k^{p-1}(\hat{z})\,\hat{\mathbf{z}} \\ L_i^p(\hat{z})L_j^p(\hat{y})L_k^{p-1}(\hat{x})\,\hat{\mathbf{x}} \\ L_i^p(\hat{z})L_j^p(\hat{x})L_k^{p-1}(\hat{y})\,\hat{\mathbf{y}} \end{cases} \quad i = 0,p;\ \ j = 1,\ldots,p-1;\ \ k = 0,\ldots,p-1 \tag{4.15}$$

This set of functions is grouped into six sub-sets, two for each face representing the contravariant basis vectors that are in the plane of that face. The index $i$ loops over the 2 faces that are normal to these basis vectors. The indices $j$ and $k$ loop over the $2p(p-1)$ basis functions per face for a total of $12p(p-1)$. Finally, there will be a total of $3p(p-1)^2$ interpolatory basis functions that are internal to the reference element (i.e. functions not shared between elements), given by

$$\hat{W}_v^I = \begin{cases} L_i^p(\hat{y})L_j^p(\hat{z})L_k^{p-1}(\hat{x})\,\hat{\mathbf{x}} \\ L_i^p(\hat{x})L_j^p(\hat{z})L_k^{p-1}(\hat{y})\,\hat{\mathbf{y}} \\ L_i^p(\hat{x})L_j^p(\hat{y})L_k^{p-1}(\hat{z})\,\hat{\mathbf{z}} \end{cases} \quad i,j = 1,\ldots,p-1;\ \ k = 0,\ldots,p-1 \tag{4.16}$$

Note that for the particular case of $p = 1$, i.e. first order basis functions, there will be no face and volume basis functions, only the 12 edge functions; hence the name “edge basis” that is commonly used. However it should be noted that in the general case of arbitrary order, this is a misnomer. Figure 4.5 gives some visual examples of 1-form interpolatory basis functions on the reference element.

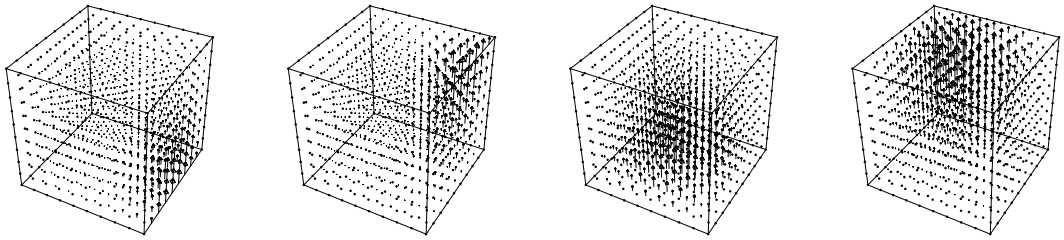

Figure 4.5: Examples of 1-form interpolatory face and cell functions of polynomial degree $p = 2$.

**Hierarchical 1-form Basis Functions**

Hierarchical basis functions will be denoted by an $H$ superscript and will make use of the scaled, normalized Legendre polynomials defined in Chapter 3 [105]. In order to satisfy the locality property, they will also make use of certain Lagrange interpolatory polynomials. Unlike interpolatory basis functions, the subsets of hierarchical basis functions will span corresponding subspaces of the full polynomial space, $\mathcal{P}^{1,p}(\hat{\Sigma})$. The hierarchical edge basis functions of polynomial degree $p$ are given by

$$\hat{W}_e^H = \begin{cases} L_i^1(\hat{y})L_j^1(\hat{z})\overline{l}^k(\hat{x})\,\hat{\mathbf{x}} \\ L_i^1(\hat{x})L_j^1(\hat{z})\overline{l}^k(\hat{y})\,\hat{\mathbf{y}} \quad i,j=0,1;\ k=0,\ldots,p-1 \\ L_i^1(\hat{x})L_j^1(\hat{y})\overline{l}^k(\hat{z})\,\hat{\mathbf{z}} \end{cases} \tag{4.17}$$

The subset $\hat{W}_e^H$ spans the subspace $\{Q_{p-1,1,1}, Q_{1,p-1,1}, Q_{1,1,p-1}\}$, which has a dimension of $12p$. This gives $p$ basis functions per edge. The hierarchical face basis functions of polynomial degree $p$ are given by

$$\hat{W}_f^H = \begin{cases} L_i^1(\hat{x})\overline{l}^j(\hat{y})\overline{l}^k(\hat{z})L_1^2(\hat{z})\,\hat{\mathbf{y}} \\ L_i^1(\hat{x})\overline{l}^j(\hat{z})\overline{l}^k(\hat{y})L_1^2(\hat{y})\,\hat{\mathbf{z}} \\ L_i^1(\hat{y})\overline{l}^j(\hat{x})\overline{l}^k(\hat{z})L_1^2(\hat{z})\,\hat{\mathbf{x}} \\ L_i^1(\hat{y})\overline{l}^j(\hat{z})\overline{l}^k(\hat{x})L_1^2(\hat{x})\,\hat{\mathbf{z}} \\ L_i^1(\hat{z})\overline{l}^j(\hat{x})\overline{l}^k(\hat{y})L_1^2(\hat{y})\,\hat{\mathbf{x}} \\ L_i^1(\hat{z})\overline{l}^j(\hat{y})\overline{l}^k(\hat{x})L_1^2(\hat{x})\,\hat{\mathbf{y}} \end{cases} \quad i=0,1;\ j=0,\ldots,p-1;\ k=0,\ldots,p-2 \tag{4.18}$$

The face basis functions of the subset $\hat{W}_f^H$ span the subspace $Q_{1,p-1,p-2} \cup Q_{1,p-2,p-1}$ for faces normal to the $\hat{\mathbf{x}}$ basis vector, $Q_{p-1,1,p-2} \cup Q_{p-2,1,p-1}$ for faces normal to the $\hat{\mathbf{y}}$ basis vector and $Q_{p-1,p-2,1} \cup Q_{p-2,p-1,1}$ for

faces normal to the $\hat{\mathbf{z}}$ basis vector. The total dimension of the face subspace is $12p(p-1)$, with $2p(p-1)$ basis functions per face. Finally, there will be a total of $3p(p-1)^2$ hierarchical basis functions that are internal to the reference element given by

$$\hat{W}_v^H = \begin{cases} \overline{l}^i(\hat{y})L_1^2(\hat{y})\overline{l}^j(\hat{z})L_1^2(\hat{z})\overline{l}^k(\hat{x})\,\hat{\mathbf{x}} \\ \overline{l}^i(\hat{x})L_1^2(\hat{x})\overline{l}^j(\hat{z})L_1^2(\hat{z})\overline{l}^k(\hat{y})\,\hat{\mathbf{y}} \quad i,j=0,\ldots,p-2;\ k=0,\ldots,p-1 \\ \overline{l}^i(\hat{x})L_1^2(\hat{x})\overline{l}^j(\hat{y})L_1^2(\hat{y})\overline{l}^k(\hat{z})\,\hat{\mathbf{z}} \end{cases} \tag{4.19}$$

The subset $\hat{W}_v^H$ spans the subspace $\{Q_{p-1,p-2,p-2}, Q_{p-2,p-1,p-2}, Q_{p-2,p-2,p-1}\}$.

In addition to their hierarchical nature, it is important to point out that by construction, the basis functions associated with a given sub-simplex are all orthogonal to each other. For example, all hierarchical basis functions associated with a given edge are mutually orthogonal, while all of the volume (or interior) hierarchical basis functions are mutually orthogonal. Figure 4.6 gives some visual examples of 1-form hierarchical basis functions on the reference element.

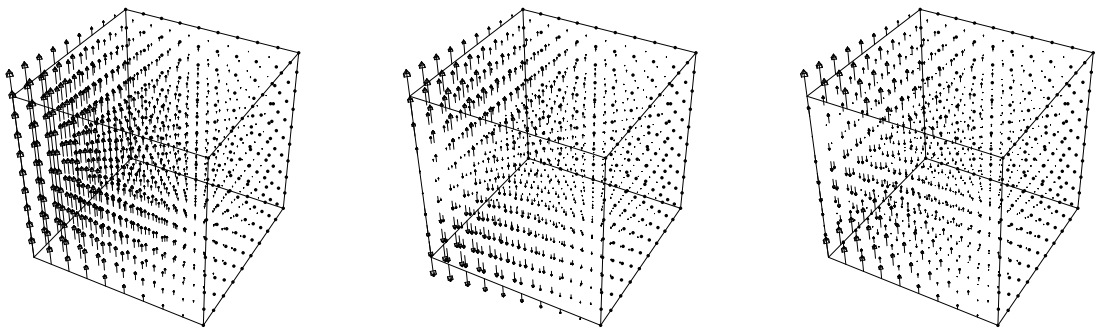

Figure 4.6: Examples of 1-form hierarchical edge functions of polynomial degree $p = 3$.

### 4.4.2 2-form Basis Functions

Let $\hat{F}$ denote a 2-form basis on the reference element, with individual basis functions denoted as $\hat{\mathbf{f}}_i$ such that $\hat{\mathbf{f}}_i \in \hat{F}$. In order to satisfy the locality property, we can break this set of basis functions into two

mutually disjoint subsets such that

$$\hat{F} = \hat{F}_f \cup \hat{F}_v \tag{4.20}$$

where the subscripts $f$ and $v$ denote the faces and volume of the reference element respectively. For 2-forms, locality implies that the face basis functions will have non-vanishing normal components along one and only one face while the volume basis functions will have no normal components along the faces.

**Interpolatory 2-form Basis Functions**

Again, due to the nature of interpolatory polynomials, each of the 2-form interpolatory basis functions are members of the full polynomial space, $\mathcal{P}^{2,p}(\hat{\Sigma})$, and therefore of maximum polynomial degree. The interpolatory face basis functions of polynomial degree $p$ are given by

$$\hat{F}_f^I = \begin{cases} L_i^p(\hat{x}) L_j^{p-1}(\hat{y}) L_k^{p-1}(\hat{z})\, \hat{\mathbf{X}} \\ L_i^p(\hat{y}) L_j^{p-1}(\hat{x}) L_k^{p-1}(\hat{z})\, \hat{\mathbf{Y}} \quad i = 0, p; \;\; j,k = 0, \ldots, p-1 \\ L_i^p(\hat{z}) L_j^{p-1}(\hat{x}) L_k^{p-1}(\hat{y})\, \hat{\mathbf{Z}} \end{cases} \tag{4.21}$$

This set of functions is grouped into three sub-sets, one for each of the covariant basis vectors. The index $i$ loops over the 2 faces that are normal to these basis vectors. The indices $j$ and $k$ loop over the $p^2$ basis functions per face for a total of $6p^2$. Finally, there will be a total of $3p^2(p-1)$ interpolatory basis functions that are internal to the reference element given by

$$\hat{F}_v^I = \begin{cases} L_i^p(\hat{x}) L_j^{p-1}(\hat{y}) L_k^{p-1}(\hat{z})\, \hat{\mathbf{X}} \\ L_i^p(\hat{y}) L_j^{p-1}(\hat{x}) L_k^{p-1}(\hat{z})\, \hat{\mathbf{Y}} \quad i = 1, \ldots, p-1; \;\; j,k = 0, \ldots, p-1 \\ L_i^p(\hat{z}) L_j^{p-1}(\hat{x}) L_k^{p-1}(\hat{y})\, \hat{\mathbf{Z}} \end{cases} \tag{4.22}$$

Again, note that for the particular case of $p = 1$, there will be no volume basis functions, only the 6 face functions; hence the name “face basis” that is commonly used. Again, for the general case of arbitrary order, this is a misnomer. Figure 4.7 gives some visual examples of 2-form interpolatory basis functions on the reference element.

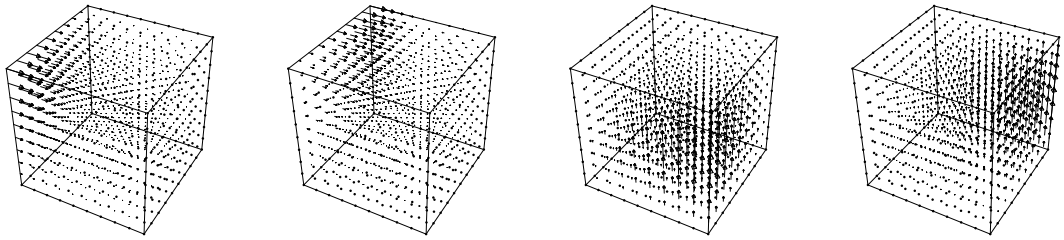

Figure 4.7: Examples of 2-form interpolatory face and cell functions of polynomial degree $p = 2$.

**Hierarchical 2-form Basis Functions**

The hierarchical face basis functions of polynomial degree $p$ are given by

$$\hat{F}_f^H = \begin{cases} L_i^1(\hat{x})\overline{l}^j(\hat{y})\overline{l}^k(\hat{z})\,\hat{\mathbf{X}} \\ L_i^1(\hat{y})\overline{l}^j(\hat{x})\overline{l}^k(\hat{z})\,\hat{\mathbf{Y}} \quad i=0,1;\;\; j,k=0,\ldots,p-1 \\ L_i^1(\hat{z})\overline{l}^j(\hat{x})\overline{l}^k(\hat{y})\,\hat{\mathbf{Z}} \end{cases} \tag{4.23}$$

The face basis functions of the subset $\hat{F}_f^H$ span the subspace $Q_{1,p-1,p-1}$ for faces normal to the $\hat{\mathbf{X}}$ covariant basis vector, $Q_{p-1,1,p-1}$ for faces normal to the $\hat{\mathbf{Y}}$ covariant basis vector and $Q_{p-1,p-1,1}$ for faces normal to the $\hat{\mathbf{Z}}$ covariant basis vector. The total dimension of the face subspace is $6p^2$, with $p^2$ basis functions per face. Finally, there will be a total of $3p^2(p-1)$ hierarchical basis functions that are internal to the reference element (i.e. functions not shared between elements), given by

$$\hat{F}_v^H = \begin{cases} \overline{l}^i(\hat{y})\overline{l}^j(\hat{z})\overline{l}^k(\hat{x})L_1^2(\hat{x})\,\hat{\mathbf{X}} \\ \overline{l}^i(\hat{x})\overline{l}^j(\hat{z})\overline{l}^k(\hat{y})L_1^2(\hat{y})\,\hat{\mathbf{Y}} \quad i,j=0,\ldots,p-1;\;\; k=0,\ldots,p-2 \\ \overline{l}^i(\hat{x})\overline{l}^j(\hat{y})\overline{l}^k(\hat{z})L_1^2(\hat{z})\,\hat{\mathbf{Z}} \end{cases} \tag{4.24}$$

The subset $\hat{F}_v^H$ spans the subspace $\{Q_{p-2,p-1,p-1}, Q_{p-1,p-2,p-1}, Q_{p-1,p-1,p-2}\}$. Figure 4.8 gives some visual examples of 2-form hierarchical basis functions on the reference element.

### 4.4.3 Basis Function Transformation Rules

The local to global mapping of (4.5) is used to transform points defined in one coordinate system to another coordinate system. The basis functions presented are defined with respect to the reference coor-

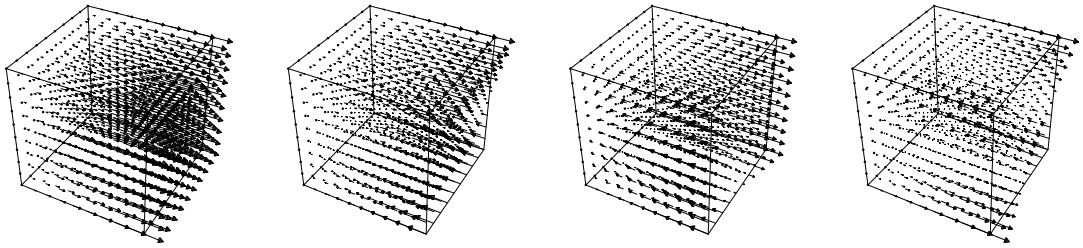

Figure 4.8: Examples of 2-form hierarchical face functions of polynomial degree $p = 2$.

dinate system. We therefore need a general procedure for transforming functions defined in one coordinate system to another system. As with the covariant and contravariant transformations of Section 4.2, in order to maintain coordinate independence, all properties of the functions defined in the reference coordinate system must be preserved under a transformation to a new global coordinate system, this property is known as invariance [86]. For example, invariance implies that 1-form basis functions defined to have non-vanishing tangential components along only one edge in the reference coordinate system, must also have non-vanishing tangential components along only the same edge in the new coordinate system. In addition, we would like the scaling of the functions to be independent of the coordinate system used to represent them. As we will see in the next section, the contravariant and covariant basis vectors are related to the degrees of freedom, $\mathcal{A}$, and are thus *dual* to the $l$-form basis. As such, they should scale inversely with the $l$-form basis in order to maintain coordinate independence. Table 4.3 and Table 4.4 give the precise transformation rules for 1-forms and 2-forms as well as the units of these transformations. The symbol $m$ denotes an arbitrary metric of distance while the symbol $\circ$ denotes composition. Note that the exterior derivative (the curl) of a 1-form transforms identically as a 2-form; which is consistent with our knowledge of the properties of the exterior derivative. Figure 4.9 and Figure 4.10 give some visual examples of these transformations applied to particular members of the 1-form and 2-form bases of polynomial degree $p = 1$. Each of the basis functions are plotted over three different elements corresponding to three different local to global mappings of geometry order $s = 0$ (i.e. the reference element), $s = 1$ and $s = 2$.

| Object | Transformation Rule | Units |
|---|---|---|
| Contravariant basis vectors | $\mathbf{v}_i = J^T\ \hat{\mathbf{v}}_i$ | $m^1$ |
| 1-form functions | $\mathbf{w} \circ \Phi = J^{-1}\ \hat{\mathbf{w}}$ | $m^{-1}$ |
| Curl of 1-form | $d\mathbf{w} \circ \Phi = \frac{1}{\lvert J \rvert}\ J^T\ d\hat{\mathbf{w}}$ | $m^{-2}$ |

Table 4.3: 1-form Transformation Rules

| Object | Transformation Rule | Units |
|---|---|---|
| Covariant basis vectors | $\mathbf{V}_i = \lvert J \rvert\ J^{-1}\ \hat{\mathbf{V}}_i$ | $m^2$ |
| 2-form functions | $\mathbf{f} \circ \Phi = \frac{1}{\lvert J \rvert}\ J^T\ \hat{\mathbf{f}}$ | $m^{-2}$ |
| Divergence of 2-form | $d\mathbf{f} \circ \Phi = \frac{1}{\lvert J \rvert}\ d\hat{\mathbf{f}}$ | $m^{-3}$ |

Table 4.4: 2-form Transformation Rules

## 4.5 $\mathcal{A}$ - Degrees of Freedom

The set $\mathcal{A}$ of degrees of freedom consists of linear functionals that map an arbitrary function, $\mathbf{g}$, onto the set of real numbers, i.e. $\mathcal{A} : \mathbf{g} \mapsto \Re$. The number of linear functionals is equal to the dimension of the polynomial space $\mathcal{P}$ . For example, if we are working with a 1-form basis of polynomial degree $p = 1$ on a hexahedron, we know that we will need 12 basis functions; the set $\mathcal{A}$ would therefore contain 12 linear functionals. The set $\mathcal{A}$ satisfies three important properties; namely

- *Unisolvence*: $\mathcal{A}$ is dual to the polynomial space $\mathcal{P}$ . For a basis $W \in \mathcal{P}$, the linear system $\mathcal{A}_i(W_j)$ must be non-singular. This allows the relation $\mathcal{A}_i(W_j) = \delta_{i,j}$ to be enforced which is necessary for basis function expansions to be valid.
- *Invariance*: Degrees of freedom remain unisolvent upon a change of variables, i.e they are valid in any coordinate system, not just the reference coordinate system.
- *Locality*: The trace of a basis function on a sub-simplex is determined by degrees of freedom associated *only* with that sub-simplex. For example, locality implies that if we project a two dimensional function onto the face of an element, we will only need to use basis functions associated with that face.

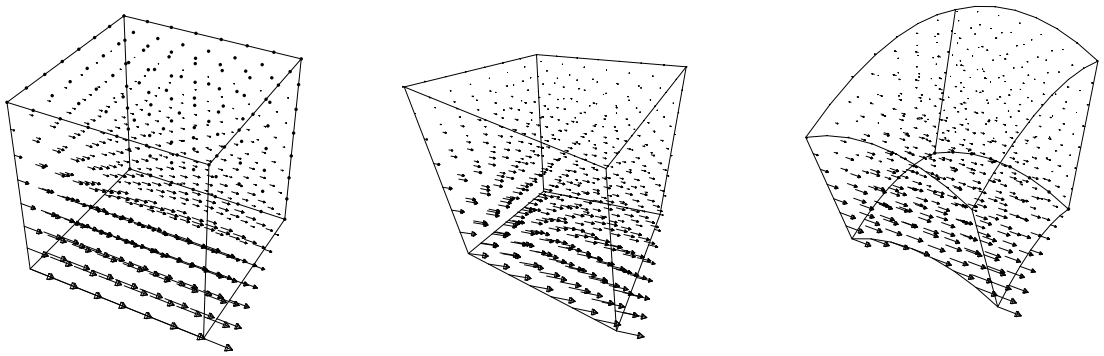

Figure 4.9: Examples of a 1-form basis function transformation for elements of geometry order $s = 0, 1$ and 2 (*left to right*).

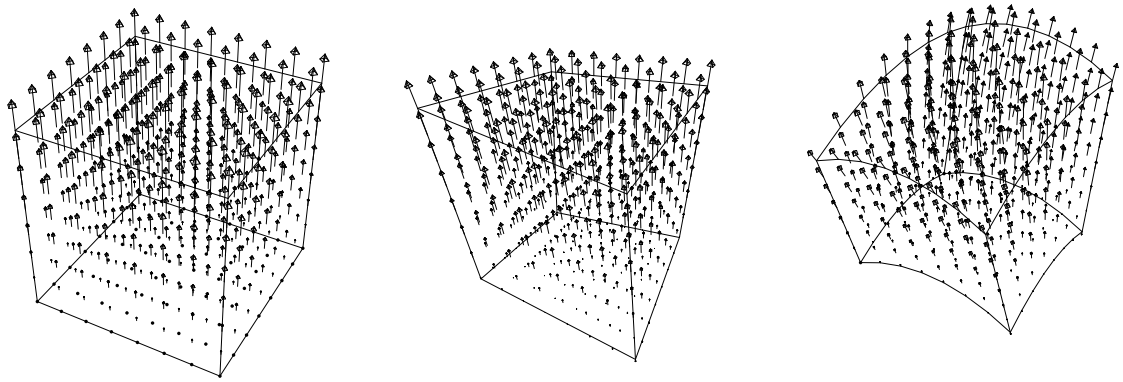

Figure 4.10: Examples of a 2-form basis function transformation for elements of geometry order $s = 0, 1$ and 2 (*left to right*).

The degrees of freedom are best understood in the following context. Suppose we have a 1-form field (for example, the 1-form electric field) that we wish to approximate using a vector basis function expansion. The expansion would be of the form

$$\mathbf{g} \approx \Pi(\mathbf{g}) = \sum_{i=1}^{dim(W)} \mathcal{A}_i(\mathbf{g})\, \mathbf{w}_i \tag{4.25}$$

where $\mathbf{w}_i$ are the 1-form basis functions and the projection operation $\Pi(\mathbf{g})$ denotes a basis function expansion of $\mathbf{g}$ (i.e. the projection of $\mathbf{g}$ onto the discrete 1-form space). The degrees of freedom act as weights in the expansion and are computed by evaluating the linear functionals $\mathcal{A}_i(\mathbf{g})$ in a manner completely analogous to computing the coefficients in a Fourier expansion. The projection operation is required by finite element

simulation codes to implement source terms, boundary conditions, mixed bilinear forms etc .... The linear functionals of $\mathcal{A}$ will return a set of real numbers representing the weights to be used in the basis function expansion. These numbers can have physical significance. For example, the electric field, **E**, has units of volts per meter $(V/m)$. In a basis function expansion of the form (4.25), the discrete 1-form basis functions will have units of inverse meters $(m^{-1})$, while the evaluation of the electric field using the degrees of freedom, $\mathcal{A}_i(\mathbf{E})$, will have units of voltage $(V)$. In general, a discrete $l$-form function will always have units of $m^{-l}$ where $m$ can be an arbitrary metric of distance. This is consistent with our understanding of $l$-forms, since an $l$-form is a differential of $l$-dimensional space. Therefore, the projection of any field onto the degrees of freedom will always carry the physical units of the field that are not related to space (such as voltage, current, charge, etc ...) and will always be spatially independent.

### 4.5.1 Integral Degrees of Freedom

The exact linear functionals from $\mathcal{A}$ that are dual to the 1-from and 2-form polynomial spaces for hexahedrons are given in [58] and are defined in terms of weighted moment integrals over sub-simplices of the element $\Sigma$ (e.g. line integrals over edges, surface integrals over faces, etc ...). If we denote a sub-simplex of an element $\Sigma$ of dimension $n$ as $\Sigma_n$, then the generalized form for the linear functional is given by [86]

$$\mathcal{A}(\mathbf{g}) = \int_{\Sigma_n} \mathbf{g} \wedge q_n \tag{4.26}$$

where $l \leq n \leq 3$ and $q_n$ is an $(n-l)$-form weighting polynomial of $n$-variables defined over the sub-simplex $\Sigma_n$. For example, the degrees of freedom for 1-forms will involve line integrals over edges weighted by 1-dimensional 0-forms, surface integrals over faces weighted by 2 dimensional 1-forms and volume integrals over the element weighted by 3-dimensional 2-forms.

#### 1-form Integral Degrees of Freedom

For a 1-form, we will require integrals over edges, faces and the volume of the element $\Sigma$ itself. Just as the case for a 1-form basis, we can break the set of 1-form degrees of freedom into three mutually

disjoint subsets

$$\mathcal{A} = \mathcal{A}_e \cup \mathcal{A}_f \cup \mathcal{A}_v \tag{4.27}$$

where the letters $e$, $f$ and $v$ denote sets of integrals over edges, faces and the volume of the element $\Sigma$ respectively. Due to the locality requirement, the dimensions of these subsets will be identical to the corresponding 1-form basis function subsets. As in the case with bilinear forms, we will find it much more convenient to integrate over the reference element $\hat{\Sigma}$, and then use the appropriate transformation rules to map the result to the actual element $\Sigma$ . The 1-form edge moments have the form

$$\mathcal{A}_e(\mathbf{g}) = \int_{\hat{e}} (\mathbf{g} \circ \Phi) \cdot J^T (\hat{\mathbf{t}} q) \tag{4.28}$$

The symbol $\hat{\mathbf{t}}$ denotes the unit tangent vector for each of the 12 edges on the reference element. In this case the weighting polynomial $q$ is a 1-dimensional 0-form such that $q \in Q_{p-1}$. This gives $p$ weighting polynomials per edge for a total of $12p$ edge integrals. The 1-form face moments will have the form

$$\mathcal{A}_f(\mathbf{g}) = \iint_{\hat{f}} (\mathbf{g} \circ \Phi) \cdot J^T (\hat{\mathbf{n}} \times \mathbf{q}) \tag{4.29}$$

The symbol $\hat{\mathbf{n}}$ denotes the unit normal vector for each of the 6 faces on the reference element. In this case the weighting polynomial $\mathbf{q}$ is a 2-dimensional 1-form (i.e. it is defined in a plane) such that $\mathbf{q} = (q_1, q_2)$. In practice, these 2-dimensional 1-forms are implemented as three dimensional vectors with one component set to zero (thus designating the plane in which it is defined). The non-zero components are such that $q_1 \in Q_{p-2,p-1}$ and $q_2 \in Q_{p-1,p-2}$. This gives $2p(p-1)$ weighting polynomials per face for a total of $12p(p-1)$ face integrals. Finally, the volume moments will have the form

$$\mathcal{A}_v(\mathbf{g}) = \iiint_{\hat{v}} (\mathbf{g} \circ \Phi) \cdot J^T \mathbf{q} \tag{4.30}$$

The weighting polynomial $\mathbf{q}$ is a 3-dimensional 2-form such that $\mathbf{q} = (q_1, q_2, q_3)$, where $q_1 \in Q_{p-1,p-2,p-2}$, $q_2 \in Q_{p-2,p-1,p-2}$ and $q_3 \in Q_{p-2,p-2,p-1}$. Equivalently, we can write $\mathbf{q} \in \mathcal{P}^{2,p-1}(\hat{\Sigma})$. This gives $3p(p-1)^2$ volume integrals. The weighting polynomials for 1-forms transform the same way as the contravariant basis vectors. This transformation is the inverse of the transformation rule for 1-form functions, therefore enforcing spatial invariance for the 1-form degrees of freedom.

**2-form Integral Degrees of Freedom**

For a 2-form, we will only require integrals over faces and the volume of the element $\Sigma$. Just as the case for a 2-form basis, we can break the set of 2-form degrees of freedom into two mutually disjoint subsets

$$\mathcal{A} = \mathcal{A}_f \cup \mathcal{A}_v \tag{4.31}$$

where the letters $f$ and $v$ denote sets of integrals over faces and the volume of the element $\Sigma$ respectively. Again, due to the locality requirement, the dimensions of these subsets will be identical to the corresponding 2-form basis function subsets. The 2-form face moments will have the form

$$\mathcal{A}_f(\mathbf{g}) = \iint_{\hat{f}} (\mathbf{g} \circ \Phi) \cdot |J| J^{-1} (\hat{\mathbf{n}} q) \tag{4.32}$$

The symbol $\hat{\mathbf{n}}$ denotes the unit normal vector for each of the 6 faces on the reference element. In this case the weighting polynomial $q$ is a 2-dimensional 0-form such that $q \in Q_{p-1,p-1}$. This gives $p^2$ weighting polynomials per face for a total of $6p^2$ face integrals. The 2-form volume moments will have the form

$$\mathcal{A}_v(\mathbf{g}) = \iiint_{\hat{v}} (\mathbf{g} \circ \Phi) \cdot |J| J^{-1} \mathbf{q} \tag{4.33}$$

The weighting polynomial $\mathbf{q}$ is a 3-dimensional 1-form such that $\mathbf{q} = (q_1, q_2, q_3)$, where $q_1 \in Q_{p-2,p-1,p-1}$, $q_2 \in Q_{p-1,p-2,p-1}$ and $q_3 \in Q_{p-1,p-1,p-2}$. Equivalently, we can write $\mathbf{q} \in \mathcal{P}^{1,p-1}(\hat{\Sigma})$. This gives $3p^2(p-1)$ volume integrals. The weighting polynomials for 2-forms transform the same way as the covariant basis vectors. This transformation is the inverse of the transformation rule for 2-form functions, therefore enforcing spatial invariance for the 2-form degrees of freedom.

### 4.5.2 Point Degrees of Freedom

While the integral degrees of freedom presented in the previous section are exact, they can be computationally expensive to implement. As such, we present a set of *discrete* degrees of freedom that are based on evaluation of a function at a point. These point degrees of freedom satisfy the properties of invariance and locality and given a set of basis functions, they can be used to enforce unisolvence using the procedure of section 4.5.3. We will discuss the nature of their discreteness in Section 4.5.5.

Let $\tilde{\mathcal{A}}$ denote the discrete point degrees of freedom. Furthermore, let $X$ denote a set of $p+1$ interpolation points over the unit interval $[0,1]$ and $X'$ denote a set of $p$ interpolation points over the same interval. These sets can be arbitrary with the exception that they must contain the endpoints of the interval $[0,1]$; this is needed to satisfy locality. The 1-form point degrees of freedom are given by

$$\tilde{\mathcal{A}}_i(\mathbf{g}) = \begin{cases} \mathbf{g}(\Phi(X'_i, X_j, X_k)) \cdot J^T \hat{\mathbf{x}} \\ \mathbf{g}(\Phi(X_k, X'_i, X_j)) \cdot J^T \hat{\mathbf{y}} \quad i = 0, \ldots, p-1; \;\; j,k = 0, \ldots, p, \\ \mathbf{g}(\Phi(X_j, X_k, X'_i)) \cdot J^T \hat{\mathbf{z}} \end{cases} \tag{4.34}$$

where $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ denote the contravariant basis vectors on the reference element. The 2-form point degrees of freedom are given by

$$\tilde{\mathcal{A}}_i(\mathbf{g}) = \begin{cases} \mathbf{g}(\Phi(X_i, X'_j, X'_k)) \cdot |J| J^{-1} \hat{\mathbf{X}} \\ \mathbf{g}(\Phi(X'_k, X_i, X'_j)) \cdot |J| J^{-1} \hat{\mathbf{Y}} \quad i = 0, \ldots, p; \;\; j,k = 0, \ldots, p-1 \\ \mathbf{g}(\Phi(X'_j, X'_k, X_i)) \cdot |J| J^{-1} \hat{\mathbf{Z}} \end{cases} \tag{4.35}$$

where $\hat{\mathbf{X}}, \hat{\mathbf{Y}}$ and $\hat{\mathbf{Z}}$ denote the covariant basis vectors on the reference element. Again, the contravariant and covariant basis vectors are equal to each other on the reference element; we make the distinction simply to emphasize their transformation properties.

### 4.5.3 Enforcing Unisolvence

Suppose we have a particular set of basis functions, $W \in \mathcal{P}$, and a set of degrees of freedom $\mathcal{A}$ that is dual to $\mathcal{P}$ and satisfies both the invariance and the locality properties. In order for basis function expansions of the form (4.25) to be valid, the following relation must hold

$$\mathcal{A}_i(W_j) = \delta_{i,j} \tag{4.36}$$

If the degrees of freedom are unisolvent, then it is a simple matter to enforce this relation. The unisolvence property requires that the matrix

$$V_{i,j} = \mathcal{A}_i(W_j) \tag{4.37}$$

be non singular. This matrix forms a linear mapping that is similar to a Vandermonde matrix. We can now apply this linear mapping to either the basis $W$ or the degrees of freedom $\mathcal{A}$ in order to enforce (4.36). In this chapter we have presented basis functions with particular properties that we would like to preserve; as such we will apply the linear mapping to $\mathcal{A}$ in order to satisfy (4.36). Because the degrees of freedom are linear functionals, we can construct a new set of degrees of freedom, denoted $\mathcal{A}'$, by the relation

$$\mathcal{A}' = (V^{-1})^T \mathcal{A} \tag{4.38}$$

Note that by construction, the interpolatory basis functions of Section 4.4 will satisfy (4.36) (up to some permutation of the basis functions). The procedure of (4.38) is therefore trivial for this particular case. However, this general procedure is valid for any proper set of basis functions including the hierarchical bases.

### 4.5.4 Validation of Basis Function Expansions

As mentioned in Chapter 1, any normed error analysis requires an explicit formulation of the expansion operator, $\Pi$, which in turn requires explicit basis functions and degrees of freedom, $\mathcal{A}$. The error in a basis function expansion of the form (4.25) is such that [58]

$$||\mathbf{g} - \Pi(\mathbf{g})|| \leq c\, h^p |\mathbf{g}| \tag{4.39}$$

where $c$ is a scalar valued constant of proportionality, $h$ is the characteristic size (or volume) of the element and $p$ is the polynomial degree of the basis functions. In order to validate the basis functions of Section 4.4, we compute the expansion error of a known vector function and its derivative for a series of $h$ and $p$ values and verify that the error convergence rate is of the form (4.39).

To validate the 1-form basis functions we choose a vector valued test function that is "sufficiently smooth", non-polynomial and has a well defined curl. Specifically, we choose

$$\mathbf{g} = \{\sin(z), \cos(x), \exp(y)\} \tag{4.40}$$

We then generate basis function expansions of this function and its curl using the expansion operator $\Pi$ and compute the error of these expansions using the $L_2$ volume norm: $||\mathbf{g}||_2 = \sqrt{\int_\Sigma (\mathbf{g}\cdot\mathbf{g})}$ . Figure 4.11 shows

logarithmic plots of the error in the expansion of (4.40) using the hierarchical 1-form basis functions of (4.17) – (4.19) for 4 levels of *h*-refinement and 6 levels of *p*-refinement. Figure 4.12 shows logarithmic plots of the error in the curl of the expansion of (4.40), i.e. $d\Pi(\mathbf{g})$, using the same basis functions and $h, p$ values.

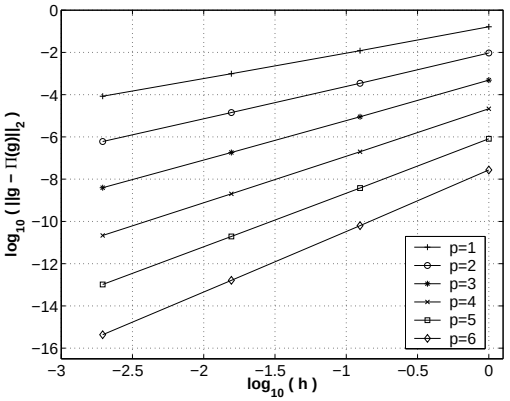


Figure 4.11: Projection error $||\mathbf{g} - \Pi(\mathbf{g})||_2$, using 1-form hierarchical basis functions with 4 levels of *h*-refinement and 6 levels of *p*-refinement.

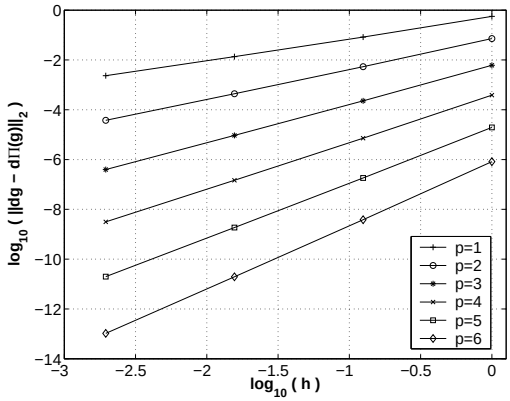


Figure 4.12: Projection error $||d\mathbf{g} - d\Pi(\mathbf{g})||_2$, using 1-form hierarchical basis functions with 4 levels of *h*-refinement and 6 levels of *p*-refinement.

To validate the 2-form basis functions we choose a vector valued test function that is "sufficiently smooth", non-polynomial and has a well defined divergence. Specifically, we choose

$$\mathbf{g} = \{\sin(x), \cos(y), \exp(z)\} \tag{4.41}$$

Figure 4.13 shows logarithmic plots of the error in the expansion of (4.41) using the interpolatory 2-form basis functions of (4.21) – (4.22) for 4 levels of *h*-refinement and 6 levels of *p*-refinement. Figure 4.14 shows logarithmic plots of the error in the divergence of the expansion of (4.41), i.e. $d\Pi(\mathbf{g})$, using the same basis functions and $h, p$ values.

### 4.5.5 Commuting Diagram Property

There is another property of the set $\mathcal{A}$ that is commonly required in the mathematics community; the so called *commuting diagram* property. Suppose we have an arbitrary 1-form function, **g**, and we compute an approximate version of it using a 1-form basis function expansion of the form (4.25), $\Pi(\mathbf{g})$. In addition,

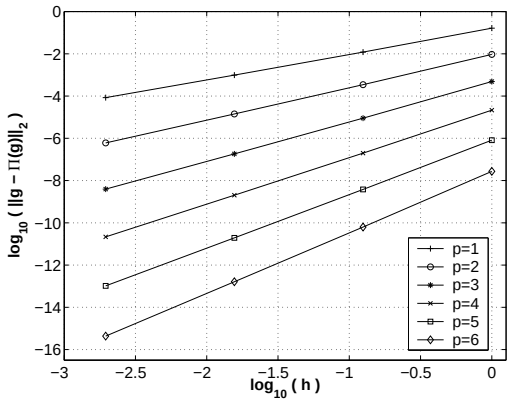


Figure 4.13: Projection error $||\mathbf{g} - \Pi(\mathbf{g})||_2$, using 2-form interpolatory basis functions with 4 levels of *h*-refinement and 6 levels of *p*-refinement.

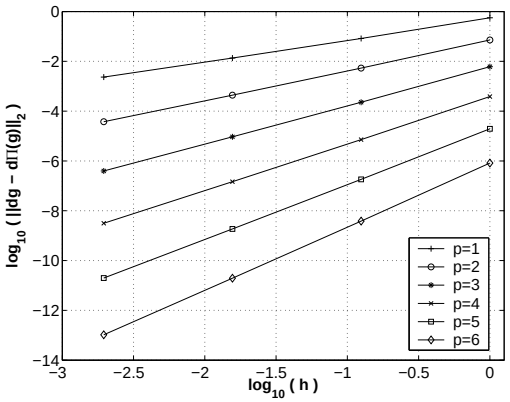


Figure 4.14: Projection error $||d\mathbf{g} - d\Pi(\mathbf{g})||_2$, using 2-form interpolatory basis functions with 4 levels of *h*-refinement and 6 levels of *p*-refinement.

suppose we take the curl of this function, $d\mathbf{g}$, the result of which will be a 2-form function, and compute an approximate version of this using a 2-form basis function expansion, $\Pi(d\mathbf{g})$. The commuting diagram states

$$d\Pi(\mathbf{g}) = \Pi(d\mathbf{g}) \tag{4.42}$$

In other words, the derivative of a projection must equal the projection of the derivative. The integral degrees of freedom of (4.28) – (4.30) and (4.32) – (4.33) satisfy this property exactly, meaning that the relation (4.42) is satisfied for any function $\mathbf{g}$. The point degrees of freedom of (4.34) and (4.35) satisfy this property in a discrete sense, meaning that as we increase the polynomial degree of the basis function expansion, the error in (4.42) converges to zero (using the $L2$ norm). Figure 4.15 gives an example of this using the function of (4.40). Likewise, the commuting diagram property using the point degrees of freedom for the *curl* operation holds exactly for any polynomial that is in the 2-form polynomial space of degree p (i.e. the *curl* of the 1-form space). It is important to point out that this property has no effect on the error convergence of the finite element solution of (2.9).

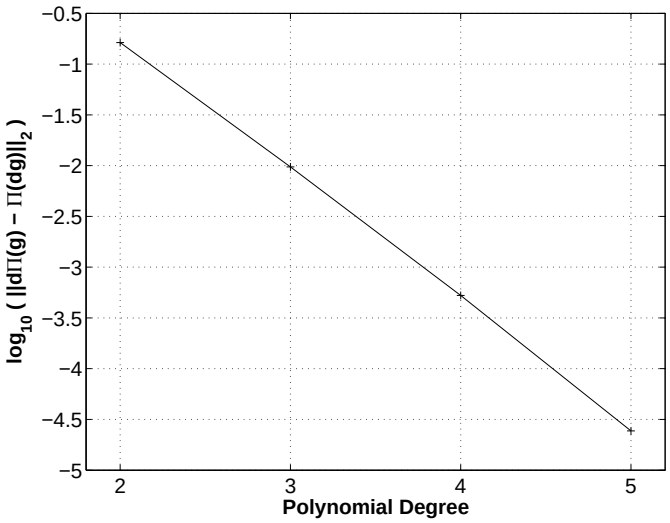


Figure 4.15: Error in commuting diagram property using the discrete point degrees of freedom.

## 4.6 Bilinear Forms

In the Galerkin finite element procedure, we require bilinear forms to construct the discrete PDE of (4.3). Typically, these bilinear forms involve integrals of basis functions, their derivatives and possibly material property functions (i.e. Hodge star functions) over the volume of mesh elements. They are used to construct objects such as global mass and stiffness matrices. In addition to the standard symmetric bilinear forms we can formulate special mixed bilinear forms which map members from one space of discrete differential forms to another space. These mixed bilinear forms are used to generate rectangular matrices which correspond to discrete versions of the exterior derivative. We can also accommodate certain radiation boundary conditions by making use of special surface bilinear forms which involve integrals over element surfaces.

### 4.6.1 Symmetric Bilinear Forms: Mass and Stiffness Matrices

We consider the general bilinear form $M_\alpha\langle\cdot,\cdot\rangle$ defined by

$$M_\alpha\langle f,g\rangle \;=\; \int_\Omega \star_\alpha\, f\wedge g \tag{4.43}$$

where $\star_\alpha$ is a generic Hodge star function associated with a symmetric definite positive tensor which can represent material properties such as electric and magnetic permeabilities and conductivities; and, $f$ and $g$ are both $l$-forms. Then by using the properties of the Hodge star and the local to global mapping (4.5), we re-write the bilinear form (4.43) as follows

$$\begin{aligned} M_\alpha\langle f,g\rangle &= \int_\Omega \star_\alpha\, f\wedge g \\ &= \sum_{\Sigma\in\Sigma_h}\int_\Sigma \star_\alpha\, f\wedge g \\ &= \sum_{\Sigma\in\Sigma_h}\int_{\hat\Sigma}(\star_\alpha\, f\wedge g)\circ\Phi\,|J| \\ &= \sum_{\Sigma\in\Sigma_h}\int_{\hat\Sigma}(\star_\alpha\, f\circ\Phi)\wedge(g\circ\Phi)\,|J| \end{aligned} \tag{4.44}$$

Equation (4.44) shows that all calculations for the bilinear forms (e.g. mass and stiffness matrices) can be performed on a standard reference element $\hat\Sigma$. The generic forms $f$ and $g$ in (4.44) are defined globally; and since the integral is performed locally, they must be replaced with transformed versions via the set of transformation rules given in Table 4.3 and Table 4.4.

In the following explicit bilinear forms, $M_\alpha$ denotes a mass matrix with a material property function $\star_\alpha$ defined over an element $\Sigma$ while $S_\gamma$ denotes a stiffness matrix with a material property function $\star_\gamma$ defined over an element $\Sigma$. The material property functions are free to be (possibly tensor valued) functions of space and will affect the scaling of each bilinear form. For 1-forms, we have the following symmetric bilinear forms

$$M_\alpha\langle \mathbf{w}_i,\mathbf{w}_j\rangle = \int_{\hat\Sigma}(\star_\alpha\circ\Phi)(J^{-1}\,\hat{\mathbf{w}}_i)\wedge(J^{-1}\,\hat{\mathbf{w}}_j)\,|J| \tag{4.45}$$

$$S_\gamma\langle \mathbf{w}_i,\mathbf{w}_j\rangle = \int_{\hat\Sigma}(\star_\gamma\circ\Phi)(\frac{1}{|J|}J^T\,d\hat{\mathbf{w}}_i)\wedge(\frac{1}{|J|}J^T\,d\hat{\mathbf{w}}_j)\,|J| \tag{4.46}$$

The 1-form mass matrix will scale as $\star_\alpha\, m^1$ while the 1-form stiffness matrix will scale as $\star_\gamma\, m^{-1}$. Finally for 2-forms, we have the following symmetric bilinear forms

$$M_\alpha\langle \mathbf{f}_i,\mathbf{f}_j\rangle = \int_{\hat\Sigma}(\star_\alpha\circ\Phi)(\frac{1}{|J|}J^T\,\hat{\mathbf{f}}_i)\wedge(\frac{1}{|J|}J^T\,\hat{\mathbf{f}}_j)\,|J| \tag{4.47}$$

$$S_\gamma\langle \mathbf{f}_i,\mathbf{f}_j\rangle = \int_{\hat\Sigma}(\star_\gamma\circ\Phi)(\frac{1}{|J|}\,d\hat{\mathbf{f}}_i)\wedge(\frac{1}{|J|}\,d\hat{\mathbf{f}}_j)\,|J| \tag{4.48}$$

The 2-form mass matrix will scale as $\star_\alpha\, m^{-1}$ while the 2-form stiffness matrix will scale as $\star_\gamma\, m^{-3}$.

In Chapter 3 we mentioned that the choice of interpolation points can directly affect the conditioning of element mass matrices. We are now ready to demonstrate this phenomenon using the interpolatory basis functions from Section 4.4 and the mass matrix bilinear forms of (4.45) and (4.47). The condition number of a matrix is defined as the ratio of its maximum and minimum eigenvalues and is used as a measure for the performance of iterative solution methods for linear systems involving the matrix (i.e. larger condition numbers mean more iterations to achieve convergence). Figure 4.16 and Figure 4.17 show plots of the condition number of single element mass matrices constructed using two different types of interpolation points: the "shifted uniform" scheme used in [78] and the Extended Chebyshev set (3.30) proposed in this dissertation. Note how the uniformly spaced interpolation points yield exponential growth of condition number as the polynomial degree of the basis is increased while the *non*-uniformly spaced points yield near logarithmic growth. In Chapter 7 we will present evidence that these local results carry over to global systems defined on a finite element mesh and hence directly affect the performance of iterative solution methods.

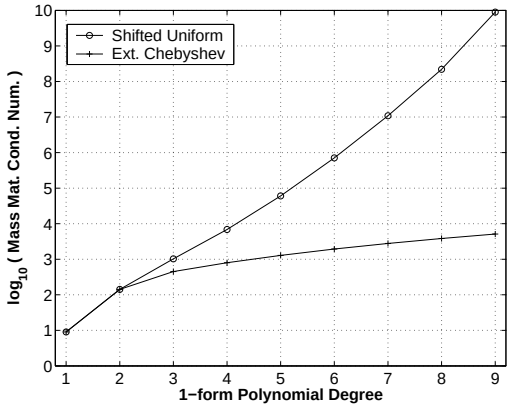


Figure 4.16: Condition number of 1-form mass matrix using two different sets of interpolation points.

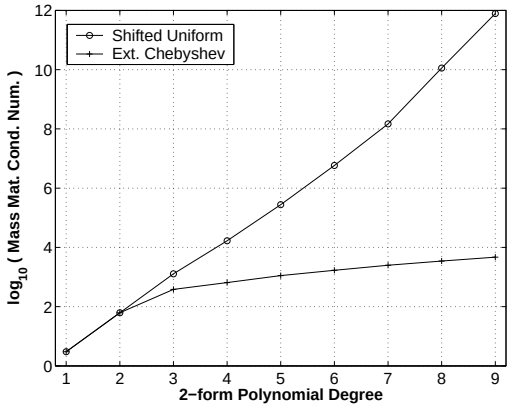


Figure 4.17: Condition number of 2-form mass matrix using two different sets of interpolation points.

### 4.6.2 Mixed Bilinear Forms

The mixed bilinear form is a very useful object which can be used to construct rectangular matrices which map members of one space of discrete differential forms to another space. The resulting rectangular matrices are in fact discrete versions of the exterior derivative. The coupled variational formulations of (3.22)

and (3.23) have terms involving both 1-form and 2-form functions; and will therefore require a mixed bilinear form. According to the exact sequence of (3.11), the curl operator links the 1-form and 2-form spaces. As such, we can construct a rectangular matrix which maps discrete 1-forms to discrete 2-forms as follows

$$C_\alpha\langle \mathbf{w}_i, \mathbf{f}_j\rangle = \int_\Sigma \star_\alpha(d\mathbf{w}_i)\wedge \mathbf{f}_j = M_\alpha\langle \mathbf{f}_i, \mathbf{f}_j\rangle\, K_{i,j} \tag{4.49}$$

resulting in a product of the 2-form mass matrix of (4.47) and a new matrix $K$ which we refer to as the topological derivative matrix. A topological derivative matrix for 1-forms and 2-forms is a discrete version of the curl operator and is independent of the element geometry, i.e. it is an incidence map between the discrete differential 1-form and 2-form degrees of freedom. Specifically, the topological derivative matrix is of the form

$$K_{i,j} = \mathcal{A}_i^2(d\mathbf{w}_j), \tag{4.50}$$

where $\mathcal{A}_i^2$ are the 2-form degrees of freedom from (4.35). In other words, we construct this matrix by projecting the exterior derivative of the 1-form basis functions from (4.13) onto the dual space of the 2-form degrees of freedom. Stated another way, we can write the exterior derivative of a 1-form as a linear combination of the 2-form basis functions. The resulting rectangular matrix contains only topological information and is independent of the mesh geometry (since the $J$ terms cancel out). It will have a number of rows equal to the dimension of the discrete 2-form basis and a number of columns equal to the dimension of the discrete 1-form basis. For the case of first order basis functions (i.e. $p = 1$), this matrix is the edge-face topological incident map commonly found in FDTD and FE methods, consisting of $\pm 1$'s and 0's [106]. Equation (4.50) is a generalization of this notion to higher-order basis functions.

### 4.6.3 Surface Bilinear Forms

The surface bilinear form is used to construct a local absorbing boundary condition (ABC) matrix for 1-form fields (such as the electric field intensity) and is of the form

$$Z\langle \mathbf{w}_i, \mathbf{w}_j\rangle = \int_{\partial\Sigma} (\hat{\mathbf{n}}\times\mathbf{w}_i)\wedge(\hat{\mathbf{n}}\times\mathbf{w}_j) \tag{4.51}$$

This bilinear form involves a surface integral over a particular face $\partial\Sigma$ of the element $\Sigma$. The ABC matrix is used to terminate boundaries for open region problems such as the radiating wave problem depicted in Figure 2.3, and is an approximation to the exact radiation boundary condition. For plane wave incidence normal to the surface of $\partial\Sigma$, the ABC matrix is exact; i.e. it will absorb 100% of the incident wave.

## 4.7 Global Assembly

Any finite element computation performed over a mesh of elements will require an *assembly* phase in which the local results on individual elements are added together to form a single linear system. When assembling a global system matrix or load vector, it is imperative that all elements which share a sub-simplex (i.e. an edge or face) agree on the ordering and possibly direction of the basis vectors associated with that sub-simplex. In this dissertation, all basis functions are defined on a single reference element. While this leads to an efficient implementation of the bases, it can lead to problems when it comes time for a global assembly process. The reason for this lies in the generality of the reference element. Given two global elements, it is not always true that they will agree on the orientation of basis vectors defined on edges and faces. If they do not agree and are assembled into the global system in this state, the resulting global matrix will be flawed and yield incorrect results. In this section, we provide a method for ensuring that any two global elements which share an edge or face will agree on their orientation by introducing an orientation standard based on the global integer IDs of an element's primary vertices (i.e. for a hexahedron, these primary vertices are simply the 8 vertices that define the corners). Given this global standard, we demonstrate how to re-orient local edges and faces on an element in order to comply with this global standard. In this section, we will denote the global coordinate system using the generic variables $(u,v,w)$.

### 4.7.1 Edge Operations

Here we define the symmetry operations of an edge. Consider an arbitrary line segment (or edge) defined by 2 generic integer IDs: $e = \{a,b\}$. We define the global $u$ orientation for this edge to be from the

*smallest* integer ID to the *largest* integer ID. We now apply the global standard to a generic edge; there are only two cases to consider:

- *Case 1 - E*: Edge remains unchanged (Identity Operation)

  $Min(e) = a; \quad e \mapsto \{a,b\}$

- *Case 2 - !E*: Edge is reversed

  $Min(e) = b; \quad e \mapsto \{b,a\}$

At most, a local edge will have to be reversed during the global assembly process in order to comply with the global standard.

### 4.7.2 Face Operations

Here we define the symmetry operations of a face. Consider an arbitrary quadrilateral face defined by four generic integer IDs: $f = \{a,b,c,d\}$. We define the global *u* orientation for this face to be from the *smallest* integer ID to its *smallest neighbor* (in a cyclical sense). The global *v* orientation for this face is defined from the *smallest* integer ID to its *largest neighbor* (in a cyclical sense). We now apply the global standard to a generic face; there are 4 cases, each with 2 sub-cases, to consider for a total of 8 distinct possibilities:

- *Case 1.1 - R0*: Rotation of 0 degrees (Identity Operation)

  $Min(f) = a$ and $MinNeighbor(a) = b; \quad f \mapsto \{a,b,c,d\}$

- *Case 1.2 - D2*: Reflection about second diagonal

  $Min(f) = a$ and $MinNeighbor(a) = d; \quad f \mapsto \{a,d,c,b\}$

- *Case 2.1 - R270*: Rotation of 270 degrees

  $Min(f) = b$ and $MinNeighbor(b) = c; \quad f \mapsto \{b,c,d,a\}$

- *Case 2.2 - V*: Reflection about vertical axis

  $Min(f) = b$ and $MinNeighbor(b) = a; \quad f \mapsto \{b,a,d,c\}$

- *Case 3.1 - R180*: Rotation of 180 degrees

  $Min(f) = c$ and $MinNeighbor(c) = d$; $\quad f \mapsto \{c,d,a,b\}$

- *Case 3.2 - D1*: Reflection about first diagonal

  $Min(f) = c$ and $MinNeighbor(c) = b$; $\quad f \mapsto \{c,b,a,d\}$

- *Case 4.1 - R90*: Rotation of 90 degrees

  $Min(f) = d$ and $MinNeighbor(d) = a$; $\quad f \mapsto \{d,a,b,c\}$

- *Case 4.2 - H*: Reflection about horizontal axis

  $Min(f) = d$ and $MinNeighbor(d) = c$; $\quad f \mapsto \{d,c,b,a\}$

These possibilities represent the 8 different symmetry operations for a square, consisting of 4 rotations and 4 reflections [107]. During the global assembly process, a local face may need to be rotated or reflected in order to comply with the global standard. The edge and face symmetry operations are summarized in Figure 4.18.

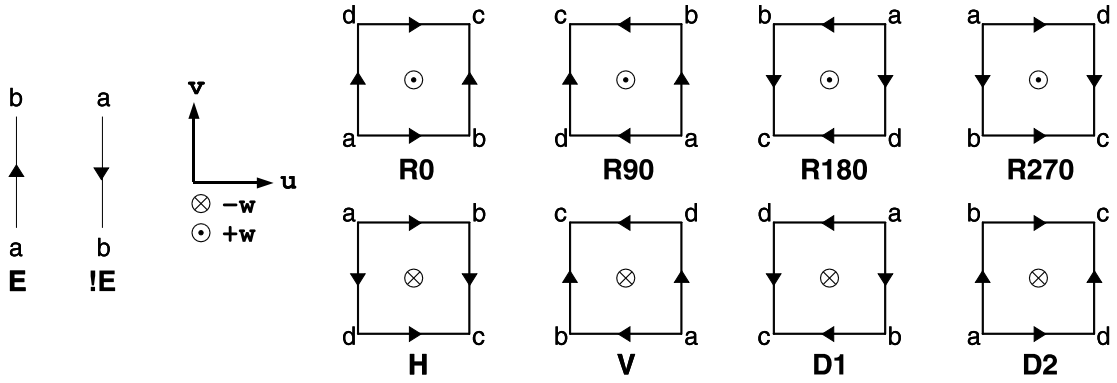


Figure 4.18: Symmetry operations for edges and faces.

### 4.7.3 Permutations

Having defined all of the possible edge and face operations, we can now describe a *permutation* process for a single element. This process will take the global IDs of an arbitrary element and construct

the local edge and face connections according to the local standards defined in Table 4.1 and Table 4.2. It will then check each local edge and face against the global standard and apply the necessary edge and face operations to enforce compliance. The specifics of this process depend on the discrete *l*-form basis and whether it is interpolatory or hierarchical. As an example, suppose two elements share an edge defined by the 2 global integer IDs 2 and 3. Furthermore, suppose that in the first element, the edge has the local orientation $e_1 = \{2,3\}$ while in the second element it has the local orientation $e_2 = \{3,2\}$. If we apply our standard to this edge then the global orientation will be $e = \{2,3\}$; thus the first element requires no change while the second requires an edge reversal. Figure 4.19 and Figure 4.20 give visual examples of this process for the case of 1-form interpolatory and hierarchical basis functions respectively.

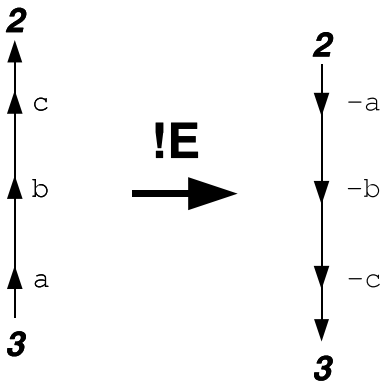


Figure 4.19: Permutation process applied to 1-form interpolatory edge basis functions of polynomial degree $p = 3$.

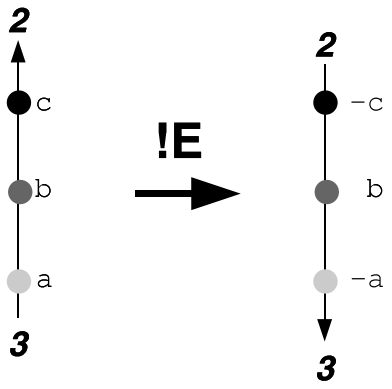


Figure 4.20: Permutation process applied to 1-form hierarchical edge basis functions of polynomial degree $p = 3$.

Now consider a 1-form interpolatory face basis of degree $p = 3$. In this case we will have 12 basis functions per face, 6 tangent to the local *u* direction and 6 tangent to the local *v* direction. Suppose two elements share a face designated by the 4 global integer IDs $2,5,8$ and 11. Furthermore, suppose that in the first element, the face has the local orientation $f_1 = \{11,8,2,5\}$ while in the second element it has the local orientation $f_2 = \{5,2,8,11\}$. If we apply our standard to this face then the global orientation will be $f = \{2,5,11,8\}$ where the IDs $\{2,5\}$ define the local *u* direction and the IDs $\{2,8\}$ define the local *v* direction. In order to comply with this global standard we must apply a rotation of

180 degrees to $f_1$ and a vertical reflection to $f_2$. Figure 4.21 gives a visual example of this process. If we label the local face basis functions (using generic IDs) with respect to the global orientation as $w = \{a,b,c,d,e,f,g,h,i,j,k,l\}$, then the face basis functions associated with the first element will be sorted as $w_1 \mapsto \{-f,-e,-d,-c,-b,-a,-l,-k,-j,-i,-h,-g\}$ while the face basis functions associated with the second element will be sorted as $w_2 \mapsto \{-c,-b,-a,-f,-e,-d,j,k,l,g,h,i\}$. The negative sign indicates that the local basis vector has changed sign as a result of the reorientation process. Figure 4.22 shows this process applied to a hierarchical face basis of degree $p = 2$.

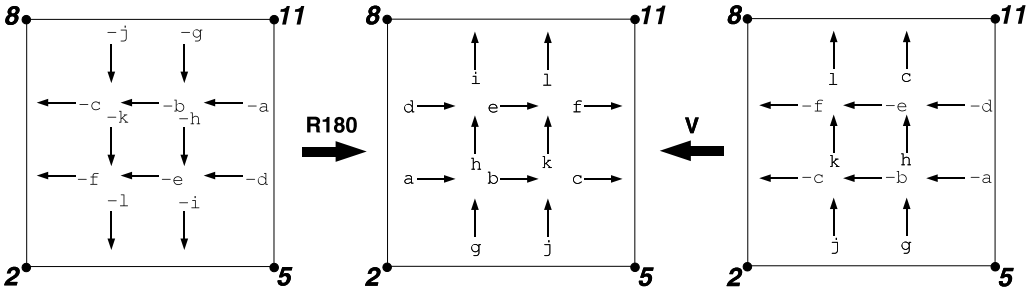


Figure 4.21: Permutation process applied to 1-form interpolatory face basis functions of polynomial degree $p = 3$. The global standard is displayed in the middle.

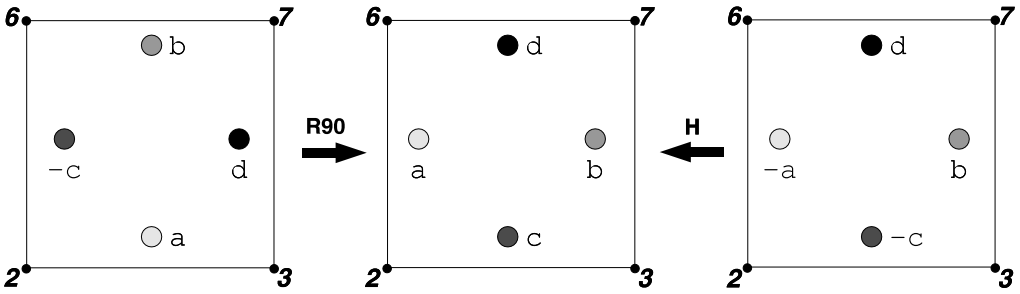


Figure 4.22: Permutation process applied to 1-form hierarchical face basis functions of polynomial degree $p = 2$. The global standard is displayed in the middle.

# Chapter 5

# Object Oriented Implementation

FEMSTER is a modular finite element class library for solving three-dimensional problems arising in electromagnetism [108]. The library was designed using a modern geometrical approach based on differential forms (or $l$-forms) and can be used for high-order spatial discretizations of well known $H(Div)$ and $H(Curl)$ conforming finite element methods [109]. The software consists of a set of abstract interfaces and concrete classes, providing a framework in which the user is able to add new schemes by reusing the existing classes or by incorporating new user-defined data types.

The philosophy of the FEMSTER library is derived from the formulation of an abstract conforming finite element method, see [93]. From the implementation point of view, such a formulation is uniquely determined by a set of 4 distinct objects $(\Sigma, \mathcal{P}, \mathcal{A}, Q)$ where:

- $\Sigma$ is a polyhedral reference element.
- $\mathcal{P}$ is a polynomial space defined on $\Sigma$ .
- $\mathcal{A}$ is the set of degrees of freedom.
- $Q$ is a numerical integration rule defined on $\Sigma$ .

This abstract formulation can be easily translated into a practical modular code by using an Object-Oriented Programming (OOP) paradigm. The C++ programming language of [110] was used in the current implemen-

tation. In this chapter we describe the classes that form the core of the FEMSTER library and give some examples of how these classes can be used in forming finite element approximations.

## 5.1 Element3D Class

The abstract class Element3D describes an interface of a reference element. It consists of a small set of methods that provide all the geometrical information needed in finite element computations, as described in Section 4.2 of Chapter 4. In Table 5.1 we present the complete interface with a brief description of each method.

| **Method** | **Description** |
|---|---|
| getOrder() | get the order of geometry |
| getNodes() | get the coordinates of the nodes |
| setNodes() | set the coordinates of the nodes |
| jacobian() | get the Jacobian matrix at a point |
| localToGlobal() | get global coordinates |
| globalToLocal() | get local coordinates |

Table 5.1: Interface of the Element3D class.

The library supports the most common types of three dimensional reference elements: tetrahedrons, hexahedrons and prisms. In Figure 5.1 we show the inheritance class diagram.

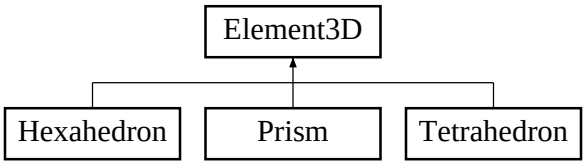


Figure 5.1: Element3D class inheritance.

The geometry of an arbitrary element is uniquely defined by the physical coordinates of its nodes, its geometrical order and a local to global mapping of the form (4.5) that transforms the reference element onto the actual element. These are commonly known in the finite element literature as sub/iso/- parametric

elements, see for example [93]. We use the term *vertex* to denote the particular nodes that define the end points of edges and the corners of faces and elements, therefore a hexahedron always has 8 vertices but may have many more nodes if it is a curved element. We also assume that every node has a unique global integer ID associated with it. All the elements of a same type and having the same geometrical order are topologically equivalent to a single reference element and can be represented through a unique object. Thus only one instantiation of a concrete reference element is needed to represent a block of elements that are topologically equivalent. Geometrical information of a particular element is obtained by first setting the coordinates of each node in the reference element, and then querying this element for the desired information.

In the following code segment we illustrate how to get the Jacobian matrix (4.8) of all the elements on a block of hexahedrons with linear geometry (i.e. $s = 1$)

```
Element3D* element = new Hexahedron(1);
R3 localPoint = R3(0.0,0.0,0.0);

for (int i = 0; i < numElementInBlock; i++) {
  R3xR3 JacobianMatrix;
  R3* nodePtr = nodeArray + i*8;
  element->setNodes(nodePtr);
  element->jacobian(localPoint,JacobianMatrix);
  ...
};
```

## 5.2 IntRule3D Class

The computation of local integrals arising in stiffness and mass matrices and local load vectors is performed numerically using high order integration rules. The abstract class IntRule3D describes a general interface for an integration rule of an arbitrary three dimensional reference element. In Table 5.2 we present the interface and in Figure 5.2 the inheritance class diagram.

The concrete classes in the bottom of Figure 5.2 are implementations of high order integration rules for the three types of elements currently available in the library. These are based on tensor products of one dimensional weighted Gauss-Jacobi quadratures. While this process allows to control the degree of exactness of a quadrature rule, in general, it may not produce optimal integration rules, i.e. with a minimum number of

| Method | Description |
|---|---|
| getNumPts() | get number of quadrature points |
| getOrder() | get order of exactness |
| getPoints() | get array of integration points |
| getWeights() | get array of integration weights |
| getRegion() | get region tag |
| getIntegral() | get approximation of the integral |

Table 5.2: Interface for integration rules in 3D.

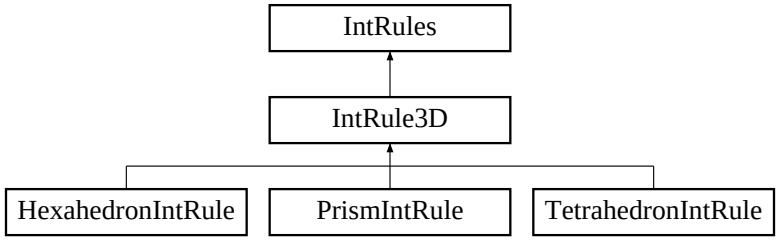


Figure 5.2: IntRule3D class inheritance.

points. However the class has been designed to be extensible, the user can provide their own integration rules if desired.

The following code segment illustrates how to compute the integral of a function $f$ on a single hexahedral element using an integration rule which is exact for polynomials of degree less than or equal to 2.

```
R3 nodes[] = {...};

Elment3D * element = new Hexahedron(1);
element->setNodes(nodes);

IntRule3D*    intRulePtr = new HexahedronIntRule(2);
int           numPoints  = intRule->getNumPts();
const R3*     point      = intRule->getPoints();
const double* weight     = intRule->getWeights();

double sum = 0.0;
for (int k = 0; k < numPoints; k++) {
  R3 x;
  element->localToGlobal(point[k],x);
  sum += f(x)*weight[k]*element->jacobian(point[k]);
};
```

Note that by using the abstract interface the same code will perform the same computation on a different type of element. The only necessary changes are the creation of the reference element and its corresponding quadrature.

## 5.3 Discrete Differential $l$-Form Class

We have a class hierarchy for each of the discrete differential $l$-form bases, the hierarchy for the 1-form class is shown in Figure 5.3. Concrete classes are presented in the lowest level of the tree. The other $l$-forms have a similar inheritance diagram. Our Silvester-Lagrange (SL) bases are similar to the bases defined in [78] which use equidistant and shifted equidistant interpolation points. The difference between our SL bases and the bases proposed in [78] is that ours are derived from the rigorous $(\Sigma, \mathcal{P}, \mathcal{A})$ definition and have well defined degrees of freedom that are scale invariant (i.e. independent of the element geometry). The uniformly spaced interpolatory bases are suitable for low order approximations, *i.e.*, $k$ = 1 to 2. However, as shown in Section 4.6, this particular choice of interpolation points produce badly conditioned mass and stiffness matrices when high order approximations are used. For this reason we have implemented spectral classes that use arbitrary sets of interpolation points, such as the Extended Chebyshev points of (3.30). The user may also experiment by passing their own set of interpolation points into the constructor of the $l$-form classes.

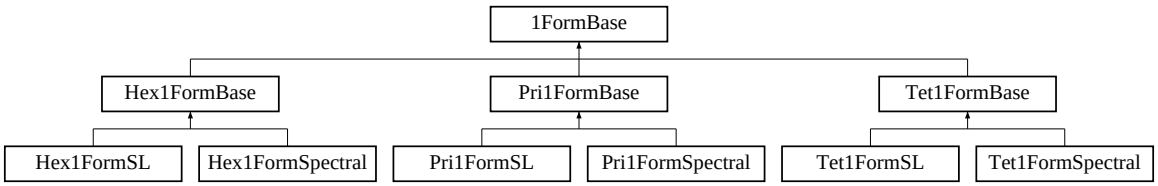


Figure 5.3: Discrete differential 1-form class inheritance.

The interface of a discrete differential $l$-form includes methods that can be used in the computation of the local matrices and vectors, such as methods to evaluate the basis functions and their derivatives at an arbitrary point, to project an arbitrary function onto the dual space spanned by the degrees of freedom, and

to compute the interpolation of an arbitrary function and its derivative. In Table 5.3 we show the general interface of an $l$-form class.

| **Method** | **Description** |
|---|---|
| getOrder() | get the order of the $l$-form |
| getDim() | get the dimension of the $l$-form |
| setElement() | set the element pointer |
| clearElement() | clear the element pointer |
| getConnectivity() | get the connectivity |
| localEvaluate() | $\hat{W}_i(\hat{x})$; $\hat{x} \in \hat{\Sigma}$; $i = 1, \ldots, n$ |
| localEvaluateD() | $d\hat{W}_i(\hat{x})$; $\hat{x} \in \hat{\Sigma}$; $i = 1, \ldots, n$ |
| globalEvaluate() | $W_i(x)$; $x \in \Sigma$; $i = 1, \ldots, n$ |
| globalEvaluateD() | $dW_i(x)$; $x \in \Sigma$; $i = 1, \ldots, n$ |
| localInterp() | $\Pi(f(\hat{x}))$; $\hat{x} \in \hat{\Sigma}$; |
| localInterpD() | $d\Pi(f(\hat{x}))$; $\hat{x} \in \hat{\Sigma}$; |
| globalInterp() | $\Pi(f(x))$; $x \in \Sigma$; |
| globalInterpD() | $d\Pi(f(x))$; $x \in \Sigma$; |
| project() | $\mathcal{A}_i(f)$ where $\Pi(f) = \sum_i \mathcal{A}_i(f)\, W_i$ |

Table 5.3: General interface for a discrete differential $l$-form class

The methods localEvaluateD() and globalEvaluateD() compute the action of a differential operator on the basis functions at a given local or global point. This operator is the exterior derivative and is uniquely determined by the $l$-form, see [111], [89] for a classical geometrical approach. In particular, this operator refers to the gradient for 0-forms; the curl for the 1-forms; and finally, the divergence for the 2-forms. To facilitate the assembly of global mass and stiffness matrices, the basis functions are locally sorted in the following order : nodal basis functions, edge basis functions, face basis functions, and interior basis functions. The getConnectivity() method returns the number of basis functions per vertex, per edge, per face, and per cell. For example, a discrete 1-form class of order $p = 2$ defined on a hexahedron element will have 0 basis functions per vertex, 2 basis functions per edge, 4 basis functions per face and 6 basis functions per cell (i.e. interior functions).

## 5.4 BilinearForms Class

The purpose of this class is to provide an interface to compute mass and stiffness matrices and load vectors and to facilitate the special cases of mixed and surface bilinear forms. As shown in (4.44), all calculations for the mass and stiffness matrices are performed on a standard reference element (i.e. the unit cube, tetrahedron, or prism). Results are then transformed to physical mesh elements (of arbitrary curvature) via the set of transformation rules of Table 4.3 and Table 4.4. Given these transformations the bases need only be evaluated on the reference element and transformed accordingly. As mentioned in Chapter 4, this specific implementation gives rise to a very computationally efficient algorithm for computing finite element approximations. For a given element topology and basis order, the basis functions only need to be computed once. Then, for every element of the same topology in the mesh, the results from the reference element can simply be mapped according to the transformation rules. This can significantly reduce computational time for a typical finite element computation. In addition, integration over the reference element is much simpler and can quite often be done exactly using Gaussian quadrature of the appropriate order. In addition, several levels of efficiency have been added in the implementation of this class. For example, the local mass and stiffness matrices are symmetric therefore only one triangular block is actually computed and the rest of the entries are copied. In Table 5.4, we show the common interface of a symmetric bilinear $l$-form.

| **Method** | **Description** |
|---|---|
| set$l$Form() | set a specific $l$-form |
| setIntRule() | set the integration rule |
| setElement() | set a specific element3D |
| initialize() | initialize internal data |
| getMassMatrix() | get the local mass matrix |
| getStiffnessMatrix() | get the local stiffness matrix |
| getLoadVector() | get local load vector |
| getUError() | get the local error |
| getQError() | get local error of derivative |

Table 5.4: Interface of the symmetric Bilinear $l$-Form class.

## 5.5 Permutation Class

As mentioned in Section 4.7 of Chapter 4, when assembling a global mass or stiffness matrix, it is imperative that all elements which share a sub-simplex agree on the ordering and possibly direction of the basis functions associated with that sub-simplex. One approach is to compute the basis functions directly on the actual element, for every element in the mesh. We disregarded this approach as it is extremely inefficient for higher-order bases. Instead, our basis functions are computed locally at the quadrature points of a reference element and then transformed as described in Section 4.4 and Section 5.4. In addition to this geometrical transformation it is necessary to perform a permutation (re-ordering) of the basis functions prior to global assembly. The purpose of this permutation is to guarantee that there exists a unique, global definition of the *i*-th basis function on a given edge or face. The details of this permutation are different for each element type, for each form and for the subsequent implementation of the from (e.g. interpolatory, hierarchical), hence the details are implemented in different concrete classes as illustrated in Figure 5.4.

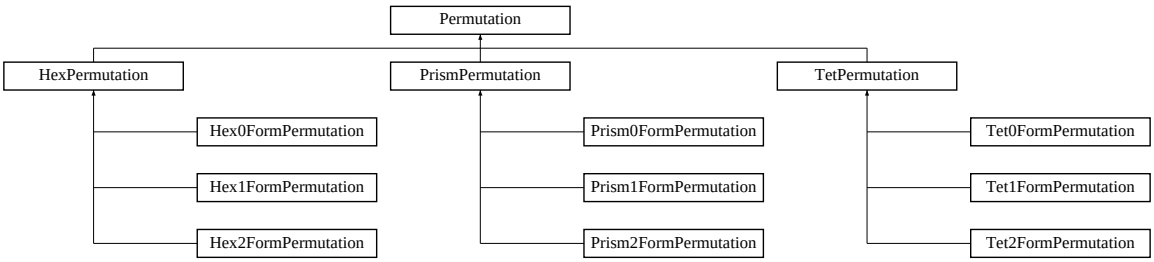


Figure 5.4: Permutation class inheritance.

The Permutation class is an abstract interface designed to allow the user to take local arrays and matrices, computed by the BilinearForms class, and reorder their contents to conform to a global standard. Our implementation of the Permutation class works only for our assumed global standard defined in Section 4.7, if a client program requires some other global standard then a new concrete Permutation class needs to be derived for this particular standard. We have adopted a standard ordering procedure for the degrees of freedom on edges and faces that is based on the global integer IDs of the vertices that define the edge or face. Our standard for edges is that an edge is directed from low global vertex ID to high global vertex ID, this

global orientation is independent of local ordering of element vertices. Our standard for faces is to define a $u-v$ coordinate system with the origin at the face vertex with lowest global vertex ID, the $u$ axis is the edge connecting the origin to the next lowest global vertex ID, and the $v$ axis is the other edge associated with the origin. The face normal is in the $u \times v$ direction. This gives every face in the mesh a global orientation that is independent of local ordering of element vertices. To summarize, the BilinearForm class computes local mass/stiffness matrices and load vectors according to a local element point of view, the Permutation class operates on these local matrices and vectors to permute the contents to the global standard. The global interface of the permutation class is shown in Table 5.5.

| **Method** | **Description** |
|---|---|
| createElementPermutation() | create permutation for a given element |
| permuteVector() | apply permutation to a vector |
| permuteMatrix() | apply permutation to a matrix |
| getNumDofPerEdge() | number of dof in edge |
| getNumDofPerFace() | number of dof in face |
| getNumDofPerCell() | number of dof in cell |
| getEdgeDofArray() | get index array of dof in edge |
| getFaceDofArray() | get index array of dof in face |
| getCellDofArray() | get index array of dof in cell |

Table 5.5: Interface of the Permutation class.

In addition to reordering local arrays and matrices, the permutation class handles the local integer IDs for the degrees of freedom associated with the sub-simplex of an element. This information is obtained by the member functions getEdgeDofArray(), getFaceDofArray() and getCellDofArray(). These methods provide the user with an integer array containing the local integer IDs for the particular sub-simplex that is queried. This information is useful when applying boundary conditions to the sub-simplices of mesh elements.

## 5.6 Example: Application of Dirichlet Boundary Conditions

The following code segment shows the usage of the Permutation class methods in the setting of Dirichlet boundary conditions to the surface of a mesh. The goal is to apply a functional value to the degrees of freedom associated with the bounding surface of a mesh. Because every discrete differential $l$-form class has an explicit project() method (i.e. an explicit formulation of $\mathcal{A}$), it is a straightforward manner to apply arbitrary functional values to the surfaces of a mesh. Since the mesh is 3-dimensional, the bounding surface is referenced by the surface faces of mesh elements. In general, a face can have degrees of freedom on its vertices, along its edges and internal to the face itself.

```
// ------------------------------------------------------------------
//    APPLICATION OF DIRICHLET BOUNDARY CONDITIONS TO A MESH SURFACE
// ------------------------------------------------------------------

for (int i = 0; i < mesh->getNumBoundaryFaces(); i++) {
  const BFaceInfo & info = mesh->getBoundaryFaceInfo(i);
  R3 nodes[8]; // enough nodes for all the element types

  int cellId = info.cellId_;
  mesh->getElement(cellId,nodes);
  element->setNodes(nodes);

  pForm->project(Source_Func,localVector);

  const int* nodeConnectivity = mesh->getNodeConnectivity(cellId);
  permutation->createElementPermutation(nodeConnectivity);
  permutation->permuteVector(localVector);
  mesh->getMapping(cellId,
                   numDofPerNode,
                   numDofPerEdge,
                   numDofPerFace,
                   numDofPerCell,
                   mapping);

  // ------------------------------------------------------------------
  //              PROCESS DOF ASSOCIATED WITH FACE NODES
  // ------------------------------------------------------------------

  int numNodes = info.numNodes_;
  for (int k = 0; k < numNodes; k++) {
    int L = info.nodeIds_[k];
    applyDirichlet(A,b,x,mapping[L],localVector[L]);
  };
```

```
  // ------------------------------------------------------------------
  //              PROCESS DOF ASSOCIATED WITH FACE EDGES
  // ------------------------------------------------------------------

  int numEdgeDof = permutation->getNumDofPerEdge(info.localId_);

  if ( numEdgeDof > 0 ) {
    int* edgeDofArray = new int[numEdgeDof];
    int numEdges = info.numEdges_;
    for (int k = 0; k < numEdges; k++) {
      permutation->getEdgeDofArray(info.edgeIds_[k],edgeDofArray);
      for (int l = 0; l < numEdgeDof; l++) {
        int L = edgeDofArray[l];
        applyDirichlet(A,b,x,mapping[L],localVector[L]);
      };
    };
  };

  // ------------------------------------------------------------------
  //             PROCESS DOF ASSOCIATED WITH FACE INTERIOR
  // ------------------------------------------------------------------

  int numFaceDof = permutation->getNumDofPerFace(info.localId_);

  if ( numFaceDof > 0 ) {
    int* faceDofArray = new int[numFaceDof];
    permutation->getFaceDofArray(info.localId_,faceDofArray);
    for (int k = 0; k < numFaceDof; k++) {
      int L = faceDofArray[k];
      applyDirichlet(A,b,x,mapping[L],localVector[L]);
    };
  };
};
```

## 5.7 Example: Vector Helmholtz Equation

The following sample code illustrates a simple driver for assembling the global mass and stiffness matrices and load vector for solving the vector Helmholtz equation in the frequency domain using the FEMSTER class library. The equation is given by

$$d(\star_\mu d\mathbf{E}) - \omega^2 \star_\varepsilon \mathbf{E} = \mathbf{f}$$

A discrete version of this equation can be written as

$$(S_\mu - \omega^2 M_\varepsilon)e = f$$

where $S_\mu$ is the 1-form stiffness matrix computed using the material property function $\star_\mu$ to represent the magnetic permeability and $M_\varepsilon$ is the 1-form mass matrix computed using the material property function $\star_\varepsilon$ to represent the dielectric properties. We assume the mesh consists of linear hexahedral elements. For this problem we use a discrete 1-form of degree 3 and the Extended Chebyshev interpolatory points of (3.30). To make the exposition clear, we have omitted all the details not related to the mathematical aspects of the finite element method.

```
// ---------------------------------------------------------------------
//                      CREATE FINITE ELEMENT OBJECT
// ---------------------------------------------------------------------

int degree = 3;

Element3D*     element = new Hexahedron(1);
p0FormBase*    pForm   = new Hex1FormSpectral(degree);
IntRule3D*     intRule = new HexahedronIntRule(2*degree+1);
Permutation*   perm    = new Hex1FormPermutation((Hex1FormBase *) pForm);
Bilinear0Form* fem     = new Bilinear1Form;

fem->setIntRule(intRule);
fem->setpForm(pForm);
fem->setElement(element);
fem->initialize();

pForm->setElement(element);

// ---------------------------------------------------------------------
//                           MESH PROCESSING PHASE
// ---------------------------------------------------------------------

int dim = pForm->getDim();

int numDofPerNode = 0;
int numDofPerEdge = 0;
int numDofPerFace[6];
int numDofPerCell = 0;

pForm->getConnectivity(numDofPerNode,
                       numDofPerEdge,
                       numDofPerFace,
                       numDofPerCell);

int numNodes = mesh->getNumNodes();
int numEdges = mesh->getNumEdges();
int numFaces = mesh->getNumFaces();
int numCells = mesh->getNumCells();
```

```
int numDofs = numNodes*numDofPerNode +
              numEdges*numDofPerEdge +
              numFaces*numDofPerFace +
              numCells*numDofPerCell;

// ------------------------------------------------------------------
//               ALLOCATE LOCAL ELEMENT MATRICES AND VECTORS
// ------------------------------------------------------------------

int*    elementMap    = new int[dim];
double* locaVector    = new double[dim];
double* localMassMat  = new double[dim*dim];
double* localStiffMat = new double[dim*dim];

// ------------------------------------------------------------------
//              ASSEMBLE GLOBAL MATRICES AND RIGHT HAND SIDE
// ------------------------------------------------------------------

CSRmat LHS;
LHS.beginAssembly(numDofs,numDofs);

for (int cellId = 0; cellId < numCells; cellId++) {
  R3 nodes[8];
  mesh->getElement(cellId,nodes);
  element->setNodes(nodes);

  fem->getLoadVector(Source_Func, localVector);
  fem->getMassMatrix(Permittivity_Func, localMassMat);
  fem->getStiffnessMatrix(Permeability_Func, localStiffMat);

  const int * nodeConnectivity = mesh->getNodeConnectivity(cellId);
  perm->createElementPermutation(nodeConnectivity);
  perm->permuteVector(localVector);
  perm->permuteMatrix(localMassMat);
  perm->permuteMatrix(localStiffMat);
  mesh->getMapping(cellId,
                   numDofPerNode,
                   numDofPerEdge,
                   numDofPerFace,
                   numDofPerCell,
                   elementMap);

  for (int k = 0; k < dim; k++) b[Map[k]] += Vec[k];
  for (int k = 0; k < dim*dim; ++k) {
    localStiffMat[k] -= (OMEGA_SQ)*localMassMat[k];
  } ;

  LHS.addSubBlock(dim,elementMap,localStiffMat);
  ...
};
```

It is important to note that after performing a permutation of the local matrices and load vector, we still need a mapping that relates local degrees of freedom with global mesh degrees of freedom. For the purposes of this example, the mapping is obtained from our generic 3D mesh class. A different mesh class could be used to perform this assembly process in parallel by distributing the elements of a mesh and the global degrees of freedom across multiple processors. In addition, once the global systems are assembled, a linear solver is required to compute the approximate values of $e$. The FEMSTER library was designed to be a modular unit that can be integrated into any global assembly / linear solver environment; thus making it very useful for massively parallel applications.

## 5.8 Example: Computation of Errors

The computation of errors can be a very useful tool for debugging purposes. They can be used to determine the accuracy of the finite element approximation whenever the exact solution of the problem is known; or to validate the numerical rates of convergence with those predicted by theoretical estimates. We have included the methods getUError() and getQError() in the bilinear form interface to compute the local error of the variable and its exterior derivative. These errors are computed in the $L_2$ norm. The following code segment illustrates the use of these methods. We assume that the linear system has already been solved and that the approximated solution is contained in $x$. Observe that after gathering the degrees of freedom of a particular element in a local vector, we need to apply the *inverse* of the permutation before we can compute the errors.

```
// ------------------------------------------------------------------
//             COMPUTE APPROXIMATION ERRORS IN THE L2 NORM
// ------------------------------------------------------------------

double errorU  = 0.0;
double errorQ  = 0.0;
for (int cellId = 0; cellId < numCells; cellId++) {
  R3 nodes[8];

  mesh->getElement(cellId,nodes);
  element->setNodes(nodes);
```

```
  // ------------------------------------------------------------------
  //        COMPUTE CONTRIBUTION OF EACH ELEMENT TO GLOBAL ERROR
  // ------------------------------------------------------------------

  const int* nodeConnectivity = mesh->getNodeConnectivity(cellId);
  permutation->createElementPermutation(nodeConnectivity);
  mesh->getMapping(cellId,
                   numDofPerNode,
                   numDofPerEdge,
                   numDofPerFace,
                   numDofPerCell,
                   mapping);

  for (int k = 0; k < dim; ++k) localVector[k] = x[mapping[k]];
  permutation->permuteVector(localVector,INVERSE_PERMUTATION);

  errorU += fem->getUError(uFunction,localVector);
  errorQ += fem->getQError(duFunction,localVector);
};
```

# Chapter 6

# High Order Temporal Discretization

In this chapter we focus our attention on the high order temporal discretization process, and we investigate the use of symplectic integration methods. Recall the spatially discretized (or semi-discrete) PDE of (4.3)

$$\begin{aligned} M_\varepsilon \frac{\partial}{\partial t} e &= K^T M_\mu\, b - M_\sigma\, e - M_\varepsilon\, j \\ \frac{\partial}{\partial t} b &= -K\, e \end{aligned}$$

This system can now be discretized in time via a finite difference method to produce a series of update steps which propagate the solutions forward in time. However, most high order numerical integration methods (e.g. Runge-Kutta, Adams-Bashforth) are dissipative. This can lead to misleading results for systems that need to be iterated for long time intervals [112], [113]. A solution is to use a symplectic time integration method that conserves energy. Therefore, in this chapter we investigate and promote the use of symplectic methods for the time integration of Maxwell's equations.

Such methods were originally developed to solve numerical systems derived from a Hamiltonian formulation and have been successfully used in the fields of astronomy and molecular dynamics where numerical accuracy and energy conservation are very important over large time integration periods [114]. Recently, these methods have been adapted for use in computational electromagnetics (CEM) in conjunction with the

finite difference method. In [115] and [116] a symplectic FDTD algorithm is presented that is implicit, 4th order accurate and valid for orthogonal 3D grids . In [117] and [118], the authors present a modified symplectic FDTD method that is up to 4th order accurate in space and time. A variation using the linear "serendipity" finite elements of [119] is also mentioned. Here, we proceed in a similar manner using high order symplectic integration methods in conjunction with the high order spatial discretization methods of Chapter 4 for use in non-orthogonal, unstructured grids [120].

## 6.1 Time Integration and Numerical Stability

To introduce the temporal discretization process, consider the very popular leap-frog method applied to (4.3) for the special case of $\sigma = 0$ (i.e. no dissipation) and $\mathbf{J} = 0$ (i.e. no source term). This is a second order accurate and energy conserving integration scheme that is commonly used in FDTD and FEM methods for CEM. We begin by approximating the first order time derivatives of the electric and magnetic fields using finite difference approximations of the form

$$\begin{aligned} \frac{\partial}{\partial t} e &\approx \frac{e_n - e_{n-1}}{\Delta t} \\ \frac{\partial}{\partial t} b &\approx \frac{b_{n+\frac{1}{2}} - b_{n-\frac{1}{2}}}{\Delta t} \end{aligned}$$

where $\Delta t$ is the discrete time step and the integer $n$ denotes the states of the fields at a particular time step. In this sense, time is discretized on a staggered grid where the 1-form electric field intensity degrees of freedom are known at whole time steps while the 2-form magnetic flux density degrees of freedom are known at half time steps. This results in the following explicit update scheme

$$\begin{aligned} e_n &= e_{n-1} + \Delta t \, M_\varepsilon^{-1} K^T M_\mu \, b_{n-\frac{1}{2}} \\ b_{n+\frac{1}{2}} &= b_{n-\frac{1}{2}} - \Delta t \, K \, e_n \end{aligned}$$

where the electric field values are updated first, then used to compute the subsequent magnetic field values. We can rewrite this system in matrix form to yield

$$\begin{bmatrix} e_n \\ b_{n+\frac{1}{2}} \end{bmatrix} = \begin{bmatrix} I & \Delta t\, M_\varepsilon^{-1} K^T M_\mu \\ -\Delta t K & I - \Delta t^2 K\, M_\varepsilon^{-1} K^T M_\mu \end{bmatrix} \begin{bmatrix} e_{n-1} \\ b_{n-\frac{1}{2}} \end{bmatrix} \tag{6.1}$$

where $I$ is the identity matrix. More generically, we can write the system as

$$\begin{bmatrix} e_n \\ b_{n+\frac{1}{2}} \end{bmatrix} = Q \begin{bmatrix} e_{n-1} \\ b_{n-\frac{1}{2}} \end{bmatrix} \tag{6.2}$$

where the matrix $Q$ is called the amplification matrix. The method is said to be stable provided it satisfies the condition [121]

$$||Q||_2 \leq 1 + O(\Delta t) \tag{6.3}$$

Note that most text book definitions of numerical stability do not include the $O(\Delta t)$ term; the necessity of which will be apparent in the next section when we discuss conservation of numerical energy. It is well known that

$$\rho(Q) \leq ||Q||_2 \tag{6.4}$$

where $\rho(Q)$ denotes the spectral radius of the amplification matrix $Q$ (i.e. its maximum eigenvalue) [122]. Thus, the relation

$$\rho(Q) \leq 1 \tag{6.5}$$

is a necessary condition for stability. The inequality of (6.5) becomes an equality if the amplification matrix $Q$ is symmetric (or is similar to a symmetric matrix). In Appendix B we prove that the amplification matrix $Q$ of (6.1) is similar to the symmetric matrix

$$\tilde{Q} = \begin{bmatrix} I - AA^T & -A \\ A^T & I \end{bmatrix} \tag{6.6}$$

Furthermore, in Appendix B we prove that the eigenvectors of the similar amplification matrix $\tilde{Q}$ form a complete eigenbasis. Therefore, the necessary condition of (6.5) becomes sufficient for stability. A tedious,

but straightforward calculation shows that (6.5) will be satisfied for the case of the second order accurate leap-frog method provided that [92]

$$\Delta t \leq \frac{2}{\sqrt{\rho(K\, M_\varepsilon^{-1} K^T M_\mu)}} \tag{6.7}$$

Note that (6.7) is essentially a Courant-Friedrichs-Lewy (CFL) [123] condition. This stability condition has a physical interpretation; for wave propagation problems, it states that the time step $\Delta t$ must be less than the time for a wave to travel halfway across one mesh element. Stated another way, the sampling frequency must be less than half the highest resonant frequency of the mesh. The stability condition of (6.7) is valid for all values of $p$, the order of the polynomial basis functions. However, as $p$ is increased, the value of $\rho(K\, M_\varepsilon^{-1} K^T M_\mu)$ will grow, thus requiring a smaller time step $\Delta t$.

## 6.2 Conservative Time Integration

For an electromagnetic problem with no physical dissipation due to conductivity or absorbing boundary conditions the total electromagnetic energy should remain constant. In this particular finite element method the instantaneous energy is the numerical version of the total energy of (2.11) stored in the electric and magnetic fields. It is computed as

$$\tilde{\mathcal{E}} = e^T M_\varepsilon\, e + b^T M_\mu\, b \tag{6.8}$$

This is similar to the discrete energy measurement of [124], where the 1-form magnetic field intensity is used instead of the 2-form magnetic flux density used in (6.8). Many time integration methods such as forward Euler, backward Euler, Runge-Kutta, Adams-Bashforth, etc. are inherently dissipative and the energy as measured by (6.8) is not conserved; given an initial condition the electromagnetic energy will decay exponentially.

It is well known that the leap frog method mentioned in the last section is both conditionally stable and non-dissipative; the energy as measured by (6.8) is conserved. Our goal is to apply higher-order energy conserving time integration methods to system (4.3). This is required to take full advantage of the higher-order finite element basis functions of Chapter 4. The resulting method is higher-order in both space and time

and will have significantly less numerical dispersion than low-order FDTD type methods, which is important for electrically large problems.

Consider a general system of ODE's, with field values $p$ and $q$ and an independent variable $t$, that is of the specific form

$$\begin{aligned} \frac{\partial}{\partial t} p &= F(q,t) \\ \frac{\partial}{\partial t} q &= G(p) \end{aligned} \tag{6.9}$$

Systems of this form have the property of being non-dissipative, i.e. the system does not lose energy as it evolves in time. Note that for Hamiltonian formulations, the system of (6.9) implies that the Hamiltonian is separable with respect to the conjugate field values. Numerical integration methods for solving system (6.9) should likewise be non-dissipative. For linear equations, such methods are typically written as an update scheme of the form

$$\begin{bmatrix} p_{n+1} \\ q_{n+1} \end{bmatrix} = Q \begin{bmatrix} p_n \\ q_n \end{bmatrix} \tag{6.10}$$

where the field values at a new state are expressed in terms of values at previous states. There are three specific cases of interest based on the matrix norm of $Q$, given by

$$||Q|| \begin{cases} > 1, & \text{unstable} \\ = 1, & \text{neutrally stable (non-dissipative)} \\ < 1, & \text{stable, dissipative} \end{cases}$$

When the eigenvalues of the update matrix all lie within the unit circle in the complex plane, the method will be stable and dissipative. Non-dissipative methods have the additional property that the eigenvalues of the update matrix all lie *on* the unit circle in the complex plane, with the additional constraint that the eignevectors are linearly independent [92] (see Appendix B for more details). The mapping is said to be symplectic if the following relation holds [125]

$$\partial Q^T \, S \, \partial Q = S \tag{6.11}$$

where

$$\partial Q = \begin{bmatrix} \partial_p \tilde{F} & \partial_q \tilde{F} \\ \partial_p \tilde{G} & \partial_q \tilde{G} \end{bmatrix}; \quad S = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$$

where $\tilde{F}$ and $\tilde{G}$ represent discretized versions of the original functions $F$ and $G$. The matrix $S$ is referred to as the symplectic matrix, where the word symplectic literally means "intertwined." Note that this definition only makes sense if the vectors of unknowns $p$ and $q$ are of the same dimension, as in the case of a Hamiltonian system where $q$ denotes the generalized coordinates and $p$ the generalized momenta. Symplectic maps are designed to preserve phase-space volume under iteration.

As a specific example, consider the case of the simple harmonic oscillator (SHO) where $F(q,t) = q$ and $G(p) = -p$. An exact solution to this simple problem is given by $p(t) = \sin(t)$ and $q(t) = \cos(t)$. We can quantify the energy of this system (i.e. a conserved or constant quantity) by the value

$$\mathcal{E} = p^2(t) + q^2(t)$$

which for this specific example is equal to 1. Applying the leap frog method to the SHO yields the following update scheme

$$\begin{bmatrix} p_n \\ q_{n+\frac{1}{2}} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\Delta t & (1-\Delta t^2) \end{bmatrix} \begin{bmatrix} p_{n-1} \\ q_{n-\frac{1}{2}} \end{bmatrix}$$

It is a straightforward calculation to show that this update scheme satisfies (6.11) and is therefore symplectic. However, it is also straightforward to show that this mapping does not conserve the exact value of $\mathcal{E}$ under iteration. This is due to the fact that symplectic maps solve *some* Hamiltonian exactly, but not the exact one of the system [114], [125]. However, as shown by Yoshida [126], the numerical value of the inexact conserved quantity $\tilde{\mathcal{E}}$ oscillates about the exact value $\mathcal{E}$ and the amplitude of this oscillation is reduced as the order of the symplectic method is increased.

To demonstrate the properties of symplectic integrators for conservative systems, we proceed to solve the SHO system numerically using both a symplectic method (the order 3 case from Table 6.1) and a non-symplectic 4th order Runge-Kutta method. In both cases, the system is propagated from $t = 0$ to $t = 250$ using a time step of $\Delta t = 0.8$ and the computed maximum global phase error will grow linearly at each time

step. Where the two cases differ is in the computation of the energy of the system. Figure 6.1 shows the computed numerical energy of the system at each time step for both methods. For the symplectic method, the numerical energy is of the form

$$\tilde{\mathcal{E}} = \delta_1 \cos(\gamma_1 \, t) \, \mathcal{E}$$

while for the non-symplectic method the energy is of the form

$$\tilde{\mathcal{E}} = \delta_2 \exp(-\gamma_2 \, t) \, \mathcal{E}$$

Figure 6.2 shows a parametric plot of the conjugate variables as a function of time. The numerical energy for the symplectic method oscillates at a fixed amplitude around the exact value, and is therefore conserved (in a time averaged sense). The energy for the non-symplectic method dissipates exponentially from the exact value, indicating spurious dampening of the system.

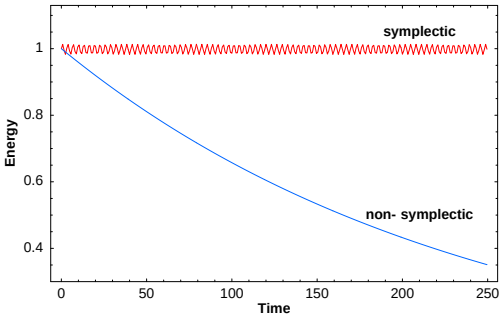


Figure 6.1: Numerical energy at each time step using a symplectic method and a non-symplectic Runge-Kutta method.

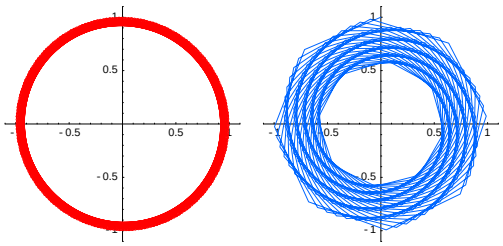


Figure 6.2: Parametric phase plots of the conjugate variables of a simple harmonic oscillator using a symplectic method (*left*) and a non-symplectic Runge-Kutta method (*right*).

Such behavior is typical of symplectic methods when applied to conservative systems, and has therefore motivated us to apply them to the particular system of (4.3). It should be noted that when a symplectic method is applied to the Maxwell system of ODEs (4.3) the result does not satisfy the symplectic property of (6.11). This is due to the fact (as mentioned previously) that the vectors $e$ and $b$ are not of the same dimension and that the matrix $K$ is rectangular. Nevertheless this does not preclude the method from being used, in fact it has been successfully used in FDTD schemes where the dimension of $e$ (the number of mesh edges) is different than the dimension of $b$ (the number of mesh faces) [115], [116]. We demonstrate

through computational experiments in Chapter 7 that high-order symplectic methods do work when applied to system (4.3) and correctly reproduce the previously mentioned features of stability, high accuracy, and no non-physical dissipation.

## 6.3 Higher Order Conservative Methods

The leap-frog method of (6.1) is second order accurate in time. Higher order conservative methods exist such as those of [112] and [127]; which were originally derived for Hamiltonian systems with applications in astrophysics and molecular dynamics. In [120], these symplectic methods are applied to finite element discretizations of Maxwell's equations of the form (4.3) using a general symplectic algorithm. In Algorithm 1, we present the inputs, procedure and outputs of the general high order symplectic integration algorithm. Numerical methods for the integration of a set of differential equations are typically characterized by the accuracy of a single step in time (the independent variable). If for some small time step $\Delta t$ the integration is performed so that it is accurate through order $\Delta t^k$, then the method is of $k$'th order. In general, a method of order $k$ will require $k$ evaluations of the functions $F$ and $G$. Therefore, as the order of the method is increased the overall computational costs will increase likewise. However, as shown in Chapter 7, higher order time integration methods can yield drastic improvements in accuracy for roughly the same computational cost as standard low order methods. The order of the method can be adjusted simply by providing the algorithm with a corresponding set of coefficients, $\alpha$ and $\beta$, each of length $k$. Table 6.1 lists exact values of the sets of coefficients for methods of order 1 through 4, as originally computed by [112] and [127].

To verify its accuracy, we apply the general high order symplectic integration algorithm to the simple harmonic oscillator example of the previous section. In Figure 6.3 we plot the computed error in the $p$ variable vs. $\Delta t$ on a log scale for each of the methods in Table 6.1 while in Figure 6.4 we plot the computed error in the $q$ variable. The slopes of these lines represent the rates of convergence of each method and hence, their orders of accuracy. Table 6.2 summarizes the various orders of accuracy for each of the symplectic methods in Table 6.1 as well as the time-staggered leap frog method. Note that the 1st order symplectic

**Algorithm 1:** General Symplectic Integration Algorithm

**input** : $k$, the order of the method
$F(b,t)$ and $G(e)$, two functions
$\alpha$ and $\beta$, two sets of coefficients
$F_0$ and $G_0$, the initial conditions
$t_0$ and $t_{fin}$, initial and final time
$\Delta t$, the time step to use

**output** : $e_{fin}$ and $b_{fin}$, the fields at time $t_{fin}$

Compute the number of time steps:
$nstep = \frac{t_{fin}-t_0}{\Delta t}$
Set initial conditions:
$e_1 \leftarrow F_0$
$b_1 \leftarrow G_0$
Begin loop over time steps:
**for** $i = 1$ ***to*** $nstep$ **do**
  Begin integration method update:
  $e_{in} \leftarrow e_i$
  $b_{in} \leftarrow b_i$
  **for** $j = 1$ ***to*** $k$ **do**
    Compute the update time for this step:
    $t_j = i\,\Delta t \; + \; \sum_{n=1}^{j-1} \alpha_n \, \Delta t$
    Update the field values:
    $e_{out} = e_{in} + \beta_j \, \Delta t \, F(b_{in}, t_j)$
    $b_{out} = b_{in} + \alpha_j \, \Delta t \, G(e_{out})$
    $e_{in} \leftarrow e_{out}$
    $b_{in} \leftarrow b_{out}$
  **end**
  Update field values for this time step:
  $e_{i+1} \leftarrow e_{out}$
  $b_{i+1} \leftarrow b_{out}$
**end**
$e_{fin} \leftarrow e_{nstep+1}$
$b_{fin} \leftarrow b_{nstep+1}$

method is 2nd order accurate in the $p$ variable while the 3rd order method is 4th order accurate in the $p$ variable. Also note that the second order symplectic method is just as accurate as the standard time-staggered leap-frog method in the field variables, but is 2nd order accurate in energy conservation. As noted by [112], the second order symplectic method is very similar to a leap-frog method.

Applying the general high order symplectic algorithm to the ODEs of (4.3) is straightforward and results in a generalized, high order update scheme of the form

$$\begin{bmatrix} e_{n+1} \\ b_{n+1} \end{bmatrix} = \left( \prod_{i=1}^{k} Q_i \right) \begin{bmatrix} e_n \\ b_n \end{bmatrix} \tag{6.12}$$

| | |
|---|---|
| **Order 1** | |
| $\alpha_1 = 1$ | $\beta_1 = 1$ |
| **Order 2** | |
| $\alpha_1 = 1/2$<br>$\alpha_2 = 1/2$ | $\beta_1 = 0$<br>$\beta_2 = 1$ |
| **Order 3** | |
| $\alpha_1 = 2/3$<br>$\alpha_2 = -2/3$<br>$\alpha_3 = 1$ | $\beta_1 = 7/24$<br>$\beta_2 = 3/4$<br>$\beta_3 = -1/24$ |
| **Order 4** | |
| $\alpha_1 = (2+2^{1/3}+2^{-1/3})/6$<br>$\alpha_2 = (1-2^{1/3}-2^{-1/3})/6$<br>$\alpha_3 = (1-2^{1/3}-2^{-1/3})/6$<br>$\alpha_4 = (2+2^{1/3}+2^{-1/3})/6$ | $\beta_1 = 0$<br>$\beta_2 = 1/(2-2^{1/3})$<br>$\beta_3 = 1/(1-2^{2/3})$<br>$\beta_4 = 1/(2-2^{1/3})$ |

Table 6.1: Symplectic Integration Coefficients for Methods of Order 1 Through 4

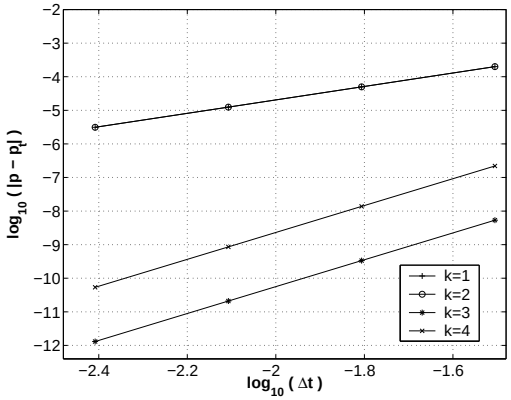


Figure 6.3: Error convergence of $|p - p_t|$ for the SHO problem using symplectic integrators of order $k = 1, 2, 3$ and 4. The computed slopes of each line are $2.00007, 2.00007, 4.00062$ and $4.00009$ respectively.

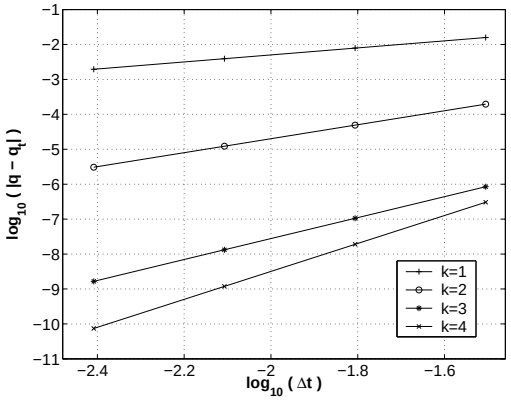


Figure 6.4: Error convergence of $|q - q_t|$ for the SHO problem using symplectic integrators of order $k = 1, 2, 3$ and 4. The computed slopes of each line are $1.00511, 2.00003, 2.99956$ and $4.00006$ respectively.

| **Method** | **Accuracy in $p$** | **Accuracy in $q$** | **Accuracy in $\mathcal{E}$** |
|---|---|---|---|
| Time-Staggered Leap-Frog | 2nd order | 2nd order | 1st order |
| Order 1 Symplectic | 2nd order | 1st order | 1st order |
| Order 2 Symplectic | 2nd order | 2nd order | 2nd order |
| Order 3 Symplectic | 4th order | 3rd order | 3rd order |
| Order 4 Symplectic | 4th order | 4th order | 4th order |

Table 6.2: Summary of the orders of accuracy for various energy conserving time integration schemes.

where $k$ is the order of the symplectic integration method and the matrices $Q_i$ are of the form

$$Q_i = \begin{bmatrix} I & \beta_i \, \Delta t \, M_\varepsilon^{-1} K^T M_\mu \\ -\alpha_i \, \Delta t K & I - \alpha_i \, \beta_i \, \Delta t^2 K \, M_\varepsilon^{-1} K^T M_\mu \end{bmatrix} \tag{6.13}$$

This results in a system amplification matrix $Q$ that is a product of the $Q_i$. For example, the third order symplectic integration method from Table 6.1 ($k = 3$) applied to (4.3) can be written in matrix form as

$$\begin{bmatrix} e_{n+1} \\ b_{n+1} \end{bmatrix} = Q_3 \, Q_2 \, Q_1 \begin{bmatrix} e_n \\ b_n \end{bmatrix}$$

In Appendix B we prove in general that each of the $Q_i$ are similar to a matrix whose eigenvectors form a complete eigenbasis. Therefore, a sufficient condition for stability of the general high order symplectic update method is

$$\rho(Q_i) \leq 1; \quad i = 1, \ldots, k \tag{6.14}$$

This in turn leads to the generalized stability condition for the high order symplectic integration method

$$\Delta t \leq \frac{2}{\sqrt{\rho(\alpha_i \, \beta_i \, K \, M_\varepsilon^{-1} K^T M_\mu)}}; \quad i = 1, \ldots, k \tag{6.15}$$

## 6.4 Conservation of Numerical Charge

### 6.4.1 Magnetic Charge

Consider the case of a source free, zero conductivity region. Magnetic charge will be conserved for all time provided that

$$d(\frac{\partial}{\partial t}\mathbf{B}) = -d(d\mathbf{E}) = 0 \tag{6.16}$$

This is simply the divergence of Faraday's law from (2.9), and based on the properties of the exterior derivative given in (3.10), it is clear that this term will always be zero. Therefore, in order for the method to conserve magnetic charge for all time, this property must be satisfied in a discrete sense. Expressing the magnetic and electric fields as basis function expansions of arbitrary polynomial degree according to (4.1), we can write (6.16) in a semi-discrete matrix form as

$$D(\frac{\partial}{\partial t}b) = -DKe$$

where the rectangular matrix $D$ is a discrete version of the divergence operator of the form $D_{i,j} = \mathcal{A}_i^3(d\mathbf{f}_j)$. This matrix is similar in construction to the discrete curl operator of (4.50), except now it is an incidence map between 2-form and 3-form degrees of freedom (i.e. for first order basis functions, this matrix is the standard face-cell topological incidence map). The discrete divergence matrix $D$ is constructed by taking the divergence of the 2-form basis functions and the term $Ke$ is simply a linear combination of the curl of 1-form basis functions. Because the Nédeléc polynomial spaces (and hence the basis functions of Chapter 4) satisfy the discrete exact sequence property of (3.27), the following matrix relation will always hold true

$$DK = 0 \tag{6.17}$$

Therefore, magnetic charge is exactly conserved (to machine precision) for all time at the semi-discrete level. For the fully discrete equation in which the time derivatives are approximated by a finite difference method, numerical charge will be conserved up to the truncation error of the time differencing scheme. For an example of the discrete Div-Curl identity of (6.17), see Section 7.2.

### 6.4.2 Electric Charge

Again, consider the case of a source free, zero conductivity region. Electric charge will be conserved for all time provided that

$$d(\star_\varepsilon \frac{\partial}{\partial t}\mathbf{E}) = 0 \tag{6.18}$$

This is simply the divergence of Ampere's law from (2.9). The discrete electric field is expressed as a linear combination of 1-form basis functions of arbitrary polynomial degree according to (4.1). As such, the element to element interfaces do not have normal continuity and the electric field is not divergence free in the classical

differential sense. Rather the field is divergence free only in the variational sense. This is required to allow for discontinuity of normal components across material interfaces. Introducing a scalar valued, piecewise continuous 0-form trial function $\phi'$, the variational form of (6.18) is

$$\int_\Omega d(\star_\varepsilon \frac{\partial}{\partial t}\mathbf{E}) \wedge \phi' = -\int_\Omega \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} \wedge d\phi' + \int_{\partial\Omega} \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} \wedge \phi'$$

Since the electric field is not required to be divergence free on the external boundary, we choose $\phi = 0$ on $\partial\Omega$ yielding the constraint

$$\int_\Omega \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} \wedge d\phi' = 0$$

This is simply the variational form of Ampere's law from (3.22) with the 1-form trial function $\mathbf{E}'$ expressed as the gradient of some scalar function $\phi'$. Setting $\mathbf{J} = \star_\sigma = 0$, we have

$$\int_\Omega \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} \wedge d\phi' = \int_\Omega \star_\mu \mathbf{B} \wedge d(d\phi')$$

Again, based on the properties of the exterior derivative given in (3.10), it is clear that the term $d(d\phi')$ will always be zero. Therefore, in order for the method to conserve electric charge for all time, this property must be satisfied in a discrete sense. This implies that the matrix relation

$$KG = 0 \tag{6.19}$$

must hold, where the rectangular matrix $G$ is a discrete version of the gradient operator of the form $G_{i,j} = \mathcal{A}_i^1(d\phi_j)$. Again, this matrix is similar in construction to the discrete curl operator of (4.50), except now it is an incidence map between 0-form and 1-form degrees of freedom (i.e. for first order basis functions, this matrix is the standard node-edge topological incidence map). The discrete gradient matrix $G$ is constructed by taking the gradient of the 0-form basis functions and the matrix $K$ is simply a linear combination of the curl of 1-form basis functions. Again, because the discrete exact sequence property is satisfied for the Nédeléc polynomial spaces, the matrix relation of (6.19) will always hold. Therefore, electric charge is conserved (to machine precision) for all time at the semi-discrete level. As before, numerical electric charge will be conserved up to the truncation error of the time differencing scheme for the fully discrete equations. For an example of the discrete Curl-Grad identity of (6.19), see Section 7.2.

## 6.5 Implicit Time Stepping and Conductivity Terms

Up until this point, we have neglected the dissipative conductivity term in (4.3). Most realistic applications will require the presence of conductive and hence, lossy materials in some or perhaps all of the problem domain. In addition, several methods for artificial wave termination in regions with infinite radiation boundary conditions require the use of artificial electric and magnetic conductivities, such as the Maxwellian perfectly matched layer (PML) technique of [128]. Typically, the absorbing PML region is defined by a series of elements which approximate a rise in electric and magnetic conductivity designed to artificially attenuate a wave before it reaches a PEC boundary condition and thus prevents non-physical reflections. Introducing the fictitious magnetic charge results in the coupled system

$$\begin{aligned} \star_\varepsilon \frac{\partial}{\partial t}\mathbf{E} &= d(\star_\mu \mathbf{B}) - \star_\sigma \mathbf{E} - \mathbf{J} \\ \frac{\partial}{\partial t}\mathbf{B} &= -d\mathbf{E} - \star_{\sigma^*}\mathbf{B} \end{aligned} \tag{6.20}$$

where the material property Hodge function $\star_{\sigma^*}$ represents the combined artificial magnetic charge and inverse magnetic permeability (i.e. $\star_{\sigma^*} = \sigma\mu^{-1}$). We can now modify the explicit update scheme of (6.1) to account for the electric and magnetic conductivity terms, resulting in the following explicit update scheme

$$\begin{aligned} e_n &= (1 - \Delta t M_\varepsilon^{-1} M_\sigma)\, e_{n-1} + \Delta t\, M_\varepsilon^{-1} K^T M_\mu\, b_{n-\frac{1}{2}} \\ b_{n+\frac{1}{2}} &= (1 - \Delta t M_{\sigma^*})\, b_{n-\frac{1}{2}} - \Delta t\, K\, e_n \end{aligned} \tag{6.21}$$

Higher order versions of this explicit scheme can be obtained simply by applying the generalized high order Symplectic algorithm of Algorithm 1. However, it is known from FDTD analysis that the explicit treatment of the conductivity term in a discretization of the coupled Ampere-Faraday laws can lead to instabilities when the conductivity term exceeds a certain threshold value. To ensure stability for all cases, the dissipative conductivity terms must be treated implicitly. For the leap-frog scheme of (6.1), this can be accomplished by splitting the conductivity terms in half and distributing them over the discrete field variables at staggered

time steps

$$\begin{aligned}(1+\frac{\Delta t}{2}M_\varepsilon^{-1}\,M_\sigma)\,e_n &= (1-\frac{\Delta t}{2}M_\varepsilon^{-1}\,M_\sigma)\,e_{n-1} + \Delta t\,M_\varepsilon^{-1}K^T M_\mu\,b_{n-\frac{1}{2}} \\ (1+\frac{\Delta t}{2}M_{\sigma^*})\,b_{n+\frac{1}{2}} &= (1-\frac{\Delta t}{2}M_{\sigma^*})\,b_{n-\frac{1}{2}} - \Delta t\,K\,e_n\end{aligned} \tag{6.22}$$

Again, higher order versions of this implicit scheme can be obtained simply by applying the generalized high order Symplectic algorithm of Algorithm 1.

As an example, consider a simple 1D FDTD analysis of EM wave propagation using the order 4 symplectic integration method from Table 6.1. The computational region is a 1D line segment that has a length of 30 units. This region is divided into discrete portions of length $dx = 0.25$ and the electric and magnetic fields are computed over this region using a simple second order accurate, staggered finite differencing of the spatial derivatives. Setting the speed of light equal to unity and choosing the discrete time step to be $\Delta t = 0.7\ dx$ results in a value of $\Delta t = 0.175$. The 1D region is truncated with a PML layer consisting of artificial electric and magnetic conductivities that increases as a function of distance. The conductivity values range from 0 to a maximum value of $\frac{3}{\Delta t}$ according to a cubic polynomial as shown in Figure 6.5.

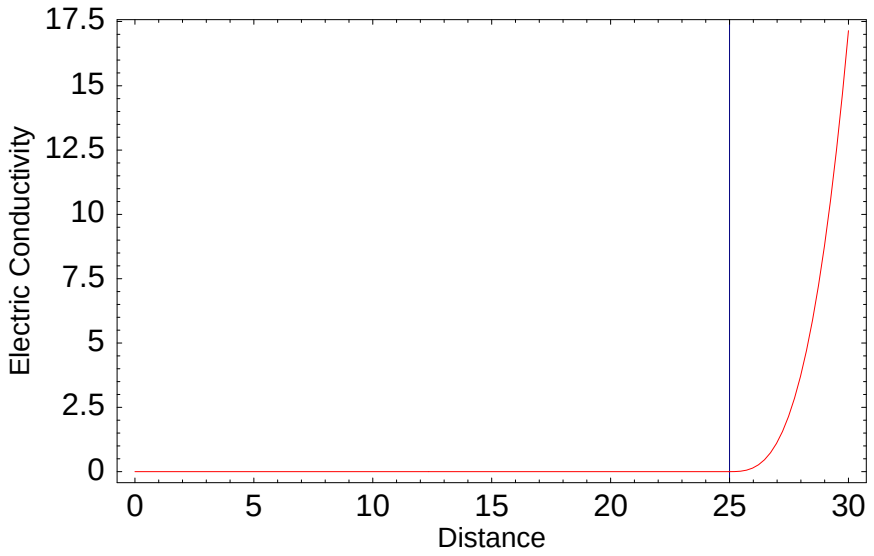


Figure 6.5: Electric conductivity as a function of propagation distance for simple 1D FDTD analysis.

An input Gaussian pulse is propagated down the 1D segment and allowed to cross into the PML region. In Figure 6.6 we show the computed electric field at various snapshots in time for the case of the

explicit 4th order symplectic integration method. Note how the computed field goes unstable as the wave impinges on the PML region. In Figure 6.7 we show the computed electric field at various snapshots in time for the case of the implicit 4th order symplectic integration method. For this case, note how the solution remains stable and the wave is attenuated in the PML region as expected.

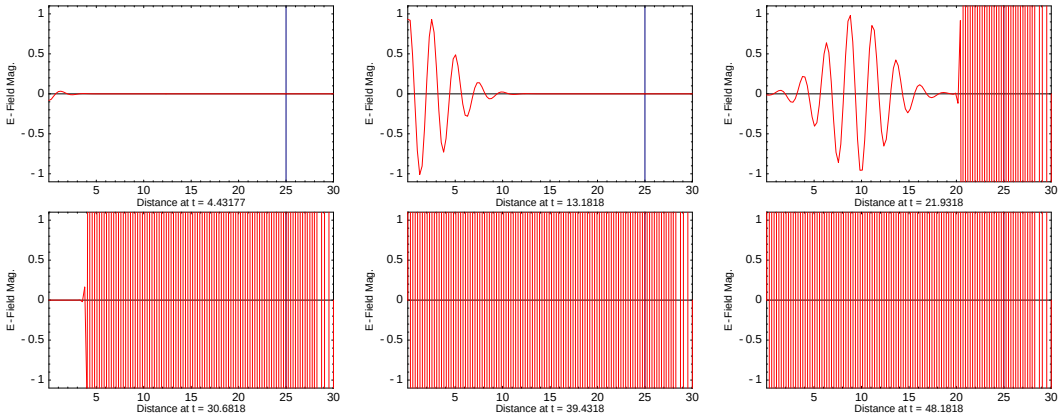


Figure 6.6: Snapshots of electric field using the explicit 4th order symplectic integration method.

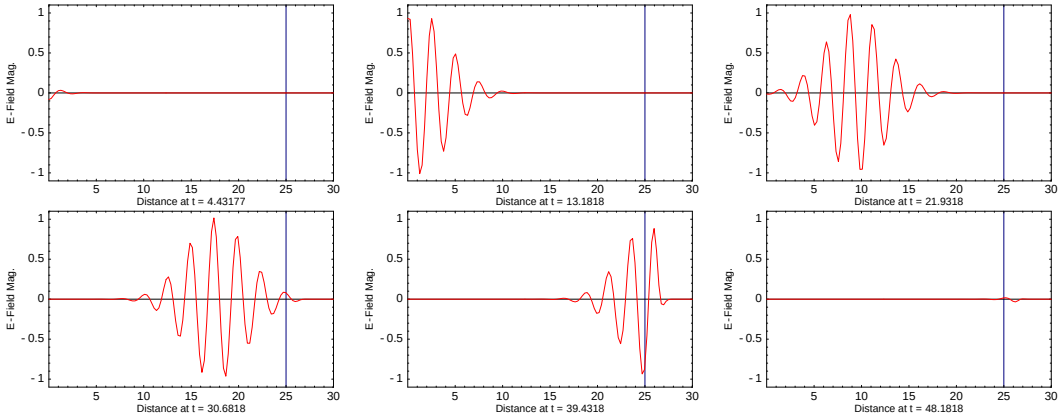


Figure 6.7: Snapshots of electric field using the implicit 4th order symplectic integration method.

# Chapter 7

# Verification

## 7.1 The Need for Rigorous Verification

In this chapter we present a series of verification experiments for EM problems with known analytic solutions. As such, these problems allow for direct quantitative as well as qualitative analysis of the method. For several reasons, this process of numerical verification is of the utmost importance for computational methods. From a software engineering perspective, verification is necessary for ensuring the quality and functionality of the computer code; i.e it is a powerful "debugging" tool. From a computational standpoint, numerical verification experiments allow for direct comparison of computed results with the predictions of theory, both physical and numerical. Much like the field of experimental physics, computational physics must correlate with theoretical numerical predictions. Oftentimes, the peak theoretical performance of a method is seldom achieved due to the complexities and numerical overhead involved in the implementation. Verification experiments allow one to quantify this discrepancy. Theoretical predictions exist for error convergence rates in time and space and for the effects of numerical dispersion.

Unlike standard low-order methods, the proposed method of this dissertation has multiple free parameters that will determine its accuracy and efficiency. Numerical verification is the only feasible way to characterize the performance of the method in order to determine optimal parameter values. In general, we have five parameters to consider when performing a finite element approximation using the proposed method: $\Delta h$, the characteristic size of the spatial discretization (i.e. element volume), $\Delta t$, the characteristic size of the temporal discretization (i.e. the discrete time step), $p$, the polynomial degree (or order) of the basis functions, $s$, the order of the element geometry and $k$, the order of the time integration method. The parameter $\Delta h$

represents the number of elements used in modeling a given problem domain; smaller values of $\Delta h$ imply more elements in the mesh and hence, more spatial accuracy. The parameter $p$ determines the level of spatial accuracy for a single element. By increasing the value of $p$, we can achieve a prescribed error tolerance with fewer elements; but by reducing the number of elements in the mesh, we may be inaccurately modeling the geometry of the problem (e.g. problems with curved surfaces), and will therefore need to increase the value of $s$, the local order of the element geometry. However, there is a significant computational cost incurred for increasing the values of $p$ and $s$. Finally, given the values of $\Delta h$, $p$ and $s$ which characterize the spatial discretization process, it is necessary to determine which values of $\Delta t$ and $k$ will yield the best accuracy in time while keeping CPU time to a minimum. A goal of this chapter is to determine values for these parameters which yield the greatest error tolerance while keeping CPU time and storage to a minimum.

Perhaps the most important reason for performing rigorous verification is based on the fundamental need for computational methods in the first place, the ability to solve problems with no known analytic solutions. When performing simulations of "real-world" physical problems which have no analytic solutions (such as those of Chapter 8), verification provides assurance that our approximations are reliable. More importantly, as we decrease the values of $\Delta h$ and $\Delta t$ and increase the values of $p, s$ and $k$, we must be assured we are converging to a real, physical solution. Verification is required to ensure the resulting computational results are trustworthy and accurately model a genuine physical phenomenon.

## 7.2 Matrix Assembly Verification

We verify the method and its components in a gradual sense by performing a series of tests which increase in complexity; as such we begin by verifying the proper assembly of global matrices over a mesh of elements. In addition, we verify the discrete exact sequence property of (3.27) for a mesh of elements. For these tests, we consider the simplest case of a two element mesh as shown in Figure 7.1, consisting of 12 unique nodes (4 of which are shared between the elements), 20 unique edges (4 shared) and 11 unique faces (1 shared). In addition, the global integer IDs of the 12 nodes are designed to exercise the permutation algorithm of Section 4.7.

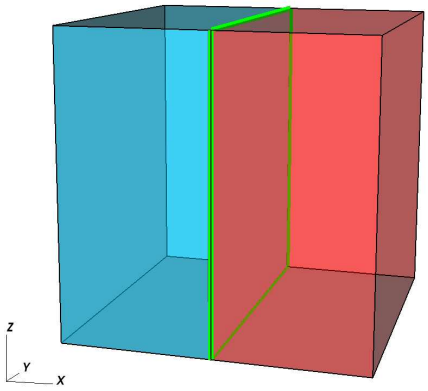


Figure 7.1: Simple two element mesh with a total of 12 nodes (4 shared), 20 edges (4 shared) and 11 faces (1 shared).

First we verify the discrete curl-gradient relation of (6.19). In Figure 7.2 we show sparse matrix plots of the non-zero entries in (6.19) for the simple two element mush using the lowest order $p = 1$ basis functions. For this case, the discrete derivative matrices are simply incidence maps between mesh nodes, edges and faces. For example, the discrete curl matrix $K$ is a rectangular 11 by 20 matrix consisting of $\pm 1$'s and 0's representing the face-edge connectivity of the mesh. In Figure 7.3 we show sparse matrix plots of the

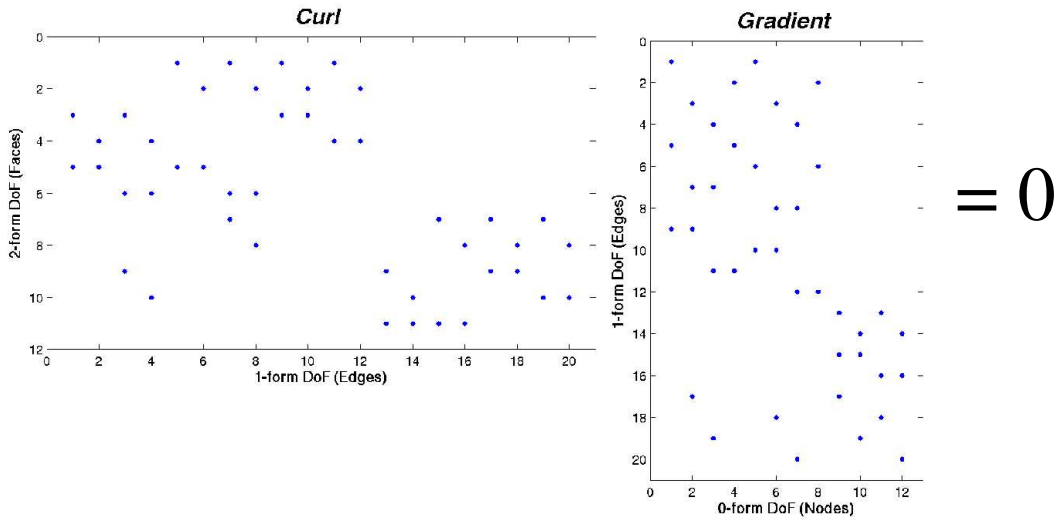


Figure 7.2: Discrete curl-gradient identity using lowest order $p = 1$ basis functions on a simple two element mesh.

non-zero entries in (6.19) for the simple two element mesh using high order $p = 3$ basis functions. In this case, the discrete derivative matrices can be viewed as incidence maps between different types of degrees of freedom. For example, the discrete curl matrix $K$ is now a rectangular 207 by 264 matrix representing the linear relation between the 2-form and 1-form degrees of freedom. The column dimension can be computed from the dimensions of the various 1-form basis function subests of (4.13) as 3 DOF per edge times 20 edges plus 12 DOF per face times 11 faces plus 36 DOF per element times 2 elements for a total of 264. The row dimension can be computed from the dimensions of the various 2-form basis function subests of (4.20) as 9 DOF per face times 11 faces plus 54 DOF per element times 2 elements for a total of 207. The non-zero entries are now floating point numbers instead of the integer values for the lowest order case, but nevertheless, the discrete identity of (6.19) holds true to machine precision. This implies that the discrete curl matrix $K$ has the correct *null* space (i.e. the set of all irrotational functions).

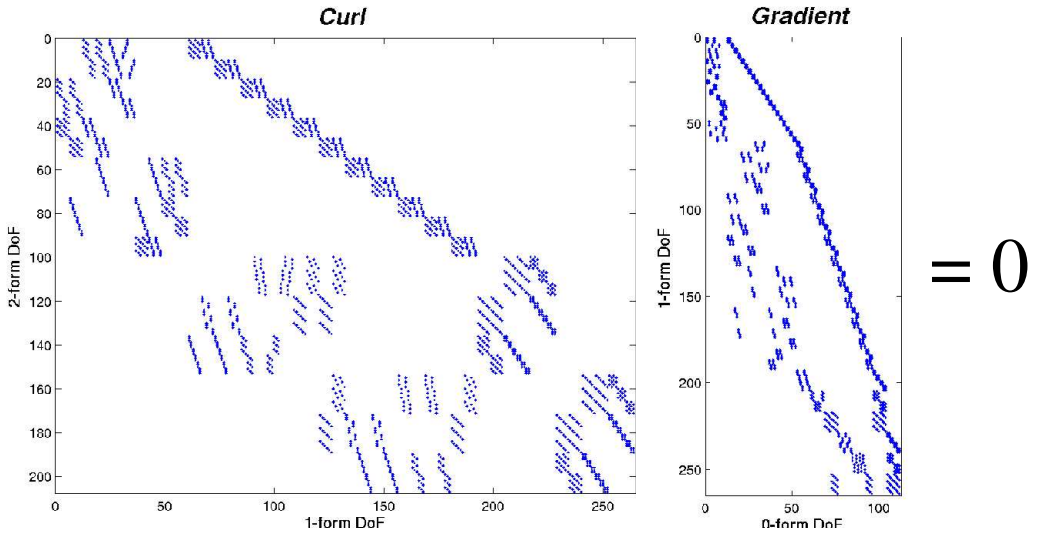


Figure 7.3: Discrete curl-gradient identity using high order $p = 3$ basis functions on a simple two element mesh.

Next we verify the discrete divergence-curl relation of (6.17). In Figure 7.4 we show sparse matrix plots of the non-zero entries in (6.17) for the simple two element mush using the lowest order $p = 1$ basis functions. Again, for the lowest order case, the discrete derivative matrices are simply incidence maps

between mesh edges, faces and elements. For example, the discrete divergence matrix $D$ is a rectangular 2 by 11 matrix consisting of $\pm 1$'s and 0's representing the cell-face connectivity of the mesh. In Figure 7.5 we show sparse matrix plots of the non-zero entries in (6.17) for the simple two element mush using high order $p = 3$ basis functions. The non-zero entries are again floating point numbers and the discrete identity of (6.17) holds true to machine precision. This implies that the discrete curl matrix $K$ has the correct *range* space (i.e. the set of all solenoidal functions).

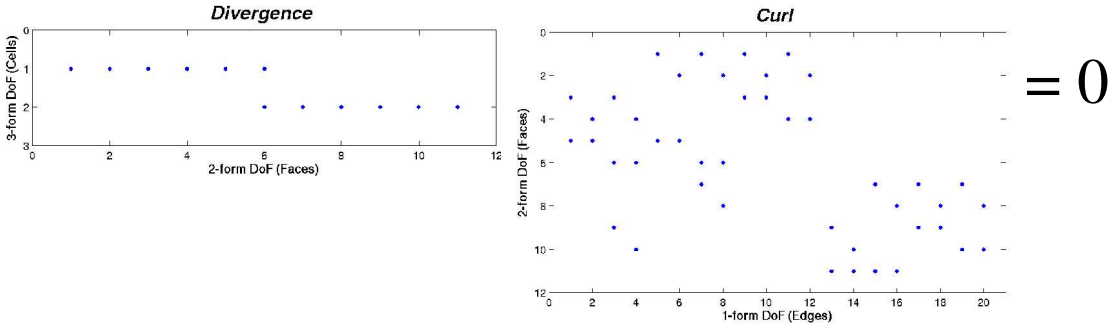


Figure 7.4: Discrete Div-Curl identity using lowest order $p = 1$ basis functions on a simple two element mesh.

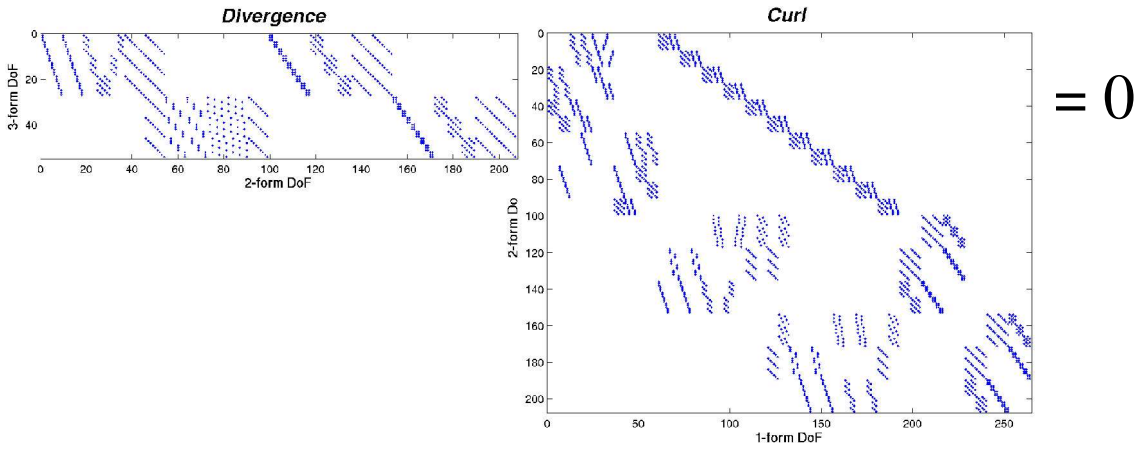


Figure 7.5: Discrete Div-Curl identity using high order $p = 3$ basis functions on a simple two element mesh.

## 7.3 Simple Frequency Domain Analysis

Here we perform a simple time-independent, frequency domain analysis. This process will be used to validate the error convergence properties of the spatial discretization process described in Chapter 4 on a global mesh of elements. We will also investigate the CPU time used in solving the resulting linear systems which make use of interpolatory basis functions.

### 7.3.1 Vector Helmholtz Equation

The vector Helmholtz equation corresponds to the second order wave equation for the electric field which can be derived from the coupled first order equation of (2.9). We assume a time harmonic electric field, thus transforming the problem to the frequency domain. In differential form we have

$$\begin{aligned} d(\star_\mu d\mathbf{E}) - \omega^2 \star_\varepsilon \mathbf{E} &= \mathbf{f} \quad \text{in } \Omega \\ \mathbf{E} \wedge \hat{n} &= \mathbf{s} \quad \text{on } \partial\Omega \end{aligned} \tag{7.1}$$

where $\mathbf{f}$ is a time harmonic current source, $\mathbf{E}$ is the time-harmonic, complex-valued 1-form field variable and $\hat{\mathbf{n}}$ is the normal direction for the surface $\partial\Omega$. In this computational experiment we validate the expected rates of convergence for both *h*-refinement and *p*-refinement by choosing a simple problem with a known, smooth solution. The computational domain is a unit cube, discretized via a series of recursively refined hexahedral meshes as shown in Figure 7.6. We choose an exact solution for the electric field identical to the testing function of (4.40), namely $\mathbf{E} = \{\sin(z), \cos(x), \exp(y)\}$, and insert this into (7.1) to form the corresponding source function $\mathbf{f}$. We then use the symmetric bilinear forms from (4.45) and (4.46) to compute the local mass and stiffness matrices and the local load vector for every element in the mesh. Given these local matrices and local vectors, the standard finite element procedure in conjunction with the process of Section 4.7 is used to assemble a global system of the form

$$(S_\mu - \omega^2 M_\varepsilon)\, e = f \tag{7.2}$$

where $S_\mu$ is the global 1-form stiffness matrix, $M_\varepsilon$ is the global 1-form mass matrix, $f$ is the global load vector, and $e$ is the unknown vector of 1-form finite element coefficients. The Dirichlet boundary condition is enforced by evaluating the solution $\mathbf{E}$ on the surface of the mesh and applying these values to the corre-

sponding surface degrees of freedom in the linear system of (7.2). The resulting linear system is then solved via a generalized minimum residual (GMRES) iterative algorithm to a residual error tolerance of $10^{-12}$ and the error of the approximate solution, $\mathbf{E}_h$, and its curl, $d\mathbf{E}_h$, are computed in the $L_2$ norm.

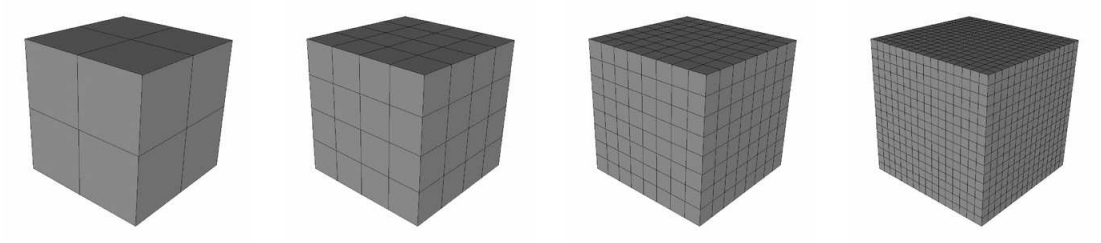

Figure 7.6: Series of recursively refined meshes of a cubic domain.

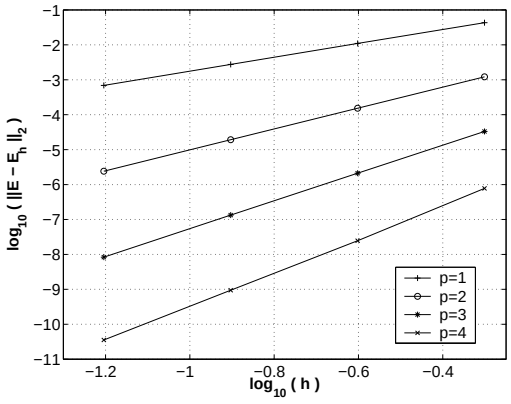


Figure 7.7: Error convergence of $||\mathbf{E}-\mathbf{E}_h||_2$ for finite element solution of discrete Helmholtz equation with 4 levels of *h*-refinement and 4 levels of *p*-refinement.

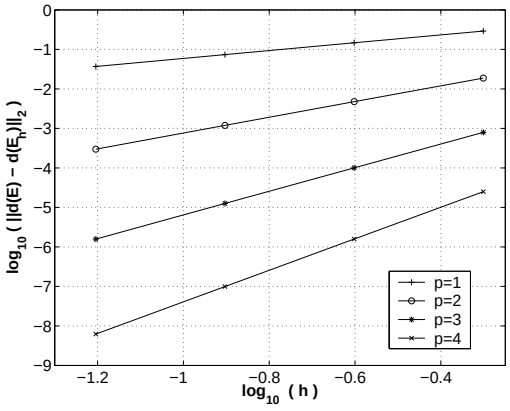


Figure 7.8: Error convergence of $||d\mathbf{E}-d\mathbf{E}_h||_2$ for finite element solution of discrete Helmholtz equation with 4 levels of *h*-refinement and 4 levels of *p*-refinement.

Figure 7.7 and Figure 7.8 show the convergence of this error for both *h*-refinement and *p*-refinement. The slopes of the lines in Figure 7.7 (based on a linear fit to the data) are $1.9899, 2.9942, 3.9864$ and $4.7947$ for $p = 1$ to 4 respectively. These results are in agreement with the theoretical predictions of [129] where an $L_2$ error estimate of $O(h^{p+1})$ is given for finite element solutions to the second order vector wave equation. The results from this numerical experiment indicate the method is converging very near the optimal rate. Note that as the computational error approaches the limit of machine precision (roughly $10^{-12}$ for a double precision floating point number), the results of this experiment begin to break down as indicated by the $p = 4$

line. The slopes of the lines in Figure 7.8 are $0.99381, 1.9936, 2.9937$ and $3.9938$ for $p = 1$ to 4 respectively. These results are in agreement with the predictions of [129] where an $L_2$ error estimate for the curl of the finite element solution is $O(h^p)$. Again the results from this numerical experiment indicate the method is converging very near the optimal rate. In Table 7.1 we summarize the results of this numerical experiment. The column "Int. Rule" indicates the order of the numerical quadrature rule that is used to compute the volume integrals of the bilinear forms. In general we use a rule of order $2p+1$ which is sufficient for computing the integral exactly whenever the geometry order of the element is $s = 1$. Note that for the $h$4-$p$1 case and the $h$2-$p$1 case, the total number of degrees of freedom is $13,872$, but the computed error for the $h$2-$p$4 case is more that 4 orders of magnitude smaller.

| | **No. Elems.** | **No. DoF** | **Int. Rule** | **No. Iterations** | $\|\|\mathbf{E}-\mathbf{E}_h\|\|_2$ | $\|\|d\mathbf{E}-d\mathbf{E}_h\|\|_2$ |
|---|---|---|---|---|---|---|
| **h1-p1** | 8 | 54 | 3 | 1 | 4.3176e-02 | 2.9161e-01 |
| **h2-p1** | 64 | 300 | 3 | 6 | 1.0973e-02 | 1.4730e-01 |
| **h3-p1** | 512 | 1,944 | 3 | 19 | 2.7553e-03 | 7.3840e-02 |
| **h4-p1** | 4,096 | 13,872 | 3 | 46 | 6.8958e-04 | 3.6944e-02 |
| **h1-p2** | 8 | 300 | 5 | 7 | 1.2162e-03 | 1.8816e-02 |
| **h2-p2** | 64 | 1,944 | 5 | 22 | 1.5346e-04 | 4.7535e-03 |
| **h3-p2** | 512 | 13,872 | 5 | 46 | 1.9229e-05 | 1.1915e-03 |
| **h4-p2** | 4,096 | 104,544 | 5 | 98 | 2.4052e-06 | 2.9808e-04 |
| **h1-p3** | 8 | 882 | 7 | 16 | 3.3213e-05 | 7.9523e-04 |
| **h2-p3** | 64 | 6,084 | 7 | 39 | 2.1173e-06 | 1.0045e-04 |
| **h3-p3** | 512 | 45,000 | 7 | 89 | 1.3323e-07 | 1.2589e-05 |
| **h4-p3** | 4,096 | 345,744 | 7 | 184 | 8.3482e-09 | 1.5746e-06 |
| **h1-p4** | 8 | 1,944 | 9 | 166 | 7.7466e-07 | 2.5056e-05 |
| **h2-p4** | 64 | 13,872 | 9 | 437 | 2.4676e-08 | 1.5823e-06 |
| **h3-p4** | 512 | 104,544 | 9 | 857 | 9.4409e-10 | 9.9141e-08 |
| **h4-p4** | 4,096 | 811,200 | 9 | 1,421 | 3.5509e-11 | 6.2103e-09 |

Table 7.1: Results for $hp$-refined finite element solutions to the vector Helmholtz equation.

In addition to this, we perform the same experiment on a fixed mesh using two different types of interpolatory 1-form basis functions (determined by the choice of interpolation points) and compare the number of iterations required to achieve a specified solver tolerance of $10^{-10}$ [99]. The goal of this experiment is to determine the iterative solver performance as a function of basis function degree and see if the results from Figure 4.16 and Figure 4.17 carry over to finite element solutions on a mesh of elements. For this

experiment we use an ILU preconditioned GMRES algorithm with a Krylov sub-space dimension (or restart length) of 2500. Figure 7.9 shows a plot of the number of iterations required to achieve the specified residual error tolerance as a function of polynomial degree for two different types of interpolatory basis functions. As expected from the results of Chapter 4, Section 4.6, the shifted-uniform interpolatory basis of [78] yields exponential growth of iteration count while the extended Chebyshev basis yields a near logarithmic growth. Note that the large jump in iteration count at $p = 6$ for the shifted-uniform basis is most likely due to the value of the restart length.

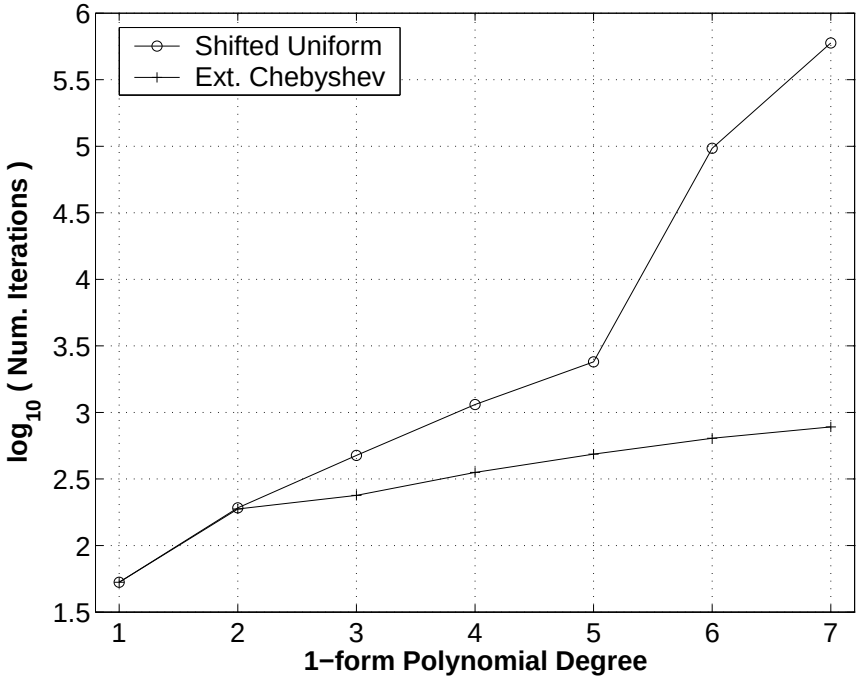


Figure 7.9: Fixed mesh iteration count vs. polynomial degree for GMRES linear solve of discrete vector Helmholtz equation using two different types of interpolatory 1-form basis functions.

### 7.3.2 Acoustic Eigenvalue Equation

Here we solve a generalized eigenvalue equation using the 2-form basis functions of Chapter 4. The acoustic eigenvalue equation is of the form

$$\begin{aligned} -d(\star_\mu d\mathbf{B}) &= \omega^2\mathbf{B} \quad \text{in } \Omega \\ \mathbf{B}\wedge\hat{n} &= 0 \quad \text{on } \partial\Omega \end{aligned} \tag{7.3}$$

where **B** represents a 2-form flux density and $\omega$ is the resonant frequency of the domain. In this computational example we compute the eigenvalues of the above equation on a fixed mesh for various values of polynomial degree $p$. We choose a problem in which the eigenmodes are known to be smooth, and thus we achieve the expected exponential convergence. The computational domain is a unit cube with the exact eigenvalues given by

$$\omega^2 = \pi^2 \left(l^2 + m^2 + n^2\right) \tag{7.4}$$

with $l, m, n \neq 0$. The domain is discretized using the 8 element hexahedral mesh from Figure 7.6. The eiegenvalue equation of (7.3) is discretized using 2-form basis functions, with the required discrete bilinear forms computed by (4.47) and (4.48). This results in a generalized linear eigenvalue problem

$$S_\mu \, b = {\omega_h}^2 M \, b \tag{7.5}$$

where $S_\mu$ and $M$ are the global 2-form stiffness and mass matrices, respectively. The vector $b$ represents the unknown coefficients of the basis function expansion of the eigenmode **B**, and $\omega_h$ is the computed resonant frequency of the eigenmode.

In this example we use Matlab to compute the entire set of eigenvalues of (7.5). The model equation of (7.3) has an infinite set of zero-valued eigenvalues, corresponding to solenoidal solutions. The discrete spectrum therefore has a large number of zero-valued (to machine precision) eigenvalues. While this is evidence that our discretization correctly models the kernel of the grad-div operator, these eigenvalues are of no interest to us, so we search the computed spectrum for the first non-zero eigenvalue which according to (7.4) should have the value $3\pi^2$. In Table 7.2 we summarize the results of this experiment for $p = 1$ to 6. The columns “Range” and “Null” correspond to the number of non-zero and zero eigenvalues respectively; there sum is equal to the total number of degrees of freedom. Also note that due to the exact sequence of (3.11) and the properties of the discrete differential form basis functions (3.27), the number of zero eigenvalues (or the null space) for this equation corresponds to the number of non-zero eigenvalues (or the range space) for the discrete 1-form Helmholtz equation. The column “No. Nonzeros” gives the total number of non-zero entries in the 2-form mass matrix.It is also interesting to point out that the filling ratio (defined as the number of

non-zero matrix entries to the total number of entries) of the mass matrix *decreases* as the polynomial degree increases. In Figure 7.10 we plot the base 10 log of the error $|\omega - \omega_h|$ of the first non-zero eigenvalue versus $p$, the degree of the finite element approximation and as expected, the error converges exponentially to zero. For very large problems in which it is not feasible to use Matlab, it is possible to develop iterative eigenvalue solvers that quickly converge to the smallest non-zero eigenvalues [130].

As before, we perform the same experiment on a fixed mesh using two different types of interpolatory 2-form basis functions and compare the number of iterations required to achieve a specified solver tolerance of $10^{-10}$. This time, the 2-form linear system is solved using a diagonally scaled preconditioned conjugate gradient (PCG) algorithm. These results are summarized in Figure 7.11. Again, note that the shifted-uniform interpolatory basis of [78] yields exponential growth of iteration count while the extended Chebyshev basis yields a near logarithmic growth.

| **Poly. Deg.** | **No. DoF** | **Range** | **Null** | **No. Nonzeros** | **Fill Ratio** | $\lvert\omega - \omega_h\rvert$ |
|---|---|---|---|---|---|---|
| $p = 1$ | 36 | 8 | 28 | 148 | 11.42% | 6.3912 |
| $p = 2$ | 240 | 64 | 176 | 5,568 | 9.67% | 0.2227 |
| $p = 3$ | 756 | 216 | 540 | 50,868 | 8.90% | 0.0040 |
| $p = 4$ | 1,728 | 512 | 1,216 | 252,928 | 8.47% | 4.0434e-5 |
| $p = 5$ | 3,300 | 1,000 | 2,300 | 892,500 | 8.20% | 2.5508e-7 |
| $p = 6$ | 5,616 | 1,728 | 3,888 | 2,524,608 | 8.01% | 1.1178e-9 |

Table 7.2: Summary of acoustic eigenvalue computations.

## 7.4 Time Domain Resonant Cavity Analysis

In these experiments, we compute the resonant modes of two different cavity geometries by directly solving the time dependent PDE of (2.9) subject to a PEC boundary condition. We begin by creating an oscillating electromagnetic field inside the cavity by applying a time dependent, vector valued current source to a random sampling of the interior DOF of the spatially discretized PDE of (4.3). The simple current source has a temporal profile equal to the first derivative of a Gaussian pulse. In addition, the pulse is randomly oriented each time it is applied to a degree of freedom; this is done to ensure that all of the modes of the cavity are excited. We then discretize this problem in time using the first order symplectic (or leap-frog) method

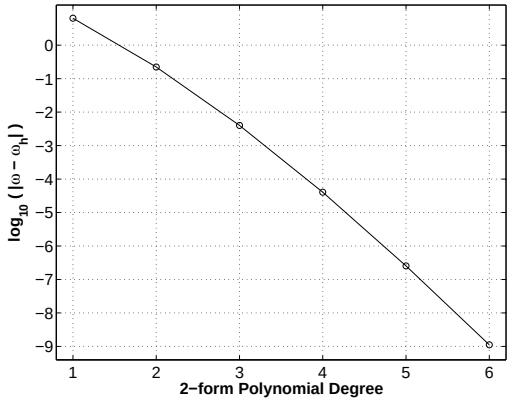


Figure 7.10: Polynomial convergence of *p*-refined solutions of the acoustic eigenvalue equation using discrete differential 2-form basis functions of degree 1 through 6.

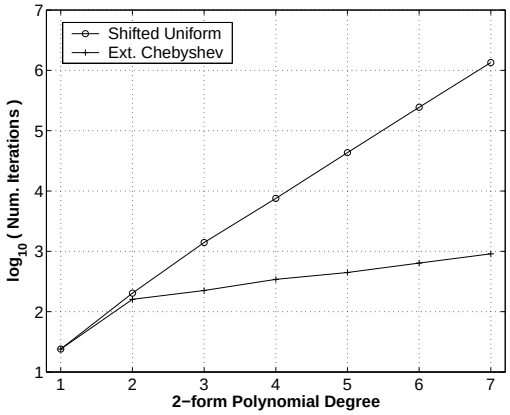


Figure 7.11: Fixed mesh iteration count vs. polynomial degree for diagonally scaled PCG linear solve of discrete acoustic equation using two different types of interpolatory 2-form basis functions.

from [120] and use a standard inverse power method to compute the largest stable time step as dictated by the stability condition of (6.7). A diagonally scaled conjugate gradient algorithm is used to invert the mass matrix at every time step. Setting the speed of light equal to unity, we let the simulation run for a physical time of 200 seconds. Upon completion, we extract the time dependent values from the discrete 1-from solution vector, a discrete version of the voltage, and Fourier transform the result to obtain both the transverse electric (TE) and transverse magnetic (TM) resonant modes of the cavity. We then compare the computed modes with their known exact values.

### 7.4.1 Cubic Cavity

We begin with the simple cavity geometry of a unit cube. The computational mesh for this problem consists of a relatively coarse $8 \times 8 \times 8$ series of hexahedral elements (the third mesh if Figure 7.6). The exact TE and TM resonant modes for a cube of this geometry are given by [131]

$$f_{l,m,n} = \frac{1}{2\pi}\sqrt{\pi^2(l^2+m^2+n^2)} \quad \text{for } l,m = 1,\ldots;\ n = 0,\ldots \tag{7.6}$$

The longer the simulation is run in time, the more accurately we can resolve the peaks of the resulting Fourier spectrum; we choose a physical time of 200 seconds which yields reasonably sharp “spikes” in the frequency domain.

In Table 7.3 we summarize the results of three different resonant cavity calculations on the same mesh, using basis functions of polynomial degree $p = 1, 2$ and 3. In Figure 7.12, Figure 7.13 and Figure 7.14 we show the respective computed Fourier spectrum for the first 27 resonant frequencies of the cavity. The vertical lines in these plots represent the locations of the exact resonant modes. Note that the vertical scale of these plots is essentially irrelevant, the height of each peak is simply a relative measure of how much this particular mode was exited by the random sampling process. Note that for the $p = 1$ case, the computed high-frequency modes are drastically "up-shifted," falling far short of their exact values. This is due the coarse nature of the mesh and the low order of approximation. As shown in Table 7.3, the computed error in all three cases increases as the frequency of the mode increases. However, the higher order methods yield a much slower rate of growth with an overall error that is orders of magnitude smaller than the standard $p = 1$ method. In addition to this, it is important to note that for all three cases presented, there are no "spurious" modes (i.e. non-physical resonant modes) or late time instabilities.

| | $p = 1$ | $p = 2$ | $p = 3$ |
|---|---|---|---|
| **No. Unknowns** | 1,194 | 13,872 | 45,000 |
| **Abs. Error in 5th Mode** | 0.03728 | 0.00104 | 0.00011 |
| **Abs. Error in 15th Mode** | 0.19809 | 0.00850 | 0.00049 |
| **Abs. Error in 25th Mode** | 0.36144 | 0.02269 | 0.00247 |

Table 7.3: Summary of cubic cavity results using the second order accurate leap-frog method.

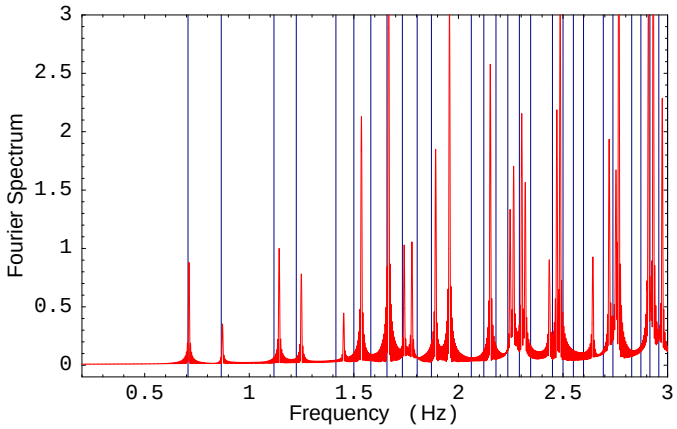


Figure 7.12: Computed resonant modes of cubic cavity using basis functions of degree $p = 1$.

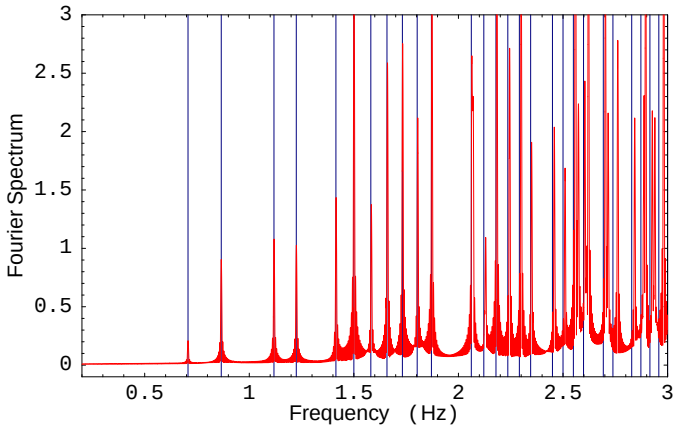


Figure 7.13: Computed resonant modes of cubic cavity using basis functions of degree $p = 2$.

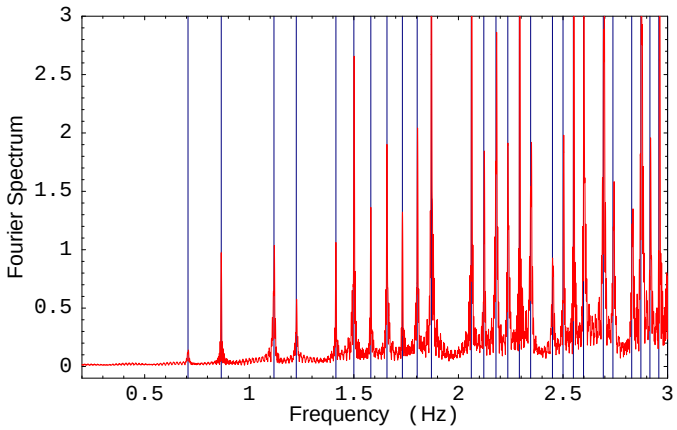


Figure 7.14: Computed resonant modes of cubic cavity using basis functions of degree $p = 3$.

In this example we demonstrate the growth of global phase error for the time integration of (4.3) using two different symplectic methods from Table 6.1. We begin by solving the general 1-form eigenvalue problem

$$S_\varepsilon \, e = \lambda M_\varepsilon \, e \tag{7.7}$$

subject to a zero flux boundary condition [130]. Here, $S_\varepsilon$ is the 1-form stiffness matrix of Chapter 4 (i.e the curl-curl matrix) and this system represents the resonant modes of the unit cube. We locate the first non-zero eigenvalue of this system (representing the first resonant mode of the cavity) and its corresponding eigenvector. Using the interpolatory 1-form basis functions from Chapter 4 of polynomial degree $p = 4$ on a coarse 8 element mesh, the first resonant mode is computed to an accuracy of $10^{-4}$. We then use the

computed eigenvector as the initial condition for the electric field in (4.3), the magnetic field will have a zero value initial condition. System (4.3) is then propagated forward in time for a total of 300 seconds (using a value of unity for the speed of light). The resulting computed electric field will be an oscillatory cosine wave with a frequency equal to the first resonant mode of the cube. We compare the global phase error in the computed solution against the exact value using both a first (leap-frog) and third order ($k = 3$) symplectic integration method. The first order method is integrated using a time step of $\Delta t = 0.005$ seconds yielding a total of $60,000$ time steps while the third order method is integrated using a time step of $\Delta t = 0.015$ seconds yielding a total of $20,000$ time steps. Since the third order method requires three linear solves per time step, the resulting computations use roughly the same total amount of CPU time to complete. The resulting global phase errors are shown in Figure 7.15 and Figure 7.16. Note that in both cases, the maximum global phase error grows linearly at each time step, but the third order method yields a much slower rate of growth with a maximum global phase error two orders of magnitude smaller than the first order method for roughly the same computational cost.

Figure 7.17 and Figure 7.18 show the computed values of the numerical energy from (6.8) at each time step for both the first and third order methods (for visual clarity, only values for the last 50 seconds are shown). Note that for both cases the numerical energy oscillates around the exact value, but for the third order case, the amplitude of this oscillation is several orders of magnitude smaller than for first order method, again for roughly the same computational cost.

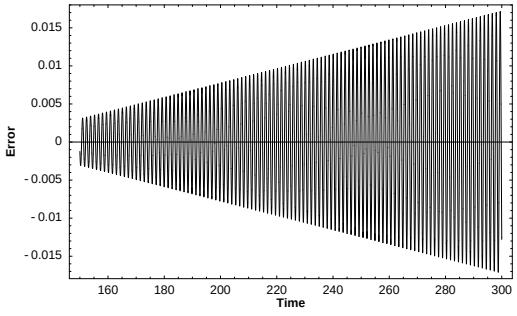


Figure 7.15: Global phase error at each time step for the first order symplectic integration method.

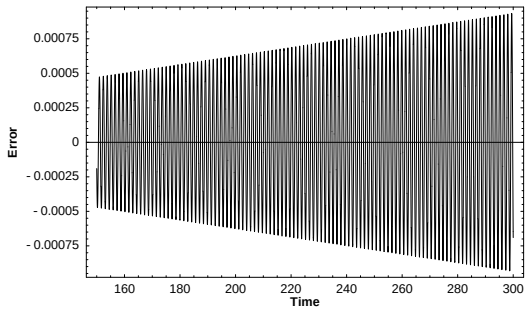


Figure 7.16: Global phase error at each time step for the third order symplectic integration method.

In this example we compute the resonant modes of the cubic cavity subject to a PEC boundary

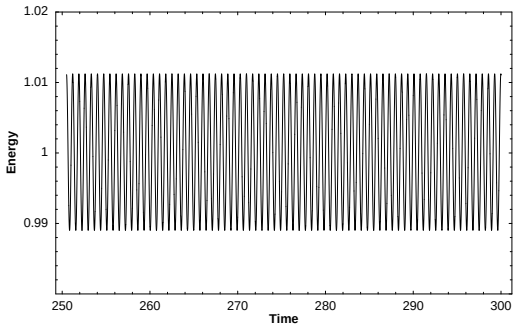


Figure 7.17: Numerical energy at each time step for the first order method.

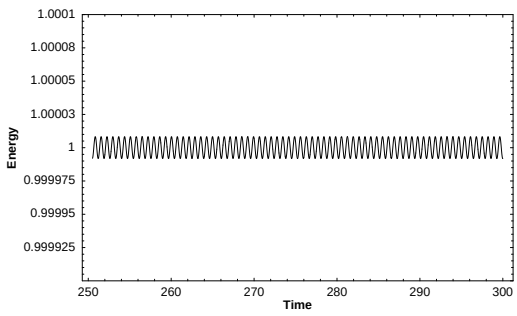


Figure 7.18: Numerical energy at each time step for the third order method.

condition using two different symplectic methods from Table 6.1 for comparison. The errors for the first 5 excited modes of the cavity are computed using both a first order (leap frog) and a third order ($k = 3$) symplectic integration method. The exact values and the computed Fourier spectrum for the case of the third order method are shown in Figure 7.19. The results for both calculations are summarized in Table 7.5. Again, note that for roughly the same computational cost, the third order method gives results that are more accurate than the first order method. We know the from eigenvalue computation of the previous example that the high order spatial discretization is capable of computing the modes to an accuracy of $10^{-4}$, and the data in Table 7.5 clearly shows this same accuracy can be achieved in the time domain only if a higher-order time integration is used.

| | **1st Order** | **3rd Order** |
|---|---|---|
| **Physical Time** | 300 sec | 300 sec |
| **Time Step** | 0.005 sec | 0.015 sec |
| **No. Steps** | 60,000 | 20,000 |
| **Avg. CPU time/step** | 0.0941365 sec | 0.297556 sec |
| **Total Run Time** | 94.1 min | 99.2 min |
| **Error in 1st Mode** | 1.3809e-3 | 1.0935e-4 |
| **Error in 2nd Mode** | 8.9125e-4 | 3.8032e-4 |
| **Error in 3rd Mode** | 5.3780e-4 | 5.3780e-4 |
| **Error in 4th Mode** | 1.5442e-3 | 2.7264e-4 |
| **Error in 5th Mode** | 3.2044e-3 | 6.1035e-4 |

Table 7.4: Comparison of results for two integration methods

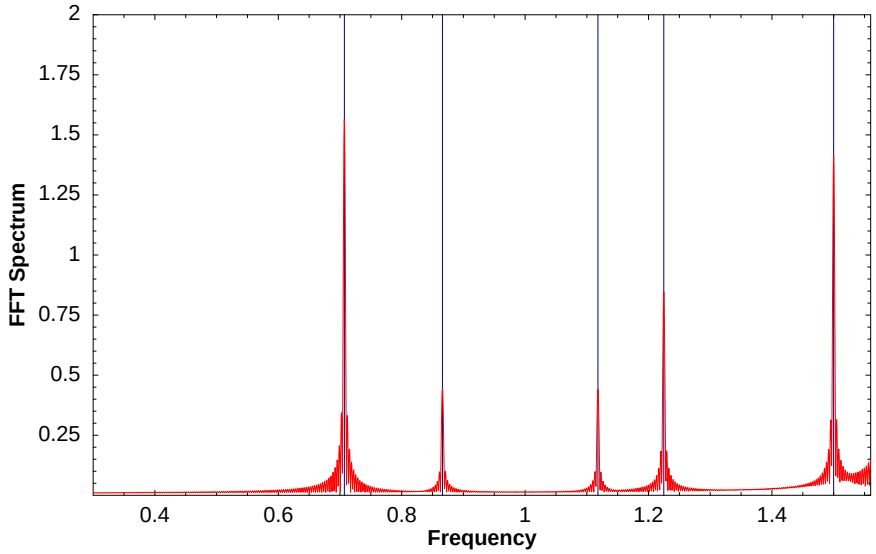


Figure 7.19: Computed resonant modes of cubic cavity using a third order symplectic method. Vertical lines represent exact values.

### 7.4.2 Spherical Cavity

In this experiment we compute the resonant modes of a spherical cavity using two different meshes: a very fine mesh (Figure 7.20) with a relatively small $\Delta h$ value and a very coarse mesh (Figure 7.21) with a large $\Delta h$ value. The fine mesh will use standard first order geometry elements ($s = 1$) while the coarse mesh will make use of curved surface elements ($s = 2$). Use of curved elements on the surface allow the mesh to be very coarse while still accurately modeling the geometric properties of the spherical surface.

Figure 7.22 and Figure 7.23 show the results of two separate calculations, one demonstrating $h$-Refinement using a discrete basis of polynomial degree $p = 1$ on the fine mesh and the other demonstrating $p$-Refinement using a discrete basis of polynomial degree $p = 3$ on the very coarse mesh with curvilinear surface elements. The results of these two calculations are summarized in Table 7.5. Qualitatively speaking, both simulations yield the same results; however, the computational costs are strikingly different. For example, to achieve a prescribed error tolerance of $10^{-3}$ in the first computed mode, using a $p$-Refinement method runs 42 times faster than a corresponding $h$-Refinement method.

We now compare the results of the linear solve performed at each time step for this spherical cavity simulation using three different types of preconditioners. Typically, the performance of a preconditioner is

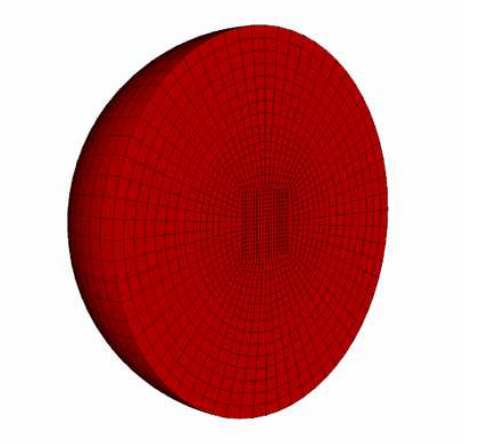

Figure 7.20: Cross section of *h*-Refined spherical mesh .

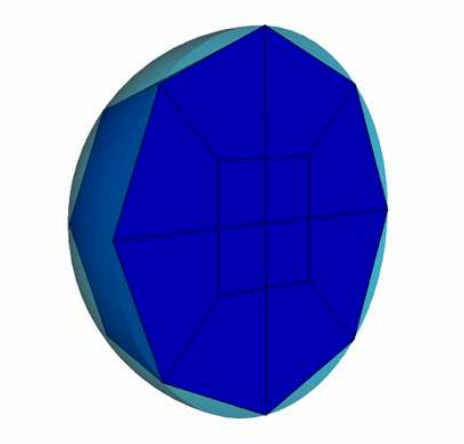

Figure 7.21: Cross section of coarse spherical mesh with curvilinear surface elements

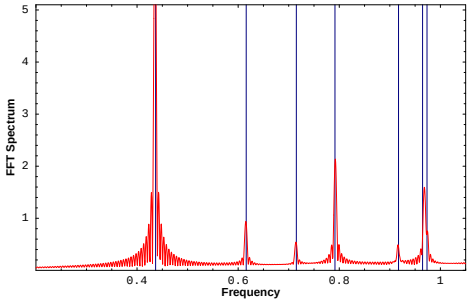


Figure 7.22: Computed resonant modes of spherical cavity using *h*-Refinement.

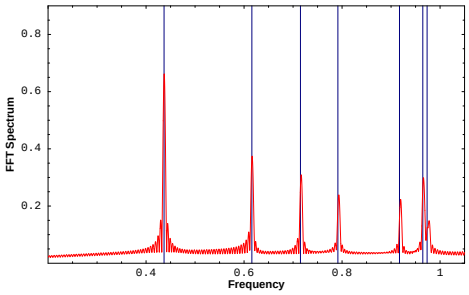


Figure 7.23: Computed resonant modes of spherical cavity using *p*-Refinement.

| | ***h*-Refinement** | ***p*-Refinement** |
|---|---|---|
| **Physical Time** | 200 sec | 200 sec |
| **Error Tol. for 1st Mode** | 1e-3 | 1e-3 |
| **Abs. Error in 1st Mode** | 7.167e-4 | 4.431e-4 |
| **No. Elements** | 28,672 | 32 |
| **No. Unknowns** | 87,632 | 2,832 |
| **No. Nonzeros** | 2,849,360 | 615,888 |
| **Fill Ratio** | 0.0371% | 7.679% |
| **Largest Stable Time Step** | 0.007 sec | 0.03 sec |
| **Number of Steps** | 28,572 | 6,668 |
| **Avg. CPU time/step** | 1.00649 sec | 0.100927 sec |
| **Total Run Time** | **479.3 min** | **11.2 min** |

Table 7.5: Comparison of computational cost for *h*-Refinement and *p*-Refinement

gauged by the number of iterations required to achieve some prescribed error tolerance. However, a reduction in iteration count does not always imply a reduction in total computational time; the cost of construction and application of the preconditioner must be taken into account as well.

For this example, the problem is discretized in space using interpolatory basis functions of degree $p = 5$ resulting in a dense 1-form mass matrix of dimension $12,640$ with a total of $9,030,800$ non-zero entries. We let the simulation run for a total of 500 time steps. In Table 7.6 we compare the results of the linear solve performed at each time step using the Silvester-Lagrange (SL) basis of [78] and the newly proposed extended Chebyshev (EC) basis using a point Jacobi (or diagonal scaling) preconditioner. This simple preconditioner requires minimal computational overhead to construct and apply, but leads to a modest reduction in iteration count. For this case, the EC basis runs roughly 11 times faster than the SL basis. In Table 7.7 we compare results for the same problem using a sparse approximate inverse preconditioner [132]; in particular, the algorithm developed by [133]. This preconditioner requires more construction and application time than point Jacobi, but leads to a more drastic reduction in iteration count; resulting in a lower total run time. In this case, the EC basis runs about 3 times faster than the SL basis. Finally, in Table 7.8 we compare the results for the same problem using a parallel ILU preconditioner [134]. Note that for this case, application of the PILU preconditioner results in essentially the same performance for each basis, substantially reducing the number of iterations required per step in comparison to the previous preconditioners. However, because of the dense nature of the linear system, construction costs and application of the preconditioner at each time step require more total CPU time than the sparse approximate inverse preconditioner.

| | **SL Basis** | **EC Basis** |
|---|---|---|
| **PreCond. Setup Time** | $\sim$0.0 sec | $\sim$0.0 sec |
| **Avg. Iterations/step** | 629 | 51 |
| **Avg. CPU time/step** | 35.9 sec | 3.27 sec |
| **Total Run Time** | **299.2 min** | **27.3 min** |

Table 7.6: Comparison of results for resonant spherical cavity simulation with point Jacobi preconditioning.

| | SL Basis | EC Basis |
|---|---|---|
| **PreCond. Setup Time** | 185.3 sec | 6.03 sec |
| **Avg. Iterations/step** | 66 | 24 |
| **Avg. CPU time/step** | 4.80 sec | 1.71 sec |
| **Total Run Time** | **43.1 min** | **14.4 min** |

Table 7.7: Comparison of results for resonant spherical cavity simulation with sparse approximate inverse preconditioning.

| | SL Basis | EC Basis |
|---|---|---|
| **PreCond. Setup Time** | 38.9 min | 39.3 min |
| **Avg. Iterations/step** | 5 | 5 |
| **Avg. CPU time/step** | 5.87 sec | 5.91 sec |
| **Total Run Time** | **87.9 min** | **88.5 min** |

Table 7.8: Comparison of results for resonant spherical cavity simulation with PILU preconditioning.

## 7.5 Guided Wave Analysis

In these computational examples we simulate the propagation of an EM wave in a coaxial guide. We use this simulation to verify several properties of the method including numerical dispersion, reflection and transmission at a dielectric interface and dissipation due to conductivity.

### 7.5.1 Numerical Dipsersion

Here we investigate the numerical dispersion properties of the method via example. It is well known that higher order methods are better at reducing the effects of numerical dispersion over standard first order *h*-refined methods [19], [20], [21], [22], [135]. For the specific case of the second order accurate leap frog method applied to time domain vector finite element solutions of Maxwell's equations (with the free space speed of light scaled to unity), the discrete dispersion relation for plane wave propagation is of the form

$$\omega^2 = (\frac{2\pi}{\lambda})^2 \left(1 + O((\frac{\Delta h}{\lambda})^{2p}) + O((\frac{\Delta t}{\lambda})^2)\right) \tag{7.8}$$

where $\lambda$ is the characteristic wavelength of the EM wave, $\omega$ is the characteristic frequency and $p$ is polynomial degree of the finite element basis functions. Thus, for a given characteristic element size $\Delta h$, an increase in the value $p$ will reduce the numerical dispersion error more than a corresponding level of $h$-refinement (i.e.

for hexahedral elements: $\Delta h \mapsto \frac{1}{8}\Delta h$).

In this example we simulate the propagation of an EM wave along a coaxial waveguide. The problem is excited with a time dependent voltage source boundary condition applied to the input cap of the mesh representing the non-dispersive TEM01 mode. The voltage source has a temporal profile equal to a ramped sine wave function and a spatial profile proportional to the inverse of the radial coordinate. A PEC boundary condition is applied to the inner and outer cylindrical walls while an absorbing boundary condition (ABC) is applied to the end cap of the mesh. An analytic (or exact) solution to this problem exists and is simply the value of the time and space dependent voltage source at the input boundary evaluated at the retarded time $t' = t - c/z$, where $c$ is the speed of light in the guide and $z$ is the propagation direction. This allows for a normed error analysis of the method, thus providing quantitative insight into the dispersion properties of the method. Scaling the speed of light equal to unity, we set the characteristic frequency of the voltage source to 0.788 while the mesh has a length of 100 units. This implies that at time $t = 100$, there will be roughly 12 full wavelengths in the coaxial mesh. Due to numerical dispersion, the computed solution will gradually get out of phase with the exact solution.

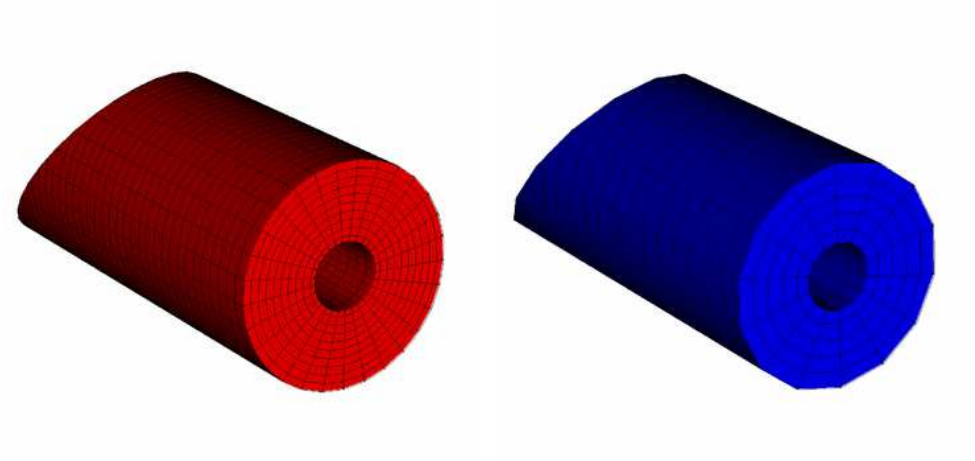

Figure 7.24: Coarse and fine coaxial waveguide meshes.

Figure 7.24 shows two meshes of the coaxial waveguide, a fine mesh ($\Delta h$) and a coarse mesh ($8\Delta h$). Figure 7.25 shows a magnitude plot of the computed electric field along with a sliced vector plot of the computed magnetic field. In Figure 7.26 we plot the maximum computed error as a function of the

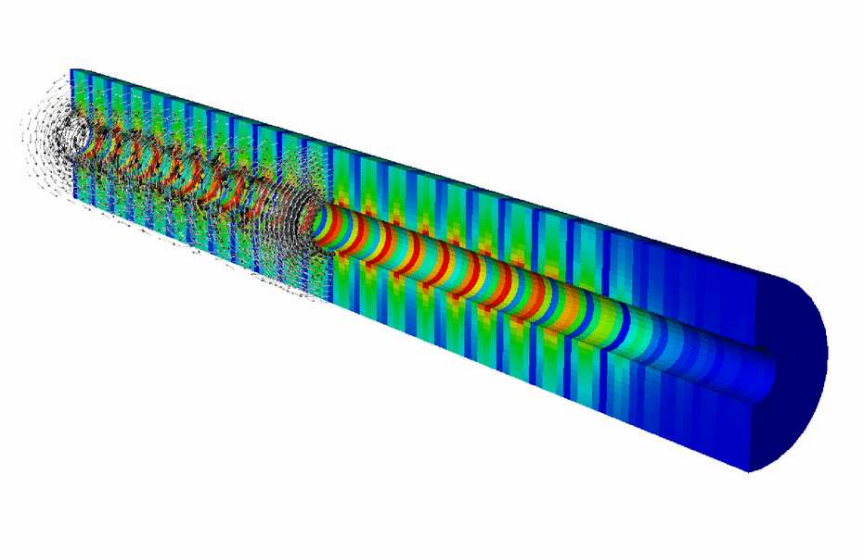

Figure 7.25: Example of the computed electric and magnetic fields for the coaxial waveguide simulation.

discrete time step for two different simulations: one using first order ($p = 1$) basis functions on the fine mesh and the other using second order ($p = 2$) basis functions on the coarse mesh with curvilinear surface elements ($s = 2$) on the inner and outer cylindrical walls. The error in the approximate electric field, $\delta = \mathbf{E} - \mathbf{E}_h$, is computed for each element in the mesh using the $L_2$ volume norm. Note that in both cases, the maximum global phase error due to numerical dispersion increases as a function of time, but the *p*-refined simulation yields a much slower rate of growth.

In Figure 7.27 we plot the base 10 log of the computed error as a function of propagation distance along the coarse mesh for a fixed time step value. We do this for the three cases $p = 1, 2$ and 3. Again, note that as $p$ is increased, the maximum value and the growth rate of the phase error due to numerical dispersion is drastically decreased. Also note that for the $p = 1$ case, the phase error begins to decrease at around $z = 60$; this is because the computed wave is now a full 180 degrees out of phase with the exact wave. It should be noted, the improved performance of *p*-refinement comes at a corresponding increase in computational cost: the total number of 1-form problem unknowns for the $p = 1$ case is $14,910$, for $p = 2$ there are $111,692$ unknowns and for $p = 3$ there are $368,466$ unknowns.

In Table 7.9 we compare the results of the linear solve performed at each time step for the SL basis

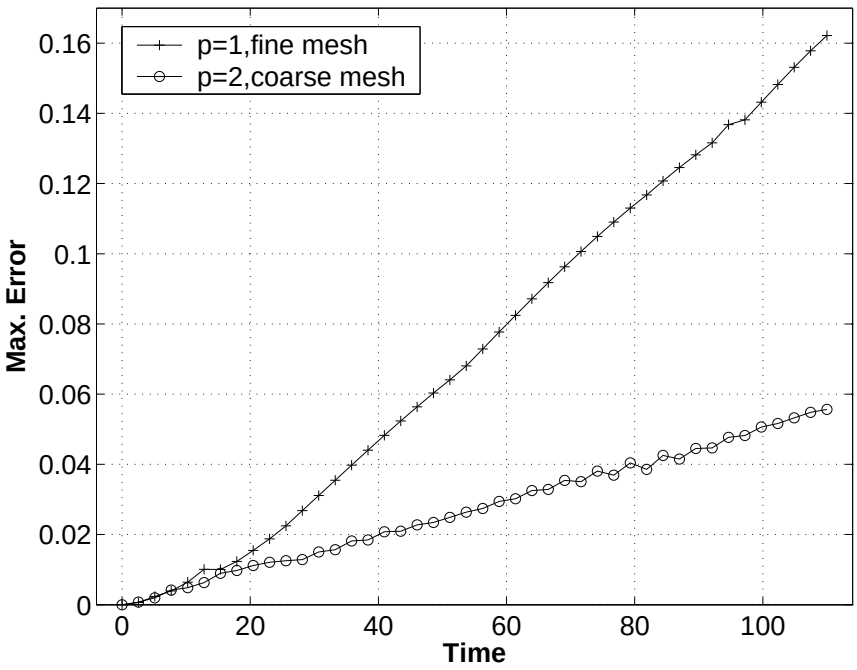


Figure 7.26: Maximum phase error at each time step for coaxial waveguide simulation.

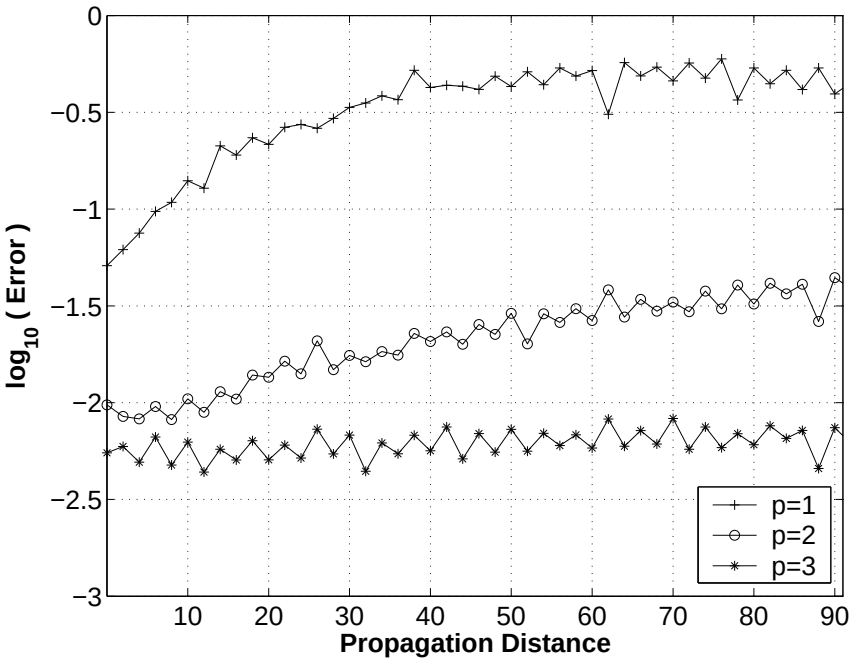


Figure 7.27: Base 10 log of computed phase error vs. propagation distance at fixed time for coaxial waveguide simulation.

| | SL Basis | EC Basis |
|---|---|---|
| **Physical Time** | 107.5 sec | 107.5 sec |
| **Time Step** | 0.05 sec | 0.05 sec |
| **No. Steps** | 2,150 | 2,150 |
| **No. Unknowns** | 80,280 | 80,280 |
| **No. Nonzeros** | 33,055,200 | 33,055,200 |
| **Avg. Iterations/step** | 225 | 35 |
| **Avg. CPU time/step** | 12.26 sec | 2.33 sec |
| **Total Run Time** | **439.3 min** | **83.5 min** |

Table 7.9: Comparison of results for coaxial waveguide simulation with point Jacobi preconditioning

and EC basis using a point Jacobi preconditioner. For this example, the coax wave guide is modeled using a very coarse 384 element mesh with second order curved elements to accurately model the geometry of the inner and outer cylindrical walls. The problem is discretized in space using interpolatory basis functions of degree $p = 4$. Note that the EC basis runs at a substantially faster rate than the SL basis.

### 7.5.2 Reflection and Transmission at a Dielectric Interface

In these numerical experiments we verify the properties of EM waves incident on a dielectric interface. We launch a single wavelength pulse down a coaxial waveguide mesh consisting of two separate dielectric regions. When a wave passes from a medium with refractive index $n_1$ to a medium with refractive index $n_2$, its wavelength will change since the frequency of radiation is fixed by the source. The relationship between the wavelengths and indices of refraction at a dielectric interface is given by

$$\frac{\lambda_2}{\lambda_1} = \frac{n_1}{n_2} \tag{7.9}$$

Therefore, if $n_2 > n_1$, the wavelength of the pulse in medium 2 will decrease. Likewise, its velocity in medium 2, given by $c_2 = \sqrt{\varepsilon_2 \mu_2}$, will decrease. Scaling the speed of light to unity, the mesh is divided into two dielectric regions of different length such that a pulse of fixed frequency will require the same net time to traverse the full length of the region. We choose $n_1 = 1$ and $n_2 = 4$ (i.e. $\varepsilon_2 = 16$) so that the speed of light in medium 2 will be 4 times slower than in medium 1. The propagation length of the mesh is 50 units with a propagation element size of $\Delta h_z = 0.5$, resulting in 40 transverse cells per wavelength in medium 1 and 10 transverse cells per wavelength in medium 2. The simulations in this section are run with a discrete time step

of $\Delta = 0.05$ for a total of 1600 time steps, enough physical time for the reflected and transmitted waves to reach the ends of the guide. Figure 7.28 shows 4 snapshots of the electric field intensity and the Poynting vector field (indicating direction of power flow) sliced parallel to the propagation direction. Note how at time $t = 40$, the wave front has arrived at the interface region at $z = 40$ as expected. At time $t = 60$, note how the initial pulse is now divided into a reflected and a transmitted wave. Finally, at time $t = 80$, note how both the reflected and transmitted pulse arrive at the the ends of the guide at the same time as expected.

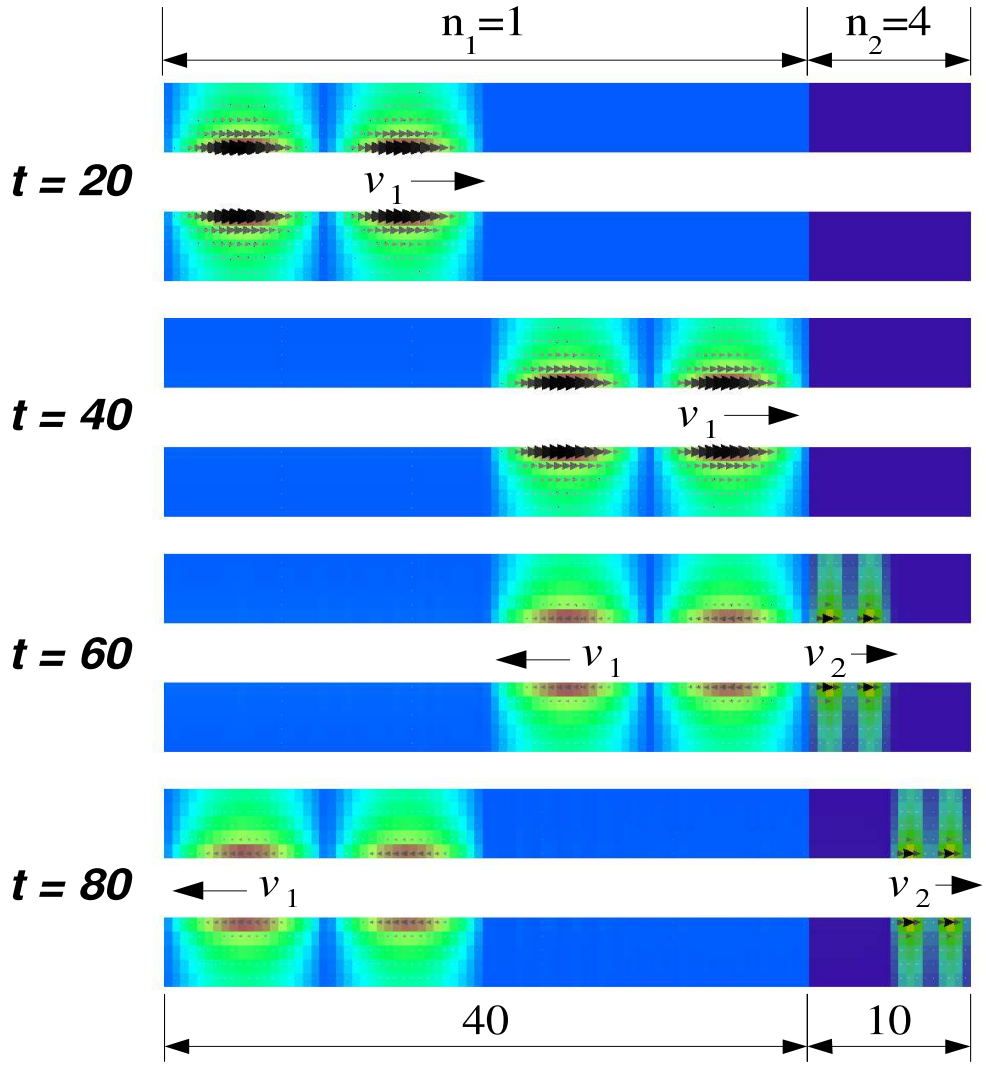


Figure 7.28: Snapshots of EM wave propagation in a coaxial waveguide with two dielectric regions. The speed of light is scaled to unity.

The reflection and transmission coefficients can be derived from the Fresnel equations [3]. For the

special case of normal incidence, these are given by

$$r_\perp = \frac{n_2 - n_1}{n_2 + n_1} \tag{7.10}$$

$$t_\perp = \frac{2n_1}{n_2 + n_1} \tag{7.11}$$

Thus for the parameters we have chosen, the reflection coefficient will be given by $r_\perp = \frac{3}{5}$ and the transmission coefficient will be given by $t_\perp = \frac{2}{5}$. The ratio of the reflected pulse amplitude to the transmitted pulse amplitude will therefore be $r/t = \frac{3}{2}$. The refelcted energy of the pulse is known as the reflectance and for the special case of normal incidence, this is given by

$$R_\perp = \left| \frac{E_{ref}}{E_{inc}} \right|^2 = r_\perp^2 \tag{7.12}$$

In Table 7.10 we compare the computed $r/t$ ratio and reflectance using a standard $p = 1$ basis and a high order $p = 2$ basis. For these measurements, the discrete reflectance is computed using the incident and reflected numerical energy as computed by (6.8). Note that the computed $r/t$ and reflectance values are much closer to the expected theoretical values when using a higher order basis.

| | **Expected Value** | $p = 1$ | $p = 2$ |
|---|---|---|---|
| $r/t$ | 1.5 | 1.532 | 1.499 |
| $R_\perp$ | 0.36 | 0.377885 | 0.36071 |

Table 7.10: Computed $r/t$ ratio and reflectance using a low order $p = 1$ basis and a high order $p = 2$ basis.

### 7.5.3 Artificial Conductivity and Absorbing Layers

In this section we investigate the properties of absorbing layers which are used to truncate the boundaries of a mesh for open region or infinite boundary problems such as those discussed in Section 6.5 of Chapter 6. Such techniques are designed to attenuate the numerical wave as it propagates through regions of artificial electric and magnetic conductivities, and are typically referred to as perfectly matched layers (PML) [14]. For finite element solutions to Maxwell's equations, we employee a variation on the PML known as a Maxwellian absorbing layer [128].

For these numerical experiments, we launch a 4 wavelength pulse down a coaxial waveguide mesh consisting of two separate regions, a lossless propagation region and a PML region consisting of electric and magnetic conductivities. The guide (and hence the direction of propagation) is oriented along the $z$-axis. In the PML region, the electric and magnetic conductivities from (6.20) are an isotropic tensor function of space given by

$$\star\sigma = \star\sigma^* = \begin{bmatrix} c_0 + c_1 z + c_2 z^2 + c_3 z^3 & 0 & 0 \\ 0 & c_0 + c_1 z + c_2 z^2 + c_3 z^3 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{7.13}$$

In other words, the conductivities are cubic polynomial functions of the propagation distance $z$ applied to the $x$ and $y$ components of the electric and magnetic vector fields.

For standard low order methods, the PML region consists of a series of elements which represent a piecewise linear rise in conductivity according to (7.13). In Table 7.11 we summarize the results of three different numerical experiments involving the use of a PML region to absorb a propagating pulse traveling down a 30 unit length propagation region. The simulations in this section are run with a discrete time step of $\Delta = 0.05$ for a total of 1100 time steps and are solved using the implicit time differencing scheme of (6.22). The maximum conductivity for each simulation is $\frac{3}{\Delta t} = 60$ which is sufficient for attenuation as demonstrated in Section 6.5.

| | $p=1$ | $p=2$ | $p=3$ |
|---|---|---|---|
| **PML Length** | 5 | 5 | 1 |
| **No. PML Layers** | 10 | 10 | 1 |
| **Layer Length** | 0.5 | 0.5 | 1 |
| **Measured Reflection:** $\frac{\lvert\mathbf{E}\rvert_{ref}}{\lvert\mathbf{E}\rvert_{max}}$ | 2.283e-3 | 7.068e-4 | 3.957e-3 |

Table 7.11: Summary of results for absorbing PML regions.

In the first two experiments, the PML region is 5 units long and consists of 10 layers. The conductivity profile used for these simulations is shown in Figure 7.29. We perform the simulation using a standard low order $p = 1$ method and a high order $p = 2$ method. Note how the measured reflection is significantly lower for the $p = 2$ method. This is because a high order polynomial basis can more accurately reproduce

the conductivity profile and is better at reducing spurious reflections at the PML layer interfaces. By using a cubic basis (i.e. $p = 3$) we can reproduce the cubic conductivity profile of (7.13) exactly over a single element. This permits a novel technique for high order methods, namely a single element PML region. The conductivity profile for the single element PML is also shown in Figure 7.29. In Figure 7.30 we show snapshots of the electric field magnitude for the single element PML simulation using $p = 3$ basis functions. Note how the incident pulse is absorbed by the PML region.

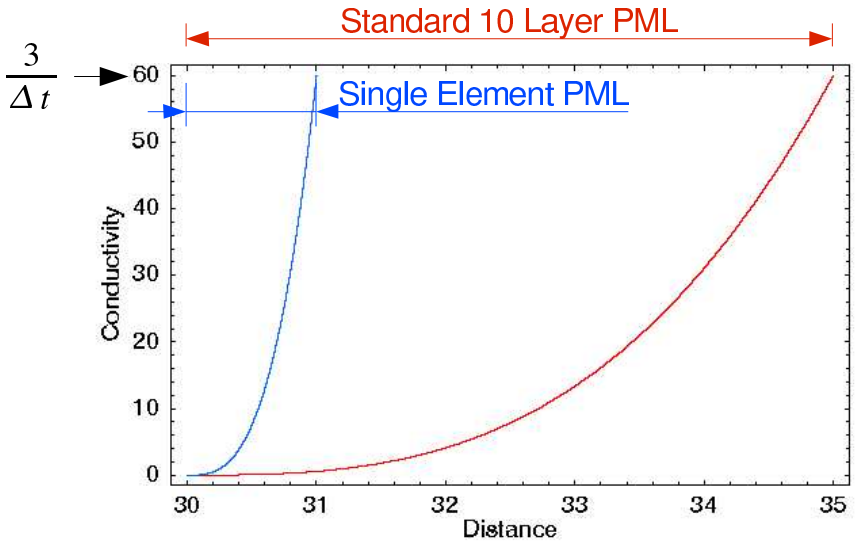


Figure 7.29: Conductivity profiles used for standard PML region and novel single layer PML.

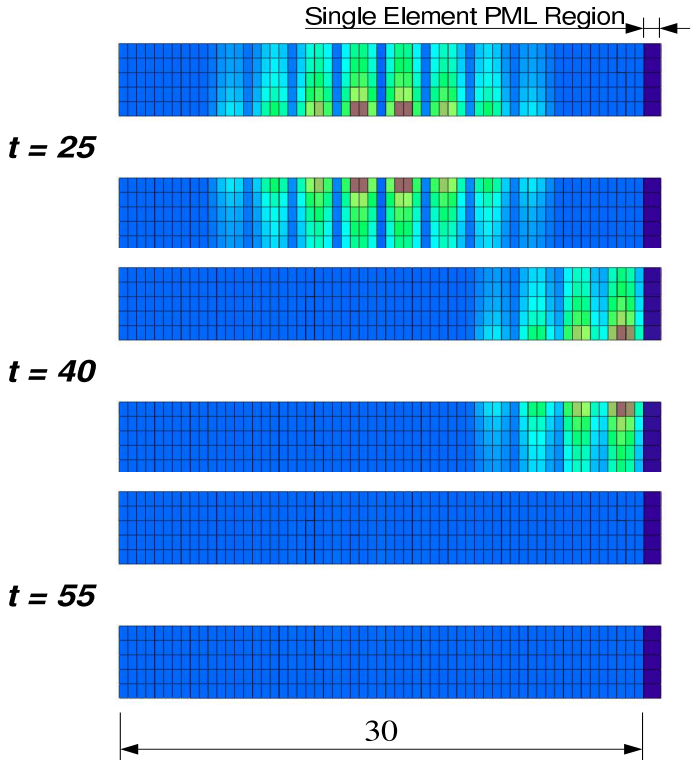


Figure 7.30: Snapshots of EM pulse incident on a single layer PML region. The speed of light is scaled to unity.

# Chapter 8

# Simulations

## 8.1 The Nature of Large Scale, Massively Parallel Simulations

Having presented all of the components of the method and demonstrated its validity using benchmark problems, we are now ready to utilize it to perform simulations of some physical and practical devices. In this chapter we present the results of massively parallel numerical simulations of electromagnetic wave propagation using the newly developed method of this dissertation. Due to the benefits of the proposed high order method (namely a significant reduction in the effects of numerical dispersion), we will apply it to a set of electrically large problems (i.e. problems with a large $\Omega/\lambda$ characteristic). Standard low order methods have difficulty in accurately simulating such devices. Due to the relatively large scale of these problems and the substantial memory requirements of high order methods, these simulations will need to be performed in a parallel computational environment.

The scale and complexity of a great many problems within traditional mathematical modeling areas such as astrophysics, structural engineering, solid state physics etc . . . has led to great technological advances in the field of large scale simulation. The capabilities of computer hardware and software have grown exponentially over the past two decades. Previously intractable problems can now be examined as modern computing power permits more computationally demanding approaches to be adopted. In addition, as evident by the results of Chapter 7, the proposed method of this dissertation is capable of delivering highly accurate results for a substantially smaller computational cost than traditional low order methods.

Parallel computing is defined as the simultaneous execution of the same task (split up and specially adapted) on multiple processors in order to obtain faster results. The term parallel processor is sometimes

used for a computer with more than one central processing unit, available for parallel processing. Systems with thousands of such processors are known as massively parallel. While a system of $n$ parallel processors is not more efficient than one processor of $n$ times the speed, the parallel system is often cheaper to build. For tasks which require very large amounts of computation, have time constraints on completion and especially for those which can be divided into $n$ execution threads, parallel computation is an excellent solution. In addition, distributed memory parallel computers are not subject to the $2GB$ memory restriction that serial machines are currently limited by. In fact, in recent years, most high performance computing systems, also known as supercomputers, have a distributed memory, parallel architecture.

In order to solve a problem in a parallel environment, the domain of the problem mus be partitioned into $n$ sub-sets of roughly equal size. This process is known as domain decomposition and there are several commercial and open-source tools available to perform this process. For the results presented in this chapter, all domain decompositions are handled by the multilevel graph partitioning schemes of the METIS program [136]. In order to maximize parallel performance, it is imperative that the domain decomposition tool partitions the problem domain into $n$ equally sized sub-sets while minimizing the amount of communication between processors. This is known as load-balancing. The goal of a load-balanced parallel algorithm is to achieve scalability, the situation where an increase in the number of processors yields an equally corresponding increase in performance.

## 8.2 Single Mode Optical Fiber

The optical fiber [137] is a modern method for the communication of data and has several applications ranging from the telecommunications industry, broadcast cable systems and defense sciences. It allows for guided transmission of optical signals which can carry a very high bandwidth of information across long physical distances. Optical fibers are typically classified as being either single or multi mode. Multimode fibers are capable of carrying many light modes and typically have core radii on the order of 20-150 micrometers [138]. Single mode fibers are best at retaining the fidelity of each light pulse over longer distances and exhibit no dispersion caused by multiple modes; thus more information can be transmitted per

unit time giving single mode fibers a higher bandwidth in comparison with multi-mode fibers. In this section a single mode step index optical fiber will be simulated. A typical single mode optical fiber has a core radius of 5-10 micrometers and a cladding radius of 120 micrometers which is in turn surrounded by buffers and mechanical cladding (such as a PVC "jacket" for protection).

### 8.2.1 Straight Optical Fiber

In this example we simulate the propagation of a pulsed TE01 mode along a $100\mu m$ section of a single mode optical fiber using third order ($p = 3$) basis functions and curvilinear surface elements ($s = 2$) at the core and cladding surfaces. The purpose of this simulation is to demonstrate the nature of pulse propagation in a straight fiber (for comparison with the next section) and to demonstrate the use of high order methods for problems of this type. To our knowledge, this is the first time such a fiber optic simulation has been performed using a high order, full wave approximation method with curvilinear surface elements. The core of the fiber has a radius of $5\mu m$ and an index of refraction of 1.471 while the cladding has a radius of $25\mu m$ and an index of refraction of 1.456. With these properties, the fiber is capable of propagating a $\lambda = 1550nm$ optical wave. The problem is excited with a space and time dependent pulsed voltage source boundary condition applied to the input cap of the mesh. The spatial dependence of the voltage source is derived from Bessel functions of the first and second kind with the appropriate transverse propagation constants to satisfy continuity across the core / cladding interface while the temporal profile is a pulsed sine wave containing 20 wavelengths as shown in Figure 8.1. A PEC boundary condition is applied to the outer cladding surface while an absorbing boundary condition (ABC) is applied to the end cap of the mesh. While typical commercial fibers of this type have a cladding radius on the order of $120\mu m$, the cladding radius chosen for this simulation is sufficient to demonstrate the propagation properties of a straight fiber as the TE01 mode dies off exponentially as a function of cladding radius as indicated in Figure 8.1.

Use of $p = 3$ basis functions and $s = 2$ curvilinear surface elements permits the use of a relatively coarse mesh, namely 1 transverse element per wavelength (i.e. a cubic polynomial can represent a whole period of a sine wave) instead of the usual 10 transverse elements for a standard low order method. For this simulation, the fiber optic mesh consists of only $8,208$ elements. Standard cell-centered visualization

methods for the fields result in a very coarse representation on such a mesh. Figure 8.2 shows a magnitude plot of the electric field vector at time step $t = 0.187ps$ sampled at 25 points per element, indicating the high degree of field resolution within in each element using high order basis functions. Note that for this straight fiber optic guide, the pulse remains confined in the core as it is guided.

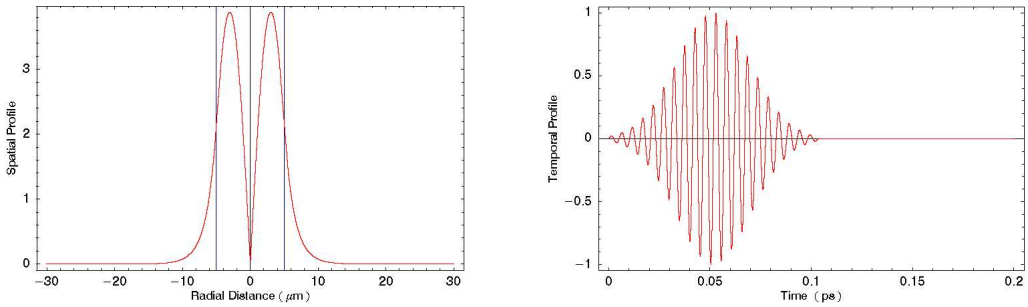


Figure 8.1: Spatial and temporal profile of pulsed voltage source used to excite fiber optic simulation.

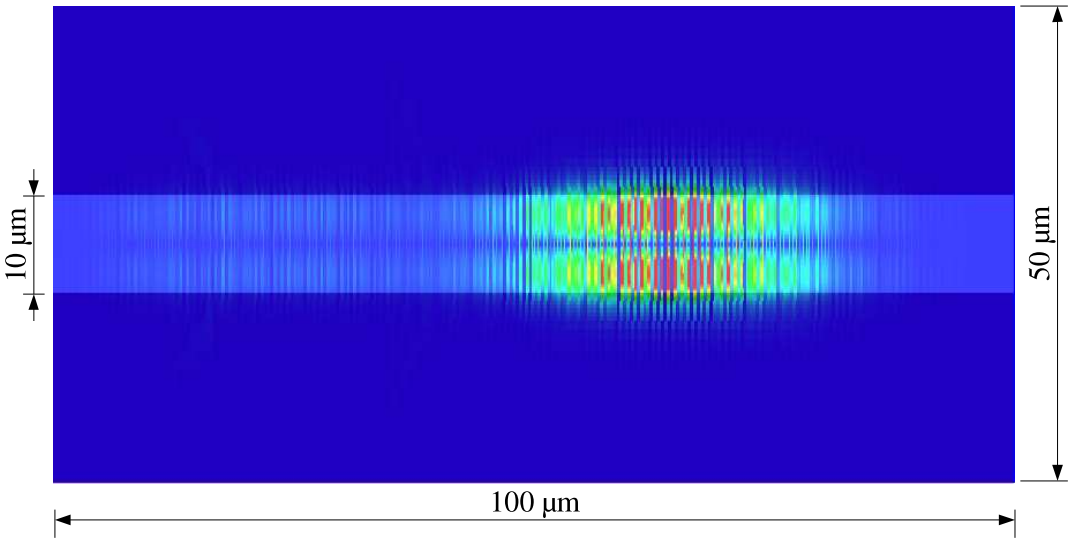


Figure 8.2: Snapshot of electric field magnitude at $t = 0.187ps$ in straight fiber optic simulation.

### 8.2.2 Transmission through a Bent Optical Fiber

Here we transmit the same $1550nm$ optical pulse from the previous example through a bent single mode optical fiber. The loss due to curvature of an optical fiber has been studied extensively and it is well well known that fiber waveguides lose power by radiation if their propagation axes are curved. It is therefore greatly important to quantify the nature of this loss for fiber optic communication systems. In particular,

engineers are most concerned with characterizing the power loss and polarization change of a signal as it traverses such a bent device. In addition, certain remote sensing techniques focus on analyzing the radiation field patterns of bent fibers.

In [139], an analytic formula for the signal loss of bent single mode fibers is derived, however there are serious limitations to this approximation. The most serious limitation to its validity is caused by using the undistorted field of a straight fiber optic guide for its derivation. As pointed out in [139], even if the standard radiation loss of the fiber is disregarded, the field changes its shape in the curved guide (known as the bend loss); the field is forced toward the outer wall in a manner resembling a centrifugal force effect. Only for very large bending radius values is it permissible to neglect this effect; for sharply curved guides the field distortion caused by the bend has a considerable influence on the curvature loss. Bending losses in fibers are therefore classified as either macro-scale or micro-scale. It is well known that losses due to macro-scale bends where the bending radius is greater than $10cm$ are essentially negligible [138]. Transmission in a fiber with a bending radius smaller than this is subject to signal loss due to radiation and bend loss [140].

In this example, we simulate a fiber bent at a $60^\circ$ angle with a bending radius of $200\mu m$. The geometry of the problem is shown in Figure 8.3. The lack of axial symmetry in the problem domain makes full wave simulations intractable for beam propagation methods (BPM) which are typically used in analyzing fiber optic waveguides. In addition, a full wave simulation for this device, where the time dependent electric and magnetic vector fields are solved for over the entire problem domain, can yield a wealth of information including radiation field patterns, electric field orientation and power loss / distribution. In addition, because the problem is electrically large, use of a high order method will reduce the effects of numerical dispersion on the pulse as it traverses the length of the bent fiber.

The computational mesh for this simulation consists of $652,700$ hexahedral elements with 4 transverse elements per wavelength yielding a value $\Delta h_z = 0.3875\mu m$ where $\Delta h_z$ is the characteristic mesh size in the direction of propagation. We use high order $p = 2$ interpolatory basis functions to represent the electric and magnetic fields in the problem domain (i.e. each period of the pulse is represented piecewise by 4 quadratic polynomials) resulting in a discrete linear system of approximately 16 million unknowns. The

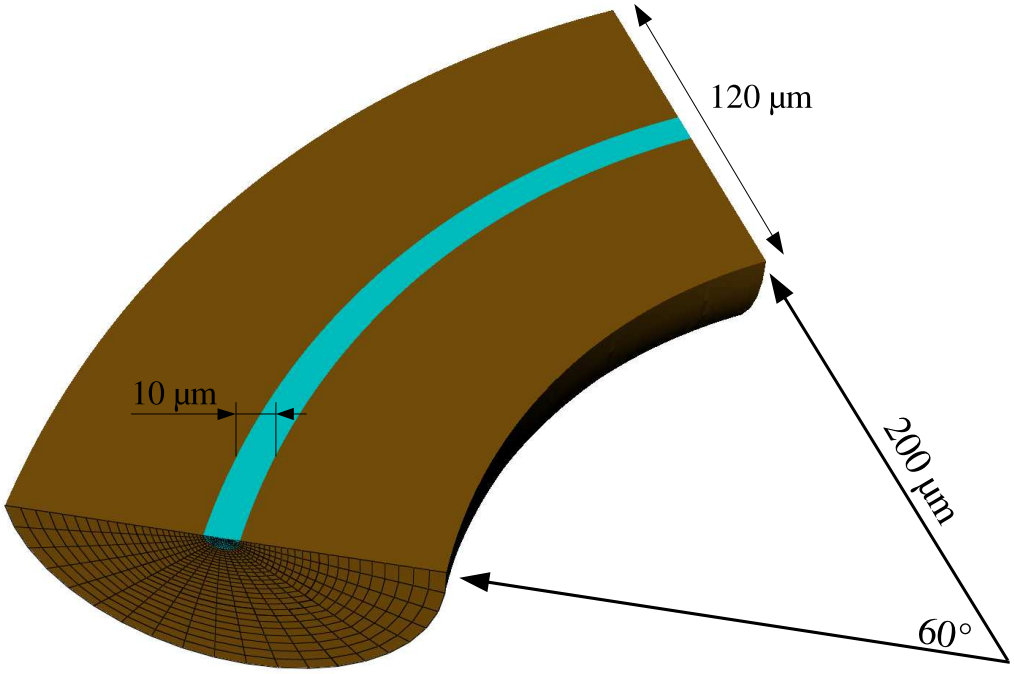


Figure 8.3: Optical fiber bent at 60 degrees with a 200 $\mu m$ bend radius.

mesh and its corresponding linear system are distributed in parallel across 256 processors. The simulation is propagated forward in time for total of $6,000$ time steps. The results of this massive computation represent the first time a high order, full wave simulation of a bent optical fiber has ever been performed and the first full wave analysis of micro-scale bend losses in an optical fiber.

In Figure 8.4 we show three separate snapshots in time of the magnitude of the Poynting-vector field (representing the transmitted power in the guide). Note how as the pulse propagates down the fiber, the signal leaks into the cladding indicating loss due to the micro-scale bend. In Figure 8.5 we plot the relative power loss (defined as the measured power in the core over the total power of the pulse) at seven separate time intervals. As the pulse traverses the bend, the majority of the power is radiated into the cladding, and the remaining power still guided in the core diminishes rapidly as a function of time. In Figure 8.6 we plot the electric field vector at two different points in time. In each case, the electric field is sliced in a plane transverse to the propagation direction of the fiber. Note that at $t = 0.13ps$ the electric field is still more or less circularly polarized as one would expect for a TE01 mode. However, as the pulse traverses the bend and

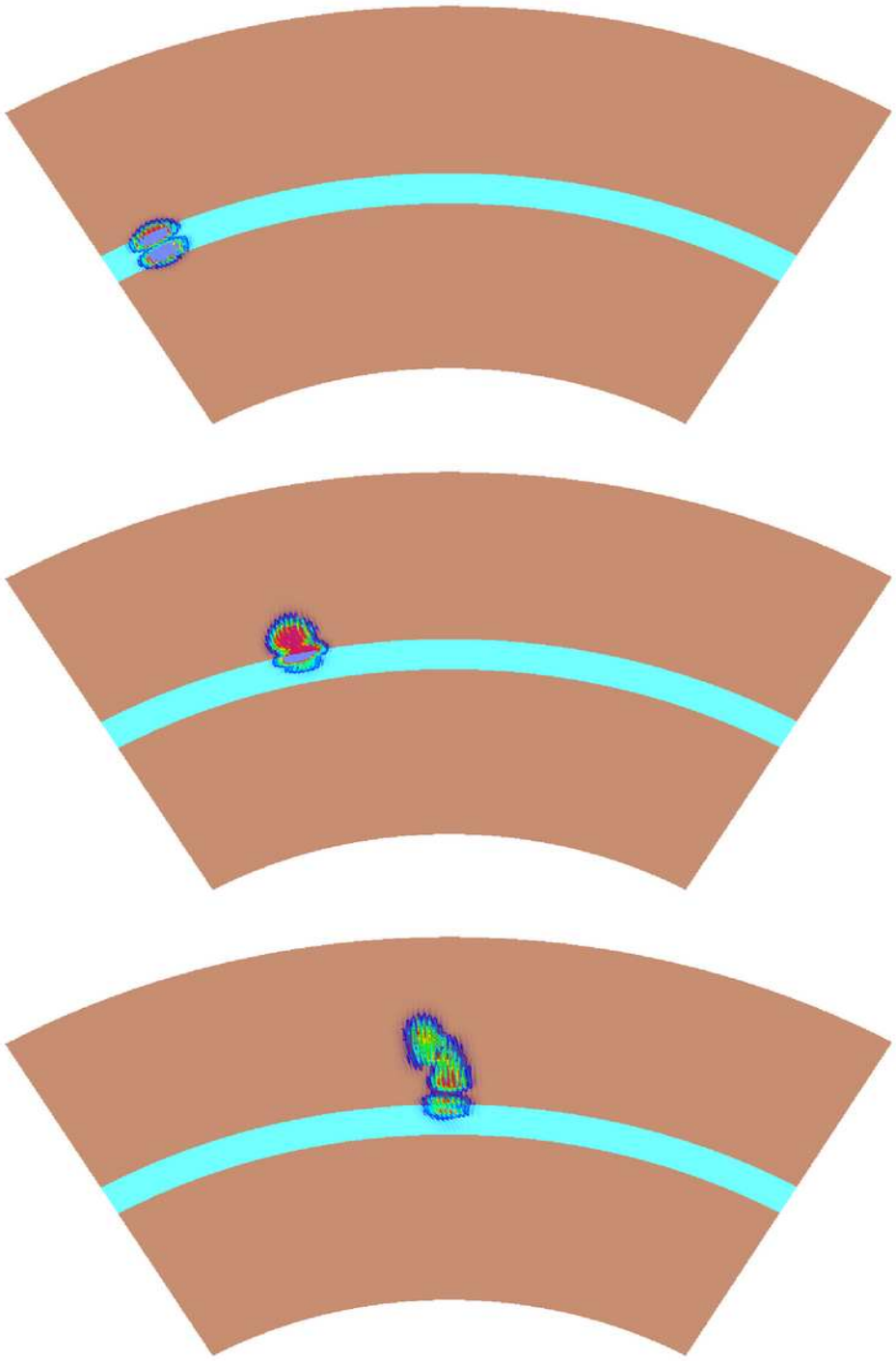

Figure 8.4: Snapshots of Poynting-vector field magnitude for bent optical fiber simulation at $0.13ps$, $0.34ps$, and $0.55ps$ indicating power loss due to micro-scale bend.

begins to radiate into the cladding, the orientation of the electric field becomes considerably more complex.

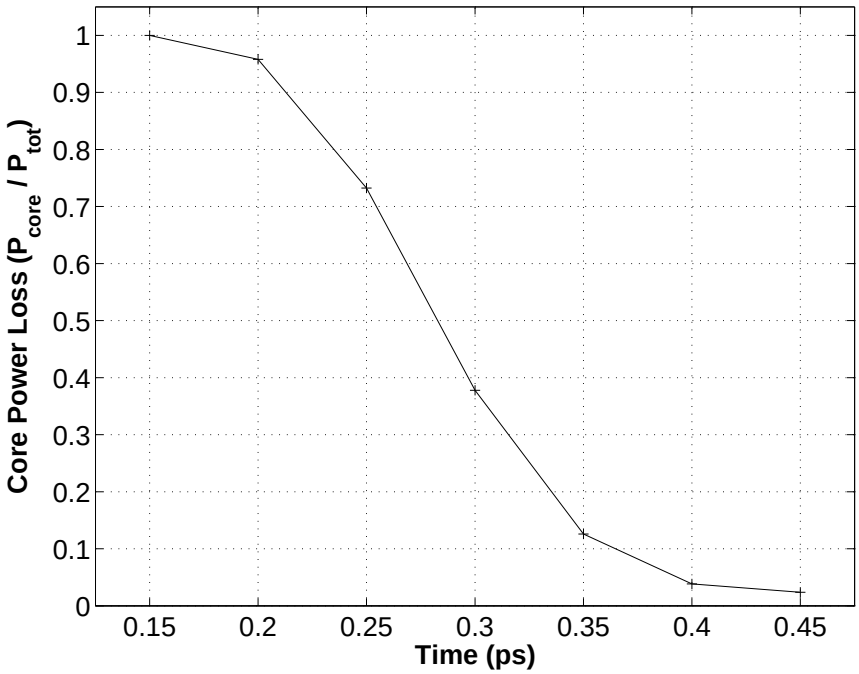


Figure 8.5: Relative core power loss as a function of time for bent optical fiber simulation.

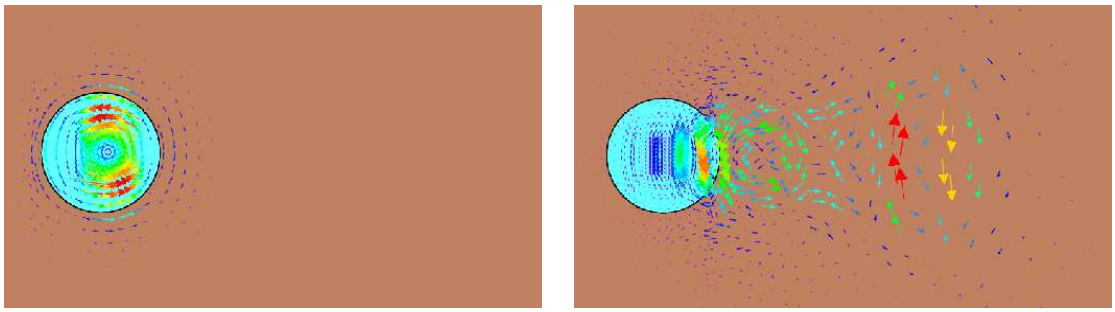

Figure 8.6: Vector plots of electric field, sliced in transverse planes, for bent optical fiber simulation at $0.13ps$ and $0.55ps$ indicating effect on polarization due to micro-scale bend.

## 8.3 Photonic Band-Gap Waveguides

In this section we simulate two different wave-guiding structures based on the notion of a photonic band-gap (PBG) structure (also known as a photonic crystal) [141], [142]. Advances in semiconductor physics have allowed for the customization of the conducting properties of certain materials, thereby initiating the transistor revolution in physics and the subsequent digital information revolution. In the last decade a new

frontier has emerged with a similar goal: to control the optical properties of materials. In a manner completely analogous with semiconductor physics, PBG materials allow for the engineering of devices that can prohibit the propagation of light, allow propagation only in certain directions at certain frequencies, or localize light in specified areas. Already, fiber optic cables like those from the previous section which simply guide light have revolutionized the telecommunications industry; photonic band-gap devices promise an even greater leap forward.

A PBG structure works in manner very similar to a currently well known and widely used optical device: the dielectric mirror (or "quarter-wave stack") consisting of alternating layers of different dielectric materials. Light of the proper wavelength, when incident on such a layered material, is completely reflected. The reason is that the light wave is scattered at the layer interfaces, and if the spacing is just right, the multiply scattered waves interfere destructively inside the material. This effect is well known – however, while such mirrors are tremendously useful, they only reflect light at normal incidence or near normal incidence to the layered material. A PBG structure is the generalization of this notion for periodic arrays of dielectric material in two and three dimensions. In general, a PBG structure is defined to be a periodic array of dielectric materials with a characteristic dimension (or lattice constant) $a$ such that incident light of wavelength $\lambda \approx 2a$ is forbidden to propagate in the structure.

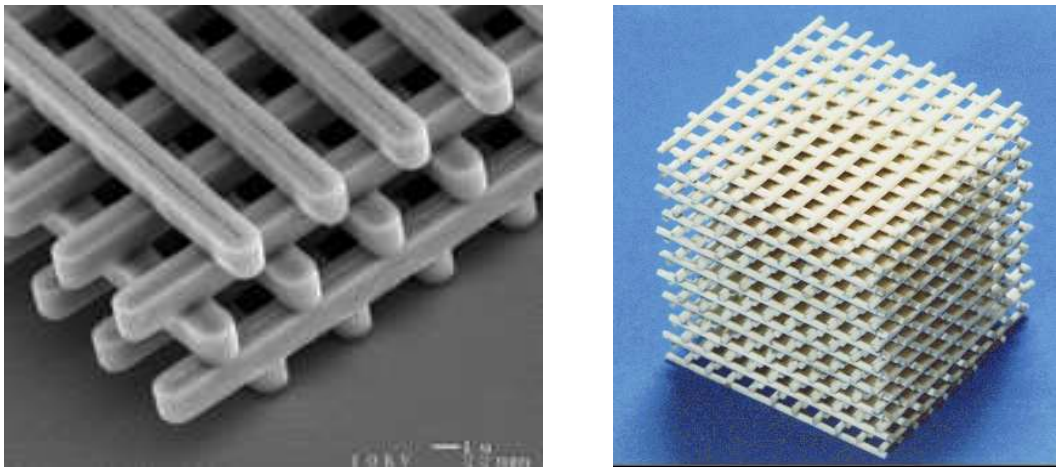

Figure 8.7: Examples of a micrometer scale (optical frequency) photonic crystal (*left*) and a centimeter scale (radio frequency) photonic crystal (*right*)

If, for some frequency range, a PBG structure reflects light of *any* polarization incident at *any* angle, then the crystal has is said to have a "complete photonic band gap" [143]. In such a crystal, no light modes can propagate if they have a frequency within that range. The simple dielectric mirror cannot have a complete band gap, because scattering occurs along only one axis. In order to create a material with a complete photonic band gap, the contrasting dielectrics must be arranged in a lattice that is periodic along three axes. Due to the nature of Maxwell's equations, a PBG structure can be scaled to any operating wavelength (or frequency). Figure 8.7 shows two examples of three dimensional PBG structures with a complete photonic band gap; one designed to operate at optical wavelengths and one designed for the RF regime.

### 8.3.1 2D Slab Optical Waveguide

Here we simulate the propagation of an optical signal around a sharp 90 degree bend in a two dimensional slab (i.e. one element thick in the *z*-direction) PBG waveguide [144]. The PBG structure consists of a 9 by 9 array of Gallium-Arsenide (GaAs) cylinders oriented normal to the *x*-*y* propagation plane, with a relative dielectric permittivity of $\varepsilon_r = 12.0$, surrounded by a square section of air ($\varepsilon_r = 1.0$). Periodic structures such as these are characterized by the ratio of the rod radius $r$ to the rod spacing, or lattice constant, $a$. For the computational mesh used in these simulations, the ratio $\frac{r}{a} = 0.18$ is chosen for an operating wavelength of $\lambda = 1.55\ \mu m$, this gives a transverse magnetic (TM) band-gap of $\omega = 0.302\frac{2\pi c}{a}$ to $\omega = 0.443\frac{2\pi c}{a}$. The rod spacing for the geometry is $a = 0.62\ \mu m$. A waveguide can be constructed by removing some of the cylinders and introducing a defect; allowing a small range of wavelengths of light around a central defect frequency to propagate through the structure [145], [146].

The computational mesh for this problem consists of $4,432$ hexahedral elements. We use high order $p = 3$ interpolatory polynomial basis functions to represent the electric and magnetic fields resulting in a total of $441,123$ 1-form and $399,888$ 2-form degrees of freedom. The mesh and its corresponding linear system are partitioned over 16 processors as shown in Figure 8.8. Table 8.1 summarizes the distribution of the various 1-form degrees of freedom over each of the 16 processors. Note that the domain decomposition algorithm has distributed the degrees of freedom very uniformly across the 16 processors resulting in a load balanced system.

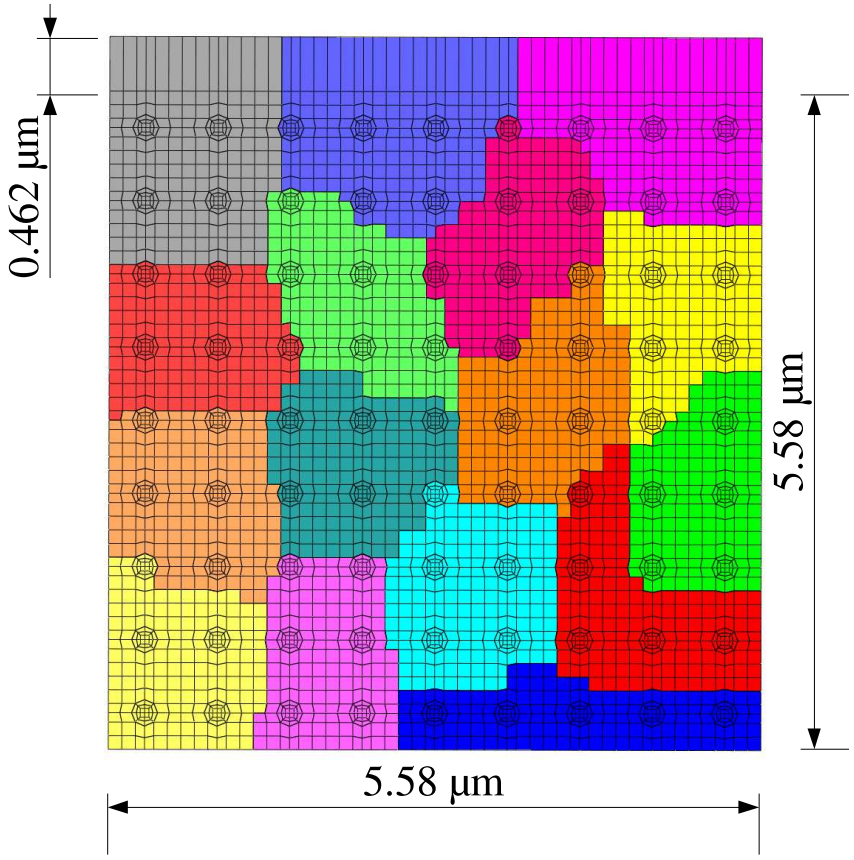


Figure 8.8: Slab PBG optical waveguide mesh partitioned over 16 processors.

To account for the open region boundary at the exit of the waveguide, a $0.462\mu m$ thick Maxwellian PML is added to the top end of the mesh. This region consists of artificial electric and magnetic conductivity terms as in (6.20), designed to attenuate the propagating wave before it hits the PEC boundary condition used to terminate the mesh. An isotropic tensor valued electric and magnetic conductivity profile is defined for this region corresponding to a third-order polynomial function of distance. The conductivity profiles increase cubically from zero to a maximum value of $\frac{3}{\Delta t}$ over the distance $5.58\mu m$ to $6.042\mu m$. Since we are using $p = 3$ polynomial basis functions for this simulation, the cubic conductivity profile will be projected onto the finite element space exactly.

For this problem, a discrete time step of $\Delta t = 0.0025$ is used and the simulation runs for a total of $15,000$ time steps. The discrete equations of (4.3) are integrated using the implicit time stepping scheme

| Proc. ID. | No. Edge DoF | No. Face DoF | No. Cell DoF | Total |
|---|---|---|---|---|
| 0 | 4,560 | 13,944 | 10,116 | 28,620 |
| 1 | 4,104 | 13,032 | 9,720 | 26,856 |
| 2 | 4,326 | 13,524 | 9,972 | 27,822 |
| 3 | 4,191 | 13,344 | 9,972 | 27,507 |
| 4 | 4,560 | 14,028 | 10,224 | 28,812 |
| 5 | 4,017 | 12,864 | 9,648 | 26,529 |
| 6 | 4,125 | 13,284 | 10,008 | 27,417 |
| 7 | 4,194 | 13,488 | 10,152 | 27,834 |
| 8 | 4,518 | 13,944 | 10,188 | 28,650 |
| 9 | 4,179 | 13,104 | 9,684 | 26,967 |
| 10 | 4,029 | 12,936 | 9,720 | 26,685 |
| 11 | 4,053 | 13,188 | 10,008 | 27,249 |
| 12 | 3,969 | 12,852 | 9,720 | 26,541 |
| 13 | 4,173 | 13,320 | 9,972 | 27,465 |
| 14 | 4,428 | 13,884 | 10,260 | 28,572 |
| 15 | 4,065 | 13,344 | 10,188 | 27,597 |

Table 8.1: Distribution of the $441,123$ high-order 1-form degrees of freedom over 16 processors for the 2D PBG simulation.

of (6.22). The problem is excited with a sinusoidally driven voltage source applied on the left hand side of the mesh. Figure 8.9 shows the results of this slab PBG waveguide simulation at four snapshots in time. Note how the PBG waveguide is capable of transmitting the electromagnetic wave around a very sharp 90 degree bend with minimal loss. In Figure 8.10 we plot the base 10 log of the electric field magnitude along a "line-out" portion of the PBG waveguide through the $y$-axis at time $t = 0.08ps$ ($9600\Delta t$). Note how the propagating wave is rapidly attenuated in the PML region.

Now if we neglect the electric and magnetic conductivity terms of (6.20) (i.e. $\sigma, \sigma^* = 0$), the numerical energy of the problem as measured by (6.8) should be conserved. We now excite the PBG waveguide with a pulsed voltage source and deposit a finite amount of energy into the problem. The pulse has a temporal length equal to 10 wavelengths as shown in Figure Figure 8.11. The resulting numerical energy as computed with the second order accurate and energy conserving leap-frog method is also shown in Figure 8.11. As expected, the energy ramps up until the pulse is switched off, at which point it remains constant for the duration of the simulation. In Figure 8.12, we show close ups of the computed numerical energy during the last half of the simulation for two different energy conserving integration methods: the standard second order accurate

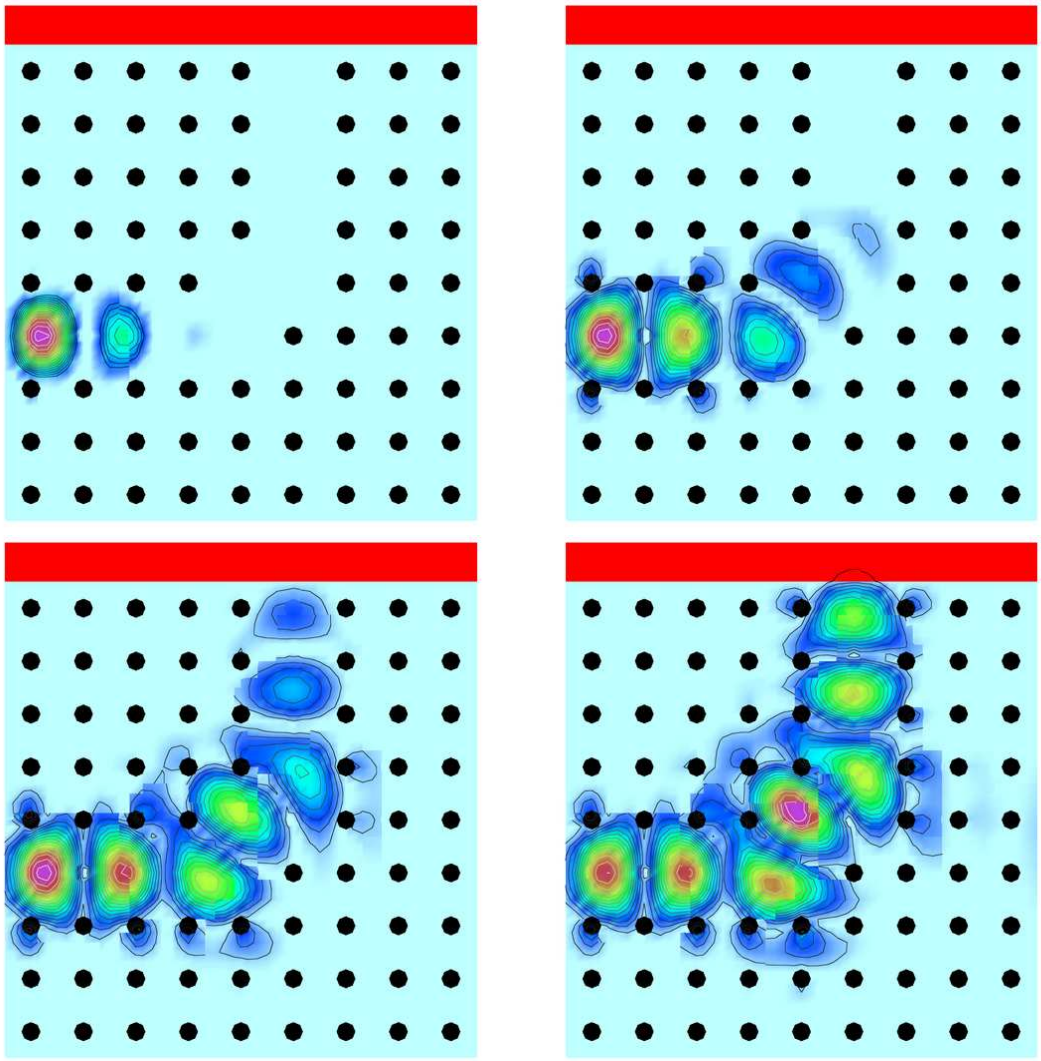

Figure 8.9: Snapshots of electric field for 2D PBG simulation at $0.02ps$, $0.035ps$, $0.05ps$ and $0.065ps$.

leap-frog method and a third order accurate ($k = 3$) symplectic integration method from Chapter 6. For the leap-frog method, a discrete time step of $\Delta t = 0.0033333$ is used and the 1-form mass matrix $M_\varepsilon$ of (4.3) needs to be solved once per time step, for a total of 12,000 time steps. For the third order symplectic method, a discrete time step of $\Delta t = 0.01$ is used and the 1-form mass matrix $M_\varepsilon$ of (4.3) needs to be solved three times per time step, for a total of 4000 time steps. The resulting computations therefore require the same total amount of time to complete. In both cases, the computed numerical energy will oscillate; however, as shown in Figure 8.12, the third order symplectic method is much better at controlling the amplitude of this

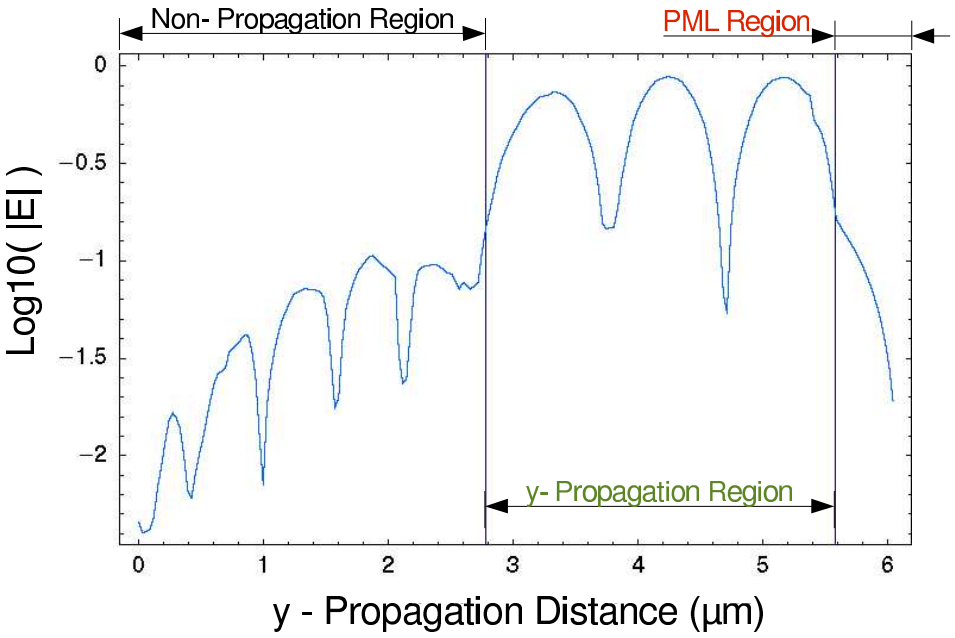


Figure 8.10: Logarithmic plot of electric field magnitude along the *y*-directed portion of the 2D PBG waveguide indicating the attenuation properties of the single element thick PML region.

oscillation, for the same computational cost as the leap-frog method.

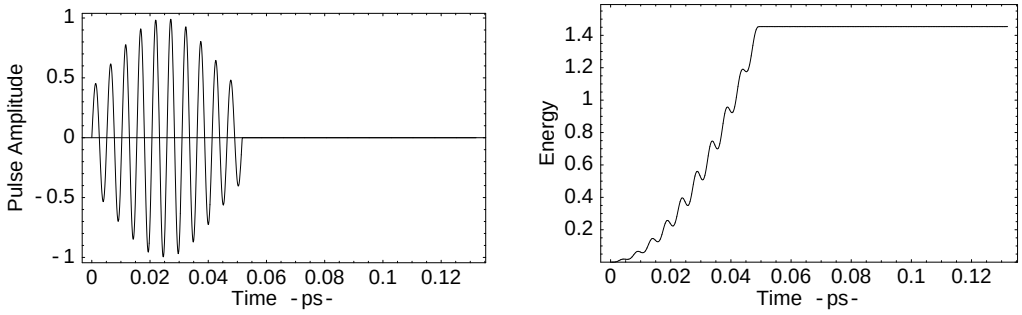


Figure 8.11: Temporal profile of pulsed voltage source (*left*) and numerical energy as a function of time (*right*) for the 2D PBG simulation.

### 8.3.2 3D "Multi-Bend" Woodpile RF Waveguide

Here we simulate a 3D PBG waveguide with a complete photonic band-gap designed to operate in the RF regime. The PBG crystal is based on the "woodpile" structure as investigated by [147] and [148]. In particular, we utilize the unit cell originally proposed by [149] which consists of a series of aluminum rods (index of refraction $= 3.1$) arranged in an alternating, stacked configuration. The lattice constant for this

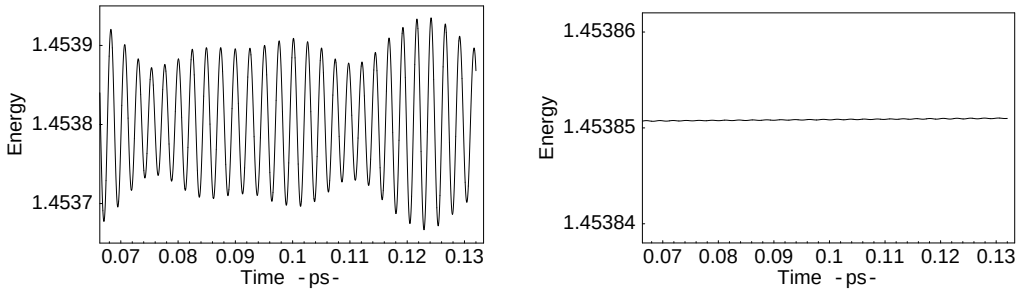


Figure 8.12: Close-up of numerical energy during last half of simulation for leap-frog method (*left*) and third order symplectic method (*right*) for 2D PBG simulation.

crystal is 1.123*cm* and the unit cell has dimensions of 1.123*cm* by 1.123*cm* by 1.272*cm* making it suitable for operation in the radio frequency regime. We construct a 3D crystal by arranging the unit cell in a 9 by 13 by 7 layer configuration as shown in Figure 8.13.

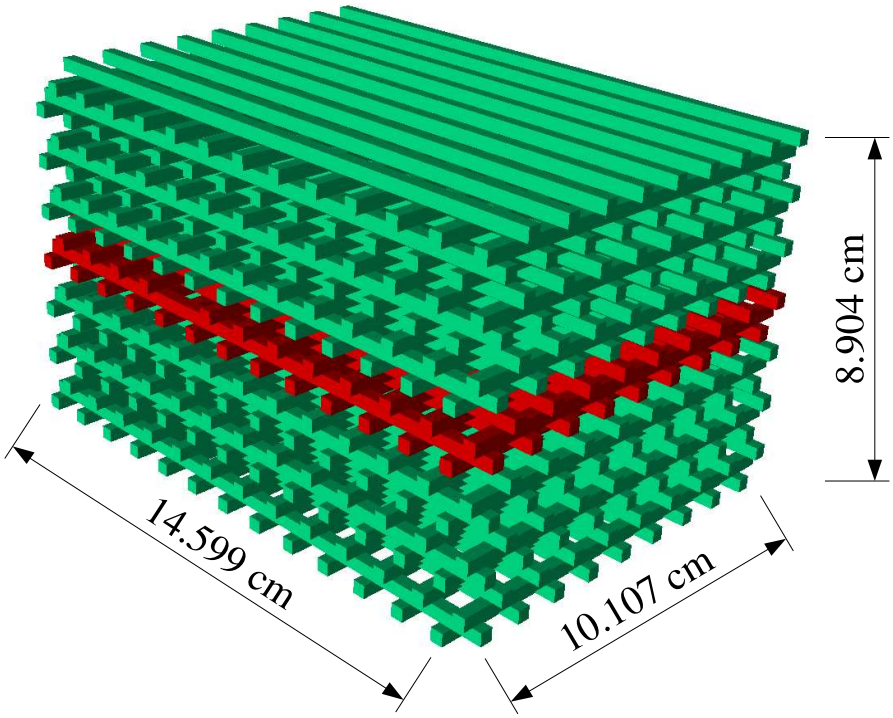


Figure 8.13: 3D PBG "woodpile" structure for RF signals.

Our goal is to exploit the complete photonic band gap of this crystal and create a "multi-bend" wave guide where we can make the radio signal traverse two separate 90 degree bends in three dimensional space. This can be accomplished by introducing three separate defects into the crystal as shown in Figure

8.14. In Figure 8.13 and Figure 8.14, the *x*-*y* defect layer is highlighted in red for clarity. First we create a 90 degree bend in the *x*-*y* plane by removing a half portion of two of the rods. While the vast majority of computational research in PBG waveguides has been performed on two dimensional structures like that of the previous section, single bends in 3D crystals like the *x*-*y* planar defects of Figure 8.14 have been studied. What makes this simulation unique is the introduction of a third *z*-defect by removing a section of rods 2 lattice constants wide from each of the stacked *z* layers as shown in Figure 8.14. Because of the 3D nature of the multi-bend, this type of simulation cannot be performed using standard 2D codes which are extensively used in the study of PBG devices. In addition, trustworthy simulations of PBG waveguides require that phase velocities of propagating waves be computed as accurately as possible. A high order method is therefore highly desirable for an electrically large waveguide such as this. Also note that this *cm* scale device is very simple to fabricate in contrast to the $\mu m$ scale optical device of the previous section.

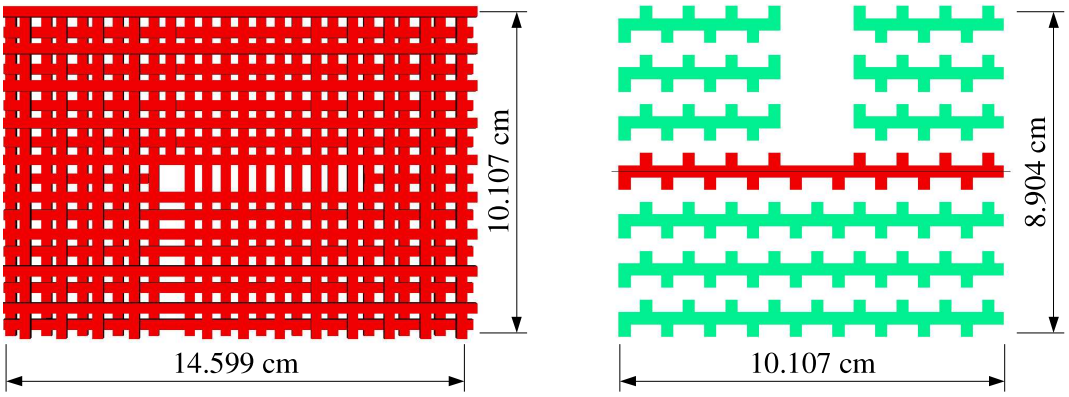


Figure 8.14: Defect layers for the 3D PBG "multi-bend" waveguide in the *x*-*y* plane (*left*) and in the *x*-*z* plane (*right*).

The computational mesh of Figure 8.13 consists of $419,328$ hexahedral elements. We excite the problem with a time dependent voltage source boundary condition applied at the *x*-*z* input plane with an operating frequency of $11GHz$. The rest of the mesh is terminated with a PEC boundary condition. We use high order $p = 2$ basis functions to represent the electric and magnetic fields resulting in a linear system with approximately 10.5 million unknowns. This large linear systems requires that the problem be distributed in

parallel across 150 processors. We let the simulation run for a total of $6,500$ time steps. In Figure 8.15 we show a three dimensional iso-surface plot of the electric field magnitude in the wave guide at the end of the simulation. Note how the wave has made two complete 90 degree bends with a negligible loss due to radiation. In Figure 8.16 we show six separate snapshots of the time dependent electric field plotted over three separate slices into the crystal. The source of the wave is originally polarized along the $z$-axis of the guide. Note how as the the wave traverses the first bend, it remains polarized along the $z$-axis but as it traverses the second bend, it becomes polarized along the $y$-axis. The results of this computation represent the first time a high order, full wave simulation of a 3D PBG waveguide has ever been performed, and to our knowledge, the first time a multi-bend PBG waveguide has ever been demonstrated.

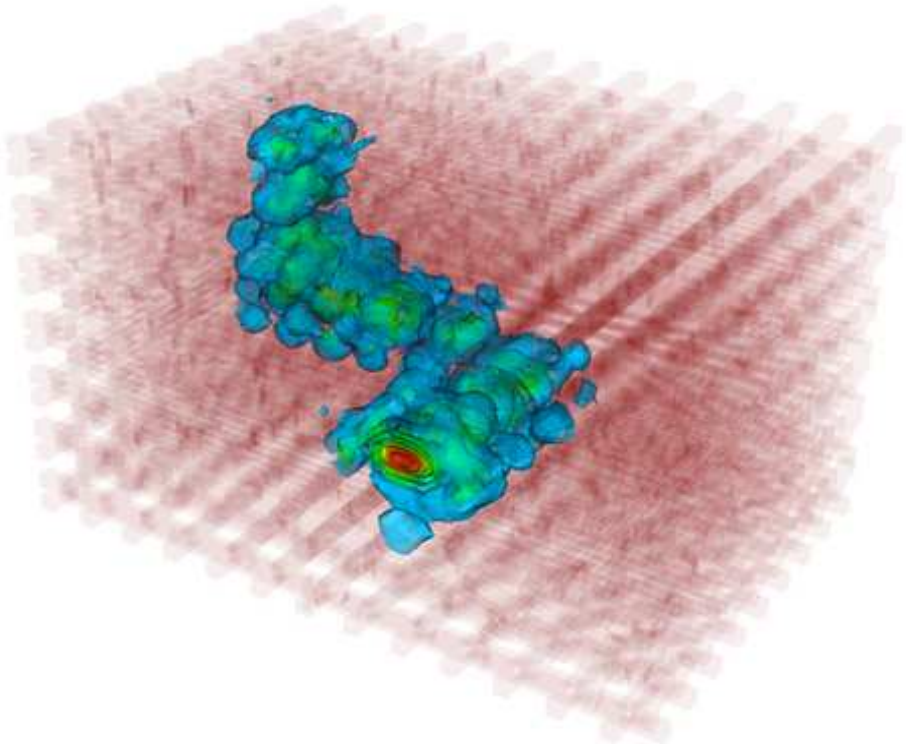

Figure 8.15: Three dimensional iso-surface plot of electric field magnitude for the 3D PBG simulation.

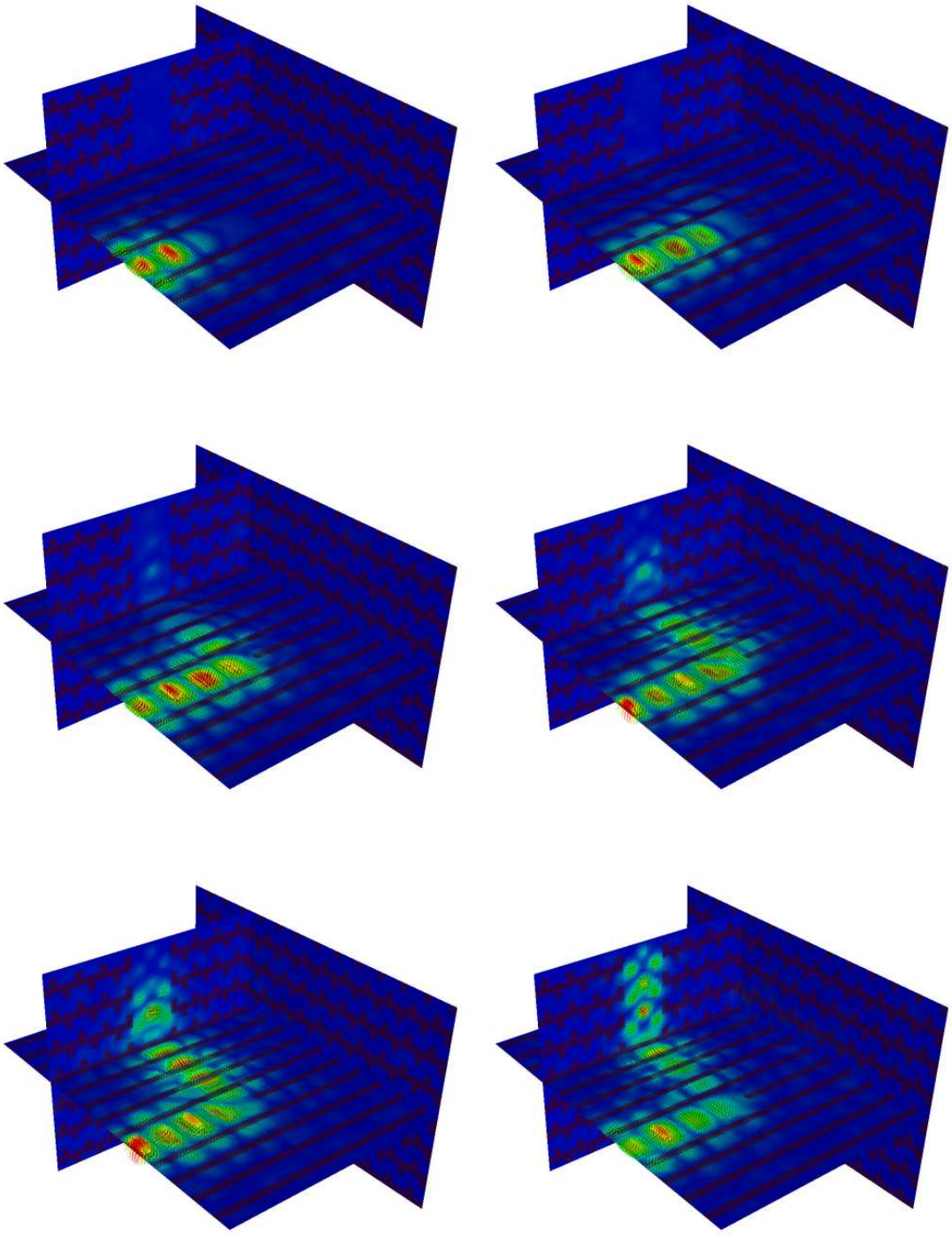

Figure 8.16: Snapshots of electric field at six separate time steps for the 3D PBG simulation.

# Chapter 9

# Conclusions and Future Work

The purpose of this dissertation has been twofold. First, a novel high order time domain vector finite element method that is suitable for simulating electrically large transmission and communication devices has been developed, verified and presented. This method has several desirable benefits, namely

- Arbitrary order accuracy in space
- Up to 4th order accurate in time
- Valid on non-orthogonal / unstructured grids
- Conditionally stable and consistent
- Charge and energy conserving
- Automatic elimination of “spurious modes”
- Fully “mimetic” - divergence and curl properties of fields satisfied exactly
- Correct modeling of field jump discontinuities
- Valid for anisotropic tensor valued media

A key point of this work is that high order methods for Maxwell’s equations excel at reducing the effects of numerical dispersion, which has plagued standard low order methods when applied to electrically large problems. In addition, it has been demonstrated that high order methods are more computational cost effective for certain problems when compared to standard low order methods. The development and verification of

this method has led to several refereed publications including [109], [102], [99], [108], [105], [120] and [150].

Second, the proposed method has been applied to modern communication devices such as optical fibers and photonic crystal wave guides via large scale, parallel simulations. In order to accurately and effectively simulate such electrically large wave guiding devices, numerical dispersion must be kept to a minimum. It has been demonstrated that the proposed high order method of this dissertation works well for such problems. Two different wave guiding structures were analyzed. To our knowledge this dissertation represents the first time a high order, full wave parallel simulation has ever been done of such devices including the bent single mode optical fiber and the 3D RF "multi-bend" guide.

Future work on this project will continue and promises several new advances in the field of computational electromagnetics and high order vector finite element methods. There is a strong need to accurately model the effects of EM wave interaction with non-linear materials and research into adapting the proposed method to deal with these types of simulations will be well spent. In addition, future work will include the introduction of reduced integration rules for the bilinear forms resulting in a significant reduction in the total number of non-zeros and for the special case of Cartesian meshes, the reduction of the mass matrix to a diagonal matrix resulting in a drastic reduction in total computational workload. Error estimators and adaptive mesh refinement will allow for adaptive $h$ and $p$-refinement schemes, leading to further computational savings. Finally, the introduction and coupling of high order methods for CEM to existing mechanical and thermal codes involving moving meshes, sliding contact surfaces and fluid dynamics will lead to advances in the fields of magnetohydrodynamics and electro-thermal-mechanical simulations.

# Bibliography


1. J. C. Maxwell. *A Treatise on Electricity and Magnetism, Vol. 1*. Dover Publications, 1954.
2. W. Greiner. *Classical Electrodynamics*. Springer-Verlag, New York, 1998.
3. M. Heald and J. Marion. *Classical Electromagnetic Radiation*. Saunders College Publishing, third edition, 1995.
4. M. N. O. Sadiku. *Numerical Techniques in Electromagnetics*. CRC Press, 1992.
5. D. A. White. *Discrete Time Vector Finite Element Methods for Solving Maxwell's Equations on 3D Unstructured Grids*. PhD thesis, University of California at Davis, Livermore, California, 1997.
6. A. Taflove ed. *Advances in Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House, 1998.
7. M. Salazar-Palma, T. K. Sarkar, L. Garcia-Castillo, T. Roy, and A. Djordjevic. *Iterative and Self Adaptive Finite-Elements in Electromagnetic Modeling*. Artech House, 1998.
8. M. V. K. Chari and S. J. Salon. *Numerical Methods in Electromagnetism*. Academic Press, 2000.
9. K. S. Yee. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Ant. Prop.*, 14(3):302–307, 1966.
10. A. Taflove and M. E. Brodwin. Numerical solution of steady-state electromagnetic scattering problems using the time-dependent Maxwell's equations. *IEEE Trans. Microwave Theory Tech.*, 23:623–630, 1975.
11. A. Taflove. Review of the formulation and applications of the finite-difference time-domain method for numerical modeling of electromagnetic wave interaction with arbitrary structures. *Wave Motion*, 10:547–582, 1988.
12. E. L. Lindman. Free-space boundary conditions for the finite-difference approximation of the time-doamin electromagnetic field equations. *J. Comput. Phys.*, 18:66–78, 1975.
13. G. Mur. Absorbing boundary conditions for the finite-difference approximation of the time-domain electromagnetic field equations. *IEEE Trans. Electromagnetic Compatibility*, 23(4):377–382, 1981.
14. J. Berenger. A perfectly matched layer for the absorption of electromagnetic waves. *J. Comput. Phys.*, 114(2):185–200, 1994.
15. R. Holland. The case against staircasing. In *Proceedings of the 6th annual Review of Progress in Applied Computational Electromagnetics*, pages 89–95, March 1990.
16. A. C. Cangellaris and D. B. Wright. Analysis of the numerical error caused by the stair-stepped approximation of a conducting boundary in FDTD simulations of electromagnetic phenomena. *IEEE Trans. Ant. Prop.*, 39(10):1518–1525, 1991.

17. R. Holland. Pitfalls of staircase meshing. *IEEE Trans. Electromagnetic Compatibility*, 35(4):434–439, 1993.

18. P. Boullard A. Deraemaeker, I. Babuska. Dispersion error and pollution of the FEM solution for the Helmholtz equation in one, two, and three dimensions. 46(4):471–499, 1999.

19. P. Monk and A. Parrot. A dispersion analysis of finite element methods for Maxwell's equations. *SIAM J. Sci. Comp.*, 15(4):916–937, 1994.

20. S. Warren and W. Scott. An investigation of numerical dispersion in the vector finite element method using quadrilateral elements. *IEEE Trans. Ant. Prop.*, 42(11):1502–1508, 1994.

21. S. Warren and W. Scott. Numerical dispersion in the finite element method using triangular edge elements. *Opt. Tech. Lett.*, 9(6):315–319, 1995.

22. M. Ainsworth and J. Coyle. Hierarchic $hp$-edge element families for Maxwell's equations on hybrid quadrilateral/triangular meshes. *Comput. Methods Appl. Mech. Engrg.*, 190:6709–6733, 2001.

23. J. Fang. *Time Domain Finite Difference Computation for Maxwell's Equations*. PhD thesis, University of California at Berkeley, Berkeley, California, 1989.

24. T. Deveze, L. Beaulie, and W. Tabbara. A fourth order scheme for the FDTD algorithm applied to Maxwell equtions. In *Proceedings IEEE Antennas and Propagat. Soc. Int. Symp.*, pages 346–349, Chicago, IL, 1992.

25. M. F. Hadi and M. Piket-May. A modified FDTD (2,4) scheme for modeling electrically large structures with high phase accuracy. *IEEE Trans. Ant. Prop.*, 45(2):254–264, 1997.

26. Z. Xie, C. Chan, and B. Zhang. An explicit fourth order orthogonal curvilinear staggered-grid FDTD method for Maxwell's equations. *J. Comput. Phys.*, 175:739–763, 2002.

27. R. F. Harrington. Matrix methods for field problems. *Proc. IEEE*, 55:136–149, 1967.

28. R. F. Harrington. *Field Computation by Moment Methods*. MacMillan, New York, 1968.

29. M. M. Ney. Method of moments as applied to electromagnetics problems. *IEEE Trans. Microwave Theory Tech.*, 33(10):972–980, 1985.

30. S. M. Rao, D. R. Wilton, and A. W. Glisson. Electromagnetic scattering by surfaces of arbitrary shape. *IEEE Trans. Ant. Prop.*, 30(3):409–418, 1982.

31. G. J. Burke and A. J. Poggio. *Numerical Electromagnetic Code (NEC) – Method of Moments*. Lawrence Livermore National Laboratory, Livermore, California, Jan. 1981.

32. F. Edelvik. A survey of finite volume time domain methods for solving Maxwell's equations. Technical Report 98:07, Parallel and Scientific Computing Institute, Royal Institute of Technology and Uppsala University, 1998.

33. N. Madsen and R. Ziolkowski. A 3 dimensional modified finite volume techniqe for Maxwell's equations. *Electromagnetics*, 10(1):147–161, 1990.

34. R. Holland, V. Cable, and L. Wilson. Finite-volume time-domain techniques for EM scattering. *IEEE Trans. Electromagnetic Compatibility*, 33(4):281–294, 1991.

35. N. K. Madsen. Divergence preserving discrete surface integral method for Maxwell's curl equations using non-orthogonal unstructured grids. *J. Comput. Phys.*, 119(1):34–45, 1995.

36. V. Shankar, A. Mohammadian, and W. Hall. A time-domain finite volume treatment for the Maxwell equations. *Electromagnetics*, 10(1):127–145, 1990.

37. A. H. Mohammadian, V. Shankar, and W. F. Hall. Computation of electromagnetic scattering and radiation using a time-domain finite volume discretization procedure. *Comput. Phys. Comm.*, 68:175–196, 1991.

38. P. Bonnet, X. Ferrieres, and F. Issac. Numerical modeling of scattering problems using a time domain finite volume method. *J. Electromagn. Waves and Appl.*, 11:1165–1189, 1997.

39. P. Silvester. Finite element solution of homogeneous waveguide problems. *Alta Frequenza*, 38.

40. O. W. Anderson. Laplacian electrostatic field calculations by finite elements with automatic grid generation. *IEEE Trans. Power Apparatus Syst.*, 92(5):1485–1492, 1973.

41. P. P. Silvester and R. L. Ferrari. *Finite Elements for Electrical Engineers*. Cambridge University Press, Cambridge, 1983.

42. M. N. O. Sadiku. A simnple introduction to finite element analysis of electromagnetic problems. *IEEE Trans. Edu.*, 32(2).

43. A. Konrad. Vector variational formulation of electromagnetic fields in anisotropic media. *IEEE Trans. Microwave Theory Tech.*, 24:553–559, 1976.

44. Z. J. Cendes and P. Silvester. Numerical solution of dielectric loaded waveguides: I – Finite element analysis. *IEEE Trans. Microwave Theory Tech.*, 18(6):1124–1131, 1971.

45. M. Hara, T. Wada, T. Fukasawa, and F. Kikuchi. Three-dimensional analysis of rf electromagnetic field by the finite element method. *IEEE Trans. Mag.*, 19:2417–2420, 1983.

46. M. Hano. Finite element analysis of dielectric-loaded waveguides. *IEEE Trans. Microwave Theory Tech.*, 32(10):1275–1279, 1984.

47. A. Bossavit. Solving Maxwell equations in a closed cavity, and the question of spurious modes. *IEEE Trans. Mag.*, 26(2):702–705, 1990.

48. D. R. Lynch and K. D. Paulsen. Origin of vector parasites in numerical Maxwell solutions. *IEEE Trans. Microwave Theory Tech.*, 39(3):383–394, 1991.

49. K. D. Paulsen, W. E. Boyse, and D. R. Lynch. Continuous potential Maxwell solutions on nodal-based finite elements. *IEEE Trans. Ant. Prop.*, 40(10):1192–1200, 1992.

50. D. R. Lynch, K. D. Paulsen, and W. E. Boyse. Synthesis of vector parasites in numerical Maxwell solutions. *IEEE Trans. Microwave Theory Tech.*, 41(8):1439–1448, 1993.

51. D. Sun, J. Manges, X. Yuan, and Z. Cendes. Spurious modes in finite element methods. *IEEE Ant. Prop. Magazine*, 37(5):12–24, 1995.

52. R. L. Lee and N. K. Madsen. A mixed finite element formulation for Maxwell's equations in the time domain. *J. Comput. Phys.*, 88:284–304, 1990.

53. D. R. Lynch and K. D. Paulsen. Time domain integration of the maxwell equations on finite elements. *IEEE Trans. Ant. Prop.*, 38:1933–1942, 1990.

54. J. J. Amborsiano, S. T. Brandon, R. Lohner, and C. R. DeVore. Electromagnetics via the Taylor-Galerkin finite element method on unstructured grids. *J. Comput. Phys.*, 110:310–319, 1994.

55. J. R. Winkler and J. B. Davies. Elimination of spurious modes in finite element analysis. *J. Comput. Phys.*, 56:1–14, 1984.

56. K. D. Paulsen and D. R. Lynch. Elimination of vector parasites in finite element Maxwell solutions. *IEEE Trans. Microwave Theory Tech.*, 39(3):395–404, 1991.

57. J. Jin. *The Finite Element Method in Electromagnetics*. Wiley, 1993.

58. J. C. Nédélec. Mixed finite elements in R3. *Numer. Math.*, 35:315–341, 1980.

59. P.A. Raviart and J.M. Thomas. A Mixed Finite Element Method for $2^{nd}$ Order Elliptic Problems. In I. Galligani and E. Mayera, editors, *Mathematical Aspects of the Finite Element Method*, volume 606 of *Lect. Notes. on Mathematics*, pages 293–315. Springer Verlag, 1977.

60. A. Bossavit. Whitney forms: a class of finite elements for three-dimensional computations in electromagnetism. *IEEE Proceedings.*, 135(8):493–500, 1988.

61. C. Crowley, P. Silvester, and H. Hurwitz. Covariant projection elements for 3D vector field problems. *IEEE Trans. Mag.*, 24(1):397–400, 1988.

62. R. Miniowitz and J. P. Webb. Covariant-projection quadrilateral elements for the analysis of waveguides with sharp edges. *IEEE Trans. Microwave Theory Tech.*, 39(3):501–505, 1991.

63. J. P. Webb and R. Miniowitz. Analysis of 3D microwave resonators using covariant-projection elements. *IEEE Trans. Microwave Theory Tech.*, 39(11):1895–1899, 1991.

64. A. Bossavit and I. Mayergoyz. Edge-elements for scattering problems. *IEEE Trans. Mag.*, 25(4):2816–2821, 1989.

65. J. P. Webb. Edge elements and what they can do for you. *IEEE Trans. Mag.*, 29(2):1460–1465, 1993.

66. A. Bossavit. Edge elements for magnetostatics. *Int. J. Numer. Model. El.*, 9(1–2):19–34, 1996.

67. K. Mahadevan, R. Mittra, and P. M. Vaidya. Use of Whitney edge and face elements for efficient finite-element time domain solution of Maxwell equations. *J. Electromagn. Waves and Appl.*, 8(9).

68. J. Lee, D. Sun, and Z. Cendes. Tangential vector finite elements for electromagnetic field computation. *IEEE Trans. Mag.*, 27(5):4032–4035, 1991.

69. P. Monk. On the $p$-extension and $hp$-extension of Nedelec curl-conforming elements. *J. Comput. Appl. Math.*, 53(1):117–137, 1994.

70. Z. J. Cendes. Vector finite elements for electromagnetic field computation. *IEEE Trans. Mag.*, 27(5):3958–3966, 1991.

71. Z. Ren and N. Ida. High order differential form-based elements for the computation of electromagnetic field. *IEEE Trans. Mag.*, 36(4):1472–1478, 2001.

72. M. Hano, T. Miyamura, and M. Hotta. Finite element eddy current analysis by novel mixed-order vector elements. *Internat. J. Appl. Electrom.*, 14(1).

73. J. S. Wang and N. Ida. Curvilinear and higher order 'edge' finite elements in electromagnetic field computation. *IEEE Trans. Mag.*, 29(2):1491–1494, 1993.

74. J. S. Savage and A F. Peterson. Higher-order vector finite elements for tetrahedral cells. *IEEE Trans. Microwave Theory Tech.*, 44(6).

75. T. V. Yioultsis and T. D. Tsiboukis. Multiparametric vector finite elements: a systematic approach to the construction of three-dimensional higher order vector shape functions. *IEEE Trans. Mag.*, 32(3):1389–1392, 1996.

76. T. V. Yioultsis and T. D. Tsiboukis. Development and implementation of second and third order vector finite elements in various 3D electromagnetic field problems. *IEEE Trans. Mag.*, 33(2):1812–1815, 1997.

77. R. Graglia, P. Savi, and D. R. Wilton. Higher order modeling for computational electromagnetics. Barcelona, September 2000. European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS.

78. R. Graglia, D. Wilton, and A. Peterson. Higher order interpolatory vector bases for computational electromagnetics. *IEEE Trans. Ant. Prop.*, 45(3):329–342, 1997.

79. L.S. Andersen and J.L. Volakis. Development and application of a novel class of hierarchical tangential vector finite elements for electromagnetics. *IEEE Trans. Ant. Prop.*, 47(1):112–120, 1999.

80. J. P. Webb. Hierarchal vector basis functions of arbitrary order for triangular and tetrahedral finite elements. *IEEE Trans. Ant. Prop.*, 47(8):1244–1253, 1999.

81. M. Ainsworth and J. Coyle. Hierarchic finite element bases on unstructured tetrahedral meshes. May 2002.

82. J. Dao and J. Jin. A general approach for the stability analysis of the time domain finite element method for electromagnetic simulations. *IEEE Trans. Ant. Prop.*, 50(11):1624–1632, 2002.

83. J. M. Hyman and M. J. Shashkov. Adjoint operators for the natural discretizations of the divergence, gradient and curl on logically rectangular grids. *Appl. Numer. Math.*, 25(4):413–442, 1997.

84. J. M. Hyman and M. J. Shashkov. Natural discretizations for the divergence, gradient, and curl on logically rectangular grids. *Comput. Math. Appl.*, 33(4):81–104, 1997.

85. J. M. Hyman and M. J. Shashkov. Mimetic discretizations for maxwell's equations. *J. Comput. Phys.*, 151(2):881–909, 1999.

86. R. Hiptmair. Canonical construction of finite elements. *Math. Comp.*, 68(228):1325–1346, 1999.

87. G. Deschamps. Electromagnetics and differential forms. *IEEE Proceedings.*, 69(6):676–687, 1981.

88. D. Baldomir. Differential forms and electromagnetism in 3-dimensional Euclidean space R3. *IEEE Proceedings.*, 133(3):139–143, 1986.

89. W. Burke. *Applied Differential Geometry: Variational Formulation*. Cambridge University Press, 1985.

90. A. Bossavit. *Computational Electromagnetism: Variational Formulation, Complementarity, Edge Elements.* Academic Press, 1998.

91. K. Warnick, R. Selfridge, and D. Arnold. Teaching electromagnetic field theory using differential forms. *IEEE Trans. Edu.*, 40(1):53–68, 1997.

92. G. Rodrigue and D. White. A vector finite element time-domain method for solving maxwell's equations on unstructured hexahedral grids. *SIAM J. Sci. Comp.*, 23(3):683–706, 2001.

93. P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. North-Holland, 1978.

94. P. Thoma. Numerical stability of finite difference time domain methods. *IEEE Trans. Mag.*, 34(5):2740–2743, 1998.

95. F. L. Teixeira and W. C. Chew. Lattice electromagnetic theory from a topological viewpoint. *J. Math. Phys.*, 40(1):169–187, 1999.

96. S. D. Gedney and J. A. Roden. Numerical stability of nonorthogonal FDTD methods. *IEEE Trans. Ant. Prop.*, 48(2):231–239, 2000.

97. G. S. Smith. *An Introduction to Classical Electromagnetic Radiation*. Cambridge University Press, Cambridge, 1997.

98. E. W. Weisstein. web page article entitled 'Exact Sequence'. From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/ExactSequence.html.

99. R. Rieben, D. White, and G. Rodrigue. Improved conditioning of finite element matrices using new high order interpolatory bases. *IEEE Trans. Ant. Prop.*, December 2004. in press.

100. L. Brutman. Lebesgue functions for polynomial interpolation - a survey. *Annals of Numerical Mathematics*, 4, 1997.

101. J. Hesthaven and C. Teng. Stable spectral methods on tetrahedral elements. *SIAM J. Sci. Comp.*, 21(6):2352–2380, 2000.

102. R. Rieben, D. White, and G. Rodrigue. Generalized high order interpolatory 1-form bases for computational electromagnetics. In *Proceedings of the 2002 IEEE International Antennas and Propagation Symposium*, volume 4, pages 686–689, San Antonio, Texas, June 2002.

103. M. R. Spiegel. *Vector Analysis and an Introduction to Tensor Analysis*. McGraww-Hill, 1959.

104. P. Monk. An analysis of Nédélec's method for the spatial discretization of Maxwell's equations. *J. Comput. Appl. Math.*, 47:101–121, 1993.

105. R. Rieben, D. White, and G. Rodrigue. Arbitrary order hierarchical vector bases for hexahedrons. In *Proceedings of the 2003 IEEE International Antennas and Propagation Symposium*, volume 2, pages 972–976, Columbus, Ohio, June 2003.

106. P. R. Kotiuga. Helicity functionals and metric invariance in three dimensions. *IEEE Trans. Mag.*, 25(4):2813–2815, 1989.

107. J. A. Gallian. *Contemporary Abstract Algebra*. fourth edition, 1998.

108. P. Castillo, R. Rieben, and D. White. FEMSTER: An object oriented class library of discrete differential forms. In *Proceedings of the 2003 IEEE International Antennas and Propagation Symposium*, volume 2, pages 181–184, Columbus, Ohio, June 2003.

109. P. Castillo, J.Koining, R. Rieben, M. Stowell, and D. White. Discrete differential forms: A novel methodology for robust computational electromagnetics. Technical Report UCRL-ID-151522, Lawrence Livermore National Laboratory, Center for Applied Scientific Computing, January 2003.

110. B. Stroustrup. *C++ Programming Language*. Addison-Wesley, Reading, MA, 1991.

111. R. Abraham, J. E. Marsden, and T. Ratiu. *Manifolds, Tensor Analysis, and Applications*. Applied Mathematical Sciences. 1996.

112. D. Ruth. A canonical integration technique. *IEEE Trans. Nuc. Sci.*, (4):2669–2671, 1983.

113. E. Forest and D. Ruth. Fourth-order symplectic integration. *Physica D*, 43(1):105–117, 1990.

114. J. M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian Problems*. Chapman and Hall, 1994.

115. I. Saitoh, Y. Suzuki, and N. Takahashi. The symplectic finite difference time domain method. *IEEE Trans. Mag.*, 37(5):3251–3254, 2001.

116. I. Saitoh and N. Takahashi. Stability of symplectic finite-difference time-domain methods. *IEEE Trans. Mag.*, 38(2):665–668, 2002.

117. T. Hirono, W. W. Lui, and K. Yokoyama. Time-domain simulation of electromagnetic field using a symplectic integrator. *IEEE Microwave and Guided Wave Lett.*, 7(9):279–281, 1997.

118. T. Hirono, W. W. Lui, K. Yokoyama, and S. Seki. Stability and numerical dispersion of symplectic fourth-order time-domain schemes for optical field simulation. *J. Lightwave Tech.*, 16(10):1915–1920, 1998.

119. O. C. Zienkiewicz. *The Finite Element Method in Engineering Science*. McGraw-Hill, London, UK, 1971.

120. R. Rieben, D. White, and G. Rodrigue. High order symplectic integration methods for finite element solutions to time dependent maxwell equations. *IEEE Trans. Ant. Prop.*, August 2004. in press.

121. R. Richtmyer and K. Morton. *Difference Methods for Initial Value Problems*. John Wiley and Sons, New York, 1976.

122. J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, second edition, 1980.

123. R. Courant, K. O. Friedrichs, and H. Lewy. Uber die partiellen differenzengleichungen der mathematisches physik. *Math. Ann.*, 100:32–74, 1928.

124. P. Monk. A mixed method for approximating Maxwell's equations. *SIAM J. Num. Anal.*, 28(6).

125. A. M. Stewart and A. R. Humphries. *Dynamical Systems and Numerical Analysis*. Cambridge University Press, 1996.

126. H. Yoshida. Symplectic integrators for Hamiltonian-systems – Basic theory. In *IAU Symposia (152)*, pages 407–411, Netherlands, 1992.

127. J. Candy and W. Rozmus. A symplectic integration algorithm for seperable Hamiltonian functions. *J. Comput. Phys.*, 92:230–256, 1991.

128. R. Ziolkowski. The design of Maxwellian absorbers for numerical boundary conditions and for practical applications using engineered artificial materials. *IEEE Trans. Ant. Prop.*, 45(4):656–671, 1997.

129. P. Monk. Analysis of finite element method for Maxwell's equations. *SIAM J. Num. Anal.*, 29:714–729, 1992.

130. D. A. White and J. M. Koning. Computing solenoidal eigenmodes of the vector Helmholtz equation: a novel approach. *IEEE Trans. Mag.*, 38(5):3420–3425, 2002.

131. C. Balanis. *Advanced Engineering Electromagnetics*. John Wiley and Sons, 1989.

132. M. Alléon, M. Benzi, and L. Giraud. Sparse approximate inverse preconditioning for dense linear systems arising in computational electromagnetics. *Numerical Algorithms*, 16(1):1–15, 1997.

133. E. Chow. A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM J. Sci. Comp.*, 21(5):1804–1822, 2000.

134. D. Hysom and A. Pothen. A scalable parallel algorithm for incomplete factor preconditioning. *SIAM J. Sci. Comp.*, 22(6):2194–2215, 2001.

135. M. Ainsworth. Dispersive properties of high-order Nedelec/edge element approximation of the time-harmonic Maxwell equations. *Philisophical Transactions of the Royal Society of London*, 362(1816):471–491, 2004.

136. G. Karypis and V. Kumar. A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. *J. Parallel Distr. Comp.*, 48(1):71–95, 1998.

137. T. Okoshi. *Optical Fibers*. Academic Press, New York, 1982.

138. D. R. Goff. *Fiber Optic Reference Guide*. second edition, 1999.

139. D. Marcuse. Curvature loss formula for optical fibers. *J. Opt. Soc. Am.*, 66(3):216–220, 1976.

140. W. A. Gambling, H. Matsumura, and C. M. Ragdale. Curvature and microbending losses in single-mode fibres. *Optical and Quantum Electronics*, 11(1):43–5, 1979.

141. J. D. Joannopoulos, R. D. MEade, and J. N. Winn. *Photonic Crystals: Molding the Flow of Light*. Princeton University Press, Princeton, New Jersey, 1995.

142. E. Yablonovitch. Photonic band-gap strucures. *J. Opt. Soc. Am. B*, 10(2):283–295, 1993.

143. E. Lidorikis, M. Povinelli, S. Johnson, and J. Joannopoulos. Polarization-independent linear waveguides in 3D photonic crystals. *Phys. Rev. Let.*, 91(2), 2003. Art. No. 023902.

144. R. Rieben, D. White, and G. Rodrigue. Application of novel high order time domain vector finite element method to photonic band-gap waveguides. In *Proceedings of the 2004 IEEE International Antennas and Propagation Symposium*.

145. A. Mekis, J. Chen, S. Fan, P. Villeneuve, and J. Joannopoulos. High transmission through sharp bends in photonic crystal waveguides. *Phys. Rev. Let.*, 77(18):3787–3790, 1996.

146. S. Fan and J. Joannopoulos. Analysis of guided resonances in photonic crystal slabs. *Phys. Rev. B*, 65(23), 2002.

147. H. S. Sözüer and J.P. Dowling. Photonic band calculations for woodpile structures. *J. Mod. Opt.*, 41(2):231–239, 1994.

148. R. Gajić, R. Meisels, J. Radovanović, and F. Kuchar. 3D photonic band gap structure for the Ka-U microwave range. In *Proceedings of the XI Telekomunikacioni Forum Telfor 2003*, Beograd, Sava Centar, November 2003.

149. E Özbay, A. Abeyta, G. Tuttle, M. Trinigides, R. Biswas, C.T. Chan, C.M Soukoulis, and K.M. Ho. Measurement of a three-dimensional photonic band gap in a crystal structure made of dielectric rods. *Phys. Rev. B*, 50(3):1945–1948, 1994.

150. R. Rieben, D. White, and G. Rodrigue. A high order mixed vector finite element method for solving the time dependent Maxwell equations on unstructured grids. *J. Comput. Phys.*, May 2004. in review.

151. L. Lapidus and G. F. Pinder. *Numerical Solution of Partial Differential Equations in Science and Engineering*. John Wiley and Sons, 1982.

152. P. Castillo, R. Rieben, and D. White. FEMSTER documentation. UCRL-WEB-147600. www.llnl.gov/casc/femster.

# Appendix A

# Variational Formulation and Galerkin's Method

Variational calculus is a field of mathematics which deals with functions of functions (called functionals), as opposed to ordinary calculus which deals with functions of numbers. Such functionals can for example be formed as integrals involving an unknown function and its derivatives. The interest is in extremal functions: those making the functional attain a maximum or minimum value. Mathematically, this involves finding stationary values of integrals of the form

$$I(y) = \int_a^b f(y, \dot{y}, x) dx$$

The fundamental theorem of variational calculus states the integral above has an extremum only if the Euler-Lagrange equation is satisfied

$$\frac{\partial f}{\partial y} - \frac{d}{dx}\left(\frac{\partial f}{\partial \dot{y}}\right) = 0$$

The fundamental lemma of Variational calculus states that, if

$$\int_a^b M(x)h(x)dx = 0$$

for all h(x) with continuous second partial derivatives, then

$$M(x) = 0$$

on the interval $[a, b]$

Now consider an elliptic PDE of the form

$$\mathcal{L}u - f = 0$$

for some differential operator $\mathcal{L}$ and function $u$. Furthermore, consider a finite series, or basis function approximation to the function $u$ of the form

$$u \approx u_h = \sum_{i=1}^{N} \alpha_i \Phi_i$$

where $\Phi_i$ denote the basis functions. Due to the finite nature of this series expansion, substitution of $u_h$ into the original PDE will result in a residual error

$$\mathcal{L}u_h - f = R$$

The objective of an approximation scheme is to select the undetermined coefficients $\alpha_i$ such that this residual is minimized in some sense [151]. This can be achieved by requiring the integral of the residual to evaluate to zero over some domain $v$

$$\int_v R dv = 0$$

Unfortunately, this yields only one equation for the $N$ unknown coefficients $\alpha_i$. We can however modify this constraint by introducing $N$ weighting functions denoted $w_i$. Setting the integral of each weighted residual to zero yields $N$ independent equations of the form

$$\int_v (R\, w_i) dv = 0; \;\; i = 1, \ldots, N$$

This is the so called method of weighted residuals for determining the expansions coefficients $\alpha_i$.

The Galerkin method results when the weighting functions $w_i$ are chosen to be the basis functions $\Phi_i$. Thus, the residual error constraint is of the form

$$\int_v (R\, \Phi_i) dv = 0; \;\; i = 1, \ldots, N$$

The basis functions $\Phi_i$ are formally required to be members of a complete set of functions. Because a complete set of functions can exactly represent any function of a given class, the series expansion $u_h$ is inherently capable of representing the exact solution as the number of terms in the series is increased.

As indicated by the fundamental lemma of variational calculus, a continuous function must be zero if it is orthogonal to every member of a complete set. The Galerkin method can therefore be viewed as a scheme in which the residual is explicitly made orthogonal to the complete set of basis functions.

# Appendix B

# Proof of Stability for Generalized Symplectic Method

***Definition***. Consider the two-level time differencing method

$$e^{n+1} = Q(\Delta t)e^n$$

where $Q(\Delta t)$ is the amplification matrix of the method. The two-level method is said to be stable in the interval $0 \leq \Delta t \leq \tau$ if there exists a constant $M$ such that

$$||Q(\Delta t)e|| \leq M||e||$$

for all non-zero vectors $e$ and all scalars $0 \leq \Delta t \leq \tau$.

***Lemma***. Consider the two-level method $e^{n+1} = Q(\Delta t)e^n$. Suppose $Q(\Delta t)$ is diagonalizable for $0 \leq \Delta t \leq \tau$. That is, there exists a non-singular eigenvector matrix $U(\Delta t)$ so that

$$U^{-1}(\Delta t)\, Q(\Delta t)\, U(\Delta t) = \Lambda(\Delta t)$$

where $\Lambda(\Delta t)$ is a diagonal matrix of eigenvalues. If there exists a constant $M$ so that there exists a non-singular eigenvector matrix $U(\Delta t)$ so that

$$\rho(\Lambda(\Delta t)) \leq M$$

for all $0 \leq \Delta t \leq \tau$, where $\rho$ denotes the spectral radius, then the method is table.

***Proof***. Let $e$ be an arbitrary vector. Since $Q(\Delta t)$ is diagonalizable, we have that

$$\rho(Q(\Delta t)) = ||Q(\Delta t)||_2$$

Thus

$$||Q(\Delta t)e|| \leq ||Q(\Delta t)||_2||e||_2 = \rho(Q(\Delta t))||e||_2 = \rho(\Lambda(\Delta t))||e||_2 \leq M||e||_2$$

***Lemma***. Let $M_1$ and $M_2$ be symmetric positive definite (SPD) matrices of order $n_1 \times n_1$ and $n_2 \times n_2$ respectively. Let $K$ be an $n_2 \times n_1$ rectangular matrix. Consider the matrix

$$Q = \begin{bmatrix} I & aM_1^{-1}K^T M_2 \\ -bK & I - abKM_1^{-1}K^T M_2 \end{bmatrix}$$

where $a = \alpha\Delta t$ and $b = \beta\Delta t$ are real scalars. If $M_1 = C^T C$ and $M_2 = G^T G$ are Cholesky factorizations of $M_1$ and $M_2$, then $Q$ is similar to the matrix

$$\tilde{Q} = \begin{bmatrix} I - abAA^T & -bA \\ aA^T & I \end{bmatrix}$$

where $A = GKC^{-1}$.

***Proof***. Let $P$ be a permutation matrix of the form

$$P = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$$

then, $PP = I$ and

$$Q^* = PQP = \begin{bmatrix} I - abKM_1^{-1}K^TM_2 & -bK \\ aM_1^{-1}K^TM_2 & I \end{bmatrix} = \begin{bmatrix} I - abKC^{-1}C^{-T}K^TG^TG & -bK \\ aC^{-1}C^{-T}K^TG^TG & I \end{bmatrix}$$

Now let

$$U = \begin{bmatrix} G^{-1} & 0 \\ 0 & C^{-1} \end{bmatrix}$$

Then,

$$\begin{aligned} \tilde{Q} &= (PU)^{-1}Q(PU) \\ &= U^{-1}PQPU \\ &= U^{-1}Q^*U \\ &= \begin{bmatrix} G & 0 \\ 0 & C \end{bmatrix} \begin{bmatrix} I - abKC^{-1}C^{-T}K^TG^TG & -bK \\ aC^{-1}C^{-T}K^TG^TG & I \end{bmatrix} \begin{bmatrix} G^{-1} & 0 \\ 0 & C^{-1} \end{bmatrix} \\ &= \begin{bmatrix} I - abGKC^{-1}C^{-T}K^TG^T & -bGKC^{-1} \\ aC^{-T}K^TG^T & I \end{bmatrix} \\ &= \begin{bmatrix} I - abAA^T & -bA \\ aA^T & I \end{bmatrix} \end{aligned}$$

**Theorem.** *Consider the matrix*

$$Q = \begin{bmatrix} I - abAA^T & -bA \\ aA^T & I \end{bmatrix}$$

*where A is an $n_2 \times n_1$ rectangular matrix with $n_2 < n_1$. Suppose*

1. *$rank(A) = m$*
2. *The eigenvalues $\mu$ of $abAA^T$ satisfy: $0 \leq |\mu| \leq 2$*

*Then,*

- *The eigenvalues of Q lie on the unit circle*
- *The eigenvectors of Q form an eigenbasis in $R^{n_1+n_2}$*

***Proof***. Since $abAA^T$ is symmetric, it has an orthogonal eigenbasis $\{x_i\}$ of $R^{n_2}$ corresponding to the real eigenvalues $\mu_i$. Also, since $rank(A^T) = rank(A) = m$, we see that $dim(null(A^T)) = dim(null(AA^T)) = n_2 - m$. Hence, there exists $n_2 - m$ orthogonal eigenvectors $x_1^{(0)}, \ldots, x_m^{(0)}$ so that $A^T x_i^{(0)} = 0$. Now consider the vectors

$$V_i^{(0)} = \begin{bmatrix} x_i^{(0)} \\ 0 \end{bmatrix}, \ i = 1, \ldots, n_2 - m$$

Then

$$QV_i^{(0)} = \begin{bmatrix} I - abAA^T & -bA \\ aA^T & I \end{bmatrix} \begin{bmatrix} x_i^{(0)} \\ 0 \end{bmatrix} = \begin{bmatrix} x_i^{(0)} \\ 0 \end{bmatrix}$$

That is, $V_i^{(0)}$, $i = 1, \ldots, n_2 - m$ are eigenvectors of $Q$ corresponding to the eigenvalue $\lambda = 1$.

Now let $\mu_k$, $k = 1, \ldots, M$ be the non-zero and distinct eigenvalues of $abAA^T$, each of multiplicity $m_k$. Then there exists $m_k$ orthogonal eigenvectors $\{x_i^{(k)}\}$ corresponding to the eigenvalue $\mu_k$. Let $\theta_k$ be such that

$$\mu_k = \pm 2(1 - \cos(\theta_k)) = \pm(2 - e^{i\theta_k} - e^{-i\theta_k})$$

where $\mu_k > 0$ if $ab > 0$ and $\mu_k < 0$ is $ab < 0$. We first consider the case when $ab > 0$. Consider the vectors

$$V_j^{(+k)} = \begin{bmatrix} x_j^{(k)} \\ -\frac{a}{\mu_k}(1-e^{i\theta_k})A^T x_j^{(k)} \end{bmatrix}, \quad j = 1, \ldots, m_k$$

Then

$$\begin{aligned} QV_j^{(+k)} &= \begin{bmatrix} x_j^{(k)} - \mu_k x_j^{(k)} + \frac{ab(1-e^{i\theta_k})}{\mu_k} AA^T x_j^{(k)} \\ aA^T x_j^{(k)} - a(\frac{1-e^{i\theta_k}}{\mu_k})A^T x_j^{(k)} \end{bmatrix} \\ &= \begin{bmatrix} x_j^{(k)} - (2 - e^{i\theta_k} - e^{-i\theta_k})x_j^{(k)} + (1-e^{i\theta_k})x_j^{(k)} \\ \frac{a}{\mu_k}(\mu_k - (1-e^{i\theta_k}))A^T x_j^{(k)} \end{bmatrix} \\ &= e^{-i\theta_k} \begin{bmatrix} x_j^{(k)} \\ -\frac{a}{\mu_k}(1-e^{i\theta_k})A^T x_j^{(k)} \end{bmatrix} \end{aligned}$$

That is, the vectors $V_j^{(+k)}$ are $m_k$ linearly independent eigenvectors of $Q$ corresponding to the eigenvalue $e^{-i\theta_k}$. Hence, $e^{-i\theta_k}$ is an eigenvalue of at least multiplicity $m_k$. Now consider the vectors

$$V_j^{(-k)} = \begin{bmatrix} x_j^{(k)} \\ -\frac{a}{\mu_k}(1-e^{-i\theta_k})A^T x_j^{(k)} \end{bmatrix}, \quad j = 1, \ldots, m_k$$

Then in the same manner as before, it can be shown that the vectors $V_j^{(-k)}$ are $m_k$ linearly independent eigenvectors of $Q$ corresponding to the eigenvalue $e^{i\theta_k}$. Hence, $e^{i\theta_k}$ is an eigenvalue of at least multiplicity $m_k$. Since

$$m_1 + m_2 + \ldots + m_M = n_2$$

we have exhibited

$$2(m_1 + m_2 + \ldots + m_M) = 2n_2$$

linearly independent eigenvectors of $Q$. For the case when $ab < 0$, then it can be shown in a similar manner as above that$V_j^{(\pm k)}$ are $2n_2$ linearly independent eigenvectors of $Q$ corresponding to the eigenvalues $-e^{\pm i\theta_k}$.

Now since $range(AA^T) = range(A)$ and $rank(A) = dim(range(A)) = m$, it follows that

$$dim(kernel(A)) = n_1 - m$$

Hence, there exists $n_1 - m$ linearly independent vectors $y_i$ so that $Ay_i = 0$. Consider the vectors

$$V_j = \begin{bmatrix} 0 \\ y_i \end{bmatrix}, \quad j = 1, \ldots, n_1 - m$$

Then

$$QV_j = V_j$$

so that these vectors are linearly independent eigenvectors of $Q$ corresponding to the eigenvalue $\lambda = 1$. Hence, we have now exhibited

$$2(m_1 + m_2 + \ldots + m_M) + (n_1 - m) + (n_2 - m) = 2n_2 + n_1 - n_2 = n_1 + n_2$$

linearly independent eigenvectors of the matrix $Q$ and the theorem is proved.

# Appendix C

# Tabulation of Second Order Interpolatory Basis Functions

The following tables contain explicit tabulations of the second order ($p = 2$) interpolatory basis functions sorted according to the FEMSTER hexahedron standard [152].

| Basis ID | Local Edge ID | 1-form Edge Basis Function |
|---|---|---|
| 1 | 1 | $-\left(\left(1-3x+2x^2\right)\left(1-3y+2y^2\right)(-1+z)\right)\hat{\mathbf{z}}$ |
| 2 | 1 | $\left(1-3x+2x^2\right)\left(1-3y+2y^2\right)z\,\hat{\mathbf{z}}$ |
| 3 | 2 | $-\left(\left(1-3x+2x^2\right)y\,(-1+2y)\,(-1+z)\right)\hat{\mathbf{z}}$ |
| 4 | 2 | $\left(1-3x+2x^2\right)y\,(-1+2y)\,z\,\hat{\mathbf{z}}$ |
| 5 | 3 | $-\left(x\,(-1+2x)\left(1-3y+2y^2\right)(-1+z)\right)\hat{\mathbf{z}}$ |
| 6 | 3 | $x\,(-1+2x)\left(1-3y+2y^2\right)z\,\hat{\mathbf{z}}$ |
| 7 | 4 | $-\left(x\,(-1+2x)\,y\,(-1+2y)\,(-1+z)\right)\hat{\mathbf{z}}$ |
| 8 | 4 | $x\,(-1+2x)\,y\,(-1+2y)\,z\,\hat{\mathbf{z}}$ |
| 9 | 5 | $-\left(\left(1-3x+2x^2\right)(-1+y)\left(1-3z+2z^2\right)\right)\hat{\mathbf{y}}$ |
| 10 | 5 | $\left(1-3x+2x^2\right)y\left(1-3z+2z^2\right)\hat{\mathbf{y}}$ |
| 11 | 6 | $-\left(\left(1-3x+2x^2\right)(-1+y)\,z\,(-1+2z)\right)\hat{\mathbf{y}}$ |
| 12 | 6 | $\left(1-3x+2x^2\right)yz\,(-1+2z))\,\hat{\mathbf{y}}$ |
| 13 | 7 | $-\left(x\,(-1+2x)\,(-1+y)\left(1-3z+2z^2\right)\right)\hat{\mathbf{y}}$ |
| 14 | 7 | $x\,(-1+2x)\,y\left(1-3z+2z^2\right))\,\hat{\mathbf{y}}$ |
| 15 | 8 | $-\left(x\,(-1+2x)\,(-1+y)\,z\,(-1+2z)\right)\hat{\mathbf{y}}$ |
| 16 | 8 | $x\,(-1+2x)\,yz\,(-1+2z)\,\hat{\mathbf{y}}$ |
| 17 | 9 | $-\left((-1+x)\left(1-3y+2y^2\right)\left(1-3z+2z^2\right)\right)\hat{\mathbf{x}}$ |
| 18 | 9 | $x\left(1-3y+2y^2\right)\left(1-3z+2z^2\right)\hat{\mathbf{x}}$ |
| 19 | 10 | $-\left((-1+x)\left(1-3y+2y^2\right)z\,(-1+2z)\right)\hat{\mathbf{x}}$ |
| 20 | 10 | $x\left(1-3y+2y^2\right)z\,(-1+2z)\,\hat{\mathbf{x}}$ |
| 21 | 11 | $-\left((-1+x)\,y\,(-1+2y)\left(1-3z+2z^2\right)\right)\hat{\mathbf{x}}$ |
| 22 | 11 | $xy\,(-1+2y)\left(1-3z+2z^2\right)\hat{\mathbf{x}}$ |
| 23 | 12 | $-\left((-1+x)\,y\,(-1+2y)\,z\,(-1+2z)\right)\hat{\mathbf{x}}$ |
| 24 | 12 | $xy\,(-1+2y)\,z\,(-1+2z)\,\hat{\mathbf{x}}$ |

Table C.1: Second order 1-form interpolatory edge basis functions on the reference hexahedron.

| Basis ID | Local Face ID | 1-form Face Basis Function |
|---|---|---|
| 25 | 1 | $4\,(-1+x)\,x\,(-1+y)\,\left(1-3z+2z^2\right)\,\hat{\mathbf{y}}$ |
| 26 | 1 | $-4\,(-1+x)\,xy\,\left(1-3z+2z^2\right)\,\hat{\mathbf{y}}$ |
| 27 | 1 | $4\,(-1+x)\,(-1+y)\,y\,\left(1-3z+2z^2\right)\,\hat{\mathbf{x}}$ |
| 28 | 1 | $-4x\,(-1+y)\,y\,\left(1-3z+2z^2\right)\,\hat{\mathbf{x}}$ |
| 29 | 2 | $4\,(-1+x)\,x\,(-1+y)\,z\,(-1+2z)\,\hat{\mathbf{y}}$ |
| 30 | 2 | $-4\,(-1+x)\,xyz\,(-1+2z)\,\hat{\mathbf{y}}$ |
| 31 | 2 | $4\,(-1+x)\,(-1+y)\,yz\,(-1+2z)\,\hat{\mathbf{x}}$ |
| 32 | 2 | $-4x\,(-1+y)\,yz\,(-1+2z)\,\hat{\mathbf{x}}$ |
| 33 | 3 | $4\,(-1+x)\,x\,\left(1-3y+2y^2\right)\,(-1+z)\,\hat{\mathbf{z}}$ |
| 34 | 3 | $-4\,(-1+x)\,x\,\left(1-3y+2y^2\right)\,z\,\hat{\mathbf{z}}$ |
| 35 | 3 | $4\,(-1+x)\,\left(1-3y+2y^2\right)\,(-1+z)\,z\,\hat{\mathbf{x}}$ |
| 36 | 3 | $-4x\,\left(1-3y+2y^2\right)\,(-1+z)\,z\,\hat{\mathbf{x}}$ |
| 37 | 4 | $4\,(-1+x)\,xy\,(-1+2y)\,(-1+z)\,\hat{\mathbf{z}}$ |
| 38 | 4 | $-4\,(-1+x)\,xy\,(-1+2y)\,z\,\hat{\mathbf{z}}$ |
| 39 | 4 | $4\,(-1+x)\,y\,(-1+2y)\,(-1+z)\,\hat{\mathbf{x}}$ |
| 40 | 4 | $-4xy\,(-1+2y)\,(-1+z)\,z\,\hat{\mathbf{x}}$ |
| 41 | 5 | $4\,\left(1-3x+2x^2\right)\,(-1+y)\,y\,(-1+z)\,\hat{\mathbf{z}}$ |
| 42 | 5 | $-4\,\left(1-3x+2x^2\right)\,(-1+y)\,yz\,\hat{\mathbf{z}}$ |
| 43 | 5 | $4\,\left(1-3x+2x^2\right)\,(-1+y)\,(-1+z)\,z\,\hat{\mathbf{y}}$ |
| 44 | 5 | $-4\,\left(1-3x+2x^2\right)\,y\,(-1+z)\,z\,\hat{\mathbf{y}}$ |
| 45 | 6 | $4x\,(-1+2x)\,(-1+y)\,y\,(-1+z)\,\hat{\mathbf{z}}$ |
| 46 | 6 | $-4x\,(-1+2x)\,(-1+y)\,yz\,\hat{\mathbf{z}}$ |
| 47 | 6 | $4x\,(-1+2x)\,(-1+y)\,(-1+z)\,z\,\hat{\mathbf{y}}$ |
| 48 | 6 | $-4x\,(-1+2x)\,y\,(-1+z)\,z\,\hat{\mathbf{y}}$ |

Table C.2: Second order 1-form interpolatory face basis functions on the reference hexahedron.

| Basis ID | 1-form Cell (Interior) Basis Function |
|---|---|
| 49 | $-16\,(-1+x)\,x\,(-1+y)\,y\,(-1+z)\,\hat{\mathbf{z}}$ |
| 50 | $16\,(-1+x)\,x\,(-1+y)\,yz\,\hat{\mathbf{z}}$ |
| 51 | $-16\,(-1+x)\,x\,(-1+y)\,(-1+z)\,z\,\hat{\mathbf{y}}$ |
| 52 | $16\,(-1+x)\,xy\,(-1+z)\,z\,\hat{\mathbf{y}}$ |
| 53 | $-16\,(-1+x)\,(-1+y)\,y\,(-1+z)\,z\,\hat{\mathbf{x}}$ |
| 54 | $16x\,(-1+y)\,y\,(-1+z)\,z\,\hat{\mathbf{x}}$ |

Table C.3: Second order 1-form interpolatory cell (or interior) basis functions on the reference hexahedron.

| Basis ID | Local Face ID | 2-form Face Basis Function |
|---|---|---|
| 1 | 1 | $(-1+x)\,(-1+y)\,\left(1-3z+2z^2\right)\,\hat{\mathbf{Z}}$ |
| 2 | 1 | $-\left(x\,(-1+y)\,\left(1-3z+2z^2\right)\right)\,\hat{\mathbf{Z}}$ |
| 3 | 1 | $-\left((-1+x)\,y\,\left(1-3z+2z^2\right)\right)\,\hat{\mathbf{Z}}$ |
| 4 | 1 | $xy\,\left(1-3z+2z^2\right)\,\hat{\mathbf{Z}}$ |
| 5 | 2 | $(-1+x)\,(-1+y)\,z\,(-1+2z)\,\hat{\mathbf{Z}}$ |
| 6 | 2 | $-\left(x\,(-1+y)\,z\,(-1+2z)\right)\,\hat{\mathbf{Z}}$ |
| 7 | 2 | $-\left((-1+x)\,yz\,(-1+2z)\right)\,\hat{\mathbf{Z}}$ |
| 8 | 2 | $xyz\,(-1+2z)\,\hat{\mathbf{Z}}$ |
| 9 | 3 | $(-1+x)\,\left(1-3y+2y^2\right)\,(-1+z)\,\hat{\mathbf{Y}}$ |
| 10 | 3 | $-\left(x\,\left(1-3y+2y^2\right)\,(-1+z)\right)\,\hat{\mathbf{Y}}$ |
| 11 | 3 | $-\left((-1+x)\,\left(1-3y+2y^2\right)\,z\right)\,\hat{\mathbf{Y}}$ |
| 12 | 3 | $x\,\left(1-3y+2y^2\right)\,z\,\hat{\mathbf{Y}}$ |
| 13 | 4 | $(-1+x)\,y\,(-1+2y)\,(-1+z)\,\hat{\mathbf{Y}}$ |
| 14 | 4 | $-\left(xy\,(-1+2y)\,(-1+z)\right)\,\hat{\mathbf{Y}}$ |
| 15 | 4 | $-\left((-1+x)\,y\,(-1+2y)\,z\right)\,\hat{\mathbf{Y}}$ |
| 16 | 4 | $xy\,(-1+2y)\,z\,\hat{\mathbf{Y}}$ |
| 17 | 5 | $\left(1-3x+2x^2\right)\,(-1+y)\,(-1+z)\,\hat{\mathbf{X}}$ |
| 18 | 5 | $-\left(\left(1-3x+2x^2\right)\,y\,(-1+z)\right)\,\hat{\mathbf{X}}$ |
| 19 | 5 | $-\left(\left(1-3x+2x^2\right)\,(-1+y)\,z\right)\,\hat{\mathbf{X}}$ |
| 20 | 5 | $\left(1-3x+2x^2\right)\,yz\,\hat{\mathbf{X}}$ |
| 21 | 6 | $x\,(-1+2x)\,(-1+y)\,(-1+z)\,\hat{\mathbf{X}}$ |
| 22 | 6 | $-\left(x\,(-1+2x)\,y\,(-1+z)\right)\,\hat{\mathbf{X}}$ |
| 23 | 6 | $-\left(x\,(-1+2x)\,(-1+y)\,z\right)\,\hat{\mathbf{X}}$ |
| 24 | 6 | $x\,(-1+2x)\,yz\,\hat{\mathbf{X}}$ |

Table C.4: Second order 2-form interpolatory face basis functions on the reference hexahedron.

| Basis ID | 2-form Cell (Interior) Basis Function |
|---|---|
| 25 | $-4\,(-1+x)\,(-1+y)\,(-1+z)\,z\,\hat{\mathbf{Z}}$ |
| 26 | $4x\,(-1+y)\,(-1+z)\,z\,\hat{\mathbf{Z}}$ |
| 27 | $4\,(-1+x)\,y\,(-1+z)\,z\,\hat{\mathbf{Z}}$ |
| 28 | $-4xy\,(-1+z)\,z\,\hat{\mathbf{Z}}$ |
| 29 | $-4\,(-1+x)\,(-1+y)\,y\,(-1+z)\,\hat{\mathbf{Y}}$ |
| 30 | $4x\,(-1+y)\,y\,(-1+z)\,\hat{\mathbf{Y}}$ |
| 31 | $4\,(-1+x)\,(-1+y)\,yz\,\hat{\mathbf{Y}}$ |
| 32 | $-4x\,(-1+y)\,yz\,\hat{\mathbf{Y}}$ |
| 33 | $-4\,(-1+x)\,x\,(-1+y)\,(-1+z)\,\hat{\mathbf{X}}$ |
| 34 | $4\,(-1+x)\,xy\,(-1+z)\,\hat{\mathbf{X}}$ |
| 35 | $4\,(-1+x)\,x\,(-1+y)\,z\,\hat{\mathbf{X}}$ |
| 36 | $-4\,(-1+x)\,xyz\,\hat{\mathbf{X}}$ |

Table C.5: Second order 2-form interpolatory cell (or interior) basis functions on the reference hexahedron.