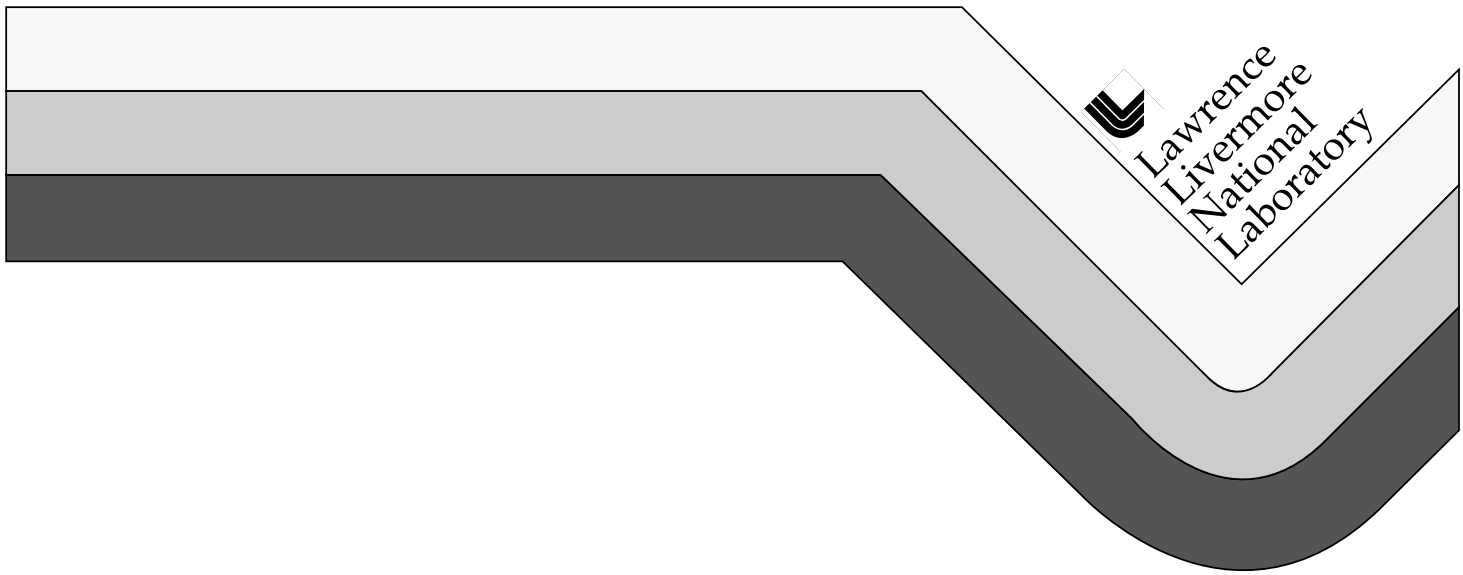
UCRL-LR-128238

Discrete Time Vector Finite Element Methods for Solving Maxwell's Equations on 3D Unstructured Grids

Daniel Arthur White
PhD Thesis

September 1997



Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.

UCRL-LR-128238
Distribution Category UC-705

Discrete Time Vector Finite Element Methods for Solving Maxwell's Equations on 3D Unstructured Grids

Daniel Arthur White

Doctor of Philosophy
Thesis

Manuscript date: September 1997

LAWRENCE LIVERMORE NATIONAL LABORATORY
University of California • Livermore, California • 94551

Daniel Arthur White
September 1997
Engineering - Applied Science

Discrete Time Vector Finite Element Methods for Solving Maxwell's Equations on 3D Unstructured Grids

Abstract

The primary goal of this research was to develop a method for solving Maxwell's equations on unstructured three dimensional grids that is provably stable, energy conserving, and charge conserving. In this dissertation a method, called the Discrete Time Vector Finite Element Method (DTVFEM) is derived, analyzed, and validated. The DTVFEM uses covariant vector finite elements as a basis for the electric field and contravariant vector finite elements as a basis for the magnetic flux density. These elements are complimentary in the sense that the covariant elements have tangential continuity across interfaces whereas the contravariant elements have normal continuity across interfaces. The Galerkin approximation is used to convert Ampere's and Faraday's law to a coupled system of ordinary differential equations. The leapfrog method is used to advance the fields in time.

Like most finite element methods the DTVFEM requires that a linear system be solved at every time step. A significant part of this dissertation addresses the solution of the large, sparse, unstructured matrices that arise in the DTVFEM. The DTVFEM was implemented in software and installed on a variety of computers, including two massively parallel supercomputers. Several computational experiments were performed to determine the accuracy and efficiency of the method.

Discrete Time Vector Finite Element Methods for Solving Maxwell's Equations on 3D Unstructured Grids

by

Daniel Arthur White

B.S.E.E. (University of California, Davis) 1985

M.S.E.E (University of California, Davis) 1986

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Engineering - Applied Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

_____

_____

_____

Committee in Charge

1997

Acknowledgments

I wish to thank my advisor Garry Rodrigue for encouraging me to apply to graduate school, for teaching me much of what I know about the numerical solution of partial differential equations, for helping me prepare for my oral examination, and for encouraging me to pursue my own research interests. I would also like to thank Niel Madsen, Grant Cook, David Steich, and Steve Ashby of Lawrence Livermore National Laboratory for many various consultations during the last three years.

I am indebted to the friendly administrative staff of the Department of Applied Science constantly reminding me of the various (and ever changing) rules, regulations, and deadlines. It would have been a bummer to get kicked out of school for forgetting to register for classes!

I am also indebted to the mysterious Student Policy Committee of LLNL for deeming me worthy of a fellowship. Without this financial assistance I would not be writing this dissertation today.

The Institute of Scientific Computing Research at LLNL provided me with the (absolute minimum) tools required to pursue to my research, such as an office, a phone, a computer, etc. The staff at ISCR was the best; they treated me as a peer rather than a lowly graduate student. I thank the staff of ISCR for the many lunch-hour debates and discussions I enjoyed during the last three years.

Life in Livermore would have been a drag without some fellow students who shared a common interest in skiing, biking, climbing, hiking, and consumption of brewed beverages. Thanks for the good times.

Finally I thank my wife Teresita for her tolerance, her moral support, and her holding down a real job during these past four years. Now I have to keep my part of the bargain.

Table of Contents

List of Figures

List of Tables

1.0 Introduction and Survey

Electromagnetic field theory is concerned with the study of charges, at rest and in motion, that produce currents and electromagnetic fields. The ancient Greeks studied magnetism and optics as two unrelated physical phenomena. Later, in the nineteenth century, the basic laws of electricity and magnetism were formulated through experiments conducted by Faraday, Ampere, Gauss, Lenz, Coulomb, Volta, and others. These basic laws of electricity and magnetism were combined into a consistent set of coupled linear partial differential equations (PDE's) by Maxwell in 1873. These equations, referred to as Maxwell's equations, completely describe the time evolution of macroscopic electromagnetic fields. These equations were later verified experimentally by Hertz. Einstein and his special theory of relativity provided further evidence that Maxwell's equations are a correct model of physical reality.

Modern day physicists still research the origin of the electromagnetic field and the interaction of fields with matter, however it is primarily electrical engineers who are concerned with the solution of Maxwell's equations. Generation and control of electromagnetic fields is of paramount importance in our society. Multifarious systems such as radio and television, telescopes and microscopes, radar systems and the global positioning system, fiber optic communication and optical CD-ROM, linear accelerators and plasma processing reactors, cardiac defibrillators and microwave hyperthermia devices all depend upon electromagnetic fields. It is extremely difficult and expensive, if not impossible, to design a modern electromagnetic system solely through trial-end-error experimentation in the laboratory. Thus electrical engineers have long sought out solutions to Maxwells equations.

But the solution of Maxwell's equations is not entirely within the domain of electrical engineers, for electromagnetic fields are central to many naturally occurring phenomena. For example geophysicists study the nature of the earth's magnetic field and its interaction with the solar wind. Astrophysicists study very large scale electromagnetic fields which are not only responsible for the beauty of galactic nebula, but may also play a part in the formation of solar systems such as our own. Perhaps the most intriguing electromagnetic fields are those that occur naturally within us. The heart, eye, and central nervous system cannot be fully comprehended without an understanding of electromagnetics. Thus the ability to solve Maxwells equations is important not only to the advancement of technology, but also to the advancement of many scientific disciplines.

Shortly after Maxwell wrote down his famous PDE's work began on solving the equations. There are many situations where one can derive an exact, closed form solution to Maxwell's equations. But many problems of interest do not admit to exact solutions and one must accept an approximate solution. Approximate solutions come in two flavors; analytical approximate solutions and numerical approximate solutions. Examples of the former include geometrical optics and physical optics, which are so called "high frequency" approximations that are valid for asymptotically large frequencies; examples of the latter include finite difference and finite element methods, which are direct numerical approximations of the PDE. Direct numerical approximations have become increasingly important of late, for two reasons: 1) analytical approximate solutions are not accurate enough for many electromagnetic design tasks, and 2) the advancement of computing technology has made direct numerical approximations feasible. This dissertation is concerned with the direct numerical solution of Maxwell's equations.

It is not possible to develop a numerical method that is suitable for every imaginable electromagnetic problem. It is necessary to classify electromagnetic problems and then develop methods applicable for all problems within a given class. Electromagnetic problems can be dichotomized into those that involve free charges and/or conducting fluids and plasma, and those that do not. This dissertation is concerned with the latter. A further dichotomization is static problems versus dynamic problems. Dynamic problems involve the generation, propagation, scattering, and absorption of electromagnetic waves. There are two distinct approaches for solving dynamic problems, one being to work in the frequency domain, the other to work directly in the time domain. The approach taken in this dissertation is to work directly in the time domain. It has been argued in the electromagnetics community that time domain methods are more general in that they are applicable to non-linear and/or time dependent problems, whereas frequency domain approaches are not. It has also been argued that time domain approaches are more computationaly efficient for many problems of interest. This author makes no such arguments. Rather, the time domain approach is taken simply because it is more interesting to the author.

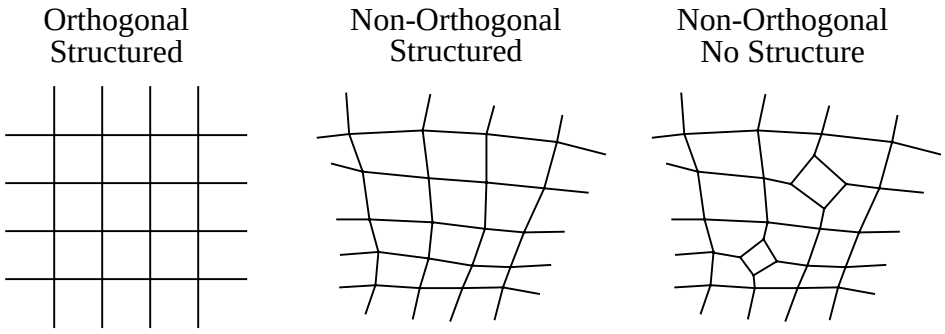
A key parameter in and classification of electromagnetic problems is the D/λ, the ratio of the characteristic size of the geometry to the characteristic wavelength. If this ratio is very much greater than unity the problem is considered a high frequency, or optics, problem. For example consider the design of a Newtonian telescope, where D is dimension of the aperture and λ is the wavelength of visible light. In this case D/λ is on the order of 10^6, therefore the design of a Newtonian telescope is most definitely considered an optics problem. As another example consider the scattering of a 10 GHz radar signal from a large air-

craft. In this case D/λ is on the order of 10^3, therefore this particular scattering problem is within the high frequency regime. The design of a fiber optic waveguide, on the other hand, is not considered a high frequency problem because D/λ is on the order of unity. As a rule of thumb high frequency problems are analyzed using approximate analytical methods such physical optics and physical theory of diffraction, and optics problems are analyzed using geometrical optics and geometrical theory of diffraction. This dissertation is not concerned with high frequency and/or optics design problems for two reasons: 1) the above mentioned approximate analytical methods are very effective for this class of problems, and 2) direct numerical approximations become prohibitively expensive as D/λ becomes large.

At the other end of the regime are static and quasi-static problems. Static problems are of course those in which the sources are constant in time and the goal is to solve for the constant field configuration. A classic example is the calculation of the capacitance of a metallic structure. In this case D/λ is zero. A quasi-static problem is one where D/λ is non-zero but quite small. Examples of quasi-static problems include Rayleigh scattering of light by dust particles and the shielding/grounding problems associated with 60 Hz power. It can be shown that it is a usually a good approximation to solve a quasi-static field problem by first solving the approximate static field problem and multiplying this solution by $\cos(\omega t)$ to obtain the true solution. This dissertation is concerned with electromagnetic design and analysis problems that fall between the quasi-static and high frequency regimes.

Many electromagnetic problems involve propagation of waves in a vacuum, while others involve material media. The variety of electric and magnetic properties of media is too vast to summarize here. Electromagnetic problems addressed in this dissertation will be restricted to a class of materials, referred to as simple media, that can be described by tensor permittivity, permeability, and (electric and magnetic) conductivity. These material properties are functions of position only, i.e. non-linear and/or dispersive media will not be considered. This raises the question, how does one describe an extremely inhomogeneous volume with complicated boundaries between regions in a way that can be easily and efficiently understood by a digital computer? The approach taken in this dissertation is to discretize the volume into an unstructured grid consisting of polyhedral cells, each cell being small enough such that the material properties assume a simple form (constant, linear, etc.) within each cell. This approach is compatible with commercially available computer aided design (CAD) packages.

FIGURE 1. Structured versus Unstructured Grids.



In the early days of computational electromagnetics it was common to develop a new computer program for every type of geometry encountered; for example cylindrical coordinates would be used for analyzing the fields within a metallic cylinder, and a body-of-

revolution program would be developed to calculate scattering from a body of revolution. Such approaches may be efficient in terms of computer resources, but they place severe restrictions on what an engineer can accurately design and analyze. The unstructured grid approach, on the other hand, places few limits on geometry. The unstructured grid approach does introduce added complexity, and in fact much of the current research in computational electromagnetics is focused on dealing with this complexity. The term unstructured is used in this dissertation to mean not necessarily structured, whereas structured means that there is a one-to-one mapping of cells, nodes, faces, and edges onto a Cartesian grid. Thus structured grids are considered to be a subset of unstructured grids. This is illustrated in Figure 1 for two-dimensional quadrilateral grids. For practical reasons the grids examined in this dissertation will be limited to hexahedral (eight node) or tetrahedral (four node) cells, rather than arbitrary polyhedral cells. According to this definition a triangular grid, or a grid with both triangles and quadrilaterals, are unstructured.

To summarize, this dissertation is concerned with the direct numerical solution of the time dependent Maxwell equations in charge free regions. The volume of interest may be inhomogeneous, consisting of several dielectric, magnetic, and metallic regions of arbitrary geometry. The material properties are assumed to be linear and non-dispersive. The volume may also contain several independent voltage and current sources. Example electromagnetic problems within this class include the design of waveguides and antennas, scattering of electromagnetic waves from automobiles and aircraft, and the penetration and absorption of electromagnetic waves by dielectric objects.

1.1 Survey of common grid-based numerical schemes

The most popular grid-based numerical scheme for solving the time-dependent Maxwell equations is the Finite Difference Time Domain method [1]-[4]. Usually this method is implemented using dual Cartesian grids, with the electric field components known on the primary grid and the magnetic field components known on the dual grid, with the curl operator approximated by the 2nd order central difference formula. The electric field is updated at whole time steps, the magnetic field at half time steps, by a 2nd order central difference in time (leapfrog). An alternative method combines the two curl operators and solves the vector wave equation for either the electric or magnetic field on a single grid. Both approaches yield a conditionally stable and consistent method for solving Maxwell's equations in the time domain. The disadvantage of these finite difference methods is that they are only defined for Cartesian grids, and it has been shown that approximating curved boundaries by a "stair step" approximation can give poor results [4][5]. Nevertheless the FDTD is extremely efficient and it is often used as a benchmark to which new methods are compared.

There have been several attempts to generalize the FDTD method to unstructured grids, most notably the modified finite volume (MFV) and discrete surface integral (DSI) methods [6]-[9]. In these methods Maxwell's equations are cast in integral form, then the subsequent volume and/or surface integrals are approximated by standard low-order integration rules. The time integration is the similar to that used in the FDTD method. In fact most of these methods reduce to the FDTD method when orthogonal grids are used. However these methods are not provably stable, and weak instabilities leading to non-

physical solution growth have been observed for non-orthogonal grids [10]. The instability is caused by the non-symmetric discretization of the curl-curl operator. Dissipative time integration schemes may be employed to counteract this non-physical solution growth, but this results in a violation of conservation of energy and charge [11].

There is another class of finite volume methods for solving Maxwell's equations that are different than the above methods in that they do not reduce to the FDTD method when implemented on Cartesian grids. In these methods Maxwell's curl equations are cast in so-called conservative form, resulting in a PDE that resembles the Euler equation of fluid dynamics [12]-[14]. Then the classic methods of computational fluid dynamics such as Lax-Wendroff or Jameson-style Runge Kutta are used to solve these equations. Typically these methods are implemented on a structured, but non-orthogonal, hexahedral grid. It has been shown that these methods are stable and consistent, thus very good accuracy can be achieved as the grid is refined. These methods rely upon dissipative time integration for stability, thus they do not conserve energy. In addition they neglect the divergence properties of the fields. It is somewhat disconcerting that these methods allow for divergent magnetic fields, i.e. magnetic monopoles, which is in direct violation of Maxwell's equations. Nevertheless these methods are very popular for the radar cross section (RCS) prediction problem. Apparently neither energy conservation nor charge conservation is essential for accurate RCS calculations.

The Finite Element Method (FEM) was developed to solve partial differential equations on unstructured grids from the onset. The original PDE is cast into an equivalent variational, Ritz-Galerkin, or Total Least Squares form. A basis function expansion is

employed for the unknown variables, and the coefficients of the expansion are solved for. Any derivatives or integrals that are required are computed exactly, or within to some numerical tolerance. Typically curved boundaries are approximated as piecewise linear, and an unstructured mesh is used within each region. The classic FEM using nodal elements has been quite successful in solving static electromagnetic problems where the continuous electrostatic potential can be employed [15]-[17]. However this approach has been unsuccessful for solving for the vector electric or magnetic fields directly. There are two problems with nodal elements: 1) they force continuity of the fields across material interfaces, even when there is supposed to be a discontinuity of the fields, and 2) they permit "spurious modes", or nonphysical solutions, which do not disappear as the grid is refined, resulting in a non-converging method. While the subject of spurious modes has been extensively investigated in the context of frequency domain electromagnetics [18]-[22], it is also a problem with recently developed time-domain methods [23]-[27]. The problem is not with the FEM per se, but rather with the choice of elements.

Several researchers have studied the FEM in conjunction the theory of differential forms, the conclusion being that different PDE's may require different finite elements in order to achieve convergence. In fact a set of coupled PDE's may require the simultaneous use of several different finite elements, this is referred to as a mixed FEM. Recently developed vector elements, also known as edge elements, Whitney 1-forms, or H(curl) elements [28]-[32], have been used to solve Maxwell's equations for the electric fields directly. These elements enforce tangential continuity of the fields but allow for jump discontinuity in the normal component of the fields. Use of these elements also eliminates spurious, divergent solutions of Maxwell's equations that were common with nodal element formu-

lations. Vector finite element methods have been successfully used in the frequency domain to analyze resonant cavities, compute waveguide modes, and perform scattering calculations [31],[33]-[35]. Vector finite elements have also been proposed to solve Maxwell's equations directly in the time domain [37]-[40]. Numerical dispersion analysis, which indicates how accurately an electromagnetic wave propagates on a given grid, indicates that vector finite element methods can be more accurate than competing FDTD and FVTD methods [41]-[43].

1.2 Discrete Time Vector Finite Element Method

While some engineers and scientists are satisfied with the above mentioned methods this author feels that there is room for improvement. The primary goal of this research was to develop a method for solving time-dependent Maxwell's equations on unstructured three dimensional grids that is provably stable, energy conserving, and charge conserving. The following is a list of features that such a method would have:

- valid for unstructured grids
- allows for tensor permittivity, permeability, and conductivity
- correctly models field continuities/discontinuities
- reduces to FDTD for Cartesian grids
- conditionally stable
- energy conserving
- charge preserving

- 2nd order accurate

In this dissertation a method, called the Discrete Time Vector Finite Element Method (DTVFEM) is derived, analyzed, and validated. This method has all of the features listed above. Thus the DTVFEM has a combination of attributes not shared by other grid-based, time-domain methods for solving Maxwell's equation. The DTFEM uses covariant vector finite elements as a basis for the electric field and contravariant vector finite elements as a basis for the magnetic flux density. These elements are complementary in the sense that the covariant elements have tangential continuity across interfaces whereas the contravariant elements have normal continuity across interfaces. The Galerkin approximation is used to convert Ampere's and Faraday's law to a coupled system of ordinary differential equations (ODE). The leapfrog method is used to advance the fields in time.

The DTVFEM described in this dissertation is different than other time domain vector finite element methods in several respects. The variational form of Maxwell's equations used in this dissertation, described in Section 3.3, is different than that used in [36]-[39]. This dissertation essentially begins with the variational form of Maxwell's equations presented in the conclusion of [28]. This form was chosen because it leads to a symmetric discretization. The method proposed in [40] is a special case of the DTVFEM. It should be noted that the use of vector finite elements is not a panacea. While variational crimes such as point-matching, collocation, or mass lumping are attractive from a computational point of view, these approximations may lead to spurious, non-divergence free solutions even if vector finite elements are used. The DTVFEM does not employ point-matching or collocation. Spurious solutions and divergence are discussed in Section 4.1 and Section 5.3,

respectively. Another issue, which is unique to time domain methods, is numerical stability. Stability of the DTVFEM is discussed in Section 5.1.

The DTVFEM, as implemented in this dissertation, requires that a sparse linear system be solved at every time step. This is a disadvantage compared to FDTD and FVTD methods. Some researchers define any method that requires a linear system to be solved as an implicit method, while methods that do not require linear system solutions as explicit. That definition is not used in this dissertation. In Section 5.0 it is shown that the DTVFEM is really an explicit method that looks like an implicit method. The second part of this dissertation addresses the solution of the large, sparse, unstructured matrices that arise in the DTVFEM. The computational effort required to solve the system depends upon how distorted the grid is. For Cartesian grids mass lumping can be used, in which case the DTVFEM reduces to the classic FDTD method. For non-Cartesian grids iterative methods are used to solve the system. It is shown that the number of iterations required to achieve a given accuracy is a constant independent of the size of the problem, thus the DTVFEM is competitive with “explicit” FDTD and FVTD methods.

The method was implemented in software and hosted on variety of computer systems, including two parallel supercomputers. The third part of this dissertation describes the parallel implementation and the resulting parallel performance. The DTVFEM is validated by comparing computed solutions to analytical solutions for a simple resonant cavity, waveguide, and antenna. While the software developed during this research effort has not undergone rigorous software quality assurance testing and it is far from user friendly, it is nevertheless a valuable by-product of this research effort.

It is important to distinguish the difference between the method and the software. For example different computer programmers can implement the DTVFEM differently, with different constraints on the form of the input and output files, different data structures used to store the matrices, different methods for parallel implementation, etc. One implementation of the DTVFEM can require more computer time or more computer memory than another. As another example, different programmers may choose to deal with radiation boundary conditions (discussed in Section 8.0) differently. The software developed during this research effort will be referred to as VFEM3D.

2.0 Maxwell's Equations

2.1 Partial Differential Equations

Maxwell's equations consist of two curl equations and two divergence equations. There is a great variety of ways to express Maxwell's equations, in this dissertation rational MKS units will be used. The literature on Maxwell's equations is vast, with examples of easily readable textbooks including [45] and [46], and more advanced textbooks exemplified by [47]-[50]. The equations are

$$\nabla\times\vec{E} = -\frac{\partial}{\partial t}\vec{B} - \sigma_M\vec{H} - \vec{M}, \tag{1}$$

$$\nabla\times\vec{H} = \frac{\partial}{\partial t}\vec{D} + \sigma_E\vec{E} + \vec{J}, \tag{2}$$

$$\nabla\bullet\vec{D} = 0, \tag{3}$$

$$\nabla\bullet\vec{B} = 0. \tag{4}$$

Equation (1) is a generalization of Faraday's law to include magnetic current density $\vec{M}$ and magnetic conductivity σ_M, while (2) is the Maxwell-Ampere law. Note that in (3) the charge density term is zero. The electric and magnetic conductivities are assumed to be symmetric positive definite tensors, which are functions of position only. Two constitutive relations are required to close Maxwell's equations. For this study the dielectric permittivity ε, the magnetic permeability μ are also positive definite tensors, which are functions of position only,

$$\vec{D} = \varepsilon \vec{E}, \quad \vec{B} = \mu \vec{H}. \tag{5}$$

In practice it is seldom necessary to solve for both electric and magnetic fields and both electric and magnetic flux densities. It is possible to use the constitutive relations to eliminate one or more of the variables, and it is possible to combine the two first-order curl equations to obtain a single second-order PDE. Of course for a well posed problem appropriate initial conditions must be specified, as well as the independent current sources and boundary conditions. For clarity in the following, two PDE's are defined:

PDE I

$$\mu^{-1}\frac{\partial}{\partial t}\vec{B} = -\mu^{-1}\nabla\times\vec{E} - \mu^{-1}\sigma_M\mu^{-1}\vec{B} - \mu^{-1}\vec{M} \text{ in } \Omega, \tag{6}$$

$$\varepsilon\frac{\partial}{\partial t}\vec{E} = \vec{\nabla}\times\mu^{-1}\vec{B} - \sigma_E\vec{E} - \vec{J} \text{ in } \Omega, \tag{7}$$

$$\nabla\bullet\varepsilon\vec{E} = 0 \text{ in } \Omega, \tag{8}$$

$$\nabla\bullet\vec{B} = 0 \text{ in } \Omega, \tag{9}$$

$$\hat{n}\times\vec{E} = \vec{E}_{bc} \text{ on } \Gamma, \tag{10}$$

$$\vec{E}(t=0) = \vec{E}_{ic}, \vec{B}(t=0) = \vec{B}_{ic}. \tag{11}$$

PDE II

$$\begin{aligned}\varepsilon\frac{\partial^2}{\partial t^2}\vec{E} + \left(\sigma_E + \mu^{-1}\sigma_M\varepsilon\right)\frac{\partial}{\partial t}\vec{E} + \mu^{-1}\sigma_M\sigma_E\vec{E} = \\ -\nabla\times\mu^{-1}\nabla\times\vec{E} - \mu^{-1}\sigma_M\vec{J} - \mu^{-1}\nabla\times\vec{M} - \frac{\partial}{\partial t}\vec{J}\end{aligned} \quad \text{in } \Omega, \tag{12}$$

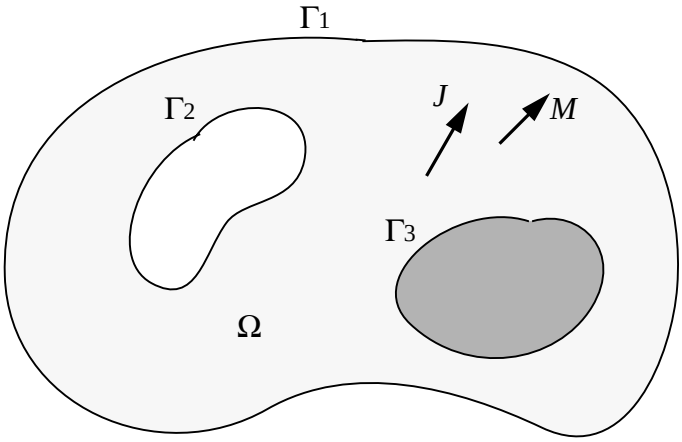
$$\hat{n} \times \grave{E} = \grave{E}_{bc} \text{ on } \Gamma, \tag{13}$$

$$\nabla \bullet \varepsilon \grave{E} = 0, \tag{14}$$

$$\grave{E}(t = 0) = \grave{E}_{ic}, \frac{\partial}{\partial t}\grave{E}(t = 0) = \frac{\partial}{\partial t}\grave{E}_{ic}. \tag{15}$$

In both PDE's Ω is the total volume of interest, which is finite, and Γ is the boundary of the volume, not necessarily simply connected. The subscript *bc* denotes boundary condition. The only boundary condition investigated will be specification of the tangential component of the electric field. The subscript *ic* denotes initial condition. Partial differential equations of the form of PDE I and PDE II are called initial boundary value problems (IBVP).

FIGURE 2. Illustration of generic inhomogeneous volume.



It is appropriate to determine the conditions under which the above IBVP's are well posed. An IBVP is well posed if: 1) the solution exists, 2) the solution is unique, and 3) the solution depends continuously upon the data. The Cauchy-Kowalewsky theorem [51] states

that solutions exist for analytic PDE's with analytic initial data. PDE's I and II above obviously qualify, since they are linear constant coefficient PDE's. However this theorem does not address boundary conditions. The existence and uniqueness of solutions to Maxwell's equations in particular is discussed in [47] where it is proved that it is necessary to provide the tangential component of the electric or magnetic field on the boundary (or tangential component of electric field on part of the boundary and tangential component of magnetic field on the remaining part). If neither field is specified on the boundary then the solution is not unique, if both are specified on the boundary then the solution may not exist. For the specific problems addressed in this dissertation there is a constraint on the independent current sources, namely

$$\nabla \bullet \vec{J} = 0 \text{ and } \nabla \bullet \vec{M} = 0. \tag{16}$$

Note that if the current sources are divergence free, then (8) and (9) will automatically be satisfied for all time if the initial data satisfy (8) and (9). In other words the initial divergence is preserved. The third requirement for being well posed is obviously satisfied since the equations are linear, constant coefficient PDE's.

As mentioned in the introduction the character of the fields in the vicinity of material discontinuity is important. It is well known that fields and flux densities have different continuity properties across material interfaces [45][46]. For completeness these properties are reviewed below.

Consider a material interface separating region 1 from region 2 as illustrated in Figure 3. A contour C surrounding area A is defined with $\hat{t}$ tangent to the contour and $\hat{n}$ normal to

the area. The material parameters are finite and no independent sources are present. Application of Stokes theorem to (1) yields

$$\oint_C \vec{E} \bullet \hat{t} dl \; = \; \frac{\partial}{\partial t} \int_A \vec{B} \bullet \hat{n} dA + \int_A \sigma_M \vec{H} \bullet \hat{n} dA \, . \tag{17}$$

As $\delta z \rightarrow 0$ the area A goes to zero, thus the right hand side goes to zero. This implies that

$$(\vec{E}_1 - \vec{E}_2) \bullet \hat{t} \; = \; 0 \, , \tag{18}$$

the tangential components of the electric field are continuous across a material interface. A similar argument applied to (2) yields

$$(\vec{H}_1 - \vec{H}_2) \bullet \hat{t} \; = \; 0 \, , \tag{19}$$

the tangential components of the magnetic field are also continuous. On the other hand the tangential components of the electric and magnetic flux densities are not continuous across a material discontinuity.

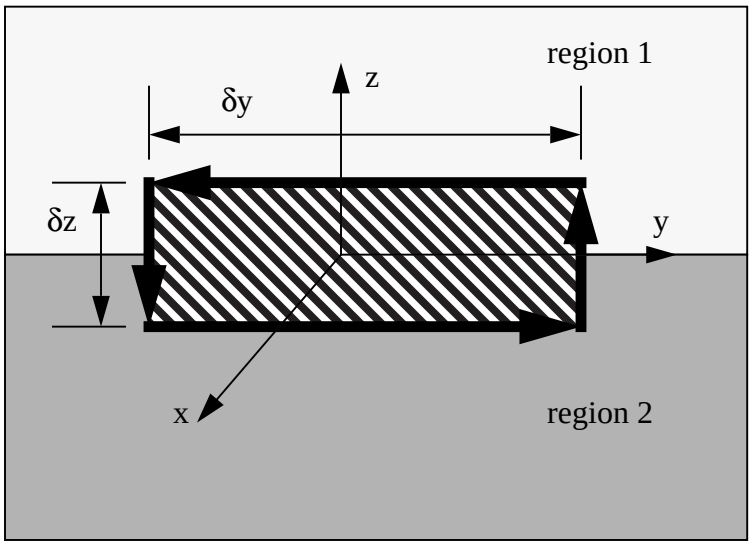
Many electromagnetic design and analysis problems involve metals with electric conductivities greater than 10^6 S/m, in which case it is practical to approximate the conductivity as infinite, i.e. a Perfect Electrical Conductor (PEC). In this case application of Stokes theorem to (2) yields

$$(\vec{H}_1 - \vec{H}_2) \bullet \hat{t} \; = \; \vec{J}_S \, , \tag{20}$$

where $\vec{J}_S$ is an infinitely thin surface current density. Inside a PEC both the electric field and the magnetic field are zero, thus (18) requires that the tangential component of the

electric field be zero, and (20) requires that the tangential component of the magnetic field be equal to the induced surface current density. In practice the induced surface current density is not known a priori, in fact this is why boundary condition (10) is used.

FIGURE 3. Geometry for boundary conditions on tangential components of fields.



To analyze the properties of the normal components of the fields a cylindrical tin can shaped volume is defined as shown in Figure 4. The surface area of the tin can is decomposed into A_1 and A_2, the area of the sides and end caps, respectively. No net charge exists within the tin can. Application of the divergence theorem to (8) yields

$$\oint_A \varepsilon \vec{E} \bullet \hat{n} dA = 0. \tag{21}$$

As $\delta z \to 0$ the area A_1 goes to zero, and (21) reduces to

$$\hat{n} A_2 \bullet \left(\varepsilon_1 \vec{E}_1 - \varepsilon_2 \vec{E}_2 \right) = 0, \tag{22}$$

thus the normal component of the electric field is discontinuous. Repeating this procedure on (9) gives

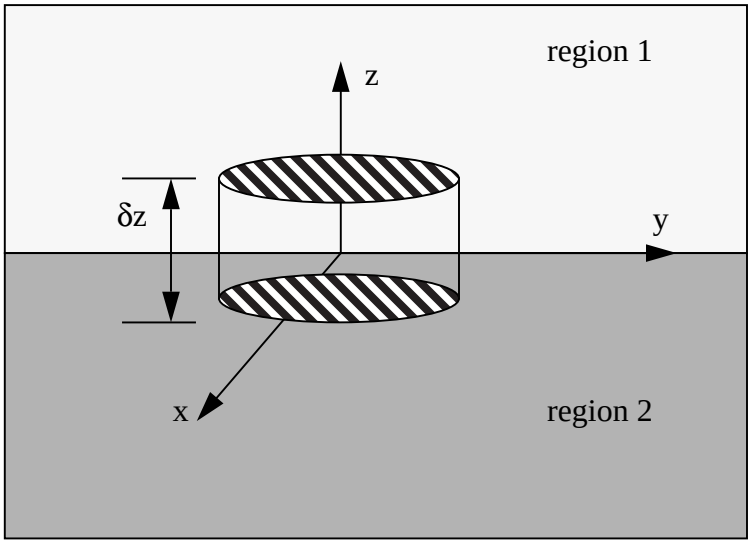
$$\hat{n}A_2 \bullet \left(\mu_1 \vec{H}_1 - \mu_2 \vec{H}_2 \right) = 0, \tag{23}$$

the normal component of the magnetic field is discontinuous. On the other hand the normal components of the electric and magnetic flux densities are continuous. If region 2 is a PEC then (22) is modified to

$$\hat{n}A_2 \bullet \varepsilon_1 \vec{E}_1 = q_S, \tag{24}$$

where q_S is the induced electric surface charge on the conductor. This surface charge is on Γ and not in Ω. This surface charge density, like the surface current density, is not known a priori, it is not considered a source.

FIGURE 4. Geometry for boundary conditions on normal components.



The electromagnetic fields described by Maxwell's equations satisfy conservation of energy. This is often referred to as Poynting's theorem. By simply manipulating Max-

well's equations and using the vector identity $\nabla\bullet(a\times b) = b\bullet(\nabla\times a) - a\bullet(\nabla\times b)$ it is easy to show that

$$\oint_{\Gamma}\mu^{-1}\vec{E}\times\vec{B}\bullet\hat{n}d\Gamma + \int_{\Omega}\mu^{-1}\vec{B}\bullet\vec{M}d\Omega + \int_{\Omega}\vec{E}\bullet\vec{J}d\Omega + \int_{\Omega}\mu^{-1}\sigma_{M}\vec{B}\bullet\vec{B}d\Omega + \int_{\Omega}\sigma_{E}\vec{E}\bullet\vec{E}d\Omega + \int_{\Omega}\mu^{-1}\vec{B}\bullet\frac{\partial}{\partial t}\vec{B}d\Omega + \int_{\Omega}\varepsilon\vec{E}\bullet\frac{\partial}{\partial t}\vec{E}d\Omega = 0\ . \tag{25}$$

The first term is the net power flow leaving the volume Ω through surface Γ. The second and third terms represent the power supplied to the volume by the magnetic and electric current sources, respectively. The fourth and fifth terms represented the power absorbed by the medium, i.e. the rate of conversion of electromagnetic energy into thermal energy. The sixth and seventh terms represent the time rate of change of stored magnetic and electric energy, respectively.

3.0 Variational Formulation of Maxwell's Equations

3.1 Some definitions from functional analysis

Before stating the variational form of PDE I and PDE II it is necessary to review some definitions used in functional analysis.

A space V is a non-empty set of elements $u, v, w, \ldots$. In general, the elements may be real numbers, complex numbers, vectors, matrices, functions of one or more variables, etc. The space of real numbers will be denoted by R, the space of three vectors by R^3.

A polynomial in the quantities $x_1, x_2, \ldots, x_n$ is an expression involving a finite sum of terms of the form $a x_1^{k1} x_2^{k2} \ldots x_n^{kn}$ where a is some scalar and $k_1, k_2, \ldots, k_n$ are non-negative integers. The degree k of a polynomial is the maximum value of $k_1 + k_2 + \ldots + k_n$ that appears in the polynomial. For example the polynomial $1 + x + y + z + xyz$ is of degree $k = 3$.

A polynomial space P is complete to order k if it contains all polynomials of degree $\leq k$. Such a space is denoted by P_k. For example the polynomial space $a_0 + a_1 x + a_2 y + a_3 z + a_4 xyz$, where $a_0, \ldots, a_4$ are arbitrary scalars, contains polynomials of degree 3 but is only complete to order 1.

A function f is continuous at a point x if for every sequence x_n whose limit is x, the sequence $f(x_n)$ converges to $f(x)$.

A multi-index α is a n-tuple of non-negative integers α_i. The length of α is given by

$$|\alpha| = \sum_{i=1}^{n} \alpha_i . \tag{26}$$

The multi-index notation for the partial derivatives of a function $f(x_1, \ldots, x_n)$ is denoted by

$$D^{\alpha} = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha 1} \ldots \partial x_n^{\alpha n}} . \tag{27}$$

A function f has continuity of order k if all partial derivatives $D^{\alpha} f$, for $0 \le |\alpha| \le k$, are continuous. The space of continuous functions of order k on some domain Ω is denoted by $C^k(\Omega)$.

The Lebesgue spaces are defined by $L^p(\Omega) = \{f : \|f\|_{L^p} < \infty\}$ where

$$\|f\|_{L^p} = \left(\int_{\Omega} |f|^p d\Omega \right)^{1/p} . \tag{28}$$

The space $L^2(\Omega)$ is the space of all functions on Ω that are square-integrable, which is often referred to as the space of functions with finite energy. For vector functions $\vec{f}: R^3 \to R^3$ the corresponding space is denoted $\left(L^2(\Omega) \right)^3$.

A function $f \in L^1(\Omega)$ is said to have a weak derivative, denoted by $D_w{}^{\alpha} f$, if there exists a function $g \in L^1(\Omega)$ such that

$$\int_{\Omega} g\phi d\Omega = (-1)^{\alpha} \int_{\Omega} f D^{\alpha}\phi d\Omega \quad \forall \phi \in \{\phi : \phi \in C^{\infty}, \phi = 0 \text{ on } \Gamma\} . \tag{29}$$

If such a g exists, then $D_w{}^{\alpha} f = g$.

A space is linear if it is characterized by the following properties: 1) addition of elements follow the rules of ordinary addition, i.e. for all $u, v \in V$ $u + v \in V$, and there exists a null element $0 \in V$ such that $0 + v = v$ for all $v \in V$, and 2) multiplication between elements v of space V and scalars c of field F follow the rules of ordinary multiplication, i.e. for all $v \in V$ and $c \in F$ the product cv is in V. In this dissertation the term linear will imply the field of real numbers.

Elements $f_1, f_2, \ldots, f_n$ of a linear space V are linearly dependent if there exists numbers $c_i \in F$, not all equal to zero, such that

$$\sum_{i=1}^{n} c_i f_i = 0 . \tag{30}$$

The elements are linearly independent if (30) implies all $c_i = 0$.

Every set S which contains a given element v is called a neighborhood of v. A subset S is open if and only if it contains with every element v also a neighborhood U_v of that element. A subset S is closed if and only if all cluster points belong to S, where an element f is called a cluster point of a set S when every neighborhood of f contains at least one element $g \in S$ which is different than f. The complementary set of a closed set of V is an open set.

A space V is a metric space if for any two elements $u, v \in V$ there is a real number $p = p(u, v)$, called the distance, with 1) $p(u, v) = 0$ if and only if $u = v$, and 2) $p(u, v) \leq p(u, w) + p(v, w)$ for all $u, v, w \in V$.

A sequence of elements $f_1, f_2, \ldots$ of a metric space, denoted by f_n, is a Cauchy sequence if for every $\varepsilon > 0$ there exists an integer N such that if $n, m > N$ then $p(f_n, f_m) < \varepsilon$. A subset S of a metric space V is complete if there is a limit element $f \in S$ to each Cauchy sequence f_n with $p(f_n, f) \to 0$.

A bilinear form $b(.,.)$ on a linear space V is a mapping $b : V \times V \to R$ such that each of the maps $v \to b(v, w)$ and $w \to b(v, w)$ is a linear form on V. It is symmetric if $b(v, w) = b(w, v)$ for all $v, w \in V$. An inner product, denoted by $(.,.)$, is a symmetric bilinear form on a linear space V that satisfies 1) $(v,v) \geq 0 \ \forall v \in V$, and 2) $(v,v) = 0 \Leftrightarrow v = 0$.

The standard inner product for real scalar functions u and v is

$$(u, v) = \int_{\Omega} uv d\Omega, \tag{31}$$

and the corresponding inner product for real vector functions is

$$(\vec{u}, \vec{v}) = \int_{\Omega} \vec{u} \bullet \vec{v} d\Omega. \tag{32}$$

The Sobolev inner product for functions u, v in $C^1(\Omega)$ is given by

$$(u, v)_k = \sum_{0 \le |\alpha| \le k} \left(D_w{}^{\alpha} u, D_w{}^{\alpha} v \right). \tag{33}$$

A linear space V together with an inner product defined on it is called an inner-product space and is denoted by $(V, (.,.))$. Two elements u and v of an inner-product space are orthogonal if $(u, v) = 0$.The associated normed inner-product space is denoted by $(V, \|.\|)$ where the induced norm is $\|.\| = \sqrt{(.,.)}$. A normed inner-product space is a metric space.

A Sobolev space is defined as $W_p{}^k(\Omega) = \{f : f \in L^1(\Omega), \|f\|_{W_p{}^k} < \infty\}$ where

$\|f\|_{W_p{}^k} = \left(\sum_{0 \le |\alpha| \le k} \left| D_w{}^{\alpha} u \right|^p \right)^{1/p}$ is the Sobolev norm. Sobolev spaces are important in the context of piecewise polynomial spaces. For example the “hat function” defined by $f(x) = 1 - |x|$, $\Omega = [-1, 1]$, is in the space $W_2{}^1$.

A bilinear form is bounded (or continuous) on a normed inner-product space if there exists a $c_1 < \infty$ such that $|b(v, w)| \le c_1 \|v\| \|w\|$ for all $v, w \in V$. A bilinear form is coercive on subspace $U \subset V$ if there exists a $c_2 > 0$ such that $|b(u, u)| \ge c_2 \|u\|^2$ for all $u \in U$.

A Hilbert space is a complete, normed, inner-product space. A standard Hilbert space is the space of square integrable functions

$$H(\Omega) = \{ u \in R; \|u\| < \infty; (v,w) = \int_{\Omega} v w d\Omega \} , \tag{34}$$

where Ω is the volume of interest. The corresponding Hilbert space for vector functions is

$$(H(\Omega))^3 = \{ \vec{u} \in R^3; \|\vec{u}\| < \infty; (\vec{v},\vec{w}) = \int_{\Omega} \vec{v} \bullet \vec{w} d\Omega \} . \tag{35}$$

It can be shown that the Sobolev spaces with $p = 2$ are Hilbert spaces, and these are denoted by $H^k(\Omega) = W_2^{\,k}(\Omega)$.

The following properties of Hilbert spaces are proved in [52]:

Property 1. If in a Hilbert space the inner product $(u, v) = 0$ for all $u \in H$, then $v = 0$.

Property 2. If $(u, v) = (w, v)$ for all $v \in H$, then $u = w$.

Property 3. Let v be an arbitrary element of H, and let U be a subspace of H. There is a unique element $u \in U$ closest to v; v can be decomposed uniquely $v = u + g$ such that $u \in U$ and $g \perp U$. The element u is called the projection of v on U.

Property 4. Any continuous linear functional L on a Hilbert space H can be uniquely represented as $L(v) = (u, v)$ for some $u \in H$. This is the Riesz Representation Theorem.

3.2 Review of variational and Galerkin formulations

Many of the laws of physics can be written in either differential form, integral form, or variational form. This can best be illustrated by example. Consider the problem of solving for the electrostatic potential due to a given charge distribution. Assume that the potential is zero on the boundary. In differential form this problem is described by Poisson's equation

$$-\nabla^2\phi = \rho \text{ in } \Omega, \ \phi = 0 \text{ on } \Gamma. \tag{36}$$

This equation specifies how the electrostatic potential must behave at every point in space. Applying the divergence theorem to (36) yields

$$\int_S \nabla\phi \bullet \hat{n} dA = -Q_{enc}, \tag{37}$$

where the integral is over any arbitrary surface and Q_{enc} is the total charge enclosed by that surface. This is an equally valid integral form of the same physics problem, but rather than specifying how the potential behaves at every point, it describes how the integral of the potential behaves over an arbitrary area. The variational form of this same problem is to find ϕ that minimizes the quadratic functional

$$I(\phi) = \int_\Omega \left(\frac{1}{2}\nabla\phi \bullet \nabla\phi + \rho\phi \right) d\Omega. \tag{38}$$

This functional is in fact the total electrostatic energy of the system, and there is one and only one ϕ that minimizes this functional. The fact that the electrostatic potential is such that the total electrostatic energy is minimized is a consequence of the fundamental principle of variational calculus. By differentiating (38) and setting this derivative to zero it is obvious that the minimum is given by the ϕ that satisfies (36).

In practice, neither (36), (37), nor (38) can be solved exactly and numerical methods are employed. Defining a grid on the volume and expanding (36) in a truncated Taylor series about each node in the grid yields a system of equations $A\vec{x} = \vec{b}$ where the vector $\vec{x}$ represents the values of ϕ at the nodes and $\vec{b}$ represents the values of ρ at the nodes. This is an example of a finite difference method. The finite volume approach begins by dividing the entire volume into sub-volumes, and then enforcing (37) on each sub-volume. This also results in a linear system $A\vec{x} = \vec{b}$, but in this case $\vec{x}$ represents the net flux through a particular cell face, and $\vec{b}$ represents the net charge within each volume. The matrix A arising from a finite volume approximation is in general different from that arising from a finite difference approximation.

The variational form of the above electrostatic problem is typically solved by the Ritz method. First it is necessary to determine the admissable space V. For this particular problem the admissable space is

$$V = \{ v \in H^1(\Omega), v = 0 \text{ on } \Gamma \}, \tag{39}$$

and the variational problem is then to find $\phi \in V$ that minimizes $I(\phi)$. In the Ritz method a finite dimension space $\tilde{V} = \{v_1, v_2, v_3, \ldots, v_n\} \subset V$ is defined and the Ritz approximation is the function $u \in V$ that minimizes $I(u)$. The functions v_i are a basis of $\tilde{V}$, the maximum number of linearly independent functions that span $\tilde{V}$. Specifically let

$$u = \sum_{i=1}^{n} x_i v_i, \tag{40}$$

then the functional is

$$I(x) = \int_{\Omega} \left(\frac{1}{2} \left(\sum_{i=1}^{n} x_i \nabla v_i \right) \bullet \left(\sum_{i=1}^{n} x_i \nabla v_i \right) + \rho \left(\sum_{i=1}^{n} x_i v_i \right) \right) d\Omega. \tag{41}$$

or more concisely

$$I(x) = \frac{1}{2} x^T A x + x^T b. \tag{42}$$

The minimum is found by differentiating with respect to the x_i and setting the derivatives to zero; this yields a linear system $A\hat{x} = \vec{b}$. The vector $\hat{x}$ is the vector of coefficients of the basis function expansion. The matrix A has the form

$$A_{ij} = \int_{\Omega} \nabla v_i \bullet \nabla v_j d\Omega, \tag{43}$$

and the vector $\hat{b}$ is the projection of the charge density ρ on the space $\tilde{V}$. The quadratic functional can be written as

$$I(v) = a(v, v) - 2(f, v) , \tag{44}$$

where $a(v, v)$ is a symmetric, bounded, coercive bilinear form. It is often called the energy inner product. Since the basis functions are linearly independent, the matrix A is positive definite, thus the solution exists and is unique. The Ritz approximation u is the projection of ϕ onto the subspace $\tilde{V}$ with respect to the energy inner product. Equivalently the error $\phi - u$ is orthogonal to $\tilde{V}$ with respect to the energy inner product. As the dimension of $\tilde{V}$ increases, the error energy $a(\phi - u, \phi - u)$ must necessarily decrease, thus the approximation converges to the exact solution in the energy inner product sense.

The Galerkin method is similar to the above Ritz method although it is more general. The Galerkin procedure begins with the variational form of the PDE. Each side of $-\nabla^2\phi = \rho$ is multiplied by a test function v and integrated over the entire domain, yielding

$$-(\nabla^2\phi, v) = (\rho, v) . \tag{45}$$

If ϕ satisfies Poisson's equation than (45) is obviously satisfied. The variational form of Poisson's equation is then to find $\phi \in S$ that satisfies

$$-(\nabla^2\phi, v) = (\rho, v) \text{ for all } v \in V, \tag{46}$$

where S is the solution space and V is referred to as the test space. Whether this uniquely defines the solution ϕ depends critically upon the choice of solution and test spaces. The Galerkin method is the obvious discretization of the variational form. In general it

involves a subspace $\tilde{V}$ of the test space and subspace $\tilde{S}$ of the solution space. Then the Galerkin approximation is the function $u \in \tilde{S}$ that satisfies

$$-(\nabla^2 u, v) = (\rho, v) \text{ for all } v \in \tilde{V}. \tag{47}$$

Note that the Galerkin method is more general than the Ritz method in that the former is applicable to non self-adjoint PDE's. The Ritz method finds the minimum of positive definite functional, whereas the Galerkin method finds a stationary point of a not necessarily positive definite functional. One popular Galerkin method is to let $\tilde{S}$ be a collection of twice differentiable functions, such as Chebyshev polynomials or Gaussian wavelets, and to let $\tilde{V}$ be a collection of delta functions. This is called the collocation, or "point matching" method. An advantage of the collocation approach is that it does not require any inner products to be evaluated; a disadvantage is that the resulting linear system is neither symmetric nor positive definite. Alternatively, one could use the same subspace for both expansion and testing. If the subspace is smooth enough, i.e. of subspace of (39), then integration by parts can be employed to yield a symmetric problem, in fact in this case the Galerkin approximation is equivalent to the Ritz approximation.

3.3 Variational formulation of Maxwell's equations

In this section the variational forms of PDE I and PDE II are derived. First it is necessary to define some Hilbert spaces, and their natural norms, that are important in the context of Maxwell's equations:

$$H(grad) = \{u : u \in L(\Omega); \nabla u \in (L(\Omega))^3\}, \tag{48}$$

$$H_0(grad) = \{u : u \in H(grad); u = 0 \text{ on } \Gamma\}, \tag{49}$$

$$\|u\|_{H(grad)} = \left(\|u\|^2 + \|\nabla u\|^2\right)^{1/2}, \tag{50}$$

$$H(div) = \{\vec{u} : \vec{u} \in (L(\Omega))^3; \nabla \bullet \vec{u} \in L(\Omega)\}, \tag{51}$$

$$H_0(div) = \{\vec{u} : \vec{u} \in H(div); \vec{u} \bullet \hat{n} = 0 \text{ on } \Gamma\}, \tag{52}$$

$$\|\vec{u}\|_{H(div)} = \left(\|\vec{u}\|^2 + \|\nabla \bullet \vec{u}\|^2\right)^{1/2}, \tag{53}$$

$$H(curl) = \{\vec{u} : \vec{u} \in (L(\Omega))^3; \nabla \times \vec{u} \in (L(\Omega))^3\}, \tag{54}$$

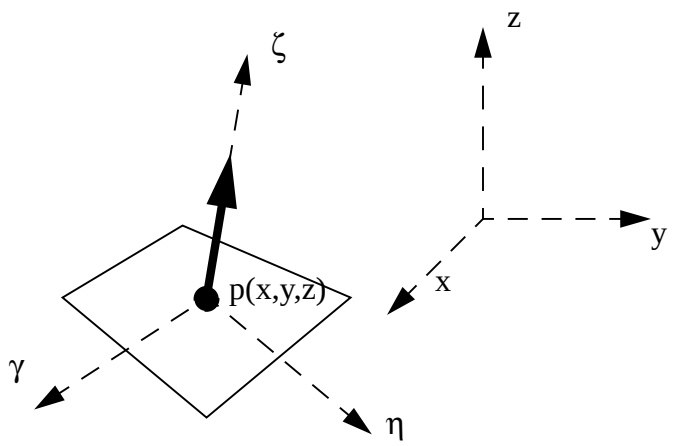
$$H_0(curl) = \{\vec{u} : \vec{u} \in H(curl); \hat{n} \times \vec{u} = 0 \text{ on } \Gamma\}, \tag{55}$$

$$\|\vec{u}\|_{H(curl)} = \left(\|\vec{u}\|^2 + \|\nabla \times \vec{u}\|^2\right)^{1/2}, \tag{56}$$

The spaces $H(div)$ and $H_0(div)$ are the solution and test spaces, respectively, for the magnetic flux density $\vec{B}$. The argument that $\vec{B} \in H(div)$ is similar to the discussion in Section 2.1 where it was shown that the normal component of $\vec{B}$ is continuous, whereas the tangential component of $\vec{B}$ is not necessarily continuous. Likewise the spaces $H(curl)$ and $H_0(curl)$ are the solution and test spaces, respectively, for the electric field $\vec{E}$. The argument that $\vec{E} \in H(curl)$ is similar to the discussion in Section 2.1 where

it was shown that the tangential component of $\vec{E}$ is continuous, whereas the normal component of $\vec{E}$ is not necessarily continuous. The tangential and normal continuity of vector functions is illustrated in Figure 5, where (ζ, η, γ) is a local Cartesian coordinate system defined at the point p. The ζ direction is in the direction of the vector function $\vec{g}(x, y, z)$, the η and γ directions are in the plane normal to $\vec{g}(x, y, z)$. The statement that $\vec{g}(x, y, z)$ has tangential continuity means that $\vec{g}(x, y, z)$ is a continuous function of η and γ at the point p, whereas the statement that $\vec{g}(x, y, z)$ has normal continuity implies that $\vec{g}(x, y, z)$ is a continuous function of ζ.

FIGURE 5. Tangential and normal continuity of vector functions.



Given the spaces defined above, the variational form of PDE I is then:

PDE I, variational form

find $\vec{B} \in H(div)$ and $\vec{E} \in H(curl)$ that satisfy

$$\frac{\partial}{\partial t}(\mu^{-1}\grave{B},\grave{B}^*) \;=\; -(\mu^{-1}\nabla\times\grave{E},\grave{B}^*) - (\mu^{-1}\sigma_M\mu^{-1}\grave{B},\grave{B}^*) - (\mu^{-1}\grave{M},\grave{B}^*), \tag{57}$$

$$\frac{\partial}{\partial t}(\varepsilon\grave{E},\grave{E}^*) \;=\; (\grave{\nabla}\times\grave{E}^*,\mu^{-1}\grave{B}) - (\sigma_E\grave{E},\grave{E}^*) - (\grave{J},\grave{E}^*), \tag{58}$$

for all $\grave{B}^* \in H_0(div)$ and $\grave{E}^* \in H_0(curl)$.

Note that the vector identity $\nabla\bullet(a\times b) \;=\; b\bullet(\nabla\times a) - a\bullet(\nabla\times b)$ is employed to derive (58), i.e.

$$\int_\Omega \nabla\times\mu^{-1}\grave{B}\bullet\grave{E}^*\,d\Omega \;=\; \int_\Omega \nabla\times\grave{E}^*\bullet\mu^{-1}\grave{B}\,d\Omega + \int_\Omega \nabla\bullet\left(\grave{E}^*\times\mu^{-1}\grave{B}\right)d\Omega, \tag{59}$$

and the divergence theorem is used to show the that the last term is zero, i.e.

$$\int_\Omega \nabla\bullet\left(\grave{E}^*\times\mu^{-1}\grave{B}\right)d\Omega \;=\; \oint_\Gamma\left(\grave{E}^*\times\mu^{-1}\grave{B}\right)\bullet\hat{n}\,d\Gamma \;=\; 0, \tag{60}$$

due to the definition of the test field $\grave{E}^*$.

One interpretation of the above variational form is that the solution fields $\grave{E}$ and $\grave{B}$ satisfy Poynting's theorem for every test field $\grave{E}^*$ and $\grave{B}^*$. This form of Maxwell's equations is a generalization of that proposed in [28] to include magnetic conductivity and magnetic current. Other variational forms have been proposed for Maxwell's equations that use $\grave{H}$ instead of $\grave{B}$, or use different test spaces [36]-[39].

The variational form of PDE II is:

PDE II, variational form

find $\vec{E} \in H(curl)$ that satisfies

$$\frac{\partial^2}{\partial t^2}(\varepsilon\vec{E},\vec{E}^*) + \frac{\partial}{\partial t}(\left(\sigma_E + \mu^{-1}\sigma_M\varepsilon\right)\vec{E},\vec{E}^*) + (\mu^{-1}\sigma_M\sigma_E\vec{E},\vec{E}^*) = (\mu^{-1}\nabla\times\vec{E},\nabla\times\vec{E}^*) - (\mu^{-1}\sigma_M\vec{J},\vec{E}^*) - (\mu^{-1}\nabla\times\vec{M},\vec{E}^*) - \frac{\partial}{\partial t}(\vec{J},\vec{E}^*), \tag{61}$$

for all $\vec{E}^* \in H_0(curl)$.

This time Green's first vector theorem is used, i.e.

$$\int_\Omega \nabla\times\mu^{-1}\nabla\times\vec{E}\bullet\vec{E}^* d\Omega = \int_\Omega \mu^{-1}\nabla\times\vec{E}\bullet\nabla\times\vec{E}^* d\Omega + \oint_\Gamma \mu^{-1}(\vec{E}^*\times\nabla\times\vec{E})\bullet\hat{n}d\Gamma, \tag{62}$$

and again the last term is zero due to the definition of the test field $\vec{E}^*$. The above variational form of the vector Helmholtz equation is a generalization of that used in [40] to include electric and magnetic conductivity.

4.0 Finite Elements

In order to approximately solve the above variational formulations of Maxwell's equations via the Galerkin method it is necessary to construct finite dimensional subspaces. As discussed in the introduction, the approach taken in this research effort is to use subspaces that are collections of finite elements. Following [28], a finite element is defined by the following:

Definition A finite element (K, P, A) consists of:

K, a polyhedral domain;

P, a space of polynomials defined on K of dimension D;

A, a set D of linear functionals defined on P called the degrees of freedom.

Definition A finite element is said to be unisolvent if

$$\forall v \in P, \alpha_i(v) = 0; \forall \alpha_i \in A \Rightarrow v = 0 \ . \tag{63}$$

In other words, the degrees of freedom determine a basis for the dual space P^*. The basis functions (also called shape functions) $\{\psi_1, \psi_2, \ldots, \psi_N\}$ are a basis for P dual to A,

$$\alpha_i(\psi_j) = \delta_{ij}. \tag{64}$$

Therefore, an arbitrary polynomial $v \in P$ can be written as

$$v = \sum_{i=1}^{D} \alpha_i(v)\, \psi_i. \tag{65}$$

Given a function f in the domain of all the α_i, the local interpolant is defined by

$$I_K f = \sum_{i=1}^{D} \alpha_i(f)\,\psi_i . \tag{66}$$

This is a local approximation to f within the polyhedral K. Two finite elements (K, P, A) and $(\tilde{K}, \tilde{P}, \tilde{A})$ are affine (or isoparametric) equivalent if there exists an invertible affine (or isoparametric) mapping $F(\hat{x}) = B\hat{x} + b$ such that

1) $\tilde{K} = F(K)$,

2) $P = F^* \tilde{P}$,

3) $\tilde{A}(\tilde{f}) = A(F^*(\tilde{f}))$,

where F^* is defined by $F^*(\tilde{f}) = \tilde{f} \cdot F$.

Consider a grid T which is a collection of polyhedrons K_i such that: 1) $K_i \cap K_j = \{\ \}$ if $i \neq j$, and 2) $\cup K_i = \Omega$. A finite element (K_i, P_i, A_i) which is equivalent to some reference element (K_0, P_0, A_0) is defined for every K_i. The approximation to f over the entire volume is called the global interpolant and is defined by

$$I_T f|_{K_i} = I_{K_i} f \text{ for all } K_i \in T, \tag{67}$$

i.e. it is the simple sum of the local interpolants over all the polyhedral volumes. The interpolant has continuity order q if $I_T f \in C^q$ for all $f \in C^m$, $m \geq q$. Of key importance is the error of this approximation.

Let h_i = diameter of K_i and $h = max\ \ h_i$ be a measure of the size of the polyhedral regions. Consider a sequence of grids parameterized by h. As $h \to 0$ the grid is said to be refined. The grid is said to be refined uniformly is as $h \to 0$ if all angles in the grid have some lower bound $\theta_0 > 0$ and some upper bound $\theta_\pi < \pi$. An important theorem of approximation theory [53][54] states that if the polynomial space P is complete to order $k-1$ and the interpolant is continuous to order q and the grid is refined uniformly, then

$$\left| f - I_T f \right|_{W_p^{\ s}} \leq C_s h^{k-s} |f|_{W_p^{\ k}} \text{ for } s \leq q. \tag{68}$$

4.1 Boundary conditions, spurious modes, and inclusion relations

Vector functions in $H(curl)$ have tangential continuity, but it is not necessary that they have normal continuity. As mentioned in Section 2.1 the normal component of the electric field across a material discontinuity is discontinuous. Thus, a finite dimensional subspace $W^h \subset H(curl)$ should force tangential continuity of basis functions but allow for normal discontinuity. Conversely, vector functions in $H(div)$ have normal continuity, but it is not necessary that they have tangential continuity. This is consistent with the properties of

magnetic flux density. Thus a finite dimensional subspace $F^h \subset H(div)$ should force normal continuity of basis functions but allow for tangential discontinuity.

In the event that the volume of interest is homogenous, then both the electric field and magnetic flux density are continuous. But as mentioned in Section 1.1 the use of continuous nodal finite elements for vector field problems can lead to spurious modes, or non-physical numerical solutions. Consider the vector Helmholtz equation

$$\nabla\times\frac{1}{\mu}\nabla\times\vec{E} = k^2\varepsilon\vec{E}, \tag{69}$$

where $k = \omega\sqrt{\mu\varepsilon}$ is the wave number. This is an eigenvalue problem, with k^2 being the eigenvalue and $\vec{E}$ the eigenvector. This equation is the frequency domain version of (12) for the special case of zero conductivity and no independent current source terms. When (69) was first solved using the finite element method [55] spurious modes were seen, and it was speculated that these modes were caused by solving (69) alone without explicitly enforcing the solenoidal nature of the field. Several attempts to correct the situation by forcing the field to be solenoidal using so called penalty methods [57][58]. These methods introduced a penalty term proportional to the divergence of the field in to the functional to be minimized. These methods reduce, but do not eliminate, spurious modes. It is now known [53] that if $\vec{V}^h$ is the set of continuous nodal finite elements, (defined in Section 4.2.1 below) defined on a arbitrary tetrahedral grid then

$$\{\vec{v} \in \vec{V}^h : \nabla\bullet\vec{v} = 0\} = \{0\}, \tag{70}$$

i.e. it is not possible to exactly represent a divergence free field on an arbitrary grid using nodal finite elements. This is intimately related to the problem of “locking” in the finite element solution of elasticity problems [59]-[61]. Many elastic materials of interest are incompressible (or nearly so) and this is equivalent to saying that the resulting displacement should be divergence free. When one forces the divergence free condition on the displacement, the result is invariably no displacement, i.e. the beam is “locked”.

In [19] it is shown that spurious finite element solutions to (69) are caused by an improper choice of finite elements. Taking the divergence of (69) yields

$$k^2 \nabla \bullet \varepsilon \vec{E} = 0, \tag{71}$$

thus either the eigenvalue k^2 is zero or the eigenvector $\vec{D} = \varepsilon \vec{E}$ is divergence free. Thus (69) itself imposes that non-static fields be divergence free, no additional equation or penalty term is required. Static solutions of (69) are not required to be divergence free. Associated with these static solutions is a scalar potential ϕ satisfying

$$\vec{E} = -\nabla \phi, \tag{72}$$

these static solutions form the null space of the curl operator since $\nabla \times \nabla \phi = 0$. The problem with nodal finite elements is that there is, in general, no scalar function ϕ that satisfies (72), thus the eigenvalue k^2 cannot be zero. The nodal finite elements provide a poor approximation to the null space of the curl operator. Thus the spurious modes are in fact static solutions to (69), which of course should not be present.

The solution to the spurious mode problem is not to force the fields to be divergence free, but rather to choose a vector finite element space $\vec{W}^h$ that includes gradients of scalar potentials. Let $I = \{u : u \in H(curl); \nabla \times u = 0\}$ be the space of irrotational functions, and let $P_I f$ be the projection of some function f onto I. Also define

$$M = \{v : v \in \vec{W}^h \subset H(curl);\ \nabla \times v = 0\}. \tag{73}$$

Then if

$$P_I v \in \vec{W}^h \text{ for all } v \in \vec{W}^h \tag{74}$$

then

$$\vec{E} \in M\perp \subset I\perp, \tag{75}$$

and then (71) can be satisfied. Static fields will have eigenvalue zero, and non-static fields will be divergence free. Equations (74) and (75) are called inclusion conditions.

The Hilbert spaces discussed in section Section 3.3 satisfy the following inclusion relations:

$$\text{If } \phi \in H(grad) \text{ then } \nabla\phi \in H(curl), \tag{76}$$

$$\text{If } \vec{E} \in H(curl) \text{ then } \nabla \times \vec{E} \in H(div), \tag{77}$$

$$\text{If } \vec{B} \in H(div) \text{ then } \nabla \bullet \vec{B} \in H(\Omega), \tag{78}$$

Finite elements that satisfy the proper continuity conditions across interfaces are said to be compatible [19] (or conforming [28]) with their Hilbert spaces. The proposition is that if finite dimension subspaces $V^h \subset H(grad)$, $\vec{W}^h \subset H(curl)$, $\vec{F}^h \subset H(div)$, and $S^h \subset H(\Omega)$ satisfy the same inclusion conditions (76)-(78) as their infinite dimensional counterparts, then spurious modes will be eliminated. The first such function spaces were developed in [56] long before the advent of computers, and then rediscovered by the finite element community in the 1980's [19],[28]-[30]. One advantage of finite element methods is that a wide variety of grids can be employed, and high order basis functions can be used to give very accurate results on a relatively course grid. In VFEM3D only tetrahedral and hexahedral grids are used, and only linear basis functions are employed. In the next section the linear tetrahedral elements and bilinear hexahedral used in VFEM3D are presented.

4.2 Tetrahedral finite elements

For the elements developed in this section the domain K is a tetrahedron consisting of 4 nodes labeled $\{1, 2, 3, 4\}$, as illustrated in Figure 6. There are 6 edges and 4 faces numbered according to Table 1.

FIGURE 6. Illustration of an arbitrary tetrahedron.

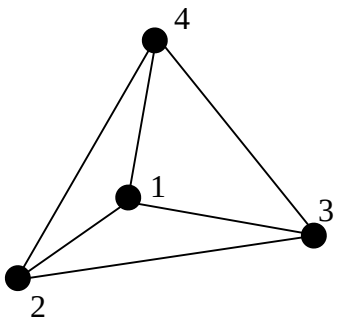


TABLE 1. Node, edge, and face numbering scheme for tetrahedral elements.

	edge	**face**
1	1-2	1-2-3
2	1-3	1-2-4
3	1-4	1-3-4
4	2-3	2-3-4
5	2-4	
6	3-4	

4.2.1 Nodal elements

The linear nodal finite elements use the space of polynomials P of the form

$$P = P_1 = \{u : u = a_0 + a_1 x + a_2 y + a_3 z; \text{ for } a_i \in R\}, \tag{79}$$

which has dimension $D = 4$. The four degrees of freedom give the value of u at the nodes, i.e.

$$A = \{\alpha_i(u) = (u, \delta_i), i = 1, \ldots, 4\}, \tag{80}$$

where $\delta_i = \delta(x - x_i)\,\delta(y - y_i)\,\delta(z - z_i)$ and (x_i, y_i, z_i) are the coordinates of node i. On the reference element defined by the four nodes $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$ the nodal basis for P determined by (64) is

$$\begin{aligned} \phi_1 &= 1 - x - y - z\,, \\ \phi_2 &= x\,, \\ \phi_3 &= y\,, \\ \phi_4 &= z\,, \end{aligned} \tag{81}$$

The basis functions above are called local basis functions since they are only defined within the given tetrahedron. Consider a tetrahedral grid consisting of N_v tetrahedrons K_i and N_n nodes, and a linear nodal finite element (K_i, P_i, A_i) associated with every tetrahedron. The global basis functions are defined to be equal to the local basis functions ϕ inside K and to be zero outside of K. The collection of all the global basis functions is called the nodal finite element space V^h. Any $f \in V^h$ is determined uniquely by the value of f at the N_n nodes, therefore the dimension of V^h is N_n. Within a given tetrahedron the value of f is determined solely by the value at the four nodes. The value of f on an arbitrary face is determined by the values at the three nodes that define the face, likewise the value on an arbitrary edge is determined by the value at the two endpoints. Thus f is a continuous function. The gradient of f is constant within each tetrahedron, but it is not continuous across two adjoining tetrahedrons. However every $f \in V^h$ does possess a weak

derivative. For this finite element space $k = 2$ and $q = 0$, and the rate of convergence is second order,

$$\left|f - I_T f\right|_{W_2^{\,o}} \leq Ch^2 |f|_{W_2^{\,2}}. \tag{82}$$

4.2.2 Edge elements

The linear edge element is defined by the polynomial space

$$P = \left\{ \vec{u} \in (P_1)^3; \frac{\partial u_i}{\partial x_i} = 0,\ i = 1, 2, 3; \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} = 0,\ i \neq j \right\} = \tag{83}$$

$$\{ \vec{u} = (\alpha_0 + \alpha_1 y + \alpha_2 z,\ \alpha_3 + -\alpha_1 x + \alpha_4 z,\ \alpha_5 + -\alpha_2 x + -\alpha_4 y),\ \alpha_i \in R \},$$

and the degrees of freedom

$$\alpha_i(v) = \int_{a_i} (v \bullet t_i)\, ds\ , \tag{84}$$

where a_i is any of the six edges and t_i is the unit tangent vector along a_i. On the reference tetrahedron defined above the space P is spanned by the six basis functions

$$\begin{aligned} \vec{W}_1 &= (1 - y - z, x, x), \\ \vec{W}_2 &= (y, 1 - x - z, y), \\ \vec{W}_3 &= (z, z, 1 - x - y), \\ \vec{W}_4 &= (-y, x, 0), \\ \vec{W}_5 &= (-z, 0, x), \\ \vec{W}_6 &= (0, -z, y), \end{aligned} \tag{85}$$

The basis functions are constructed by forcing (64) to hold. It is common in practice to use the above finite element as a reference element, and apply a linear transformation to generate the elements for other tetrahedra. In order to remain affine equivalent the basis functions must transform covariantly,

$$\begin{aligned} \hat{x} &= B\tilde{x} + \hat{b}\,, \\ \hat{W} &= (B^{*})^{-1}\tilde{W}\,. \end{aligned} \tag{86}$$

Vector functions that transform this way are referred to as covariant or polar vectors. The basis functions above are called local basis functions since they are only defined within the given tetrahedron.

Consider a tetrahedral grid consisting of N_v tetrahedrons K_i and N_e edges, and a linear edge finite element (K_i, P_i, A_i) defined on every tetrahedron. The global basis functions are defined to be equal to the local basis functions W inside K and to be zero outside of K. The collection of all the global basis functions is called the edge finite element space W^h. Any $f \in W^h$ is determined uniquely by the value of f along the N_e edges, therefore the dimension of W^h is N_e. Within a given tetrahedron the value of f is determined solely by the value along the six edges. By construction the tangential component of f is a continuous function across faces. The curl of f is constant with each tetrahedron, and is discontinuous across faces. The rate of convergence is first order,

$$\left| f - I_T f \right|_{H(curl)} \le C h^1 |f|_{H^1}\,. \tag{87}$$

4.2.3 Face elements

The linear face element is defined by the polynomial space

$$F = \{ (P_0)^3 + P_0 \bullet \vec{r} \} = \{ \vec{u} = (\alpha_0 + \alpha_3 x, \alpha_1 + \alpha_3 y, \alpha_2 + \alpha_3 z), \alpha_i \in R \}, \tag{88}$$

and the degrees of freedom are the flux through each face

$$\alpha_i (v) = \int_{A_i} (v \bullet n_i) \, dA, \tag{89}$$

where A_i is any of the four faces and n_i is the unit normal vector to A_i. On the reference tetrahedron defined above the space F is spanned by the four basis functions

$$\begin{aligned} \vec{F}_1 &= (2x, 2y, -2 + 2z), \\ \vec{F}_2 &= (2x, -2 + 2y, 2z), \\ \vec{F}_3 &= (-2 + 2x, 2y, 2z), \\ \vec{F}_4 &= (2x, 2y, 2z). \end{aligned} \tag{90}$$

These basis functions are constructed by forcing (64) to hold. It is common in practice to use the above elements as reference elements, and apply a linear transformation to generate the elements for other tetrahedra. In order to remain affine equivalent the elements must transform contravariantly,

$$\begin{aligned} \vec{x} &= B\tilde{x} + \vec{b}, \\ \vec{F} &= B\tilde{F}. \end{aligned} \tag{91}$$

Vector functions that transform this way are referred to as contravariant or axial vectors. The basis functions above are called local basis functions since they are only defined within the given tetrahedron.

Consider a tetrahedral grid consisting of N_v tetrahedrons and N_f faces, and a linear face finite element defined on every tetrahedron. The global basis functions are defined to be equal to the local basis functions F inside K and to be zero outside of K. The collection of all the global basis functions is called the face element space F^h. Any $f \in F^h$ is determined uniquely by the value of f at the N_f faces, thus the dimension of S^h is N_f. Within a given tetrahedron the value of f is determined solely by the value at the four faces. By construction, the normal component of f is continuous across faces. The divergence of f is constant with each tetrahedron. The rate of convergence for these elements is first order,

$$\left|f - I_T f\right|_{H(div)} \leq Ch^1 |f|_{H^1}. \tag{92}$$

4.2.4 Volume elements

The lowest order finite element for the volume element is trivial, it is simply a constant scalar within each tetrahedron. Let the collection of all such basis functions on a tetrahedral grid be denoted by S^h. These functions have no continuity at all, and the rate of convergence is first order,

$$\left|f - I_T f\right|_{W_2{}^0} \leq Ch^1 |f|_{W_2{}^1}. \tag{93}$$

The tetrahedral node, edge, face, and volume elements defined above are conforming on their infinite dimensional Hilbert spaces, and they satisfy the inclusion relations

$$\text{If } \phi \in V^h \text{ then } \nabla\phi \in W^h, \tag{94}$$

$$\text{If } \vec{E} \in W^h \text{ then } \nabla\times\vec{E} \in F^h, \tag{95}$$

$$\text{If } \vec{B} \in F^h \text{ then } \nabla\bullet\vec{B} \in S^h. \tag{96}$$

It is interesting to note that the dimension of these finite element spaces satisfies Euler's equation

$$N_n - N_e + N_f - N_v = \chi, \tag{97}$$

where χ is the Euler characteristic of the domain Ω. The edge and face elements for a tetrahedron are shown in Figure 8 and Figure 9. The Mathematica script used to generate these figures in included in Section 11.1.

4.3 Hexahedral finite elements

For the elements developed in this section the domain K is a hexahedron consisting of 8 nodes labeled $\{1, 2, 3, 4, 5, 6, 7, 8\}$, as illustrated in Figure 7. There are 12 edges and 6 faces numbered according to Table 2.

FIGURE 7. Illustration of an arbitrary hexahedron.

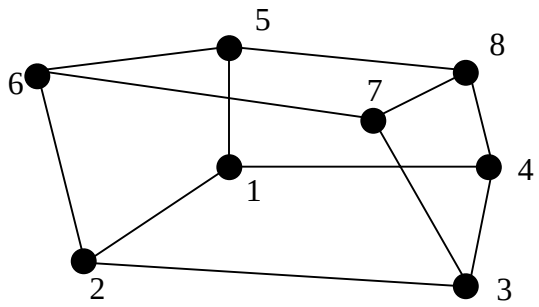


TABLE 2. Node, edge, and face numbering scheme for hexahedrons.

	edge	face
1	1-2	1-4-8-5
2	4-3	2-6-7-3
3	5-6	1-5-6-2
4	8-7	4-8-3-7
5	1-4	1-2-3-4
6	5-8	5-6-7-8
7	2-3	
8	6-7	
9	1-5	
10	2-6	
11	4-8	
12	3-7	

4.3.1 Nodal elements

First, define the polynomials $Q_{l,\,m,\,n}$ in three variables (x, y, z) the maximum degree of which are respectively, l in x, m in y, n in z. For bilinear nodal elements the polynomial space P is,

$$Q_{1,\,1,\,1} = a_0 + a_1 x + a_2 y + a_3 z + a_4 xy + a_5 xz + a_6 yz + a_7 xyz, \tag{98}$$

which has dimension $D = 8$. The eight degrees of freedom give the values of u at the nodes, i.e.

$$A = \{\alpha_i(u) = (u,\delta_i),\ i = 1, \ldots, 8\}, \tag{99}$$

where $\delta_i = \delta(x - x_i)\,\delta(y - y_i)\,\delta(z - z_i)$ and (x_i, y_i, z_i) are the coordinates of node i. On the reference hexahedron defined by the eight nodes $(0, 0, 0)$, $(1, 0, 0)$, $(1, 1, 0)$, $(0, 1, 0)$, $(0, 0, 1)$, $(1, 0, 1)$, $(1, 1, 1)$, $(0, 1, 1)$, the nodal basis for P determined by (99) is

$$\begin{aligned}
\phi_1 &= (1 - x)\,(1 - y)\,(1 - z),\\
\phi_2 &= x\,(1 - y)\,(1 - z),\\
\phi_3 &= xy\,(1 - z),\\
\phi_4 &= (1 - x)\,y\,(1 - z),\\
\phi_5 &= (1 - x)\,(1 - y)\,z,\\
\phi_6 &= x\,(1 - y)\,z,\\
\phi_7 &= xyz,\\
\phi_8 &= (1 - x)\,yz\,.
\end{aligned} \tag{100}$$

The basis functions above are called local basis functions since they are only defined within the given tetrahedron. Consider a hexahedral grid consisting of N_v hexahedrons K_i and N_n nodes, and a linear nodal finite element (K_i, P_i, A_i) defined on every hexahedron. The global basis functions are defined to be equal to the local basis functions ϕ inside K and to be zero outside of K. The collection of all the global basis functions is called the nodal finite element space V^h. Any $f \in V^h$ is determined uniquely by the value of f at the N_n nodes, therefore the dimension of V^h is N_n. Within a given hexahedron the value of v^h is determined solely by the value at the eight nodes. The value of v^h on an arbitrary

face is determined by the values at the four nodes that define the face, likewise, the value on an arbitrary edge is determined by the value at the two endpoints. Thus v^h is a continuous function. For these elements $k = 2$ and $q = 0$, the rate of convergence is second order,

$$\left|f - I_T f\right|_{H^1} \leq Ch^2 |f|_2 . \tag{101}$$

4.3.2 Edge elements

The edge elements are first defined on the reference element. The space P used for bilinear edge elements is

$$P = \{\vec{u} : u_1 \in Q_{0,1,1}; u_2 \in Q_{1,0,1}; u_3 \in Q_{1,1,0}\} , \tag{102}$$

which has dimension $D = 12$.The degrees of freedom are

$$\alpha_i (v) = \int_{a_i} (v \bullet t_i)\, ds , \tag{103}$$

where a_i is any of the twelve edges and t_i is the unit tangent vector along a_i. On the reference hexahedron defined above, the basis for P defined by (103) is

$$
\begin{aligned}
\vec{W}_1 &= (1 - y - z + yz,\ 0,\ 0)\,, \\
\vec{W}_2 &= (y + yz,\ 0,\ 0)\,, \\
\vec{W}_3 &= (z + yz,\ 0,\ 0)\,, \\
\vec{W}_4 &= (yz,\ 0,\ 0)\,, \\
\vec{W}_5 &= (0,\ 1 - x - z - xz,\ 0)\,, \\
\vec{W}_6 &= (0,\ x - xz,\ 0)\,, \\
\vec{W}_7 &= (0,\ z - xz,\ 0)\,, \\
\vec{W}_8 &= (0,\ xz,\ 0)\,, \\
\vec{W}_9 &= (0,\ 0,\ 1 - x - y + xy)\,, \\
\vec{W}_{10} &= (0,\ 0,\ x - xy)\,, \\
\vec{W}_{11} &= (0,\ 0,\ y - xy)\,, \\
\vec{W}_{12} &= (0,\ 0,\ xy)\ .
\end{aligned}
\tag{104}
$$

In order to remain affine equivalent, the elements must transform covariantly,

$$
\begin{aligned}
\vec{x} &= B\tilde{x} + \vec{b}, \\
\vec{W} &= (B^*)^{-1}\tilde{W}.
\end{aligned}
\tag{105}
$$

Vector functions that transform this way are referred to as covariant or polar vectors. The basis functions above are called local basis functions since they are only defined within the given hexahedron.

Consider a hexahedral grid consisting of N_v hexahedrons K_i and N_e edges, and a linear edge finite element (K_i, P_i, A_i) defined on every hexahedron. The global basis functions are defined to be equal to the local basis functions W inside K and to be zero outside of K. The collection of all the global basis functions is called the edge finite element space

W^h. Any $v^h \in W^h$ is determined uniquely by the value of v^h at the N_e edges, therefore the dimension of P^h is N_e. Within a given hexahedron the value of v^h is determined solely by the value at the twelve edges. By construction the tangential component of v^h is a continuous function across faces. The rate of convergence is first order,

$$\left|f - I_T f\right|_{H(curl)} \leq C h^1 |f|_{H^1}. \tag{106}$$

4.3.3 Face elements

The face elements are first defined on the reference element. The space P used for bilinear face elements is

$$P = \{\vec{u} : u_1 \in Q_{1,0,0}; u_2 \in Q_{0,1,0}; u_3 \in Q_{0,0,1}\}, \tag{107}$$

and the degrees of freedom are the flux through each face

$$\alpha_i(v) = \int_{f_i} (v \bullet n_i)\, ds\ , \tag{108}$$

where f_i is any of the six faces and n_i is the unit normal vector to f_i. On the reference hexahedron the space P is spanned by the six basis functions

$$\begin{aligned}
\vec{\tilde{F}}_1 &= (-1+x, 0, 0), \\
\vec{\tilde{F}}_2 &= (x, 0, 0), \\
\vec{\tilde{F}}_3 &= (0, -1+y, 0), \\
\vec{\tilde{F}}_4 &= (0, y, 0), \\
\vec{\tilde{F}}_5 &= (0, 0, -1+z), \\
\vec{\tilde{F}}_6 &= (0, 0, z) \ .
\end{aligned} \tag{109}$$

In order to remain affine equivalent, the elements must transform contravariantly,

$$\begin{aligned}
\hat{x} &= B\tilde{x} + \hat{b}, \\
\vec{\hat{W}} &= B\vec{\tilde{W}}.
\end{aligned} \tag{110}$$

Vector functions that transform this way are referred to as contravariant or axial vectors. The basis functions above are called local basis functions since they are only defined within the given tetrahedron.

Consider a hexahedral grid consisting of N_v hexahedrons K_i and N_f faces, and a linear face finite element (K_i, P_i, A_i) defined on every hexahedron. The global basis functions are defined to be equal to the local basis functions F inside K and to be zero outside of K. The collection of all the global basis functions will be denoted by F^h. Any $v^h \in F^h$ is determined uniquely by the value of v^h at the N_f faces, therefore the dimension of S^h is N_f. Within a given hexahedron the value of v^h is determined solely by the value at the six faces. By construction, the normal component of v^h is continuous across faces. The rate of convergence for these elements is first order,

$$\left|f - I_T f\right|_{H(div)} \leq Ch^1 |f|_{H^1}. \tag{111}$$

4.3.4 Volume elements

The lowest finite element for the volume element is trivial, it is simply a constant scalar within each hexahedron. Let the space of all such elements on a hexahedral grid be denoted by S^h. These elements have no continuity at all, and the rate of convergence is first order,

$$\left|f - I_T f\right|_{H^0} \leq Ch^1 |f|_1. \tag{112}$$

The hexahedral node, edge, face, and volume elements defined above are conforming on their infinite dimensional Hilbert spaces, and they satisfy the inclusion relations:

$$\text{If } \phi \in V^h \text{ then } \nabla\phi \in W^h, \tag{113}$$

$$\text{If } \vec{E} \in W^h \text{ then } \nabla\times\vec{E} \in F^h, \tag{114}$$

$$\text{If } \vec{B} \in F^h \text{ then } \nabla\bullet\vec{B} \in S^h. \tag{115}$$

It is interesting to note that the dimension of these finite element spaces satisfies Euler's equation

$$N_n - N_e + N_f - N_v = \chi, \tag{116}$$

where χ is the Euler characteristic of the domain Ω. The edge elements for a hexahedron are shown in Figure 10 through Figure 12, the face elements are shown in Figure 13. The Mathematica script used to generate these figures is included in Section 11.2.

FIGURE 8. Linear covariant elements (edge elements) defined on a tetrahedron.

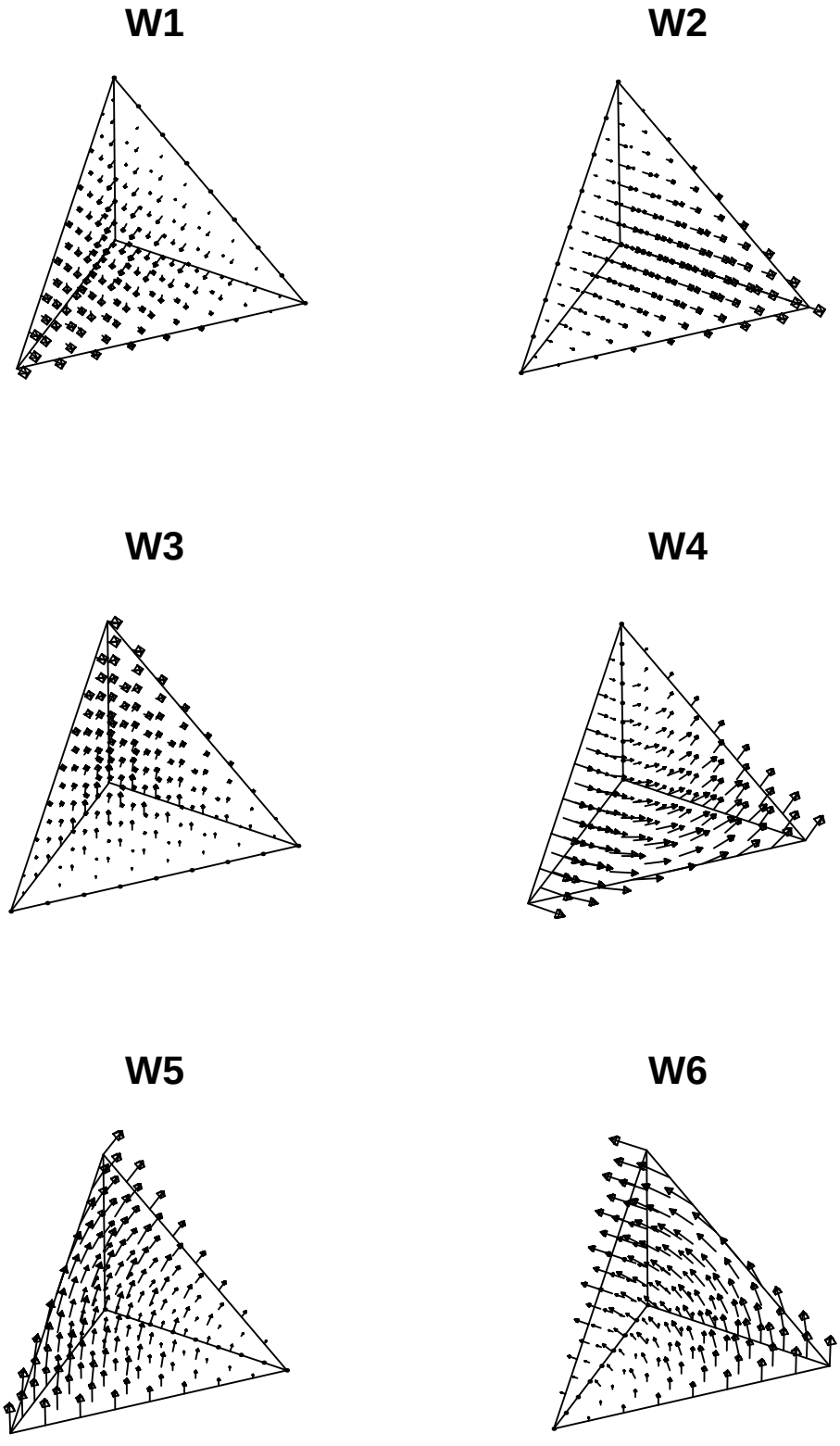


FIGURE 9. Linear contravariant elements (face elements) defined on a tetrahedron.

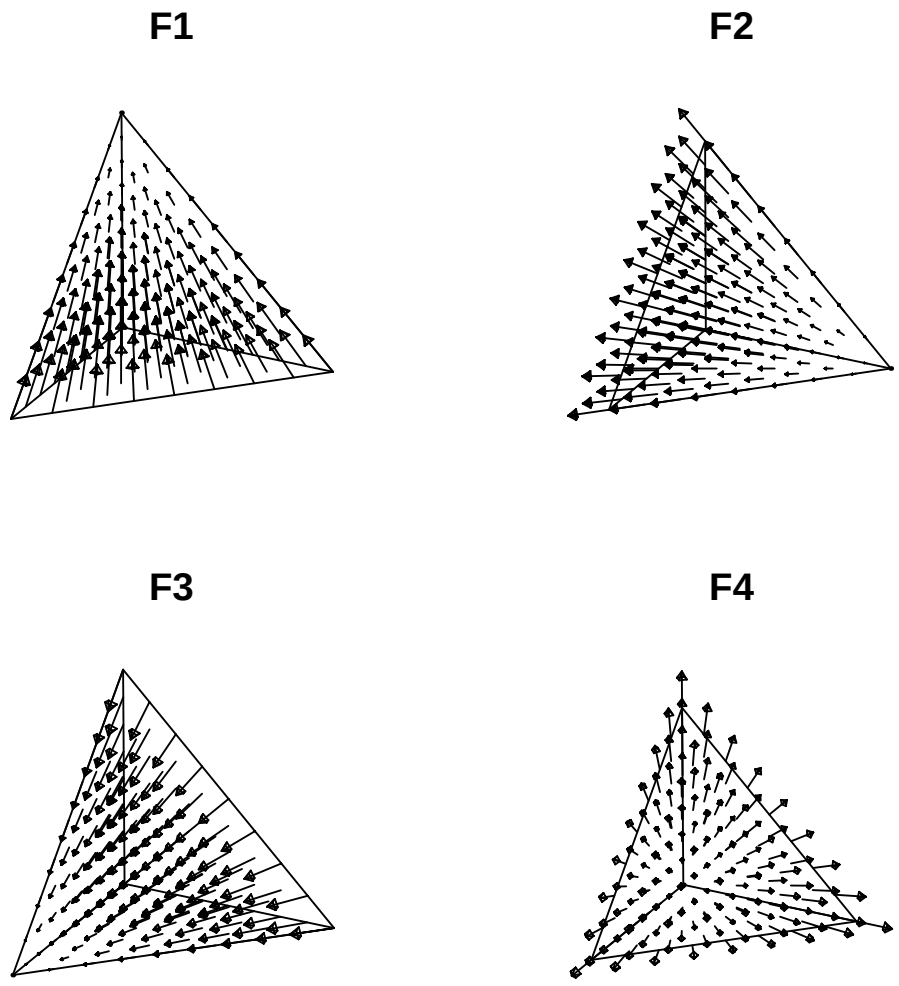


FIGURE 10. Linear covariant (edge elements) defined on a hexahedral (part 1).

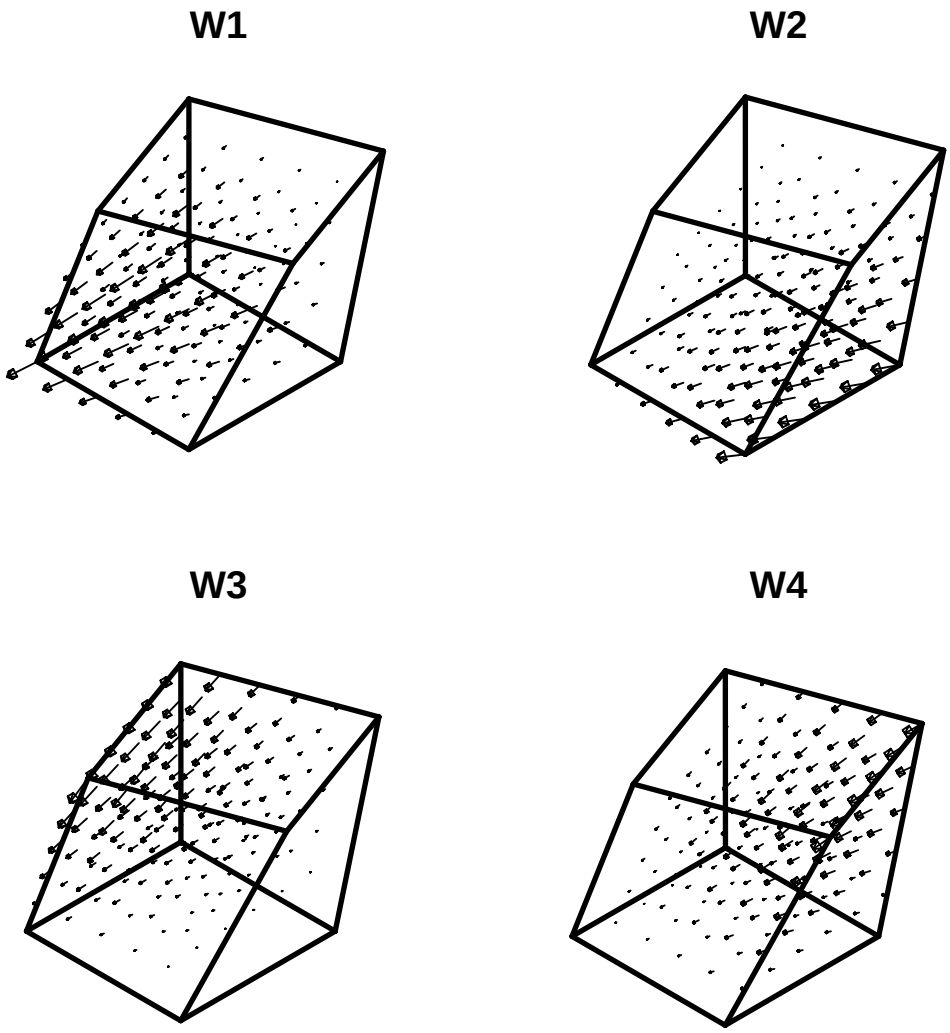


FIGURE 11. Linear covariant (edge elements) defined on a hexahedron (part 2).

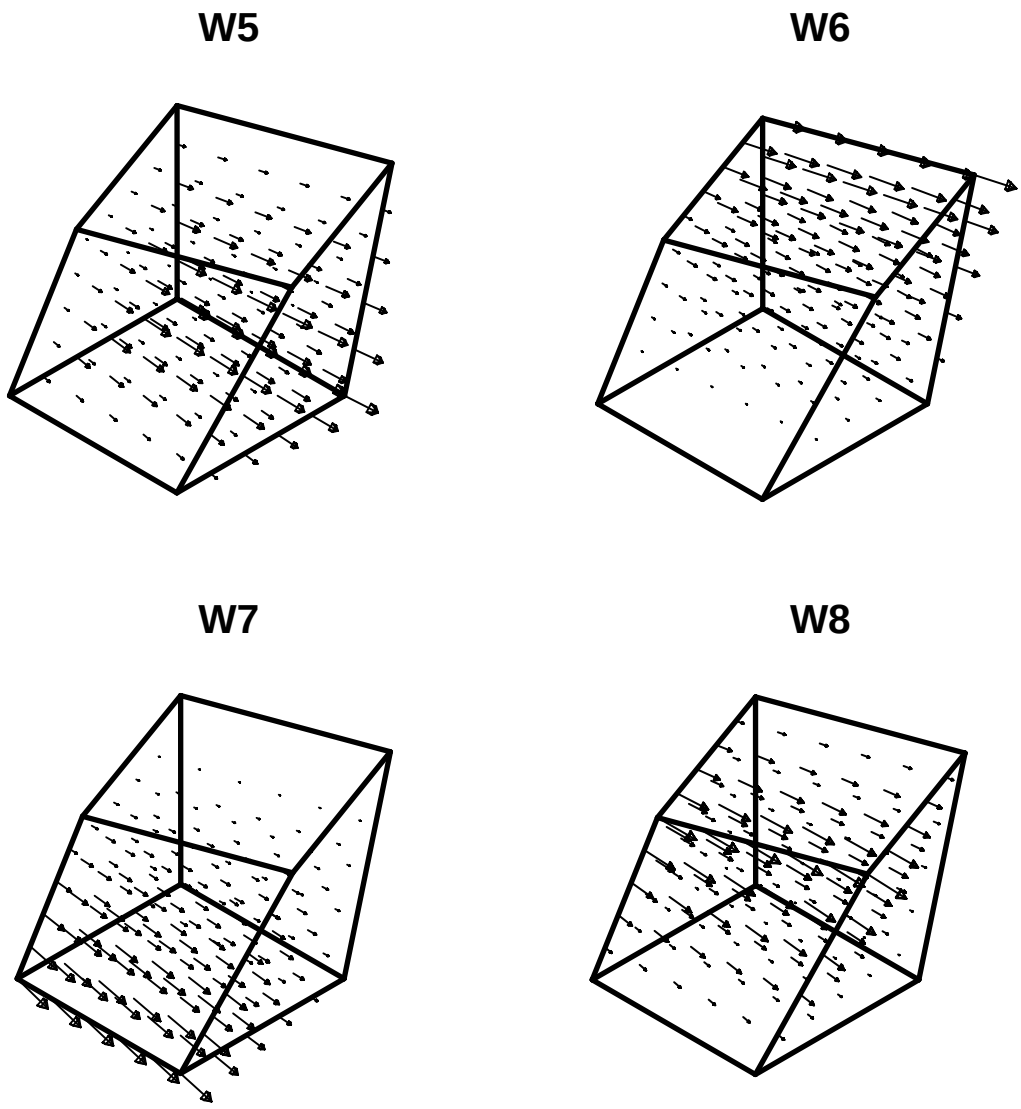


FIGURE 12. Linear covariant (edge elements) defined on a hexahedron (part 3).

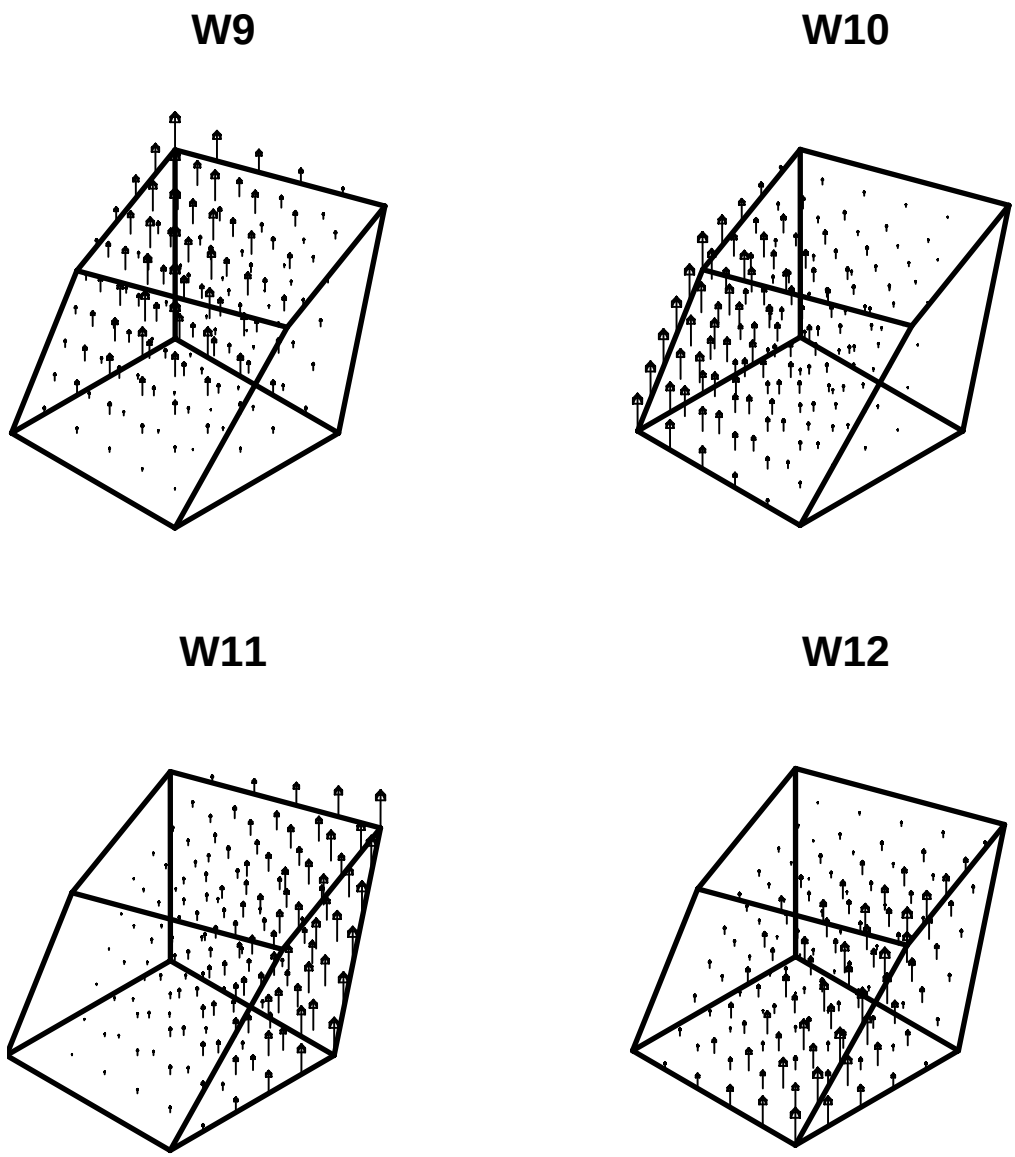
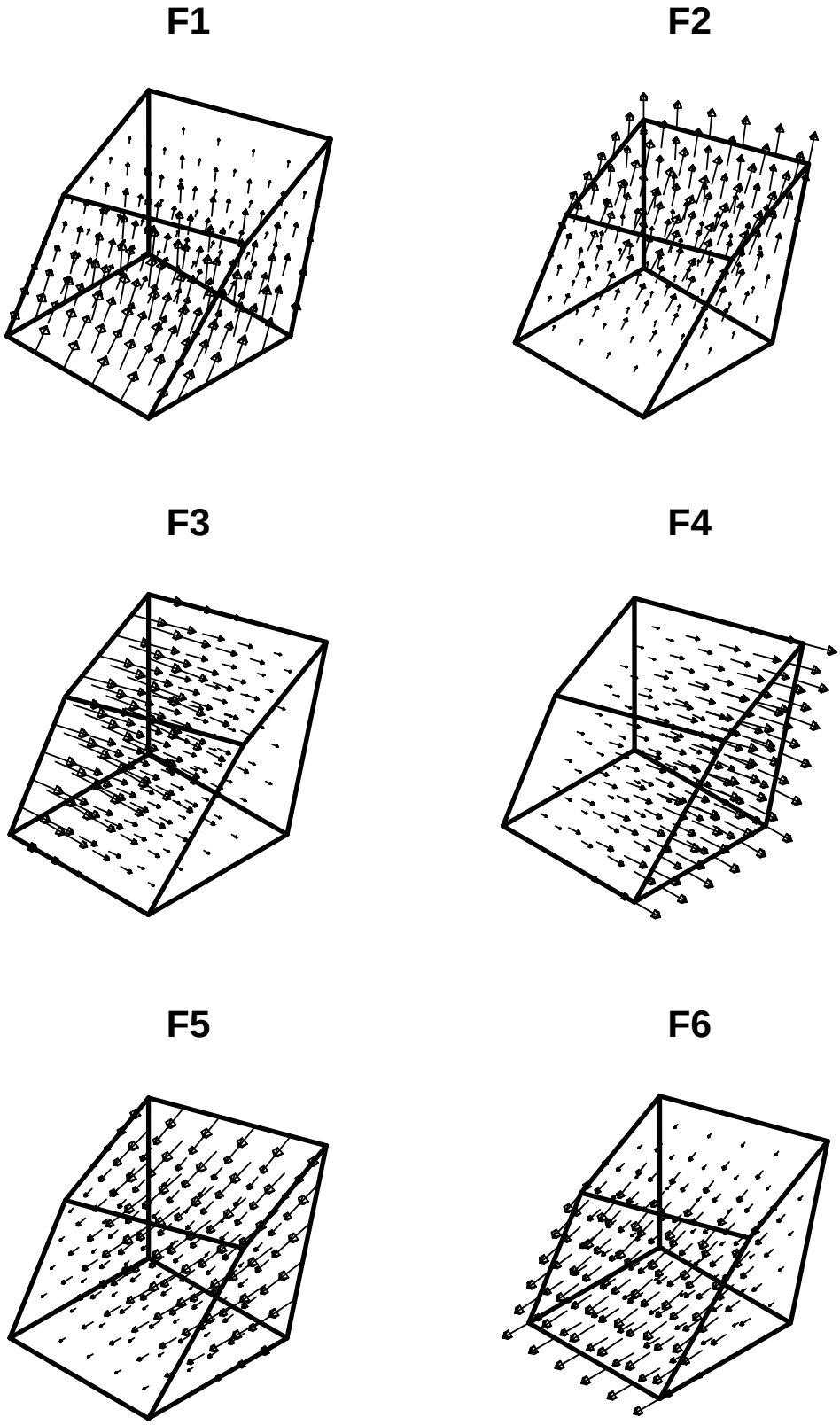


FIGURE 13. Linear contravariant (face elements) defined on a hexahedron.



4.4 Differential forms

In this dissertation, the laws of physics are expressed using classical vector calculus. There is an alternative method for expressing the same laws, namely differential forms. Differential forms have been used extensively in theoretical physics but have only recently been adopted by the engineering community. The language of differential forms allow the laws of physics to be expressed independent of any particular coordinate system. A classic textbook on differential forms is [62], and a more modern view is presented in[63]. The primary source for the above vector finite elements [28] uses differential forms extensively. An interesting connection between the above vector finite elements and differential forms will be examined in this section. Much of the following is from [64].

A differential form is an expression upon which integration operates. Forms of degree p, or p-forms, are expressions that occur in p-fold integrals over domains. For example the work done on a unit charge by the electric field is given by

$$W = \int_{\gamma} E_x dx + E_y dy + E_z dz, \tag{117}$$

where the quantity under the line integral is the one-form E. The total current flowing through a surface is given by

$$I = \int_{\Gamma} (J_x dy dz + J_y dz dx + J_z dx dy), \tag{118}$$

where the quantity under the surface integral is the two-form J. The total charge in a volume is given by

$$Q = \int_{\Omega} q dx dy dz , \tag{119}$$

where the quantity under the volume integral is the three-form ρ. Scalar functions of position, such as the electrostatic potential Φ, are considered zero-forms. They integrate over a domain of dimension zero, i.e. a point.

$$\Phi = \int_{P} \Phi . \tag{120}$$

Mathematicians have developed an algebra and a calculus for differential forms, often referred to as exterior algebra and exterior calculus, respectively. This is a generalization of the traditional vector algebra and vector calculus. The gradient, divergence, and curl of vector calculus are replaced by a single differential operator d, called the exterior derivative. Likewise the dot product and cross product are replaced by a single exterior product. The term exterior is used because the derivative of a p-form is not a p-form, it lies outside (hence exterior) of the space of p-forms. Likewise the product of two p-forms is not a p-form.

Maxwell's equations written using differential forms are:

$$\frac{\partial D}{\partial t} = dH - J , \tag{121}$$

$$\frac{\partial B}{\partial t} = -dE - M , \tag{122}$$

$$dD = \rho , \tag{123}$$

$$dB = 0, \tag{124}$$

where the electric flux density D, the magnetic flux density B are two-forms, the current densities J and M are two-forms, the electric field E and magnetic field H are one-forms, and the charge density ρ is a three-form. For simplicity the conductivity is assumed to be zero. The operator d is unambiguous since there is only one way to differentiate a p-form; the derivative of a p-form is a $(p+1)$-form. The constitutive relations are

$$D = \varepsilon E \qquad B = \mu H, \tag{125}$$

where ε and μ are not simple scalars, but rather operators that convert a two-form to a one-form. These operators define the metric of the space in which (121)-(124) are defined. Thus the "measure" of the electric field E and magnetic field H is

$$E\varepsilon E, \tag{126}$$

$$H\mu H, \tag{127}$$

where the product is unambiguous, since the product of a p-form and a q-form is a $(p+q)$-form. Thus [126] and [127] represent the electric and magnetic energy densities, which are three-forms. Multiplying [121] by E and [122] by H yields

$$d(EH) + \frac{1}{2}\frac{\partial}{\partial t}(E\varepsilon E + H\mu H) + EJ + HM = 0, \tag{128}$$

which is Poynting's theorem of energy conservation. The product EH is a two-form which represents the power flow, the three-forms EJ and HM represent power density supplied by the sources.

In three dimensional space there are four differential forms: zero-forms, one-forms, two-forms, and three-forms. These forms can be associated with the Hilbert spaces defined in Section 3.3 and the finite elements defined in Section 4.2 and Section 4.3 above. The connection between differential forms, Hilbert spaces, electromagnetic variables, and finite elements is shown in Table 3. The proposition is that by expressing a PDE using the language of differential forms it becomes obvious which type of finite elements should be used to approximate the variables.

TABLE 3. Connection between forms, electromagnetics, and finite elements.

	zero form	**one from**	**two form**	**three form**
integral	point	line	surface	volume
derivative	grad	curl	divergence	none
continuity	total	tangential	normal	none
Hilbert space	H(grad)	H(curl)	H(div)	L2
electromagnetics	potential	fields	fluxes, currents	charge density
finite element	V (nodal)	W (edge)	F (face)	S (volume)

4.5 Galerkin formulation of Maxwell's equations

The vector finite elements $\vec{W}$ and $\vec{F}$ developed above are used as basis functions for the electric field and the magnetic flux density, respectively. Let $\vec{W}_i$ be the basis function associated with edge i and $\vec{F}_i$ be the basis function associated with face i, then the electric field given by

$$\vec{E} = \sum_{i=1}^{N_e} e_i \vec{W}_i , \tag{129}$$

and the magnetic flux density is given by

$$\vec{B} = \sum_{i=1}^{N_f} b_i \vec{F}_i . \tag{130}$$

According to the definition of $\vec{W}$ in Section 4.2.2 and Section 4.3.2, the degrees of freedom e_i have units of volts and can be interpreted as the voltage along edge i of the grid. Likewise according the definition of $\vec{F}$ in Section 4.2.3 and Section 4.3.3 the degrees of freedom b_i have units of webers and can be interpreted as the magnetic flux through face i. The independent current sources are also expanded in terms of basis functions

$$\vec{J} = \sum_{i=1}^{N_f} j_i \vec{F}_i , \tag{131}$$

$$\vec{M} = \sum_{i=1}^{N_f} m_i \vec{F}_i . \tag{132}$$

The test spaces $H_0(curl)$ and $H_0(div)$ consist of all edge and face elements for edges and faces not on the boundary Γ. Now, using these solution and test spaces, the Galerkin form of PDE I and PDE II become systems of ordinary differential equations.

PDE I, Galerkin form

$$G\frac{\partial b}{\partial t} = -Ke - Pb - LIm, \tag{133}$$

$$C\frac{\partial e}{\partial t} = K^T b - Se - Qj, \tag{134}$$

where the matrices are given by

$$\begin{aligned}
G_{ij} &= \left(\mu^{-1}\vec{F}_j, \vec{F}_i\right),\\
K_{ij} &= \left(\vec{\mu}^{-1}\nabla\times\vec{W}_j, \vec{F}_i\right),\\
P_{ij} &= \left(\mu^{-1}\sigma_M\mu^{-1}\vec{F}_j, \vec{F}_i\right),\\
C_{ij} &= (\varepsilon\vec{W}_j,\vec{W}_i),\\
K^T{}_{ij} &= \left(\vec{\mu}^{-1}\nabla\times\vec{W}_i, \vec{F}_j\right),\\
S_{ij} &= (\sigma_E\vec{W}_j,\vec{W}_i),\\
Q_{ij} &= (\vec{F}_j,\vec{W}_i).
\end{aligned} \tag{135}$$

Of key importance is the fact that (133) involves K while (134) involves K^T, therefore the discrete equations have the same hyberbolicity as the original PDE's [67]. This property is not shared with some other proposed methods that use different variational forms of Maxwell's equations. This property will allow stable, non-dissipative time integration. The matrices C and G are symmetric positive definite, they have units of farads and inverse henries and can be interpreted as the capacitance and reciprocal inductance of the grid, respectively. The matrices S and P have units of siemens and siemens meter-cubed per square henry and can be interpreted as the electric and magnetic conductivity of the grid. While (133) and (134) may look unusual at first, they do in fact have an interesting physi-

cal interpretation. Multiplying (133) by $L = G^{-1}$ and defining electric current $i = Gb$, (133) and (134) become analogous to the classic telegraphists equations

$$L\frac{\partial i}{\partial t} = -\frac{\partial}{\partial x}e - Ri, \tag{136}$$

$$C\frac{\partial e}{\partial t} = \frac{\partial}{\partial x}i - Se, \tag{137}$$

for the current and voltage on a one-dimensional transmission line. The solution to these equations is of course a propagating wave with exponential decay.

PDE II, Galerkin form

$$C\frac{\partial^2 e}{\partial t^2} + T\frac{\partial e}{\partial t} + Ue = -Ae - Vj - K^T m - Q\frac{\partial j}{\partial t}, \tag{138}$$

where the matrices are given by

$$\begin{aligned} T_{ij} &= \left(\left(\sigma_E + \mu^{-1}\sigma_M \varepsilon \right) \vec{W}_j, \vec{W}_i \right), \\ U_{ij} &= \left(\mu^{-1}\sigma_M \sigma_E \vec{W}_j, \vec{W}_i \right), \\ A_{ij} &= \left(\vec{\mu}^{-1} \nabla \times \vec{W}_j, \nabla \times \vec{W}_i \right), \\ V_{ij} &= (\mu^{-1}\sigma_M \vec{F}_j, \vec{W}_i) \, . \end{aligned} \tag{139}$$

Although (138) looks complicated at first, it is really just a wave equation, with dissipation, for the voltage e due to electric and magnetic sources j and m. Note that (138) can be derived either from the variational form of PDE II directly, or it can be derived by simply eliminating the variable b from (133) and (134). It can be shown that

$$A = K^T L K \ , \tag{140}$$

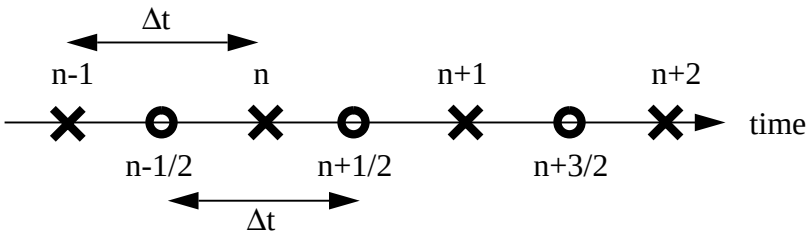
where A is obviously symmetric positive definite, preserving the hyperbolicity of the original PDE.

In VFEM3D the matrices in (135) and (139) are evaluated using an exact closed form expression for tetrahedral volumes and Gaussian quadrature for hexahedral volumes. The Gaussian quadrature was constrained to use not more than ten points in each dimension.

5.0 Time Integration

Using vector finite elements, the variational form of Maxwell's equations are converted to finite dimensional systems of ODE's. These ODE's are integrated in time using standard second order accurate finite difference formulas. For PDE I, time is discretized such that the electric degrees of freedom will be known at whole time steps, the magnetic degrees of freedom will be known at the half time steps, as illustrated in Figure 14. This is often referred to as a leapfrog method.

FIGURE 14. Staggered time scheme for electric and magnetic degrees of freedom.



The second order accurate finite difference formula for the first derivative is

$$\left(\frac{\partial x}{\partial t}\right)^n = \frac{x^{n+1/2} - x^{n-(1/2)}}{\Delta t} + O\left(\Delta t^2\right), \tag{141}$$

and the dissipative terms are handled implicitly using the second order accurate finite difference formula for the average

$$x^n = \frac{x^{n+1/2} + x^{n-(1/2)}}{2} + O\left(\Delta t^2\right) \quad . \tag{142}$$

Using these formulas in (133) and (134) gives the DTVFEM for PDE I,

PDE I, discrete time

$$\begin{aligned}(G+\Delta tP/2)\,b^{n+1/2} &= -\Delta tKe^{n}+(G-\Delta tP/2)\,b^{n-1/2}-\Delta tGm^{n}\\ (C+\Delta tS/2)\,e^{n+1} &= \Delta tK^{T}b^{n+1/2}+(C-\Delta tS/2)\,e^{n}-\Delta tQj^{n+1/2}\end{aligned}. \tag{143}$$

For PDE II, time is discretized such that the electric degrees of freedom are known at whole time steps. The second order accurate finite difference formula for the second derivative is

$$\left(\frac{\partial^{2}x}{\partial t^{2}}\right)^{n} = \frac{x^{n+1}-2x^{n}+x^{n-1}}{\Delta t^{2}}+O\left(\Delta t^{2}\right). \tag{144}$$

Using the above finite difference formulas for the second and first derivative (138) gives the DTVFEM for PDE II,

PDE II, discrete time

$$\begin{aligned}(C+\Delta tT/2)\,e^{n+1} = {} & Ce^{n}-\Delta t^{2}(A+U)\,e^{n}-(C-\Delta tT/2)\,e^{n-1}-\\ & \Delta t^{2}\left(Vj^{n}+K^{T}m^{n}+Q\frac{\partial j^{n}}{\partial t}\right)\end{aligned}. \tag{145}$$

The above finite difference equations (FDE) are consistent. As the time step Δt approaches zero the FDE's reduce to the original ODE's. But consistency does not tell the whole story. A discrete time integration method is convergent if the solution to the discrete equations converge to the solution of the original ODE's as the time step approaches zero. It is well known that for a time stepping method to be convergent the method must be both consistent and stable. A qualitative definition of stability from [68] is

A finite difference approximation of an ODE is stable if it produces a bounded solution when the exact solution is bounded, and it is unstable if it produces an unbounded solution when the exact solution is bounded.

Note that this definition is appropriate only for equations for which the exact solution is known to be bounded. However this definition is fine for the problems addressed in this dissertation. A more quantitative definition of stability is developed in the next section.

5.1 Stability

In Section 2.1, two different PDE's were presented. These problems were named PDE I and PDE II. PDE I is a coupled system of first order PDE's for both the electric field and the magnetic flux density, whereas, PDE II is a second order PDE for the electric field alone. In Section 3.3, the variational form of these problems was presented, and the discrete time version of these two problems was presented above. Both problems give identical values of the electric field, the only difference between the two problems is that the first one uses the magnetic flux density as an intermediate variable, while the second problem does not. Thus the coupled first order system (143) will have the exact same stability as the single second order equation (145). It is easier to prove stability of (145), however stability of the system (143) will be derived because it leads to an important conservation of energy result.

The most difficult equations to integrate are those that do not have any physical loss mechanism. In a region of space in which there is no electric or magnetic conductivity, a wave will simply propagate unattenuated, and numerical errors may build up in such a way that

the method is unstable. Alternatively, in a region of space in which there exists electric or magnetic conductivity a wave will be attenuated exponentially as it propagates. Any numerical errors will also be attenuated exponentially. For this reason, the stability condition will be derived by assuming no electric or magnetic conductivity. In this case the discrete equations become

$$\begin{aligned} e^{n+1} &= \Delta t C^{-1} K^T b^{n+1/2} + e^n \\ b^{n+1/2} &= -\Delta t G^{-1} K e^n + b^{n-1/2} \end{aligned}, \tag{146}$$

where the source terms have also been neglected. These equations can be expressed in matrix form as

$$\begin{bmatrix} e^{n+1} \\ b^{n+1/2} \end{bmatrix} = \begin{bmatrix} \left(I - \Delta t^2 C^{-1} K^T G^{-1} K \right) & \Delta t C^{-1} K^T \\ -\Delta t G^{-1} K & I \end{bmatrix} \begin{bmatrix} e^n \\ b^{n-1/2} \end{bmatrix}, \tag{147}$$

or more generically as

$$\dot{x}^{n+1} = Q \dot{x}^n . \tag{148}$$

The matrix Q in (148) is called the amplification matrix of the method. The general condition for stability is that

$$\|Q\| \le 1 , \tag{149}$$

where $\| \ \|$ denotes the two norm. Note that some texts [67] use the stability condition

$$\|Q\| \le 1 + O(\Delta t) , \tag{150}$$

which allows for growth of the solution. However for the DTVFEM (149) is the appropriate condition, as is discussed in Section 5.2. If $\chi(Q)$ is the spectral radius of Q, the relation

$$\chi(Q) \leq \|Q\| \tag{151}$$

is known to hold. Thus,

$$\chi(Q) \leq 1 \tag{152}$$

is a necessary condition for stability. This is referred to as the von Neumann condition. A tedious but straightforward calculation shows that the eigenvalues of Q are given by

$$\lambda = \frac{\tilde{\lambda} \pm i\sqrt{4-\tilde{\lambda}^2}}{2}, \tag{153}$$

where $\tilde{\lambda} = 2-\Delta t^2 \zeta$ and ζ is an eigenvalue of the matrix $C^{-1}K^T G^{-1}K$. Equivalently, ζ and $\hat{x}$ satisfy the generalized eigenvalue problem

$$C\hat{x} = \zeta K^T G^{-1} K\hat{x}. \tag{154}$$

The matrix C is symmetric positive definite and the matrix $K^T G^{-1}K$ is symmetric positive semi-definite. Thus $\zeta \geq 0$ and the eigenvalues λ of the amplification matrix Q will have unit magnitude if and only if

$$\Delta t \leq \frac{2}{\sqrt{max(\zeta)}} \quad . \tag{155}$$

This is the necessary stability condition. Note that the von Neumann condition is not in general sufficient to prove stability, but in this case the amplification matrix Q is similar, via a complex similarity transformation, to a diagonal matrix, therefore if (155) is satisfied the method is stable. The stability condition (155) can be interpreted as the statement that the sampling frequency must be less than one-half the highest resonant frequency of the grid.

In Section 1.1 it was mentioned that some finite difference and finite volume methods are unstable when implemented on unstructured grids. The operator $\nabla\times\nabla\times$ is self-adjoint (with the boundary condition $n\times E = 0$), however some finite difference and finite volume methods do not result in a symmetric discrete approximation to $\nabla\times\nabla\times$. If the matrix representing the discrete approximation to $\nabla\times\nabla\times$ is not symmetric the eigenvalues ζ may be complex, in which case $\chi(Q) > 1$ for any Δt and the method is unconditionally unstable. However this is not a problem for the DTVFEM since the matrix $K^T G^{-1} K$, which is the discrete approximation the $\nabla\times\nabla\times$ operator, is symmetric for any grid.

5.2 Conservation of Energy

It is shown above that the eigenvalues of the amplification matrix have unit magnitude as long as the stability condition (155) is satisfied. If the eigenvalues had magnitude less than unity, the method would still be stable, but the solution would decay with time. This is often referred to as numerical dissipation since there is no physical loss mechanism, i.e. there is no electric or magnetic conductivity to absorb the fields. A method for which the

eigenvalues of the amplification matrix are all of unit magnitude is called non-dissipative, or neutrally stable. In Section 1.1, it was mentioned that some finite volume methods do not conserve energy. This is because they employ dissipative time integration methods; in fact they require dissipative time integration in order to be stable. In this section it is shown that because the DTVFEM is non-dissipative, a variational form of Poynting's theorem of energy conservation is satisfied.

From Section 2.1, Poynting's theorem is

$$\oint_{\Gamma} \mu^{-1} \hat{E} \times \hat{B} \bullet \hat{n} d\Gamma + \int_{\Omega} \mu^{-1} \hat{B} \bullet \hat{M} d\Omega + \int_{\Omega} \hat{E} \bullet \hat{J} d\Omega + \int_{\Omega} \mu^{-1} \sigma_M \hat{B} \bullet \hat{B} d\Omega + \int_{\Omega} \sigma_E \hat{E} \bullet \hat{E} d\Omega + \int_{\Omega} \mu^{-1} \hat{B} \bullet \frac{\partial}{\partial t} \hat{B} d\Omega + \int_{\Omega} \varepsilon \hat{E} \bullet \frac{\partial}{\partial t} \hat{E} d\Omega = 0 . \tag{156}$$

The first term can be written as

$$\oint_{\Gamma} \mu^{-1} (\hat{E} \times \hat{B}) \bullet \hat{n} d\Gamma = \int_{\Omega} \mu^{-1} \hat{B} \bullet \nabla \times \hat{E} d\Omega - \int_{\Omega} \mu^{-1} \hat{E} \bullet \nabla \times \hat{B} d\Omega . \tag{157}$$

Using the degrees of freedom e, b, and the matrices defined in (135) and (139), Poynting's theorem can be written as

$$b^T K e - e^T K^T b + b^T G m + e^T Q j + b^T P b + e^T S e + b^T G \frac{\partial b}{\partial t} + e^T C \frac{\partial e}{\partial t} = 0 . \tag{158}$$

The combined first two terms represent the power flowing into the domain, which depends upon the boundary condition. Assume that $\hat{E}$ is zero on the boundary, thus there is not net power flow into the domain. The terms $b^T G m$ and $e^T Q j$ represent the power put into the

domain by the independent magnetic and electric current sources, respectively. The terms $b^T Pb$ and $e^T Se$ represent the power dissipated due to magnetic and electric conductivity, respectively. The last two terms $b^T G\frac{\partial b}{\partial t}$ and $e^T C\frac{\partial e}{\partial t}$ represent the time rate of change of stored magnetic and electric energy. Again consider the case where electric and magnetic conductivity is zero. Poynting's theorem reduces to

$$b^T G\frac{\partial b}{\partial t} + e^T C\frac{\partial e}{\partial t} = 0 , \tag{159}$$

the total energy in the domain is a constant. Using the result that the eigenvalues of the amplification matrix all have unit magnitude, it can be shown that

$$\left(e^{n+1}\right)^T Ce^{n+1} + \left(b^{n+1/2}\right)^T Gb^{n+1/2} + \left(e^n\right)^T Ce^n + \left(b^{n-1/2}\right)^T Gb^{n-1/2} , \tag{160}$$

is a constant for all time. Thus the DTVFEM conserves energy in a time-average sense. This relation is valid for any grid and for any stable time step.

5.3 Conservation of Charge

5.3.1 Magnetic charge

Recall (129) and (130), which define the electric field and the magnetic flux density in terms of the edge and face vector finite elements,

$$\vec{E} = \sum_{i=1}^{N_e} e_i \vec{W}_i \qquad \vec{B} = \sum_{i=1}^{N_f} b_i \vec{F}_i . \tag{161}$$

Again, consider the case of an electromagnetic wave propagating in a source free, zero conductivity region. Magnetic charge will be conserved if

$$\nabla \bullet \frac{\partial}{\partial t} \vec{B} = 0, \tag{162}$$

or alternatively

$$\oint_{\Gamma} \frac{\partial}{\partial t} \vec{B} \bullet \hat{n} d\Gamma = 0, \tag{163}$$

for every tetrahedral or hexahedral volume in the grid. In terms of the degrees of freedom, this can be expressed as

$$\sum_{i}^{n} \frac{\partial b_i}{\partial t} = 0, \tag{164}$$

since the degrees of freedom b_i are precisely the net magnetic flux through face i. Define the subspace

$$\vec{\tilde{F}}^{h}{}_{0} = \{ \vec{u} \in \vec{\tilde{F}}^{h}, \nabla \bullet \vec{u} = 0 \}. \tag{165}$$

An important property of the vector finite element spaces $\vec{W}^{h}$ and $\vec{\tilde{F}}^{h}$ is that the operator $\nabla \times$ is surjective from $\vec{W}^{h}$ onto $\vec{\tilde{F}}_{0}{}^{h}$. This means that the curl of any edge element can be written as a linear combination of face elements with a net magnetic flux of zero. This is illustrated in Figure 15. The electric field inside a tetrahedron is given by

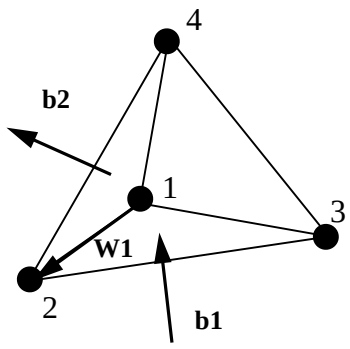
$$\vec{\vec{E}} = \sum_{i=1}^{6} e_i \vec{\vec{W}}_i , \tag{166}$$

and the time rate of change of the magnetic flux density is given by

$$\begin{gathered} \frac{\partial}{\partial t}\vec{\vec{B}} = \sum_{i=1}^{6} e_i \nabla\times\vec{\vec{W}}_i = e_1 (F_2 - F_1) + e_2 (F_1 - F_3) + \\ e_3 (F_3 - F_2) + e_4 (F_4 - F_1) + e_5 (F_2 - F_4) + e_6 (F_4 - F_3) = \\ F_1 (e_2 - e_1 - e_4) + F_2 (e_1 + e_5 - e_3) + F_3 (e_3 - e_2 - e_6) + F_4 (e_4 + e_6 - e_5) \ . \end{gathered} \tag{167}$$

It is obvious that for any electric field the magnetic degrees of freedom sum to zero, thus (162) - (164) are satisfied exactly. Magnetic divergence is preserved. If the initial magnetic charge in each polyhedral region is zero it will remain zero for all time. An analogous argument holds for the hexahedral elements as well.

FIGURE 15. The curl of an arbitrary edge element is divergence free.



The variational form of Faradays law in a source free, zero conductivity region is given by

$$\frac{\partial}{\partial t}(\mu^{-1}\vec{\vec{B}},\vec{\vec{B}}^*) = -(\mu^{-1}\nabla\times\vec{\vec{E}},\vec{\vec{B}}^*) . \tag{168}$$

This can be interpreted as a projection of $\nabla\times\vec{E}$ onto the space $\vec{B}\in H(div)$. The above analysis simply shows that this projection is exact when using edge elements for the electric field and face elements for the magnetic flux density.

5.3.2 Electric charge

The electric field is assumed to be a linear combination of edge elements. Since these elements do not have normal continuity across cell faces, 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. The variational form is

$$\int_{\Omega}\phi\,(\nabla\bullet\varepsilon\vec{E})\,d\Omega \;=\; -\int_{\Omega}\varepsilon\vec{E}\bullet\nabla\phi\,d\Omega+\oint_{\Gamma}\phi\varepsilon\vec{E}\bullet\hat{n}\,d\Gamma\,, \tag{169}$$

where ϕ is a continuous piecewise linear function. Since the field is not required to be divergence free on the external boundary we can choose $\phi = 0$ on Γ, thus the last term in (169) is zero. Using a similar argument to that used in Section 5.3.1 above, the requirement for charge conservation is

$$\left(\frac{\partial}{\partial t}\varepsilon\vec{E},\,\nabla\phi\right) \;=\; 0\,, \tag{170}$$

and the variational form of Amperes law in a source free, zero conductivity region is

$$\frac{\partial}{\partial t}(\varepsilon\vec{E},\vec{E}^{*}) \;=\; (\vec{\nabla}\times\vec{E}^{*},\mu^{-1}\vec{B})\,. \tag{171}$$

As discussed in Section 4.1 the space $\vec{W}^h$ includes $\nabla\phi$. Since (171) must hold for all $\vec{E}^* \in \vec{W}^h$, it must hold for $\vec{E}^* = \nabla\phi$. Therefore

$$\frac{\partial}{\partial t}(\varepsilon\vec{E},\nabla\phi) = (\vec{\nabla}\times\nabla\phi,\mu^{-1}\vec{B}) = 0, \tag{172}$$

and divergence is preserved for all time in the variational sense. Again, this is a consequence of the inclusion relations discussed in Section 4.1, and charge is not conserved when other basis functions, such as nodal basis functions, are used for the electric field.

5.4 Numerical Dispersion

Consider PDE II in an infinite, source free, zero conductivity region. In this case PDE II becomes

$$\varepsilon\frac{\partial^2}{\partial t^2}\vec{E} = -\nabla\times\mu^{-1}\nabla\times\vec{E}, \tag{173}$$

which is simply the vector wave equation. If μ and ε are constant scalars, the general solution to (173) is a plane wave of the form

$$\vec{E} = \vec{E}_0 e^{I(\vec{k}\bullet\vec{x}-\omega t)}, \tag{174}$$

where ω is the radian frequency, $\vec{k}$ is the wave vector, and $\vec{E}_0$ is a constant vector perpendicular to $\vec{k}$ that determines the polarization of the wave. The plane wave is a solution to the vector wave equation only if the dispersion relation

$$\omega^2 = c^2k^2 \tag{175}$$

holds, where $c = 1/(\sqrt{\mu\varepsilon})$ is the speed of light and $k = |\vec{k}|$ is the wave number. The phase velocity of the wave is defined as

$$v = \frac{\omega}{k} \tag{176}$$

which equals the speed of light c.

In many media μ and ε are not constant, thus the phase velocity is not constant. If the phase velocity depends upon ω (or equivalently, k) then the medium is said to be dispersive. A narrow pulse propagating in such a medium will spread out, or disperse, because each Fourier component of the propagates at a different velocity. If the phase velocity depends upon $\hat{k}$ the medium is said to be anisotropic. In an anisotropic medium plane waves propagate at different velocities in different directions.

The DTVFEM, like other grid based methods for solving Maxwell's equations, exhibits numerical dispersion and numerical anisotropy due to the finite grid and finite time sampling. The numerical anisotropy for frequency domain vector finite element methods has been derived for a variety of two dimensional triangular grids [41]-[43], and for two dimensional Cartesian grids [44]. In this section the numerical dispersion relation for a plane wave propagating on an infinite, uniform, distorted, three dimensional hexahedral grid will be derived. The main result of this analysis is that regardless of the distortion, the numerical dispersion relation is second order accurate,

$$\omega^2 = c^2 k^2 \left(1 + O\left((k\Delta h)^2 \right) + O\left((\omega \Delta t)^2 \right) \right), \tag{177}$$

as the grid is refined and the time step is reduced the numerical phase velocity approaches the speed of light c. The analysis will be performed using the Galerkin form of PDE II for simplicity, although the results are applicable to the Galerkin form of PDE I as well.

The Galerkin form of (173) is

$$C \frac{\partial^2 e}{\partial t^2} = -Ae, \tag{178}$$

where the electric degrees of freedom are given by

$$e_i = \int_{a_i} \vec{E} \bullet \hat{t} dl \tag{179}$$

with $\hat{t}$ the unit tangent vector to edge a_i, and the matrices C and A are defined by (139). The grid is assumed to be composed of identical hexahedral cells, which may be distorted. For this analysis the distortion is such that edges e_1 - e_4 are parallel, edges e_5 - e_8 are parallel, and edges e_9 - e_{12} are parallel. This is illustrated in Figure 16. Since the solution to (178) is assumed to be a plane wave, the vector e has only three independent components, denoted as X, Y, and Z. Edges e_1 - e_4 are related by

$$\begin{aligned} e_1 &= X \\ e_2 &= Xe^{I\left(\hat{k} \bullet \vec{\Delta_2} - \omega\Delta t\right)} \\ e_3 &= Xe^{I\left(\hat{k} \bullet \vec{\Delta_3} - \omega\Delta t\right)} \\ e_4 &= Xe^{I\left(\hat{k} \bullet \vec{\Delta_4} - \omega\Delta t\right)} \end{aligned} \tag{180}$$

where Δ_i is the vector from the midpoint of edge e_1 to the midpoint of edge e_i. Edges e_5 - e_8 are related by

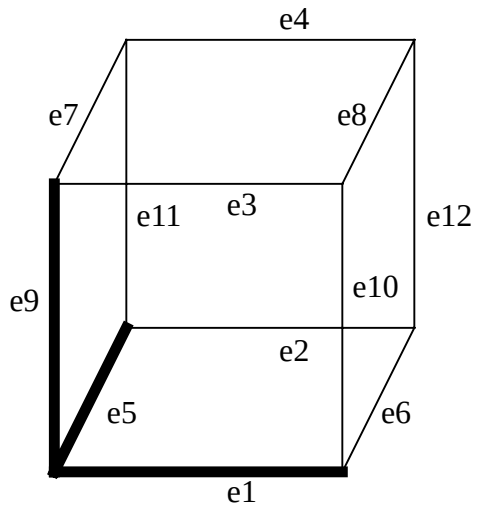
$$\begin{aligned} e_5 &= Y \\ e_6 &= Ye^{I\left(\hat{k} \bullet \vec{\Delta_6} - \omega\Delta t\right)} \\ e_7 &= Ye^{I\left(\hat{k} \bullet \vec{\Delta_7} - \omega\Delta t\right)} \\ e_8 &= Ye^{I\left(\hat{k} \bullet \vec{\Delta_8} - \omega\Delta t\right)} \end{aligned} \tag{181}$$

where Δ_i is the vector from the midpoint of edge e_5 to the midpoint of edge e_i. Edges e_9 - e_{12} are related by

$$
\begin{aligned}
e_9 &= Z \\
e_{10} &= Ze^{I\left(\hat{k}\bullet\overrightarrow{\Delta_{10}}-\omega\Delta t\right)} \\
e_{11} &= Ze^{I\left(\hat{k}\bullet\overrightarrow{\Delta_{11}}-\omega\Delta t\right)} \\
e_{12} &= Ze^{I\left(\hat{k}\bullet\overrightarrow{\Delta_{12}}-\omega\Delta t\right)}
\end{aligned}
\tag{182}
$$

where Δ_i is the vector from the midpoint of edge e_9 to the midpoint of edge e_i.

FIGURE 16. Edge numbering for numerical dispersion analysis



The time derivative in (173) is approximated by the second order central difference formula (144), therefore

$$
\left(\frac{\partial^2 e_i}{\partial t^2}\right)^n \approx \frac{e_i^{\,n+1} - 2e_i^{\,n} + e_i^{\,n-1}}{\Delta t^2} = \frac{\Psi e_i}{\Delta t^2}, \quad \Psi = 2\left(\cos\left((\omega\Delta t) - 1\right)\right). \tag{183}
$$

Combining (183) with (178) and (180)-(182) yields a homogeneous system of equations

$$
(\Psi F + \eta G)\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = 0, \tag{184}
$$

for X, Y, and Z. In (184) $\eta = c^2 \frac{\Delta t^2}{\Delta h^2}$ is a given constant. The 3 by 3 matrices F and G are functions of the wave vector $\vec{k}$ and the matrices C and A, respectively. The matrices F and G are complicated and are not shown here. Instead a Mathematica script that generates these matrices is provided in Section 11.3.

The numerical dispersion relation is given by

$$det(\Psi F + \eta G) = 0. \tag{185}$$

This is a complicated non-linear relationship between the wave vector $\vec{k}$ and the radian frequency ω. There are three roots; one is zero which does not represent anything physical, and the other two correspond to the two distinct polarizations. The roots can be expanded in a Taylor series about $\Delta h = 0$, the result is

$$F = k^2 + O\left((\Delta h)^2 \right). \tag{186}$$

The $O\left((\Delta x)^2 \right)$ term depends upon the distortion of the grid and upon $\vec{k}$. For the special case of a uniform Cartesian grid, with $\vec{k} = a\hat{x} + b\hat{y} + c\hat{z}$, the Taylor expansion is

$$F = k^2\left(1 + \frac{1}{12}(k\Delta h)^2 + \frac{1}{360}\left((a\Delta h)^4 + (b\Delta h)^4 + (c\Delta h)^4 \right) + O\left(\Delta h^6 \right) \right), \tag{187}$$

but in general the expression is much more complicated. The Taylor series of F expanded about $\Delta t = 0$ is

$$\omega^2\left(1 - \frac{1}{12}(\omega\Delta t)^2 + \frac{1}{360}(\omega\Delta t)^4 + O\left((\omega\Delta t)^6 \right) \right). \tag{188}$$

The general numerical dispersion relation is then

$$\frac{\omega^2}{k^2} = c^2 \frac{1 + O\left((k\Delta h)^2 \right)}{\left(1 - \frac{1}{12} (\omega \Delta t)^2 + O\left((\omega \Delta t)^4 \right) \right)}, \tag{189}$$

which is consistent with the physical dispersion relation (175). The numerator is the anisotropic part of the numerical dispersion relation, the denominator is the isotropic part. A Mathematica script that performs the series expansion is provided in Section 11.4.

Given matrices C and A, it is possible to pick a value of $\hat{k}$ and solve numerically for the value of ω that satisfies (185). Then the numerical phase velocity is then given by (176). This process is performed for several different grids below.

5.4.1 Numerical dispersion for two-dimensional shear distortion.

Consider a unit square that is sheared in the x direction by an amount θ, as illustrated in Figure 17. The value $\theta = 0°$ corresponds to no distortion, i.e. a Cartesian grid. Let

$$\hat{k} = k (\hat{x} \cos(\phi) + \hat{y} \sin(\phi)) \tag{190}$$

be the wave vector as a function of the polar angle ϕ. Given a value of k, it is possible to compute the numerical phase velocity as a function of polar angle ϕ. In the computational experiments below $c = 1$, $\Delta h = 1$, and $\Delta t = 1/3$. Figure 18 through Figure 21 are polar plots of the phase velocity error for shear angles of $\theta = 0°$, $\theta = 15°$, $\theta = 30°$, and $\theta = 45°$, respectively. Each figure shows the velocity error for $k = 2\pi/5$, $k = 2\pi/10$, $k = 2\pi/15$, and $k = 2\pi/20$. The Mathematica script that was used to generate these velocity error curves is provided in Section 11.5.

The phase velocity error is defined as $v - c$. The phase velocity error was always positive for these grids, indicating that the numerical phase velocity is slightly greater than c. It is interesting to note that the numerical phase velocity for the classic FDTD method is always less than c [3][4]. This difference is analogous with a result from continuum mechanics, where the resonant frequencies of a simply supported beam are over-estimated

by a consistent mass matrix finite element method, and under estimated by a lumped mass matrix finite element method [65].

FIGURE 17. Definition of shear angle for distorted quadrilateral.

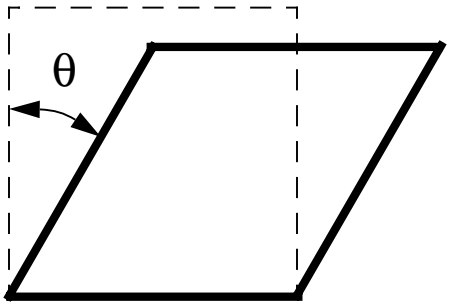


FIGURE 18. Phase velocity error for $\theta = 0°$ quadrilateral grid. The curves correspond to $k = 2\pi/5$, $k = 2\pi/10$, $k = 2\pi/15$, and $k = 2\pi/20$ respectively. The larger error corresponds to larger k.

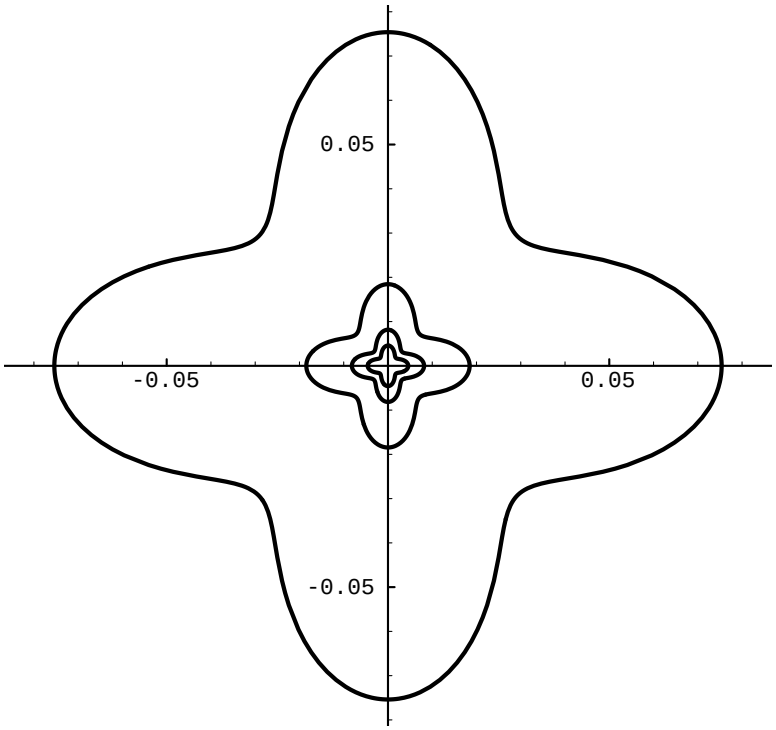


FIGURE 19. Phase velocity error for $\theta = 15°$ quadrilateral grid. The curves correspond to $k = 2\pi/5$, $k = 2\pi/10$, $k = 2\pi/15$, and $k = 2\pi/20$ respectively. The larger error corresponds to larger k.

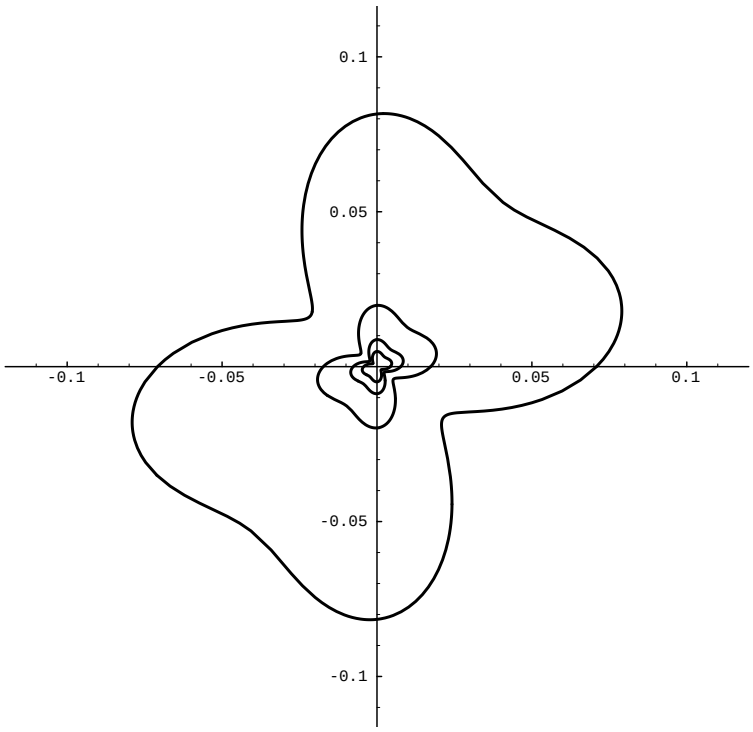


FIGURE 20. Phase velocity error for $\theta = 30°$ quadrilateral grid. The curves correspond to $k = 2\pi/5$, $k = 2\pi/10$, $k = 2\pi/15$, and $k = 2\pi/20$ respectively. The larger error corresponds to larger k.

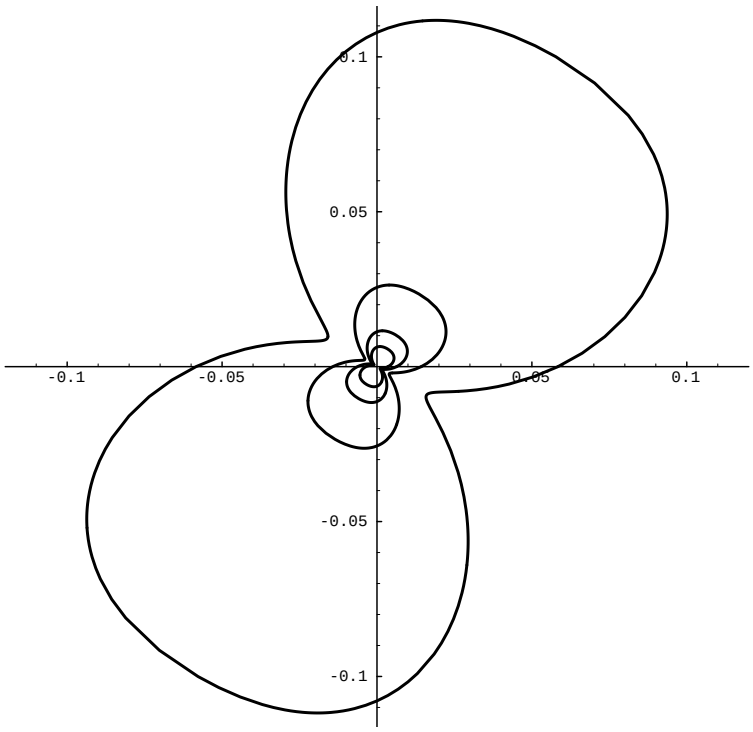
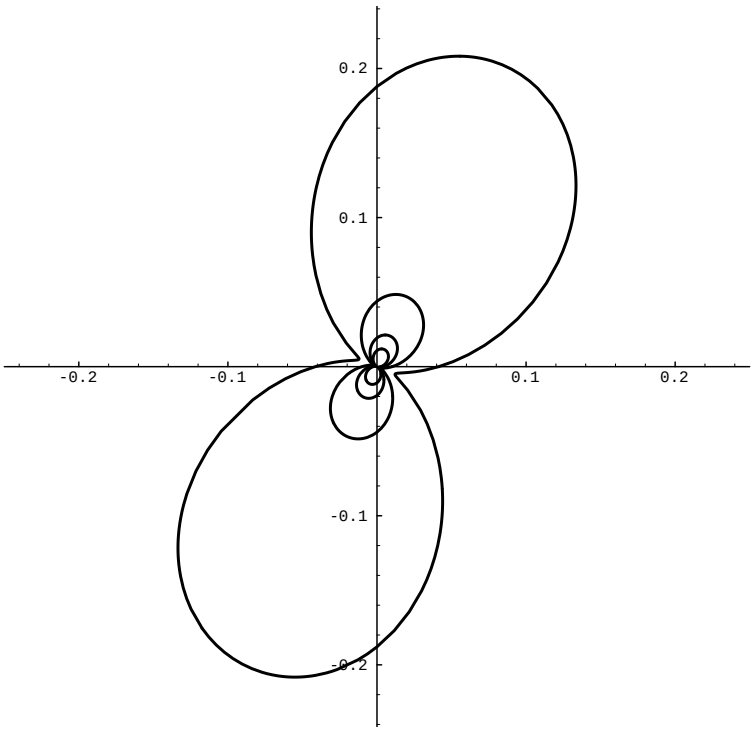


FIGURE 21. Phase velocity error for $\theta = 45°$ quadrilateral grid. The curves correspond to $k = 2\pi/5$, $k = 2\pi/10$, $k = 2\pi/15$, and $k = 2\pi/20$ respectively. The larger error corresponds to larger k.



The phase velocity error shown in Figure 18 is comparable to the phase error shown in [44] for a frequency domain implementation of a vector finite element method on a uniform Cartesian grid. The results shown in Figure 19 through Figure 21 are new, they indicate that as the grid is distorted the method becomes more anisotropic (greater variation of velocity with direction). The maximum velocity, minimum velocity, and anisotropy ratio are tabulated below as a function of k for each of the four grid distortions.

TABLE 4. Phase velocity and anisotropy ratio versus k for $\theta = 0°$ quadrilateral grid.

k	**max** v	**min** v	**ratio**
$2\pi/5$	1.07538	1.04137	1.03266
$2\pi/10$	1.01845	1.0101	1.00824
$2\pi/15$	1.00816	1.00448	1.00366
$2\pi/20$	1.00458	1.00252	1.00266

TABLE 5. Phase velocity and anisotropy ratio versus k for $\theta = 15°$ quadrilateral grid.

k	**max** v	**min** v	**ratio**
$2\pi/5$	1.0817	1.02764	1.05325
$2\pi/10$	1.01979	1.0067	1.01301
$2\pi/15$	1.00874	1.00297	1.00575
$2\pi/20$	1.0049	1.00107	1.00323

TABLE 6. Phase velocity and anisotropy ratio versus k for $\theta = 30°$ quadrilateral grid.

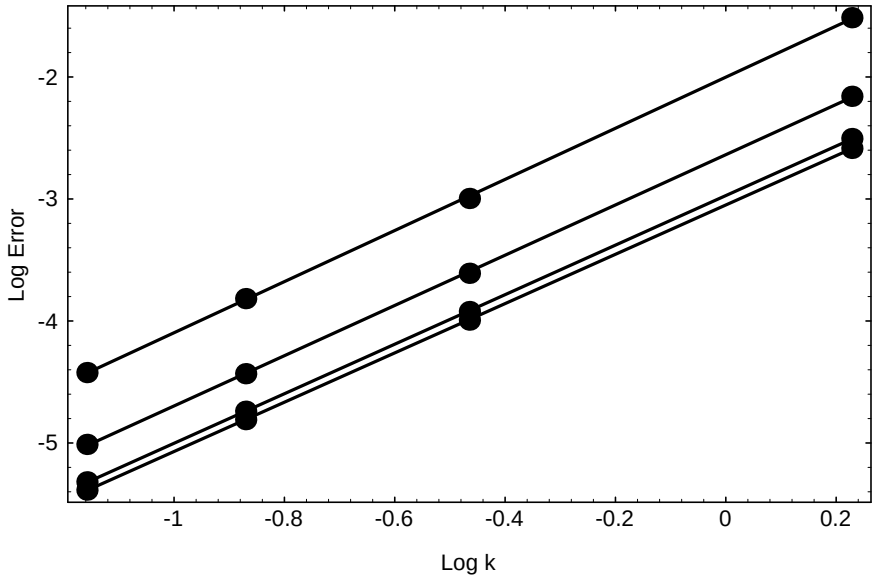
k	**max** v	**min** v	**ratio**
$2\pi/5$	1.115	1.01845	1.0953
$2\pi/10$	1.02708	1.00458	1.0224
$2\pi/15$	1.01189	1.00203	1.00984
$2\pi/20$	1.00666	1.00114	1.00551

TABLE 7. Phase velocity and anisotropy ratio versus k for $\theta = 45°$ quadrilateral grid.

k	**max** v	**min** v	**ratio**
$2\pi/5$	1.22	1.0131	1.203
$2\pi/10$	1.05	1.0032	1.047
$2\pi/15$	1.022	1.0014	1.0208
$2\pi/20$	1.012	1.0008	1.0116

It is possible to determine the rate of convergence of the numerical dispersion relation for distorted quadrilateral grids by applying a least-square fit to the above data. The logarithm of the error versus the logarithm of k is shown in Figure 22 for each of the four grids, along with a least-square linear fit. The least-square fit is applied to the maximum velocity error. For each grid the slope of the linear fit is approximately 2 (from 2.02 to 2.09), indicating second order convergence.

FIGURE 22. Least-square fit of phase velocity error indicating second order accuracy for distorted quadrilateral grids with shear $\theta = 0°$, $\theta = 15°$, $\theta = 30°$, and $\theta = 45°$, respectively. The larger error corresponds to the larger shear angle.



5.4.2 Numerical dispersion for three-dimensional shear distortion.

The same process described above for two-dimensional quadrilateral grids was applied to three-dimensional hexahedral grids. A unit cube was sheared by an amount θ in the x direction and by the same amount θ in the z direction. The value $\theta = 0°$ corresponds to no distortion, i.e. a Cartesian grid. A shear of $\theta = 45°$ is illustrated in Figure 23 below. Let

$$\grave{k} = k\,(\hat{x}\cos(\phi)\sin(\Phi) + \hat{y}\sin(\phi)\sin(\Phi) + \hat{z}\cos(\Phi)) \tag{191}$$

be the wave vector as a function of the spherical angles ϕ and Φ. Given a value of k, it is possible to compute the numerical phase velocity as a function of the spherical angles ϕ and Φ. In the computational experiments below $c = 1$, $\Delta h = 1$, and $\Delta t = 1/3$. Figure 24 through Figure 27 show surfaces of the phase velocity error for shear angles of $\theta = 0°$, $\theta = 15°$, $\theta = 30°$, and $\theta = 45°$, respectively. Each figure shows the velocity

error for $k = 2\pi/5$, where the velocity error is defined as $v - c$. The shape of the velocity error surface remains the same as k is decreased, thus it is not necessary to display different surfaces. Note that the scale is different for each plot. The Mathematica script that was used to generate these velocity error surfaces is provided in Section 11.6.

FIGURE 23. Illustration of a cube distorted in the x and z directions by an amount $\theta = 45°$.

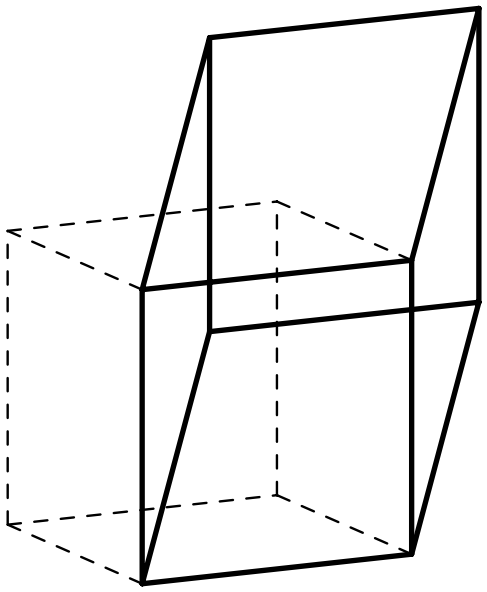


FIGURE 24. Phase velocity error for $\theta = 0°$ hexahedral grid. The surface corresponds to $k = 2\pi/5$. The length of the axes are 0.15.

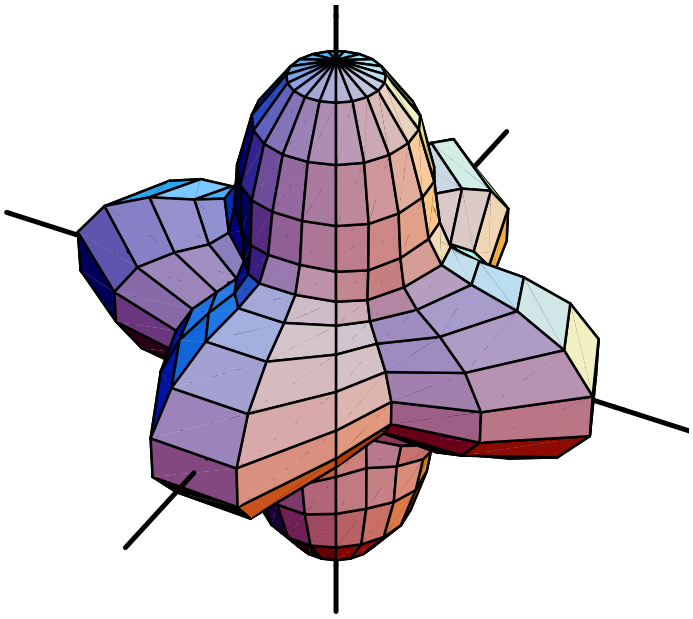


FIGURE 25. Phase velocity error for $\theta = 15°$ hexahedral grid. The curves correspond to $k = 2\pi/5$. The length of the axes are 0.25.

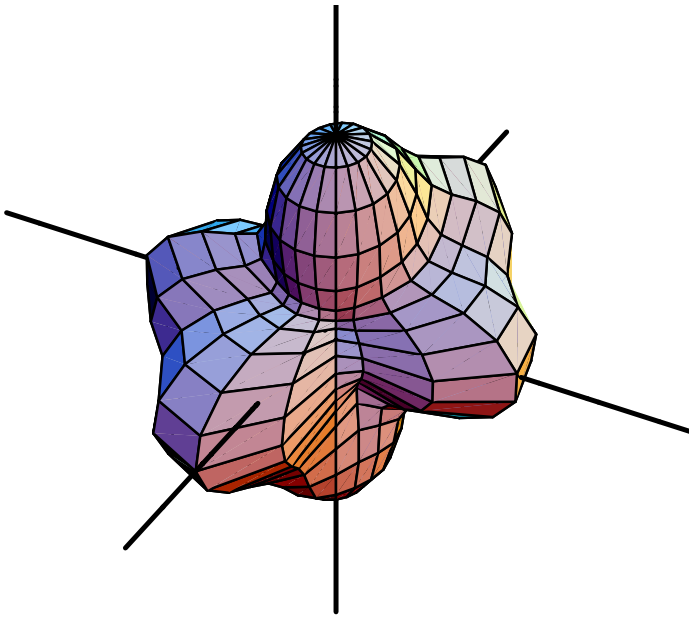


FIGURE 26. Phase velocity error for $\theta = 30°$ hexahedral grid. The surface corresponds to $k = 2\pi/5$. The length of the axes are 0.35.

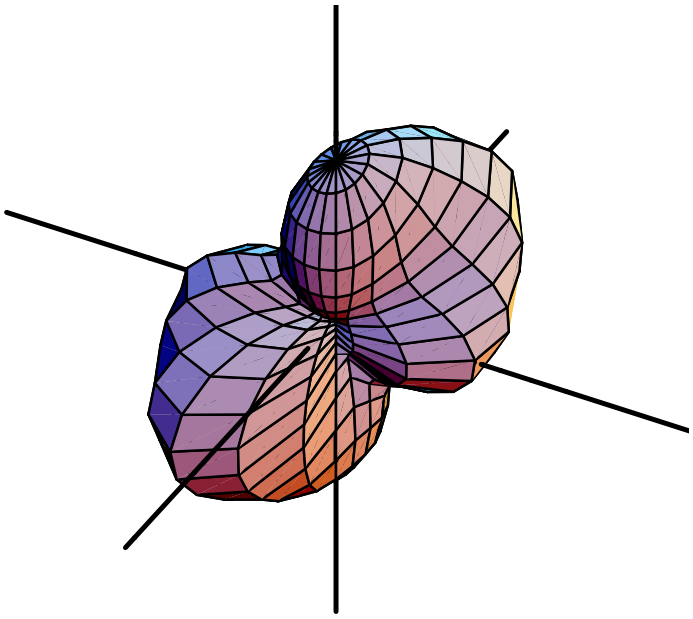
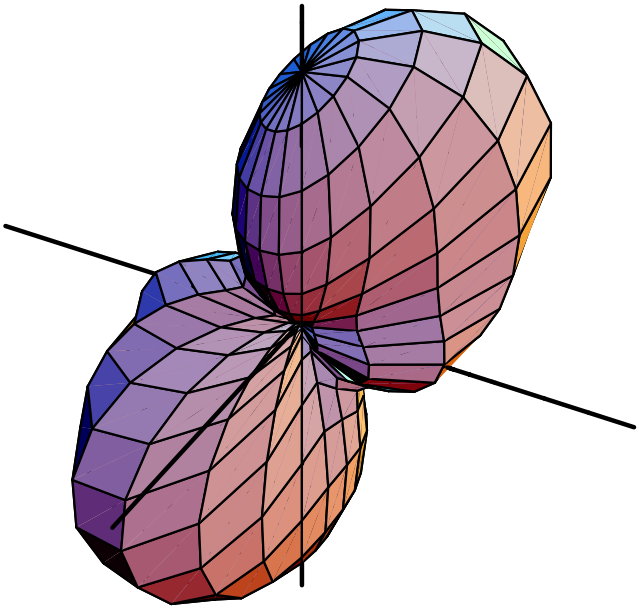


FIGURE 27. Phase velocity error for $\theta = 45°$ hexahedral grid. The surface corresponds to $k = 2\pi/5$. The length of the axes are 0.35.



The maximum velocity, minimum velocity, and anisotropy ratio are tabulated below as a function of k for each of the four grid distortions. The results are similar to the quadrilateral results above, as the grid becomes more distorted the numerical dispersion relation becomes more anisotropic.

TABLE 8. Phase velocity and anisotropy ratio versus k for $\theta = 0°$ hexahedral grid.

k	max v	min v	ratio
$2\pi/5$	1.07538	1.03002	1.04404
$2\pi/10$	1.01845	1.00736	1.01101
$2\pi/15$	1.00816	1.00326	1.00488
$2\pi/20$	1.00458	1.00183	1.00274

TABLE 9. Phase velocity and anisotropy ratio versus k for $\theta = 15°$ hexahedral grid.

k	max v	min v	ratio
$2\pi/5$	1.08797	1.01709	1.06969
$2\pi/10$	1.02113	1.00423	1.01682
$2\pi/15$	1.00931	1.00188	1.00742
$2\pi/20$	1.00522	1.00106	1.00416

TABLE 10. Phase velocity and anisotropy ratio versus k for $\theta = 30°$ hexahedral grid.

k	max v	min v	ratio
$2\pi/5$	1.14536	1.00913	1.135
$2\pi/10$	1.03401	1.00227	1.0316
$2\pi/15$	1.01493	1.00101	1.0139
$2\pi/20$	1.00836	1.0057	1.00779

TABLE 11. Phase velocity and anisotropy ratio versus k for $\theta = 45°$ hexahedral grid.

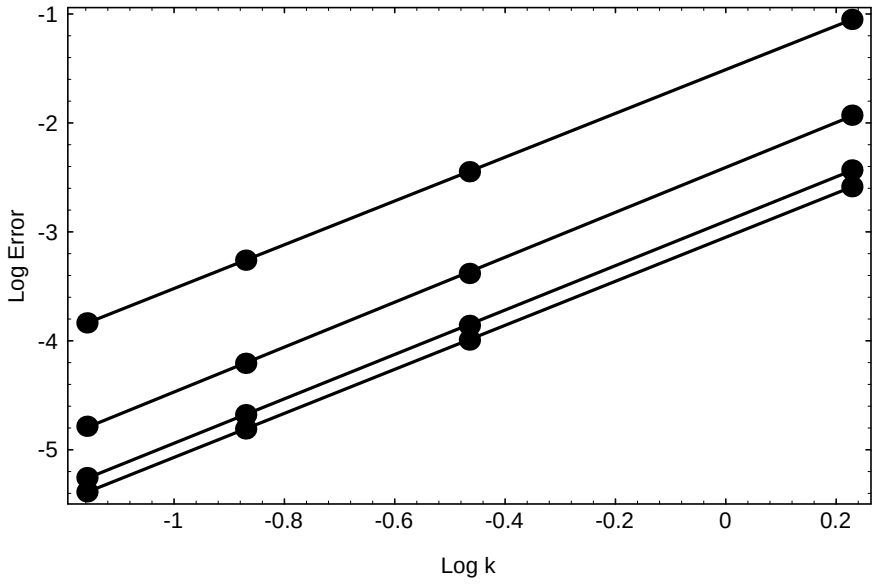
k	max v	min v	ratio
$2\pi/5$	1.35058	1.00333	1.34609
$2\pi/10$	1.08656	1.00083	1.08566

TABLE 11. Phase velocity and anisotropy ratio versus k for $\theta = 45°$ hexahedral grid.

k	**max** v	**min** v	**ratio**
$2\pi/15$	1.03845	1.00037	1.03807
$2\pi/20$	1.02163	1.00021	1.02142

It is possible to determine the rate of convergence of the numerical dispersion relation for distorted hexahedral grids by applying a least-square fit to the above data. The logarithm of the error versus the logarithm of k is shown in Figure 28 for each of the four grids, along with a least-square linear fit. The least-square fit is applied to the maximum velocity error. For each grid the slope of the linear fit is approximately 2 (from 2.02 to 2.09), indicating second order convergence.

FIGURE 28. Least-square fit of phase velocity error indicating second order accuracy for distorted hexahedral grids with shear $\theta = 0°$, $\theta = 15°$, $\theta = 30°$, and $\theta = 45°$, respectively. The larger error corresponds to the larger shear angle.



6.0 Linear System Solution Methods

The DTVFEM as defined by either (143) for coupled first order equations or (145) for a single second order equations requires that a large, sparse, unstructured linear system be solved at every time step. The system will be written generically as $Cx = y$. Since C is a Gram matrix it is symmetric positive definite. In Section 6.1, this system will be solved for the special case of an orthogonal Cartesian grid. For this special case it will be shown that if mass lumping is employed the DTVFEM reduces to the classic FDTD method. In addition, it will be shown that system can be solved exactly, in $O(n)$ operations, using a direct method. This results in a method that is both very efficient and very accurate. For general unstructured grids, neither mass lumping or direct methods are employed. Section 6.2 describes several iterative methods that are used to solve the system in this case. All of the different methods discussed in this section are implemented in VFEM3D. This was done for two reasons: 1) it allows one to easily experiment with different methods, and 2) it gives the user the freedom to choose between speed and accuracy.

6.1 Cartesian grids

In this section (146) will be analyzed in detail for the special case of an orthogonal Cartesian grid. The grid is assumed to have a uniform spacing $h = 1$. The electric degrees of freedom, e, are associated with the edges of the grid, the magnetic degrees of freedom, b, are associated with the faces of the grid. The degrees of freedom are illustrated in Figure 29. Note the similarity to the classic FDTD method which uses two grids, with the

electric field known on the edges of the primary grid and the magnetic field known on the edges of the dual grid. The DTVFEM employs only one grid, but two function spaces, the $\vec{W}^h$ or edge space and the $\vec{F}^h$ or face space. The matrices defined in (135) are easily evaluated for the special case of a Cartesian grid, and the update equation (146) simplifies to

$$\begin{aligned} b_1^{\,n+1/2} &= b_1^{\,n+1/2} - \Delta t\left(e_4^n + e_{10}^n - e_5^n - e_{12}^n \right), \\ b_2^{\,n+1/2} &= b_2^{\,n+1/2} - \Delta t\left(e_5^n + e_{11}^n - e_6^n - e_{13}^n \right), \end{aligned} \tag{192}$$

for the magnetic degrees of freedom, and

$$\begin{aligned} \frac{4}{9}e_5^{n+1} + \frac{1}{9}\left(e_4^{n+1} + e_6^{n+1} + e_8^{n+1} + e_2^{n+1} \right) + \frac{1}{36}\left(e_1^{n+1} + e_3^{n+1} + e_7^{n+1} + e_9^{n+1} \right) = \\ \frac{4}{9}e_5^{n} + \frac{1}{9}\left(e_4^{n} + e_6^{n} + e_8^{n} + e_2^{n} \right) + \frac{1}{36}\left(e_1^{n} + e_3^{n} + e_7^{n} + e_9^{n} \right) + \\ \Delta t\left(b_2^{\,n+1/2} - b_1^{\,n+1/2} + b_4^{\,n+1/2} - b_3^{\,n+1/2} \right), \end{aligned} \tag{193}$$

for the electric degrees of freedom, where the numbering scheme is illustrated in Figure 29 and Figure 30.

6.1.1 Capacitance lumping

To obtain the value e_5^{n+1} in (193) it is necessary to solve a linear system. The mass lumping approximation is to approximate the matrix C by a diagonal matrix $\tilde{C}$. There are a variety of methods of approximation that could be employed, the method used in this dissertation is construct $\tilde{C}$ via

$$\tilde{C}_{ii} = \sum_j \alpha_j C_{ij} \qquad i = 1, \ldots, N_e$$
$$\tilde{C}_{ij} = 0 \qquad i \neq j \tag{194}$$

where the coefficients α_j are such that

$$\sum_j \alpha_j \vec{W}_j \bullet \vec{W}_i = \int_\Omega \vec{E} \bullet \vec{W}_i d\Omega, \tag{195}$$

for a constant electric field $\vec{E}$ in the direction of edge i. In other words the integral in the weak form is approximated by a midpoint rule. For a uniform Cartesian grid $\alpha_j = 1$, each diagonal term of the matrix $\tilde{C}$ is equal to the row-sum of C. This results in the new update equation

$$e_5^{n+1} = e_5^n + \Delta t \left(b_2^{\,n+1/2} - b_1^{\,n+1/2} + b_4^{\,n+1/2} - b_3^{\,n+1/2} \right), \tag{196}$$

which does not require a linear system to be solved. The computational cost is $4N_e + 4N_f$ floating point operations per time step. Therefore, the method is $O(n)$, where n is the number of degrees of freedom. If the grid is parameterized by a grid spacing h, then the method requires $O\left(h^4\right)$ floating point operations to simulate T seconds of physical time. This includes a factor of h for each dimension, and another factor of h for the reduction of the time step required for stability.

FIGURE 29. Electric degrees of freedom on Cartesian grid.

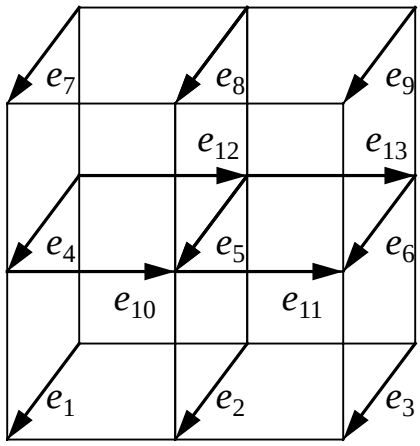
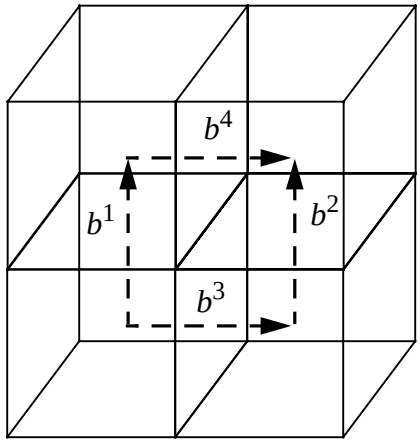


FIGURE 30. Magnetic degrees of freedom on Cartesian grid.



The term mass lumping comes from computational continuum mechanics, where the mass matrix is approximated by a diagonal matrix such that the total mass is the same. This can be visualized by considering the compression of spring, see Figure 31. The continuous spring is approximated by a series of point masses connected by zero mass springs. The first case results in an equation for the displacement of the form

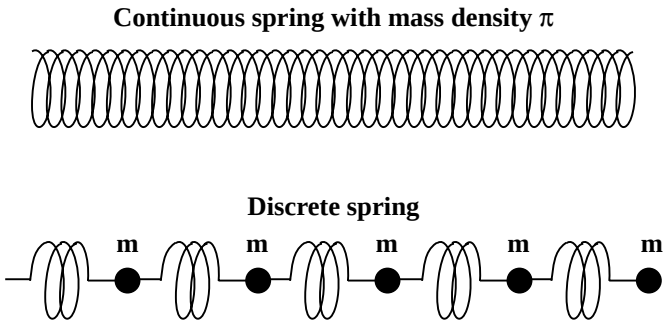
$$M\frac{\partial^2 x}{\partial t^2} = Kx, \tag{197}$$

where M is the mass matrix and K is the stiffness matrix. The second case results in the equation

$$\frac{\partial^2 x}{\partial t^2} = \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & m_n \end{bmatrix}^{-1} Kx. \tag{198}$$

Obviously the computer implementation of (198) will be more efficient than (197), since the former requires the solution of a linear system, while the latter does not. Of course, in one dimension, the matrix M is tri-diagonal and can be solved relatively efficiently. In three space dimensions, M is, in general, a large, sparse, unstructured matrix. In the mathematics literature, mass lumping is referred to as a variational crime, since it makes most of the classic convergence proofs invalid [54]. Nevertheless it is used quite frequently in the engineering community, and it some special cases it can be shown to give very reasonable results [65].

FIGURE 31. Illustration of mass lumping for a a spring.



In the electromagnetic, case the approximation of (193) by (196) should be referred to as capacitance lumping since the matrix C is the capacitance matrix. Note that using the capacitance lumping approximation gives the classic FDTD method, (192) and (196). This method has been extensively analyzed and it is known to be stable, energy conserving, charge conserving, and second order accurate approximation to Maxwell's equations. Thus, the DTVFEM should not be considered an alternative to the FDTD method, but rather as generalization of the FDTD method.

6.1.2 Cholesky decomposition

Consider the system $Cx = y$ where C is the capacitance matrix of a uniform Cartesian grid. Since C is symmetric positive definite is has a Cholesky decomposition $C = L^T L$, where L is a lower triangular matrix. For the special case of a Cartesian grid, the Cholesky decomposition has the exact same sparsity structure as C itself, i.e. there is no zero-fill in the course of the Cholesky decomposition. Therefore, the calculation of x requires a back-substitution and a forward-substitution,

$$\begin{aligned} Lz &= y, \\ L^T x &= z\,. \end{aligned} \tag{199}$$

Since the matrix C has at most nine non-zero entries per row, the matrix L has on average 9/2 non-zero entries per row, and the solution cost for (199) is approximately $9n$ floating point operations for a vector x of length n. Thus the total cost for the DTVFEM on a Cartesian grid is $13N_e + 4N_f$ floating point operations per time step. If the grid is parameter-

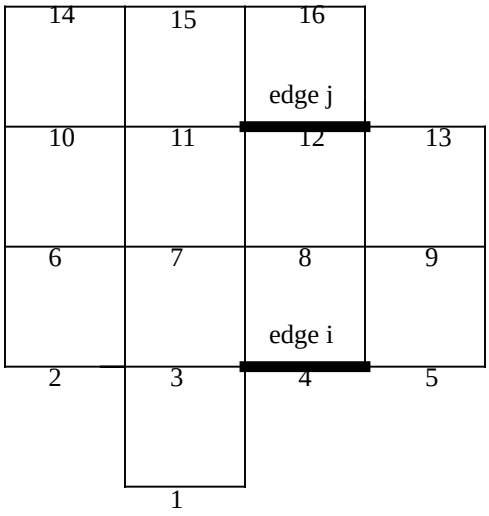
ized by a grid spacing h, then the method requires $O\left(h^4 \right)$ floating point operations to simulate T seconds of physical time, the same as in the FDTD method. While this direct solution method is less efficient than capacitance lumping, it may result in a more accurate answer. This is examined in Section 8.0 below.

That there is no zero fill during the course of the Cholesky decomposition of C can be seen by carefully examining the inner most loop of the decomposition (see algorithm 4.2.2 in [66])

$$C_{ij} = C_{ij} - C_{ik}C_{jk}, \tag{200}$$

where $k < j \leq i$. Assume that $C_{ij} = 0$, which means that there is no interaction between edges i and j. There will be zero fill only if there is another edge k that interacts with both edges i and j. Numbering the edges sequentially precludes this possibility. Thus, the Cholesky decomposition has the exact same sparsity structure of C itself. This is illustrated on a two dimensional grid in Figure 32.

FIGURE 32. Grid numbering scheme for Cartesian grid.



6.2 General unstructured grids

For a general unstructured hexahedral or tetrahedral grid neither capacitance lumping nor direct solution are effective for solving the linear system $Cx = y$. Two general methods were investigated for solving the system, stationary iteration and preconditioned conjugate gradient. In the VFEM3D program, a variety of solution methods are implemented, the user chooses which method he/she wants to use. For some grids, the simple methods are adequate, for highly distorted grids or problems with wildly varying material properties, the more sophisticated methods are required.

6.2.1 Stationary iteration

Let the matrix C be decomposed as

$$C = M - N, \tag{201}$$

which is referred to as a splitting. The matrix M is called the preconditioner. The stationary iteration is then

$$Mx^{k+1} = Nx^{k} + y. \tag{202}$$

The method is called stationary because the matrices M and N are independent of k. A starting value x^{0} must be supplied to start the iteration. Let the error be defined as

$$\delta^{k} = x^{k} - x. \tag{203}$$

It is clear that the error satisfies the equation

$$M\delta^{k+1} = N\delta^{k}. \tag{204}$$

If the error approaches zero as the number of iterations increases, the stationary iteration is said to converge. The stationary iteration converges if and only if

$$\rho\left(M^{-1}N \right) < 1, \tag{205}$$

where $\rho(A)$ denotes the spectral radius of matrix A. The smaller the spectral radius, the faster the stationary iterations will converge.

A similar approach to the above stationary iteration is Neumann polynomial approximation. (201) can be written as

$$C = M - N = M\left(I - M^{-1}N \right), \tag{206}$$

and then the inverse of C is

$$C^{-1} = \left(I - M^{-1}N \right)^{-1} M^{-1}. \tag{207}$$

The Neumann approximation is then

$$C^{-1} = \left(I + M^{-1}N + \left(M^{-1}N \right)^2 + \left(M^{-1}N \right)^3 + \ldots \right) M^{-1}. \tag{208}$$

Note that the series converges if and only if (205) is satisfied. It is common practice to use a fixed number of terms in the polynomial approximation. This is equivalent to taking a fixed number of iterations of (202) with a starting value of $x^0 = 0$. The performance of the stationary iteration and the Neumann polynomial approximation depend upon the choice of the splitting. Some common splittings are described below. Let $C = L + D + L^T$, where L is a lower triangular matrix and D is a diagonal matrix.

Jacobi. The Jacobi method uses $M = D$. If M is a diagonal matrix different than the diagonal elements of C then it is a Jacobi-like method. For example M_{ii} could equal a linear combination of terms in row i of C, i.e. a mass lumping preconditioner.

Gauss Seidel. The Gauss Seidel method uses $M = D + L$. The symmetric Gauss Seidel method, which results in a symmetric Neumann approximation, is given by

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

Successive Overrelaxation. This is a generalization of Gauss Seidel. The matrix M is given by $M = \frac{1}{\omega}(D + \omega L)$ where ω is called the relaxation parameter. The optimal value of ω depends upon the matrix C. The symmetric version is given by

$$M = \frac{1}{(2-\omega)}\left(\frac{D}{\omega} + L\right)\left(\frac{D}{\omega}\right)^{-1}\left(\frac{D}{\omega} + L\right)^{T}.$$

Incomplete Cholesky. The matrix C has a Cholesky decomposition $C = LL^{T}$. The matrix M is given by $M = \tilde{L}\tilde{L}^{T}$ where $\tilde{L}\tilde{L}^{T}$ is the incomplete Cholesky decomposition. There are different incomplete Cholesky decompositions, in this dissertation only ILU(0) factorization, in which $\tilde{L}$ has the same sparsity pattern as C, is used.

An important property of the above stationary methods is that the matrix C is not really inverted, rather the system $Cx = y$ is solved approximately. But the approximation is such that

$$x = \tilde{C}^{-1}y, \tag{209}$$

where $\tilde{C}^{-1}$ is a symmetric matrix. independent of y. Thus the stability analysis in Section 5.1 is still valid, with the term C^{-1} is simply replaced by $\tilde{C}^{-1}$. As long as the preconditioner M is symmetric, the DTVFEM is provably stable. It should be noted using a fixed number of stationary iterations is considered a variational crime [53] in the mathematics community, since the computed solution is no longer a projection of the exact solution onto the finite element space. However, this approximation is commonly used in

computational continuum mechanics without serious problems. In Section 8.0, the above stationary methods will be applied to a variety of problems.

6.2.2 Conjugate gradient

The conjugate gradient method [66] is a non-stationary iterative method for solving $Cx = y$. The method is non-stationary in the sense that miscellaneous constants change from one iteration to the next. This also means that at iteration k, the approximate solution x^k cannot be expressed as shown in (209). Thus if the DTVFEM is to be stable, the conjugate gradient must not be applied for a fixed number iterations, but rather applied until the solution converges to within some numerical tolerance. The conjugate gradient method is investigated here because it can be much more efficient for certain problems than the above stationary iterations. The conjugate gradient is well known and it will not be derived here.

It can be shown that the number of iterations required in order to achieve a given relative error is proportional to $\sqrt{\chi(C)}$, where $\chi(C)$ is the condition number of the matrix C. The number of iterations can be reduced by employing a preconditioner to the linear system. Rather than solving $Cx = y$ via conjugate gradient, the system $M^{-1}Cx = M^{-1}y$ is solved, where M is the preconditioner. If

$$\sqrt{\chi\left(M^{-1}C\right)} < \sqrt{\chi(C)}\,, \tag{210}$$

then the solution of the pre-conditioned system will require fewer iterations. A good pre-conditioner is one for which $M \approx C$. However, it is necessary to solve a system $Mz^k = r^k$ within the conjugate gradient algorithm, thus it is also important that the preconditioner be easy to solve. All of the symmetric preconditioners described in Section 6.2.1 can be used in a pre-conditioned conjugate gradient algorithm. The results for various preconditioners are reported in Section 8.0.

7.0 Parallel Implementation

Parallel computers represent the state-of-the-art in computer technology. These computers provide the power (speed, precision, memory, data transfer) to solve previously intractable problems. These computers are becoming common at universities, government laboratories, and in industry. Some algorithms are essentially parallel, and their implementation on a parallel computer is trivial. Other algorithms are essentially serial, they cannot be implemented on a parallel machine in an efficient manner. In this section, the parallel implementation of the DTVFEM used in VFEM3D is described. As described in Section 6.0, the DTVFEM requires that a large, sparse, unstructured linear system be solved at every time step. This is the crux of the parallel implementation. Several methods were investigated for solving this problem. There are variety of parallel architectures. VFEM3D was designed to run only on a certain class of machines. It is shown that on these machines the parallel performance is satisfactory, i.e. the implementation makes good use of the computer for reasonable combinations of problem size and computer size. Before discussing the parallel implementation, it is necessary to define some terminology used by the parallel computing community. Most of the following is from [70].

7.1 Review of parallel computing

The traditional serial computer consists of a single processor and memory. This computer is exemplified by the typical personal computer or engineering workstation. These computers execute a single sequence of instructions and operate on a single sequence of data. They are referred to as Single Instruction Single Data (SISD) computers.

Parallel computers can be dichotomized according to their control mechanism. A Single Instruction Multiple Data (SIMD) computer has one control unit that dispatches instructions to several processing units. The same instruction is executed synchronously by each processor. Each processor operates on different data. The classic SIMD computer is the Connection Machine 2. Parallel computers, in which each processor is capable of executing a different program independent of the other processors, are called Multiple Instruction Multiple Data (MIMD) computers. Examples of MIMD computers include the Cray-YMP, Cray T3D, and the Meiko CS2. A network of workstations can also be considered a MIMD computer.

Parallel computers can also be characterized as to how the memory is organized and addressed. A shared memory computer provides hardware support for read and write access by all processor to a shared address space. A MIMD computer with shared memory is called a multiprocessor, and is exemplified by the Cray-YMP and some high performance Silicon Graphics and Sun workstations. In a distributed memory computer, each processor has its own private memory, which is directly accessible only by that processor. Processors interact with each other only by passing messages, thus this architecture is sometimes referred to as a message passing architecture. A MIMD distributed memory computer is called a multicomputer. Examples include the Cray T3D and the Meiko CS2.

The DTVFEM could be efficiently implemented on any of the above described computers. However, the VFEM3D program was designed to run either on a typical SISD workstation or on a message passing multicomputer. There are many different ways to physically connect the processors in a multicomputer, such as ring, mesh, tree, hypercube, etc. The inter-

connection network determines how data is communicated from one processor to another, and it is possible to optimize a message passing program for a given interconnection network. However, this was not done in VFEM3D. Rather, VFEM3D was developed for a model multicomputer in which any given processor can send data to and receive data from any other processor in equal time. Some multicomputers approach this idealization for large messages. This approach was taken because 1) the software is easy to develop and maintain, and 2) the software will have good performance on all architectures, rather than excellent performance on one machine and poor performance on another.

Given an algorithm such as the DTVFEM and a multicomputer, it is still not obvious how to develop the software. It is necessary to establish a paradigm for the implementation. The Single Program Multiple Data paradigm is used in VFEM3D. In this paradigm there is only one program which is executed an all the processors. No one processor is more important than any other, they all read input files, write output files, and communicate with each other. There is no master or host processor to manage the computation. Of course, the processors operate on different data, and there can be data dependent conditional statements in the program, so the processors may end up executing different statements.

The serial run time of a program is the time elapsed between the beginning and end of its execution, measured in processor seconds, not wall clock seconds. The parallel run time is the time elapsed from the moment the parallel computation starts to the moment the last processor finishes. The ratio of the time taken to solve a problem on a single processor to the time taken to solve the same problem in parallel with p processors is called speedup.

An ideal parallel computer with p processors can achieve a speedup of p. A practical parallel system (algorithm and computer) cannot achieve a speedup of p due to overhead. The efficiency of a parallel system is a measure of the fraction of time for which the processors are usefully employed. It is defined as the ratio of speedup to the number of processors. The ability to maintain efficiency at a fixed value by simultaneously increasing the size of the problem and the number of processors is called scalability.

All causes of non-optimal efficiency are referred to as overhead. One source of overhead is the serial component of the algorithm. Amdahl's law [71] states that if a problem of size W has a serial component of size W_s then the maximum possible speedup is W/W_s. Another source of overhead is interprocessor communication time. For example, consider an iterative algorithm with a stopping criteria dependent upon the norm of a vector, where the vector is distributed among all the processors. Each processor must agree upon the value of the norm, which requires interprocessor communication. Functions such as minimum, maximum, root mean square, etc. that reduce distributed data to a single number are referred to as global reduction. The cost of global reduction, terms of interprocessor communication time, is hardware dependent. A load imbalance occurs when the work is not assigned equally among the processors, and one or more processors have to wait for the rest to catch up. Load balancing is addressed in more detail in section Section 7.2 below. The final cause of overhead is due to extra computation. The fastest known algorithm for solving a problem on a serial computer may be difficult or impossible to parallelize, thus a slower but more parallelizable algorithm must be used.

7.2 Domain decomposition

The VFEM3D program uses the SPMD paradigm, executing the same program on each processor in parallel. However, each processor has only part of the data. The data is distributed among all the processors using a domain decomposition approach. There are different ways of decomposing the data. The data could be decomposed into disjoint sets, or into overlapping sets. The data could be decomposed in a purely mathematical way, or in a way that relates to the physics of the problem. The domain decomposition approach used in VFEM3D is best explained via example.

Consider a two dimensional triangular grid with a dual graph as illustrated in Figure 33. The grid is denoted with solid lines, the graph with dashed lines. The dual graph connects the midpoints of the triangular volumes, which are denoted by the circles. In this example it is assumed that there are two processors. The data is decomposed by cutting the dual graph in two. This is creates two disjoint groups of volumes, all data associated with the dark group is associated with processor 0 and all data with the light group is associated with processor 1.

FIGURE 33. A triangular grid with a dual graph.

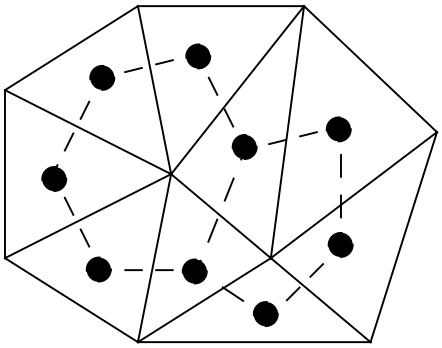
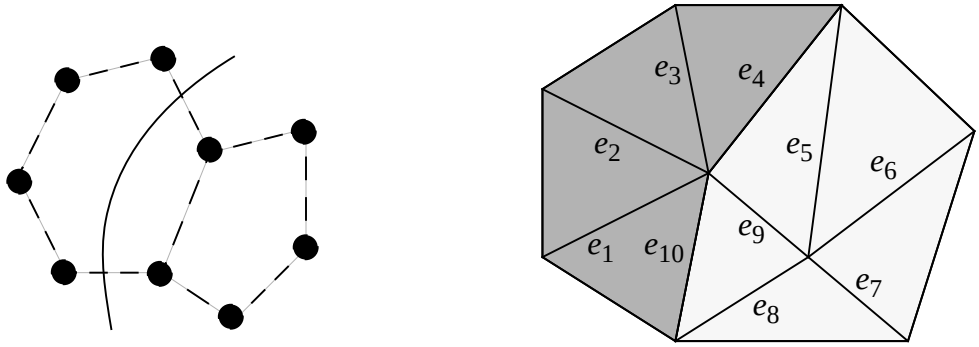


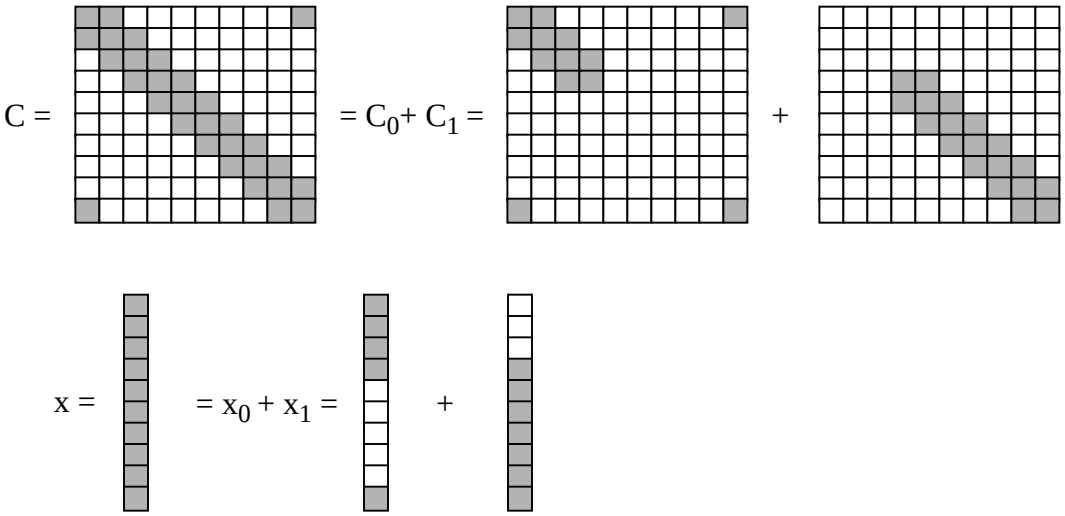
FIGURE 34. Partitioning the dual graph.



The above decomposition is a physical decomposition of the problem, the domain Ω is cut into two disjoint domains Ω_0 and Ω_1 which do not overlap, but do share a boundary. This is how a person would cut a piece of cake in two. Note that in the DTVFEM the degrees of freedom are associated with the edges and faces of the grid, thus the degrees of freedom on the boundary joining Ω_0 and Ω_1 reside on both processors. Thus from a data point of view the decomposition is not disjoint, but slightly overlapping.

Consider the 10×10 capacitance matrix C and the vector of degrees of freedom x which is of length 10 and numbered as shown in Figure 34. The matrix and the vector are decomposed as shown in Figure 35. The degrees of freedom e_4 and e_{10} are shared by both processors. Note that the workload is slightly unbalanced, processor 0 is responsible for five degrees of freedom, while processor 1 is responsible for seven degrees of freedom. However, the amount of communication is small; the processors must agree upon the value of e_4 and e_{10} which requires that each processor send and receive a message of length two. Other decompositions would have a different load balance and different communication costs.

FIGURE 35. Decomposition of the data vectors and matrices.

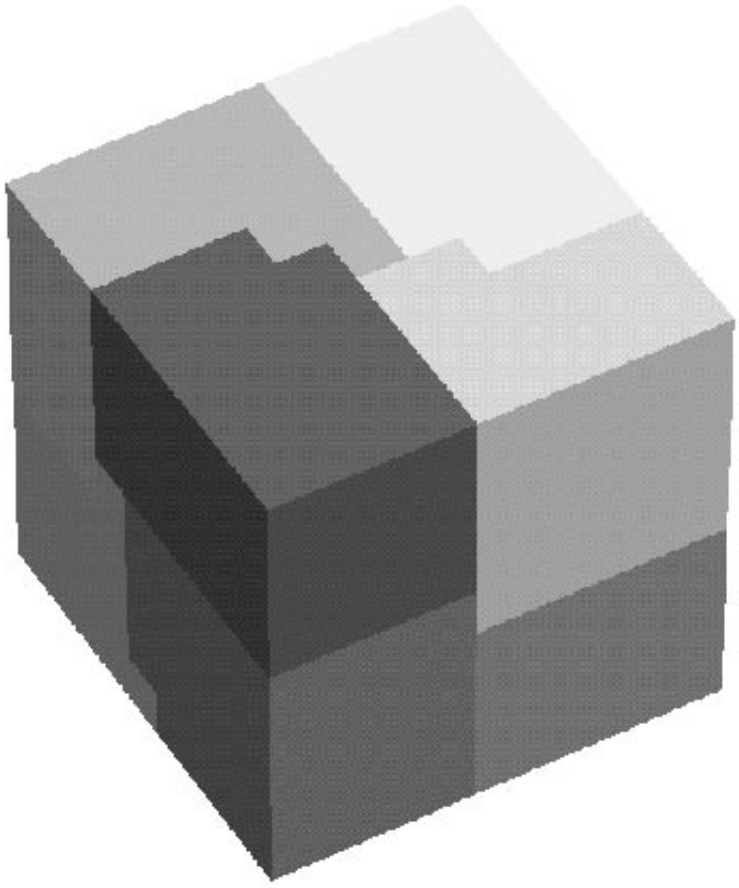


The above example illustrates the complexity of domain decomposition for unstructured grid problems. The goal is to distribute the workload evenly among all the processors while also minimizing the amount of communication required. VFEM3D allows for completely arbitrary partitioning, the user supplies a color vector of length N_v. This vector associates a color with each tetrahedral or hexahedral volume in the grid, where the color corresponds to a processor number. Determining the optimal color vector for an unstructured grid is a difficult combinatorial problem and is not part of this research effort. There are several different methods for generating a good color vector, notably the Recursive Spectral Bisection (RBS) algorithm [72] and multilevel k-way schemes [73][74].

As an example of the application of the RSB algorithm consider a $9 \times 9 \times 9$ grid, and assume there are eight processors. If the grid was $10 \times 10 \times 10$, an obvious decompositions is to divide the cube into eight $5 \times 5 \times 5$ blocks. However for the $9 \times 9 \times 9$ case it is not obvious what the optimal decomposition is. The RSB algorithm generated a decomposition with a maximum load of 92, and a minimum of 91. The maximum communication

vector length was 80, the minimum was 63. The decomposition is shown in Figure 36. Note that this is for illustration only, a parallel computer would not be used for such a small problem.

FIGURE 36. Eight processor partitioning of a 9 by 9 by 9 cube.



7.3 Parallel linear system solution methods

Some of the linear system solution methods described in Section 6.0 are easily parallelizable, others are not. Methods that only require matrix-vector multiplication are easy to parallelize. The Jacobi fixed point iteration and the Jacobi preconditioned conjugate gradient were implemented in parallel. These two approaches require nearest neighbor communication to perform the matrix-vector multiplication and global reduction communication to compute norms. These two types of communication are common and are detailed in [70]. Methods that require forward and/or backward substitutions are quite difficult to parallelize. While there has been some success in developing parallel backward substitution for particular matrices, it is an intractable problem for matrices that arise from unstructured grids. Thus parallel versions of Gauss Seidel, successive over-relaxation, and incomplete Cholesky were not developed.

One approach to parallel preconditioning is to use block preconditioners. The degrees of freedom can be divided into two groups, those that are internal to the sub-domains Ω_i and those that lie on the domain boundaries Γ_i. Let the internal variables be denoted by I and the domain boundary variables be denoted by B. The variables are then re-numbered as

$$x = \begin{bmatrix} x_I \\ x_B \end{bmatrix}, \qquad b = \begin{bmatrix} b_I \\ b_B \end{bmatrix}, \tag{211}$$

and the linear system $Ax = b$ can be expressed as

$$\begin{bmatrix} A_I & A_{IB} \\ A_{IB}^T & A_B \end{bmatrix} \begin{bmatrix} x_I \\ x_B \end{bmatrix} = \begin{bmatrix} b_I \\ b_B \end{bmatrix}.$$

(212)

Note that since the sub-domains are uncoupled, the matrix A_I is block-diagonal. A suitable block preconditioner is then

$$M = \begin{bmatrix} A_I & 0 \\ 0 & diag(A_B) \end{bmatrix}. \tag{213}$$

This is a variant of the so called block Jacobi preconditioner. Both the fixed point iteration and the pre-conditioned conjugate gradient require that the linear system $Mz = r$ be solved once per iteration. For the block preconditioner defined by (213), this linear solve is totally parallel, no communication is required. Each processor solves the system

$$A_i x_i = b_i, \tag{214}$$

where the vectors x_i and b_i are inside the domain Ω_i. Of course, the systems of equations defined by (214) are still large, sparse, and unstructured, so a direct method is still impractical. But any of the iterative methods described in section Section 6.2 can be used to solve (214). This results in nested iterative scheme, the outer iteration is over the entire grid and requires nearest neighbor communication and global reduction communication, the inner iteration is local to each processor and requires no communication. A sophisticated inner iteration will reduce the number of outer iterations, thus reducing the amount

of communication. Of course, the total run time may not be significantly reduced, this is very much problem dependent.

To summarize, the following parallel linear system solution methods were incorporated into VFEM3D.

Outer Iteration Methods

Jacobi iteration, block Jacobi iteration, Jacobi pre-conditioned conjugate gradient, block Jacobi pre-conditioned conjugate gradient.

Inner Iteration Methods

Symmetric Gauss Seidel iteration, incomplete Cholesky iteration, Jacobi pre-conditioned conjugate gradient, symmetric Gauss Seidel pre-conditioned conjugate gradient, incomplete Cholesky pre-conditioned conjugate gradient.

8.0 Validation

In this section VFEM3D is validated by comparing computed solutions to exact solutions for several simple electromagnetics problems. There are several reasons for doing this. The obvious reason is to make sure that there are no bugs in the program, but that is really a software engineering issue and not a computational issue. The second reason is to determine the absolute accuracy of the method. It was shown, via numerical dispersion analysis in Section 5.4, that on a regular hexahedral grids the DTVFEM is at least second order accurate. A similar proof for an unstructured three-dimensional grid is intractable.Therefore, in this section, computer experiments are used to determine the accuracy of the method on unstructured grids for select problems. These results can then be used to estimate the error for problems in which the exact solution is not known.

Naturally, different grids for the same geometry will give rise to different computed solutions. However, an important property of the DTVFEM is that it gives similar results for any reasonable grid. The issue of grid imprinting is investigate in this section. Grid imprinting is the phenomena in which the computed solution adopts a structure similar to the underlying grid. For example, if a uniform grid gives rise to an accurate field solution, and a distorted grid gives rise to a field solution that exhibits that same distortion, then the method is said to exhibit grid imprinting. Grid imprinting is an undesirable, but not disastrous, phenomena exhibited by many numerical methods. It will be shown via computer experiments that the DTVFEM does not exhibit grid imprinting.

Another issue that is investigated via computer experiment is that of radiation (or absorbing) boundary conditions. Many electromagnetic problems, such as antenna design and radar cross section involve infinite domains. Since an infinite grid is not feasible the infinite domain must somehow be simulated on a finite grid. The goal is to eliminate non-physical reflections from the artificial truncation of the domain. The subject of radiation boundary conditions is still a controversial subject, with plenty of opportunities for research. The method used in VFEM3D is a variant of the so-called Perfectly Matched Layer (PML) method. The original PML method derived in [75] is applicable only for the classic Cartesian grid FDTD method, but many variants have been proposed for unstructured grids [77]-[79]. The general idea is to surround the domain by several layers of anisotropic conductive media, using both electric and magnetic conductivity. Grading the layers from low conductivity to high conductivity creates a broadband impedance match, thus eliminating (or nearly eliminating) front face reflections. As the outgoing wave propagates though the PML it is absorbed by the medium. The PML is not really perfect, a small amount of energy will be reflected from the boundary. But the reflection is an exponential function of layer thickness and can be made arbitrarily small. Since the DTVFEM allows for arbitrary tensor material properties VFEM3D is capable of using the PML technique without modification. In Section 8.3 and Section 8.4 results for specific PML's are shown.

Finally, the computer CPU time for various problems is tabulated. The CPU time depends upon the type of grid, the distortion of the grid, and upon the method used to solve the linear systems. On a parallel machine, the computer CPU time is also a function of the num-

ber of processors and the particular domain decomposition employed. Parallel speedup and efficiency results are presented for two different parallel supercomputers.

8.1 Rectangular Cavity

In this section, VFEM3D is used to compute the resonant frequencies of a rectangular cavity of dimensions $a \times b \times c$. In the frequency domain this electromagnetics problem is described by the vector Helmholtz equation

$$\begin{aligned} \nabla \times \frac{1}{\mu} \nabla \times \vec{E} + \omega^2 \varepsilon \vec{E} &= 0 \text{ in } \Omega, \\ \hat{n} \times \vec{E} &= 0 \text{ on } \Gamma. \end{aligned} \tag{215}$$

where ω is a resonant frequency of the cavity. (215) is an eigenvalue problem, where ω is the eigenvalue and $\vec{E}$ is the corresponding eigenfunction. The different eigenfunctions, which may be degenerate, are referred to as the cavity modes. The solution to (215) is straightforward and can be found in most textbooks [46]. The steady state electric field inside a rectangular cavity can be decomposed into Transverse Electric TE_z modes and Transverse Magnetic TM_z modes, where a mode is an eigenfunction of (215). The two modes are independent. The TE_z mode is given by

$$\begin{aligned} E_x &= \frac{\beta_y}{\varepsilon} A_{mnp} \cos(\beta_x x) \sin(\beta_y y) \sin(\beta_z z), \\ E_y &= -\frac{\beta_x}{\varepsilon} A_{mnp} \sin(\beta_x x) \cos(\beta_y y) \sin(\beta_z z), \\ E_z &= 0, \end{aligned} \tag{216}$$

where the resonant frequency given by

$$\omega_{mnp} = 2\pi f_{mnp} = \frac{1}{\sqrt{\mu\varepsilon}}\sqrt{\beta_x^2 + \beta_y^2 + \beta_z^2} \qquad \begin{matrix} m = 0, 1, \ldots \\ n = 0, 1, \ldots \\ p = 1, 2, \ldots \end{matrix} \qquad m = n \neq 0 . \tag{217}$$

The TM_z solution is

$$\begin{aligned} E_x &= -j\frac{\beta_x\beta_z}{\omega\mu\varepsilon}A_{mnp}\cos(\beta_x x)\sin(\beta_y y)\sin(\beta_z z), \\ E_y &= -j\frac{\beta_x\beta_z}{\omega\mu\varepsilon}A_{mnp}\sin(\beta_x x)\cos(\beta_y y)\sin(\beta_z z), \\ E_z &= A_{mnp}\sin(\beta_x x)\sin(\beta_y y)\cos(\beta_z z), \end{aligned} \tag{218}$$

where this time the resonant frequency is

$$\omega_{mnp} = 2\pi f_{mnp} = \frac{1}{\sqrt{\mu\varepsilon}}\sqrt{\beta_x^2 + \beta_y^2 + \beta_z^2} \qquad \begin{matrix} m = 1, 2, \ldots \\ n = 1, 2, \ldots . \\ p = 0, 1, \ldots \end{matrix} \tag{219}$$

In both cases, the wavenumbers are given by

$$\beta_x = m\pi/a, \ \beta_y = n\pi/b, \text{ and } \beta_z = p\pi/c . \tag{220}$$

In the examples below, $a = 29m$, $b = 23m$, and $c = 19m$. The lowest resonant frequencies are shown in Table 12.

TABLE 12. Exact value of resonant frequencies below f = 0.5 Hz.

110	**101**	**011**	**111**	**210**	**201**	**120**	**211**
0.027746	0.031461	0.034134	0.038241	0.040763	0.043377	0.046772	0.048519

In the time domain the electric field within the cavity is described by PDE II, (6)-(11). In this case $\sigma_E = \sigma_M = 0$ and $\mu = \varepsilon = 1$ within the cavity. The exact solution is of the form

$$\vec{E} = \sum_{mnp} A_{mnp} \vec{E}^{TE}_{mnp} \cos(\omega_{mnp} t + \phi_{mnp}) + \sum_{mnp} B_{mnp} \vec{E}^{TM}_{mnp} \cos(\omega_{mnp} t + \theta_{mnp}), \qquad (221)$$

where the sum is over all the modes, and A, B, ϕ, and θ depend upon the initial conditions. In this computational experiment the electric field is excited by a pulsed current source (described below) which is designed to excite all of the cavity modes. Once the source is turned off the electric field within the cavity obeys (221) for all time.

The rectangular cavity was discretized using five different grids. Grid 1 is a uniform Cartesian grid. Grid 2 is a non-uniform Cartesian grid, i.e. the grid spacing is not constant but the grid is still orthogonal. Grid 3 is a random hexahedral grid that was generated by adding random noise to the nodes of grid 3, thus this grid can be considered structured but it is non-Cartesian. Grids 1, 2, and 3 are logically 9 cells by 9 cells by 9 cells. Grid 4 is an unstructured tetrahedral grid. These four grids have the same number of nodes, the same average node spacing, and the same average edge length. Finally grid 5 is a 17 by 17 by 17 uniform Cartesian grid. The Cartesian grids were generated using TrueGrid [80], while the tetrahedral grid was generated using GEOMPACK [81].

FIGURE 37. Rectangular cavity Grid 1, a uniform Cartesian grid.

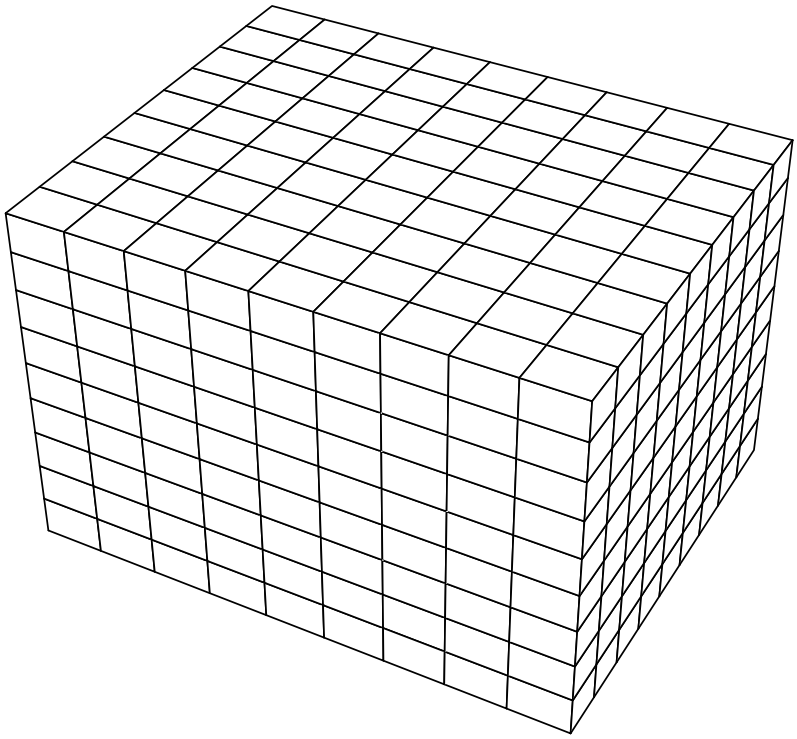


FIGURE 38. Rectangular cavity grid 2, a non-uniform Cartesian grid.

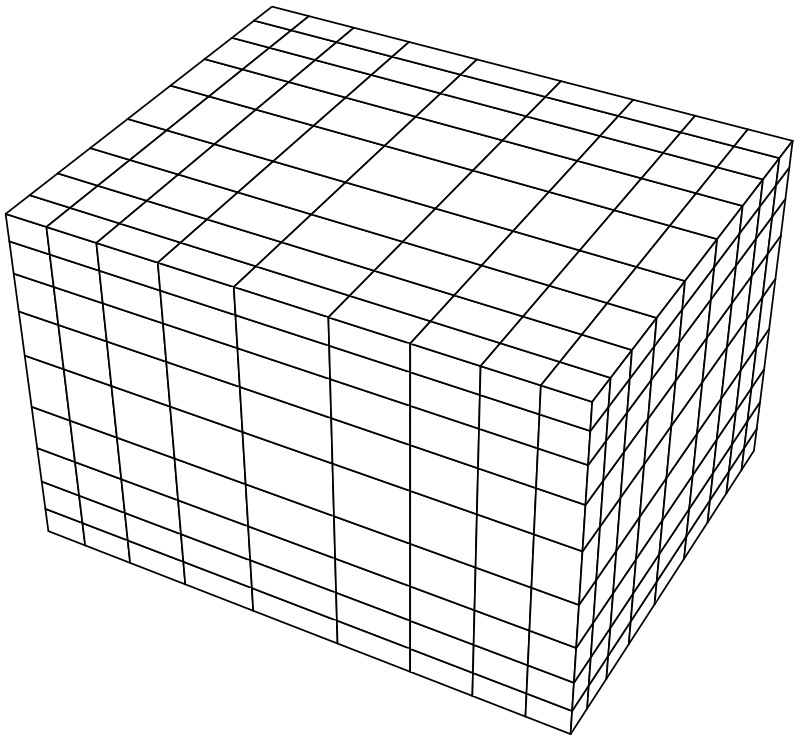


FIGURE 39. Rectangular cavity grid 3, a non-uniform, non-Cartesian hexahedral grid.

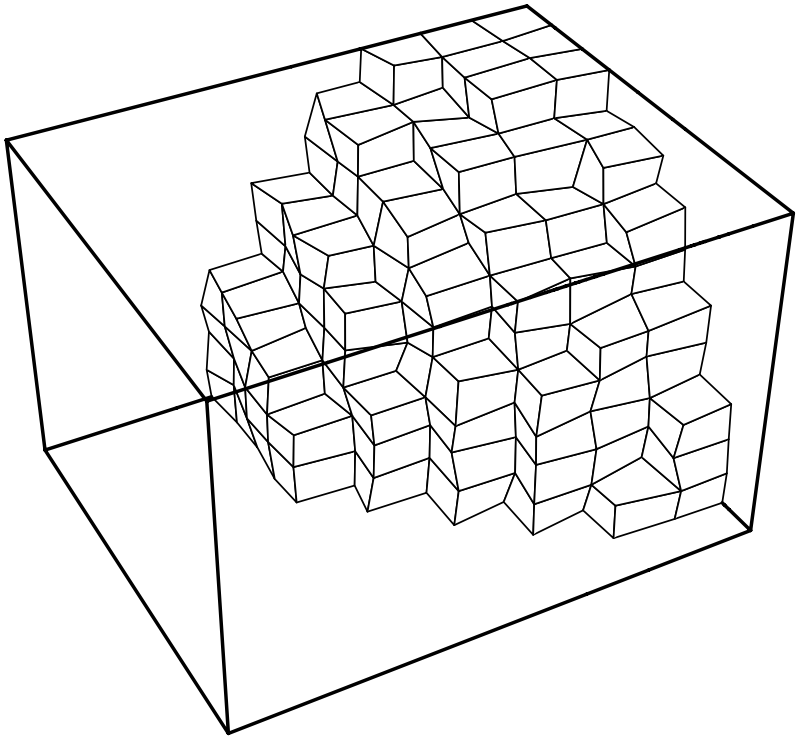
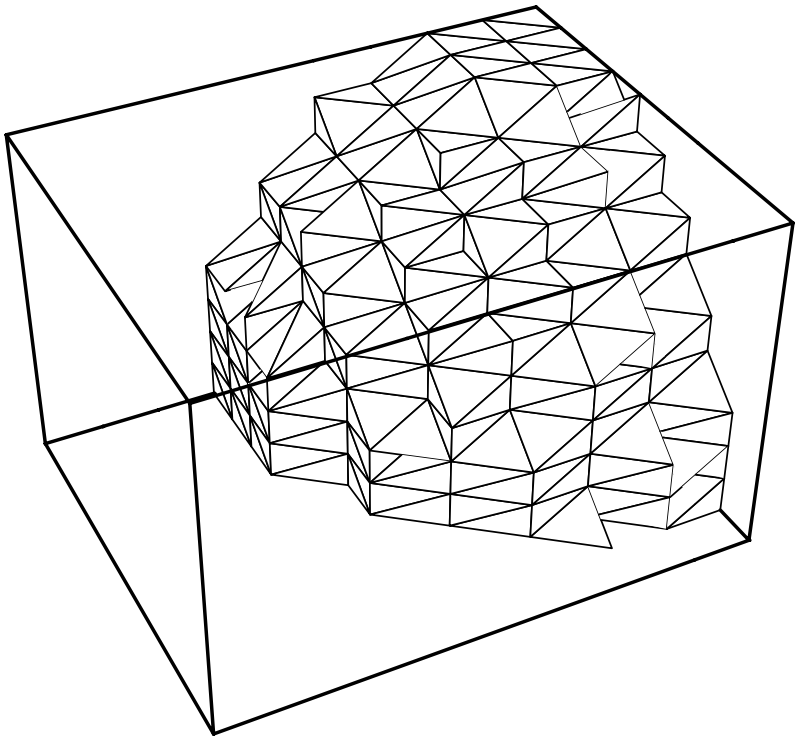


FIGURE 40. Rectangular cavity grid 4, a tetrahedral grid.



The resonant frequencies of the rectangular cavity were computed as follows. A cell near the $(0, 0, 0)$ corner of the cavity was randomly selected. A time dependent current source was put in the cell. The current source as a function of time is described by the second derivative of a Gaussian. This current source excites all of the cavity modes. The electromagnetic fields are updated for 5000 time steps using a step size of 0.5 seconds. Note that $\mu = \varepsilon = 1$ for this calculation, i.e., the velocity of light is unity. An edge within the cavity was selected at random and the electric field along this edge was written to disk at every time step. This signal was weighted by a Hamming window and then the signal was zero-extended to 32768 samples and then Fourier transformed. The magnitude of the Fourier transform is the power spectrum of the signal. The location of the peaks of the power spectrum should correspond to the resonant frequencies of the cavity. The width of the peaks is inversely proportional to the total time window. Note that the location of the peaks in the power spectrum is independent of the location of the current source and the selected output edge, whereas the amplitude of the peaks does depend upon location of the source location and the output location.

The computed power spectra for the calculations using grids 1, 2, and 3 are shown in Figure 41 - Figure 43 below. Note the grids 1, 2, and 3 give very similar results, indicating that there is no disadvantage to using a non-uniform or non-Cartesian grid. This is not true of the classic FDTD method. A careful examination of Figure 41 reveals that the first four frequencies are fairly accurate, while the higher frequencies are less accurate. Since the grid is 9 by 9 by 9 the lowest four modes, which have a half wavelength of a, b, or c, are sampled approximately 20 times per wavelength (10 edges per one-half wavelength)

whereas the next highest modes are sampled at 10 times per wavelength. Thus it is expected the lowest four lowest resonant frequencies be more accurate the higher frequencies. The relative error versus mode number is shown in Figure 44 using the computed resonant frequencies from grid 1. The average error for the first four modes is 0.006139, the error for the second four modes is 0.02608. This indicates that the error is proportional to $(\Delta x/\lambda)^2$, where the wavelength is $\lambda = 2\pi/\beta$, since a change in wavelength by a factor of two reduced the error by a factor of four. This preliminary evidence indicates that the DTVFEM is second order accurate.

FIGURE 41. Computed versus exact resonant frequencies for grid 1 using ICCG.

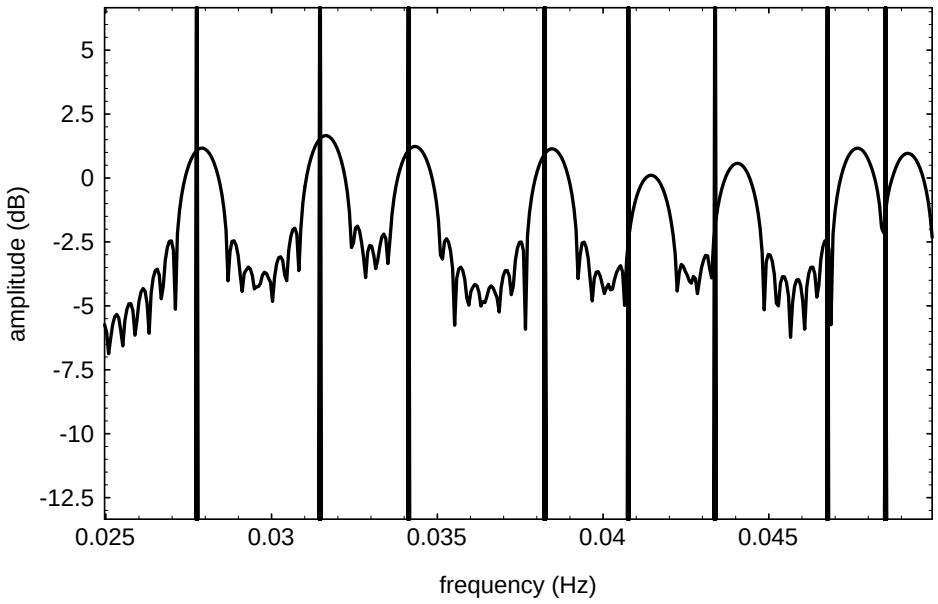


FIGURE 42. Computed versus exact resonant frequencies for grid 2 using ICCG.

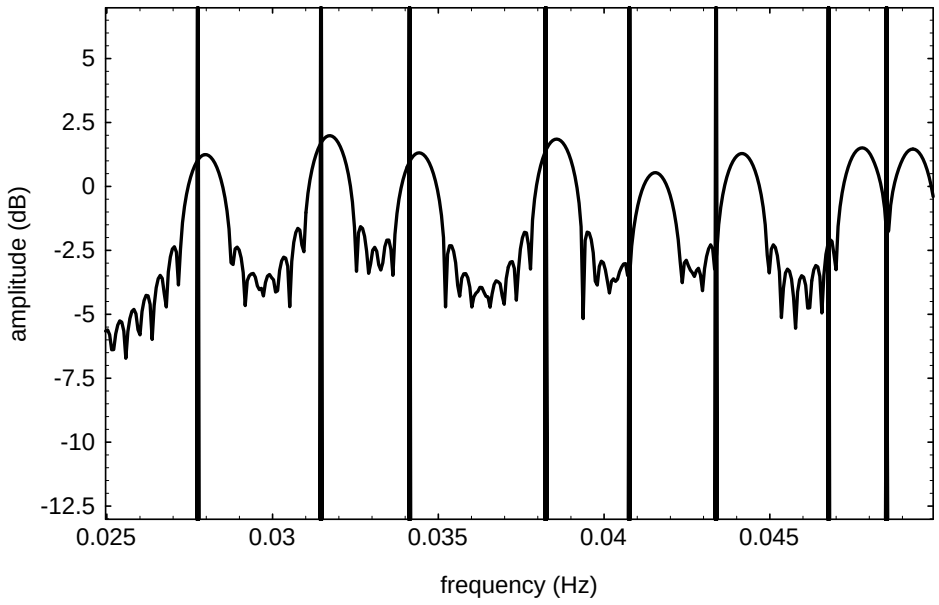


FIGURE 43. Computed versus exact resonant frequencies for grid 3 using ICCG.

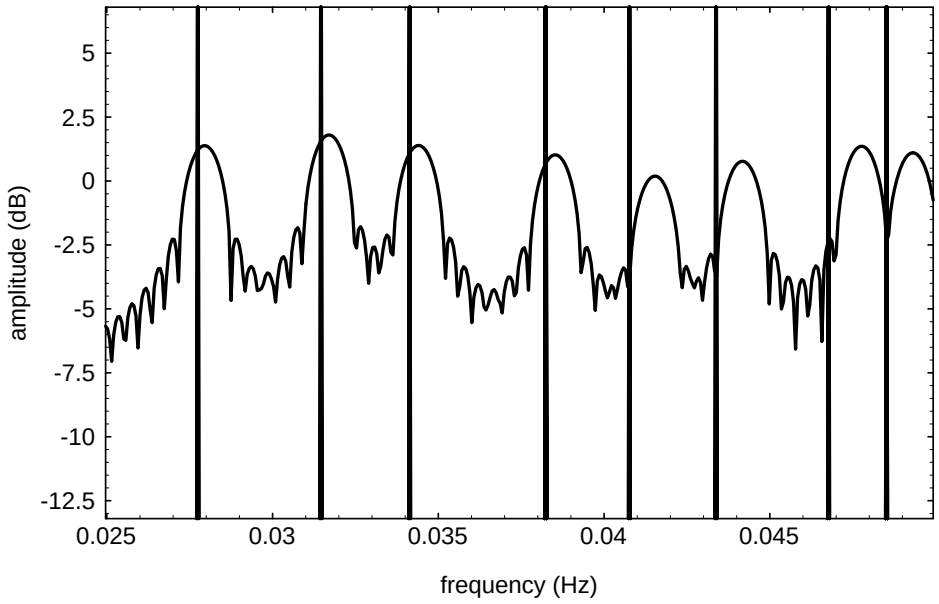
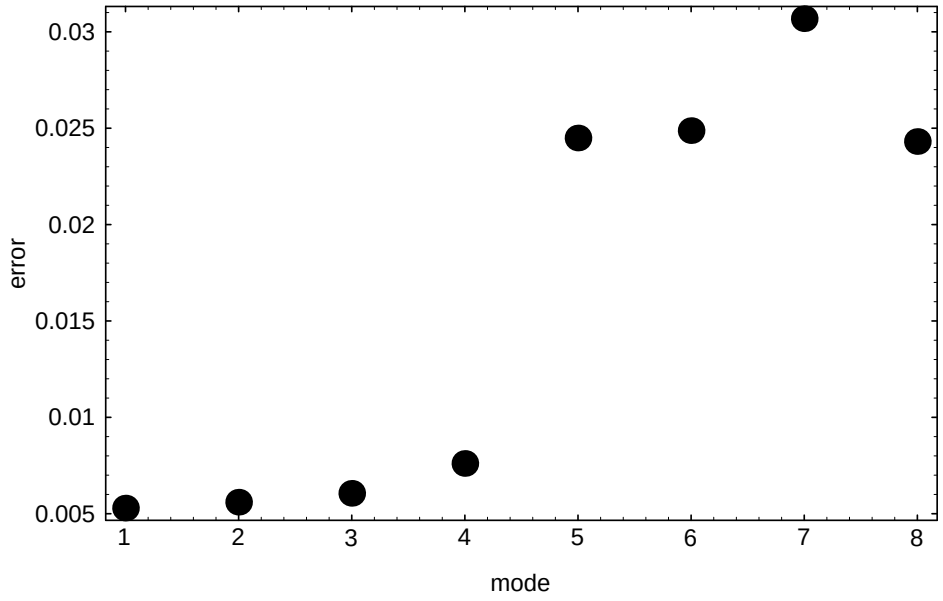


FIGURE 44. Relative error versus mode number using grid 1.



The results for grid 4 are shown in Figure 45. The computed frequencies are much more accurate for the tetrahedral grid than for the hexahedral grids. This is due in part to the fact that there are 4401 internal edges for grid 4 and 1728 internal edges for grids 1, 2, and 3. Thus the tetrahedral grid has approximately 2.5 times more degrees-of-freedom than the hexahedral grid. More degrees-of-freedom should result in a more accurate answer. Of course, more degrees-of-freedom imply more computer time. It is interesting to note that some other methods give less accurate results for tetrahedral grids than they do for Cartesian grids of the same problem [11], but this is not the case for the DTVFEM. The average relative error (eight lowest modes) and computer CPU time are tabulated for all four grids in Table 13. The incomplete Cholesky pre-conditioned conjugate gradient (ICCG) was used in each case. As mentioned in Section 6.1.2, the incomplete Cholesky decomposition is in fact equal to the complete Cholesky decomposition for Cartesian grids, thus grids 1

and 2 required only a single iteration for each time step, whereas grid 3 required on average 5 ICCG iterations, and grid 4 required on average 8 ICCG iterations. The CPU time shown in Table 13 is for VFEM3D running on a Silicon Graphics 8000 workstation (64 bit, 300 MFLOPS, SPECfp92 310). The CPU time is for the time stepping portion of the methods only, it does not include disk I/O or computing the matrices. The time required to compute the matrices and write them to disk is approximately equal to the time required for 100 time steps. For the above computational experiments the matrix fill time was 1/50 of the total CPU time.

FIGURE 45. Computed resonant frequencies for grid 4 using ICCG.

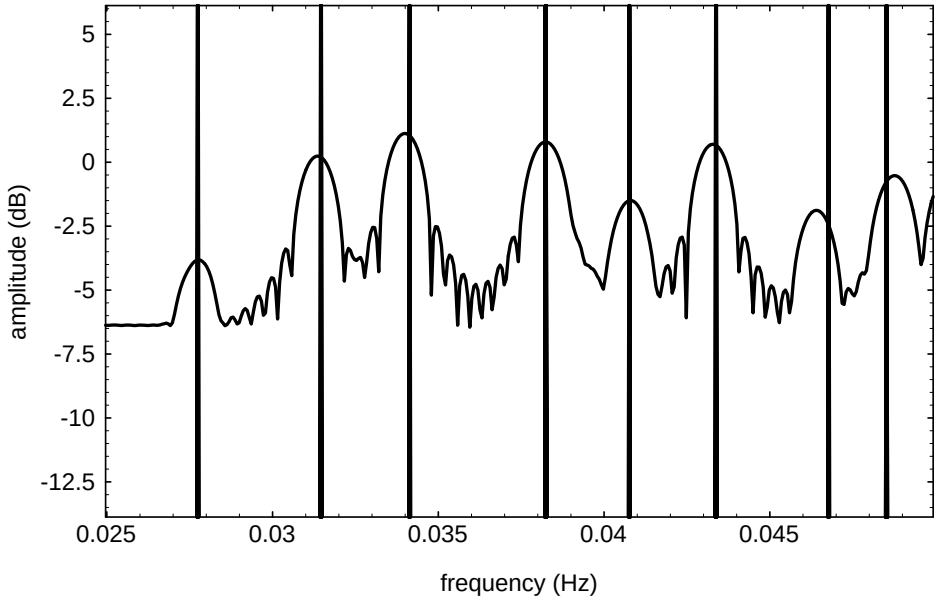


TABLE 13. VFEM3D error and CPU time for rectangular resonant cavity using ICCG.

	Grid 1	**Grid 2**	**Grid 3**	**Grid 4**	**Grid5**
# nodes	1000	1000	1000	1000	5832
# edges	1728	1728	1728	4401	10800
ICCG iter.	1	1	5	8	1

TABLE 13. VFEM3D error and CPU time for rectangular resonant cavity using ICCG.

	Grid 1	**Grid 2**	**Grid 3**	**Grid 4**	**Grid5**
Error	0.01607	0.017443	0.016939	0.004613	0.00478
CPU	204.9	204.9	641.4	1439.8	1563.0

In order to directly compare the tetrahedral grid to the uniform Cartesian grid, the grid spacing was reduced until the error became (approximately) the same for the two grids. This resulted in a new 17 by 17 by 17 uniform Cartesian grid for the same rectangle. This is referred to as grid 5, it is not shown because it looks exactly like grid 1 except for more cells. Grid 5 has 10800 degrees-of-freedom. The time step was reduced to $\Delta t = 0.45$ and the number of time steps increased to 5555. The total CPU time and average error for the lowest eight resonant frequencies is shown in the last column of Table 13. These results indicate that for this particular experiment, a 5832 node uniform Cartesian grid has similar accuracy and CPU time as a 1000 node tetrahedral grid, although the tetrahedral results are slightly better.

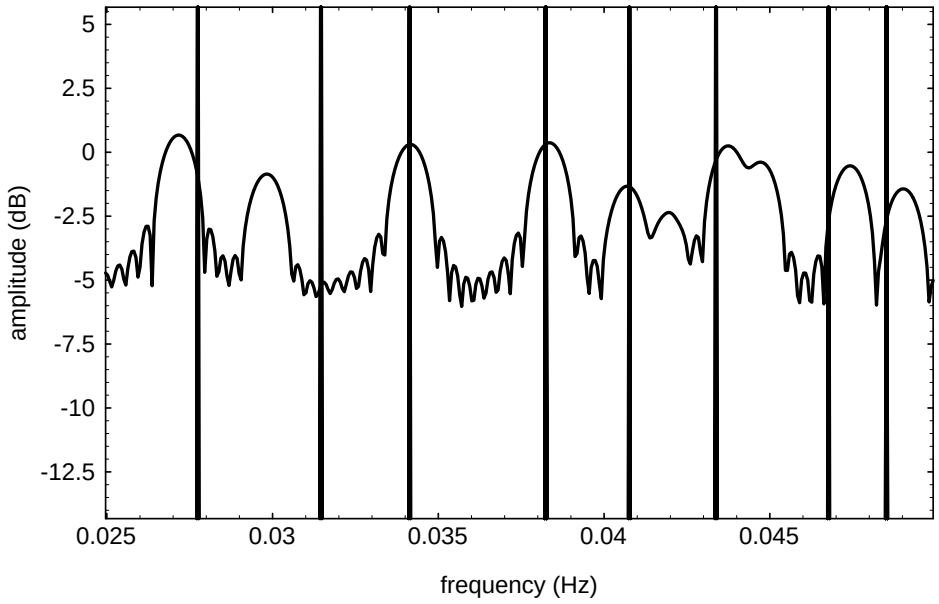
As mentioned in Section 6.1.1, capacitance lumping can be used on Cartesian grids, in which case the DTVFEM reduces to the classic FDTD method. Capacitance lumping is more efficient since it does not require that a linear system be solved at each time step, it can be considered a zero iteration method. The results for all five grids using capacitance lumping are shown in Table 14. For the Cartesian grids there is a slight degradation of the error, but this may be tolerated since the CPU time was reduced significantly. Hence the popularity of the classic FDTD method for Cartesian grids. However for the non-Cartesian grids there is a significant degradation of the error. For example on grid 4 the error is

500% larger when capacitance lumping is employed, which is not acceptable. The computed power spectrum for grid 4 using capacitance lumping is shown in Figure 46.

TABLE 14. VFEM3D error and CPU time for rectangular resonant cavity using capacitance lumping.

	Grid 1	**Grid 2**	**Grid 3**	**Grid 4**	**Grid5**
# nodes	1000	1000	1000	1000	5832
# edges	1728	1728	1728	4401	10800
Error	0.0252	0.224	0.0388	0.021	0.00645
CPU	111.8	111.8	159.39	217.39	1078.0

FIGURE 46. Computed resonant frequencies for grid 4 using capacitance lumping.



While capacitance lumping was ineffective on grid 3 and grid 4, the incomplete Cholesky stationary iteration described in Section 6.2.1 was very effective. Unlike ICCG this iteration is stable regardless of how few iterations are used. The minimum number of iterations required was determined via trial and error. On grid 3 a single iteration was sufficient, which resulted in a CPU time 373.9 seconds. This is almost twice as fast as ICCG applied

to the same problem. For grid 4 only three IC stationary iterations were required to achieve the same error as ICCG for this grid. This reduced the CPU by almost a factor of two. The conclusion is that the linear system need not be solved exactly at every time step in order to get comparable results. Of course the number of stationary IC iterations required to achieve good results is grid dependent and not known a priori.

TABLE 15. VFEM3D number of required IC stationary iterations for various grids.

Grid	**1**	**2**	**3**	**4**
CPU time	190.7	190.7	373.9	892.5
Error	0.01607	0.017443	0.016939	0.004613
# iterations	1	1	1	3

8.2 Spherical Cavity

In this section a perfectly conducting spherical cavity is analyzed using VFEM3D and the computed solutions are compared to the exact analytical solution. The frequency domain electric field in the cavity is a solution of the vector Helmholtz equation (215). The fields can be decomposed into independent modes TE_r and TM_r which can be found in many textbooks [47]. The TE_r solution is given by

$$\begin{aligned} E_r &= 0, \\ E_\theta &= -A_{mnp}\frac{m}{\varepsilon}\frac{1}{r\sin\theta}\tilde{J}_n(\beta r)\,P_n^m(\cos\theta)\,[-C\sin(m\theta) + D\cos(m\theta)], \\ E_\phi &= A_{mnp}\frac{1}{\varepsilon r}\tilde{J}_n(\beta r)\frac{\partial}{\partial\theta}\Big(P_n^m(\cos\theta)\Big)[C\sin(m\theta) + D\cos(m\theta)], \end{aligned} \tag{222}$$

where

$$\beta = \frac{\zeta_{np}}{a} \qquad \begin{matrix} n = 1, 2, 3, \ldots \\ p = 1, 2, 3, \ldots \end{matrix}, \tag{223}$$

and ζ_{np} is the pth zero of the spherical Bessel function of order n. The resonant frequencies of the TE_r mode are

$$f_{mnp} = \frac{\zeta_{np}}{2\pi a\sqrt{\mu\varepsilon}} \qquad \begin{matrix} m = 0, 1, 2, \ldots, n \\ n = 1, 2, 3, \ldots \\ p = 1, 2, 3, \ldots \end{matrix} . \tag{224}$$

The resonant frequencies of the TM_r mode are given by

$$f_{mnp} = \frac{\zeta'_{np}}{2\pi a\sqrt{\mu\varepsilon}} \qquad \begin{matrix} m = 0, 1, 2, \ldots, n \\ n = 1, 2, 3, \ldots \\ p = 1, 2, 3, \ldots \end{matrix} , \tag{225}$$

where ζ'_{np} represents the pth zero of the derivative of the spherical Bessel function of order n. The exact resonant frequencies for cavity of radius $a = 0.05855m$, assuming $\mu = 1$ and $\varepsilon = 1$, are shown in Table 16.

TABLE 16. Exact value of resonant frequencies below f = 20 Hz.

TM11	TM21	TE11	TM31	TE21	TM41	TM12	TE31
7.4589	10.5665	12.2132	13.518	15.6654	16.4782	16.6277	18.9953

In the time domain the electric field within the cavity is described by PDE II, (6)-(11). In this case $\sigma_E = \sigma_M = 0$ and $\mu = \varepsilon = 1$ within the cavity. The exact solution is of the form

$$\vec{E} = \sum_{np} A_{np}\vec{E}^{TE}_{np}\cos(\omega_{np}t + \phi_{np}) + \sum_{np} B_{np}\vec{E}^{TM}_{np}\cos(\omega_{np}t + \theta_{np}) , \tag{226}$$

where the sum is over all the modes, and A, B, ϕ, and θ depend upon the initial conditions. In this computational experiment the electric field is excited by a pulsed current source (described below) which is designed to excite all of the cavity modes. Once the source is turned off the electric field within the cavity obeys (226) for all time.

8.2.1 Hexahedral grid results

The perfectly conducting spherical cavity of radius $a = 0.05855m$ was modeled using a sequence of hexahedral grids ranging from a coarse grid with 4 cells per radius to a fine grid with 12 cells per radius. The grids were generated using TrueGrid [80]. Figure 47 and Figure 48 are cut away views of the 256 hexahedral and 2048 hexahedral grids, respectively. As in the rectangular cavity case, the electromagnetic fields in the cavity were excited by a pulsed current source, the pulse having the shape of the second derivative of a Gaussian. The initial electric and magnetic fields within the cavity were zero. The simulation was run for $T = 6.71315s$ which corresponds to fifty periods of the lowest mode. The time step and the number of steps was different for each grid due to different stability requirements. The power spectrum was computed in the same manner as for the rectangular cavity case. The power spectrum for the 256 hexahedral case and the 2048 hexahedral case are shown in Figure 49 and Figure 50, respectively.

TABLE 17. Relative error of TM_{31} resonant frequency versus grid size for hexahedral grid.

h/a	1/4	1/6	1/8	1/10	1/12
# nodes	321	997	2273	4341	7393
# cells	256	864	2048	4000	6912
# edges	688	2400	5792	11440	19920
error	0.09846	0.03960	0.02342	0.01589	0.009693

Naturally the power spectrum corresponding to the higher resolution grid is more accurate then the power spectrum corresponding the lower resolution grid. It is possible to perform a least-square fit to the data to determine, experimentally, the order of accuracy of the method. For this fit, the error was defined to be the difference between the exact TM_{31} frequency and the computed TM_{31}. This error is shown in Table 17 as a function of grid size, where h is the average cell size. The logarithm of the error versus the logarithm of (h/a) is shown in Figure 55, along with a linear least-square fit. The slope of the line is 2.028, indicating that the method is second order accurate. This agrees with the numerical dispersion analysis in Section 5.4.

TABLE 18. CPU time for cavity calculation versus grid size for hexahedral grid.

# edges	**688**	**2400**	**5792**	**11440**	**19920**
Δt	.0035	.002	.0015	.001	.001
# steps	1918	3356	4475	6713	6713
# ICCG iter.	7.8	7.8	7.8	7.8	7.8
CPU sec.	107	731	3255	11962	22490

The CPU time for the hexahedral calculations is shown in Table 19. The CPU time is for the time stepping part of the calculation only. For the above experiments the matrix fill time is approximately 1/50 of the total CPU time. The CPU time is for VFEM3D running on a Silicon Graphics 8000 workstation. The computer time increase for two reasons; the number of degrees of freedom increases and the number of time steps increases. Note that the number of ICCG iterations does not increase as the grid is refined. The calculations required 7.8 ICCG iterations on average. The stopping criteria for the ICCG was

$$\|r\|/\|b\| \leq 10^{-9}, \tag{227}$$

where $\|r\|$ is the Euclidean norm of the residual and $\|b\|$ is the Euclidean norm of the right hand side. Therefore the computational cost per time step proportional to the number of degrees of freedom.

The number of ICCG iterations does not increase as the grid is refined because the condition number of the capacitance matrix remains constant. For a uniform Cartesian grid it can be shown that the condition number of the capacitance matrix is 9, independent of how large the matrix is. The capacitance matrix is a Gram matrix, and if the condition number of a Gram matrix remains constant as the number of basis functions increases the basis functions are said to be uniformly linearly independent. This is true for the linear nodal finite elements [54] and it appears to be true for linear vector finite elements as well. The condition number of the capacitance matrix does however depend upon variations in dielectric constant, and the condition number of a highly distorted grid will be larger than that of a regular grid. The conditioning of the capacitance matrix as a function of grid distortion has been analyzed for two dimensional grids [43], but a similar analysis for three dimensional grids is probably intractable.

Additional experiments using the stationary iteration described in Section 6.2.1 were performed. For these experiments the preconditioner M was constructed by the capacitance lumping method. The results obtained using a single iteration, which corresponds to simple capacitance lumping, were quite poor. The error was, on average, 500% larger than that obtained with ICCG. The number of stationary iterations was varied until the error in the computed solution was comparable to that obtained when using ICCG. It was experimentally determined that four iterations were sufficient for all of the hexahedral grids.

This is equivalent to a four-term Jacobi-like polynomial approximation for the inverse of the Capacitance matrix. For these experiments, using the four-term polynomial approximation reduced the CPU time by more than a factor of two, as shown in Table 19.

TABLE 19. CPU time for cavity calculation versus grid size for hexahedral grid using four-term polynomial approximation for approximately inverting the capacitance matrix.

# edges	**688**	**2400**	**5792**	**11440**	**19920**
CPU sec.	50	318	1256	4975	9670

FIGURE 47. Internal view of 256 hexahedral grid of sphere.

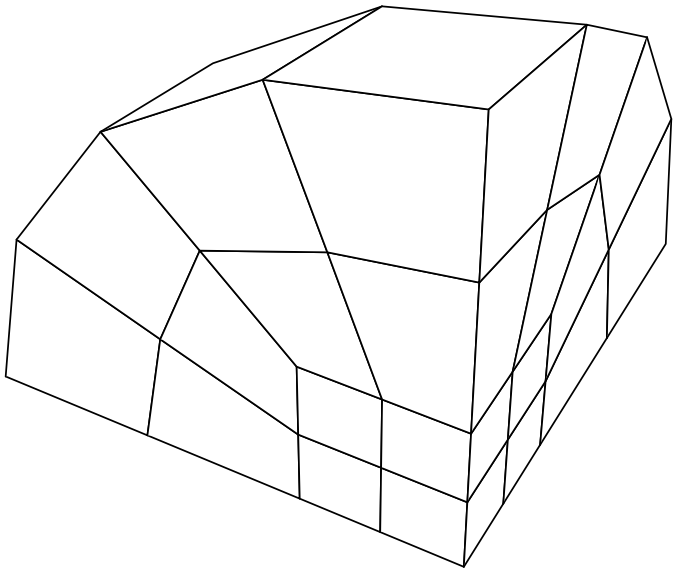


FIGURE 48. Internal view of 2048 hexahedral grid of sphere.

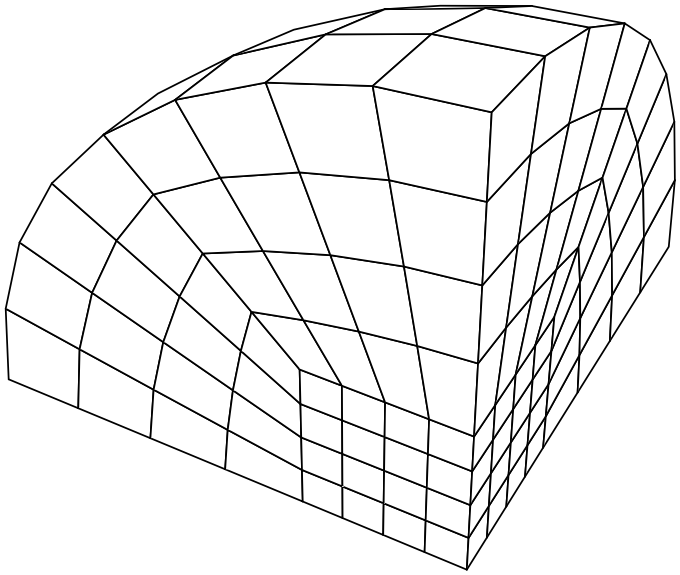


FIGURE 49. Computed power spectrum versus exact for 256 hexahedral sphere

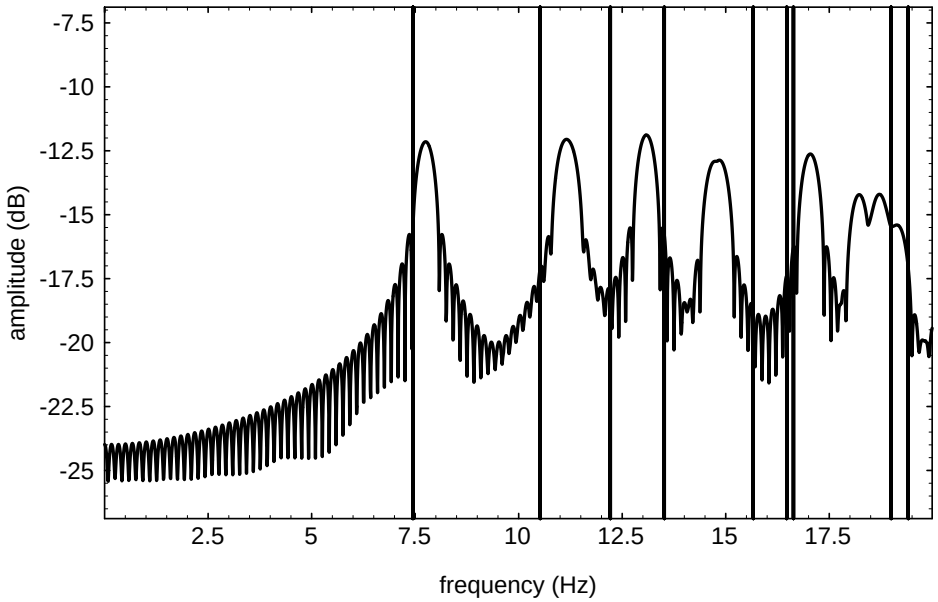
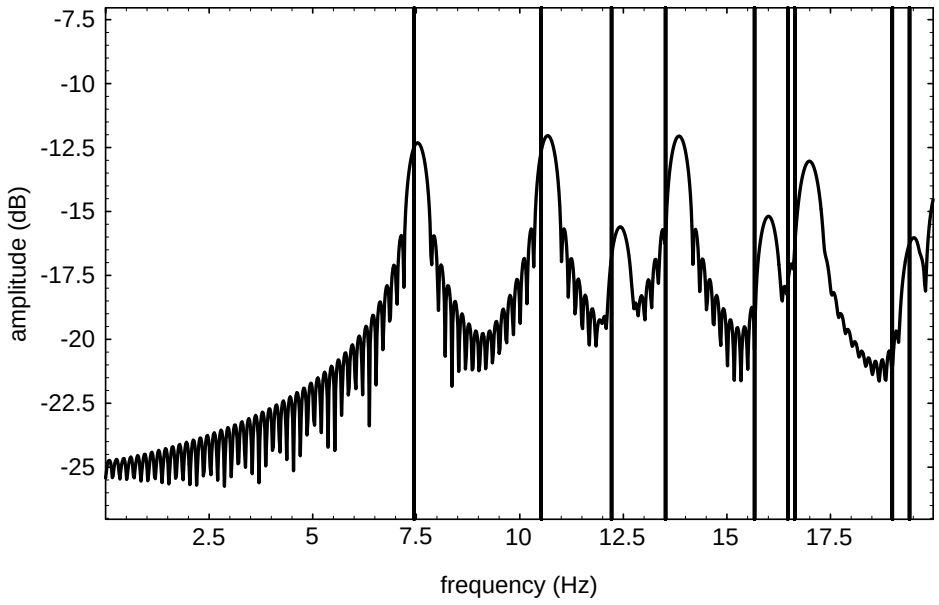


FIGURE 50. Computed power spectrum for 2048 hexahedral sphere.



8.2.2 Tetrahedral grid results

The same procedure described above was performed on a sequence of tetrahedral grids. The tetrahedral grids were constructed using GEOMPACK [81] with the exact same node locations as in the above hexahedral grids. The error versus grid spacing is shown in Table 17. The results indicate that for a given number of nodes the tetrahedral grid is more accurate than the corresponding hexahedral grid. The logarithm of the error versus the logarithm of (h/a) is shown in Figure 55, along with a linear least-square fit. The slope of the line is 2.17, indicating that the method is second order accurate. The CPU time for the tetrahedral grid is shown in Table 19. The data shows that for the same number of nodes the CPU time for a tetrahedral grid is approximately 3 times greater. There are several reasons for the increase in CPU time. The main reason is that the tetrahedral grid has more

edges, and hence more degrees of freedom, than the corresponding hexahedral grid. Another reason is that the time step was reduced by about 30%, which required 30% more time steps to simulate the same amount of physical time. Finally, for this particular experiment, the ICCG required approximately three more iterations to meet the same convergence criteria (227).

TABLE 20. Relative error of TM_{31} resonant frequency versus grid size for tetrahedral grid.

(h/a)	1/4	1/6	1/8	1/10	1/12
# nodes	321	997	2273	4341	7393
# cells	1536	5162	12248	23907	41040
# edges	1952	6374	14904	28847	49296
error	0.04951	0.017408	0.01138	0.00660	0.004266

TABLE 21. CPU time for cavity calculation versus grid size for tetrahedral grid.

# edges	**688**	**2400**	**5792**	**11440**	**19920**
Δt	.00225	.0015	.0010	.00085	.00080
# steps	2983	4475	6712	7897	8390
# ICCG iter.	10.1	10.1	10.1	10.1	10.1
CPU sec.	349	2172	11216	29153	58237

Additional experiments using the stationary iteration described in Section 6.2.1 were performed. For these experiments the preconditioner M was the incomplete Cholesky decomposition, rather than the lumped-capacitance matrix. The number of stationary iterations was varied until the error in the computed solution was comparable to that obtained when using ICCG. It was experimentally determined that four iterations were sufficient for all of the tetrahedral grids. For these experiments the CPU time was reduced by more than a factor of two, as shown in Table 19.

TABLE 22. CPU time for cavity calculation versus grid size for tetrahedral grid using incomplete Cholesky stationary iteration.

# edges	**688**	**2400**	**5792**	**11440**	**19920**
CPU sec.	174	1130	5508	14119	24597

FIGURE 51. Internal view of 1356 cell tetrahedral grid of sphere.

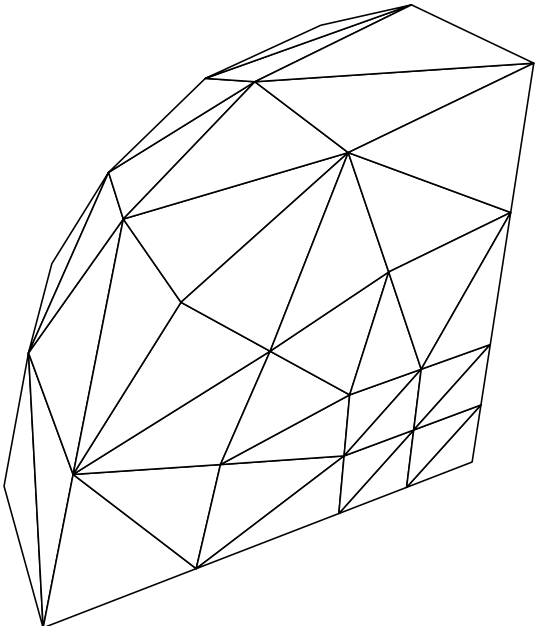


FIGURE 52. Internal view of 12248 cell tetrahedral grid of sphere.

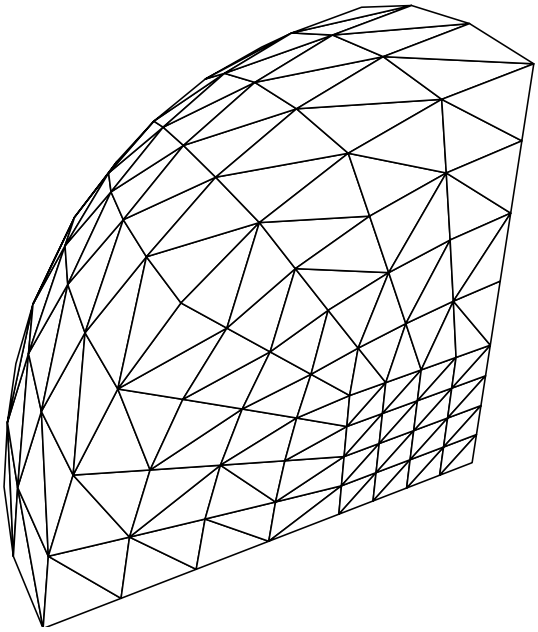


FIGURE 53. Computed power spectrum versus exact for 1536 cell tetrahedral sphere.

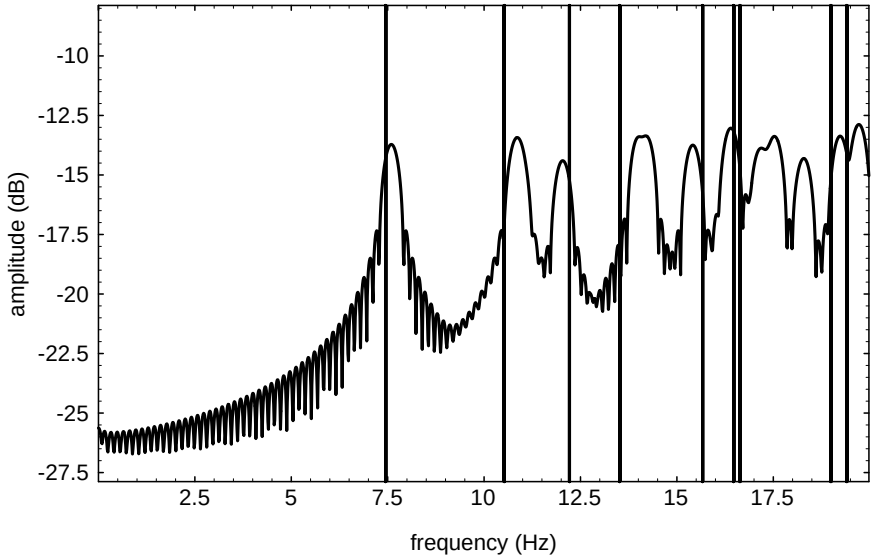


FIGURE 54. Computed power spectrum versus exact for 12248 cell tetrahedral sphere.

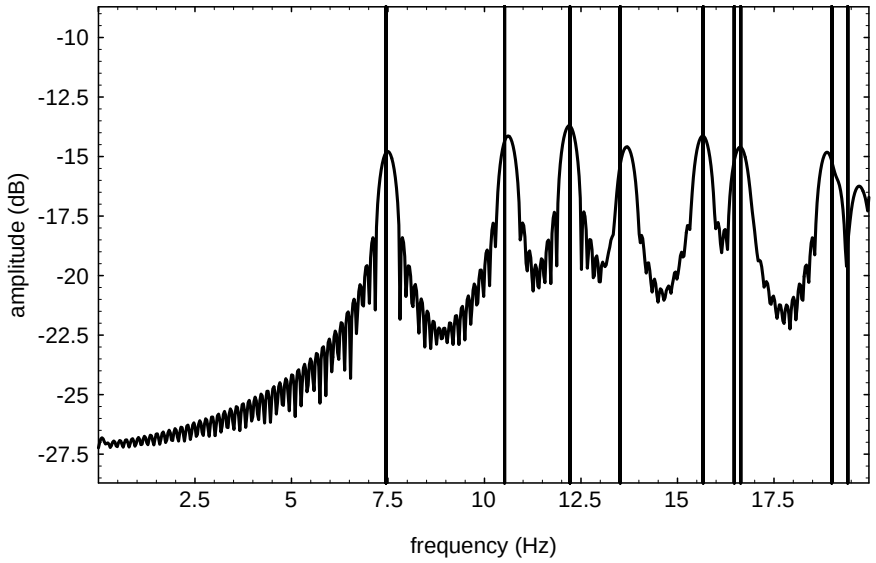
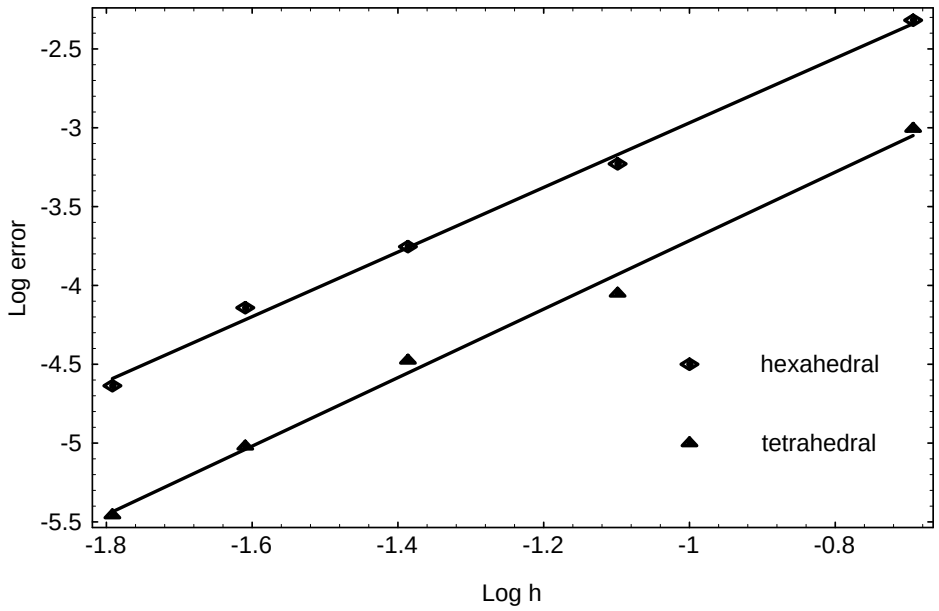


FIGURE 55. Log error versus Log h indicating second order accuracy.



8.3 Rectangular Waveguide

In this section, VFEM3D is used to compute the electromagnetic fields in a rectangular waveguide. The exact solution is well known and can be found in many textbooks [46]. The computed solution is compared to the exact solution for different grids. This problem is different from the resonant cavity problems above because the waveguide is infinite in one dimension, thus a PML is employed to simulate an infinite guide. A time varying boundary condition is used at the other end of the wave guide to launch the wave.

As in the rectangular cavity problem in Section 8.1 the exact solution can be expressed as a linear combination of modes. Unlike the resonant cavity examples above, in which the excitation was specifically chosen to excite all the modes, in this problem the excitation is chosen to excite only a single mode. Let the rectangular waveguide have width a in the $\hat{x}$

direction, height b in the $\hat{y}$ direction, and be infinite in the $\hat{z}$ direction. The lowest mode is the TE_{10} which is given by

$$\begin{aligned} E_x &= 0, \\ E_y &= A\sin(\pi x/a)\sin(\omega t - \beta_z z), \\ E_z &= 0, \end{aligned} \tag{228}$$

$$\begin{aligned} H_x &= -\frac{\beta_z}{\omega\mu}A\sin(\pi x/a)\sin(\omega t - \beta_z z), \\ H_y &= 0, \\ H_z &= A\frac{(\pi/a)^2}{\omega\mu}\cos(\pi x/a)\cos(\omega t - \beta_z z), \end{aligned} \tag{229}$$

where the wave number is

$$\beta_z = \sqrt{\omega^2\mu\varepsilon - (\pi/a)^2}. \tag{230}$$

Note that if $\omega^2 < \omega_c^2 = \frac{1}{\mu\varepsilon}\left(\frac{\pi}{a}\right)^2$, the wave number is imaginary, then (228) describes diffusion rather than propagation of the electromagnetic field. Assuming that $\omega > \omega_c$ the electromagnetic field propagates down the guide with a velocity and wavelength given by

$$\begin{aligned} v_z &= \frac{\omega}{\beta_z}, \\ \lambda_z &= 2\frac{\pi}{\beta_z}. \end{aligned} \tag{231}$$

The waveguide to be analyzed has a cross section $a = 1m$ by $b = 1/2m$. The infinite waveguide is approximated by a finite length waveguide of length $10m$. The waveguide is

modeled using a sequence of grids in order to determine the accuracy of the method. The coarsest grid has $h = a/6$, the finest has $h = a/14$, where h is the average cell size. All of the grids have a chevron pattern to them. This pattern does not represent anything physical; the purpose is to demonstrate that the DTVFEM is stable for distorted grids. Several finite volume methods have been shown to be unconditionally unstable for this particular grid [10]. Two of the grids are illustrated in Figure 56 and Figure 57.

FIGURE 56. Rectangular waveguide model using 1080 chevron cells.

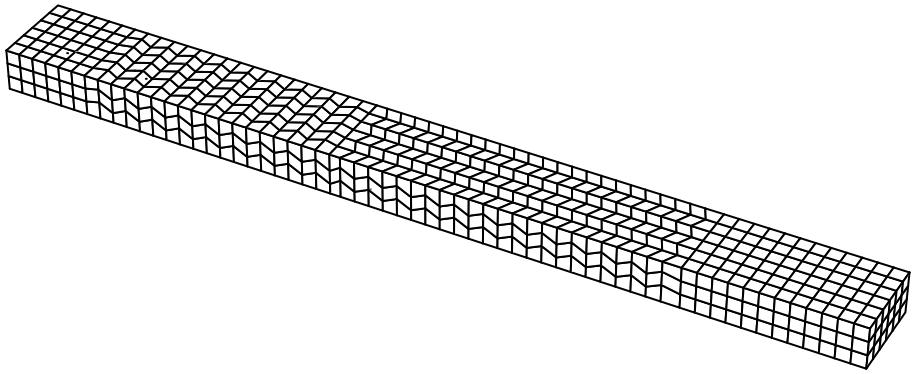
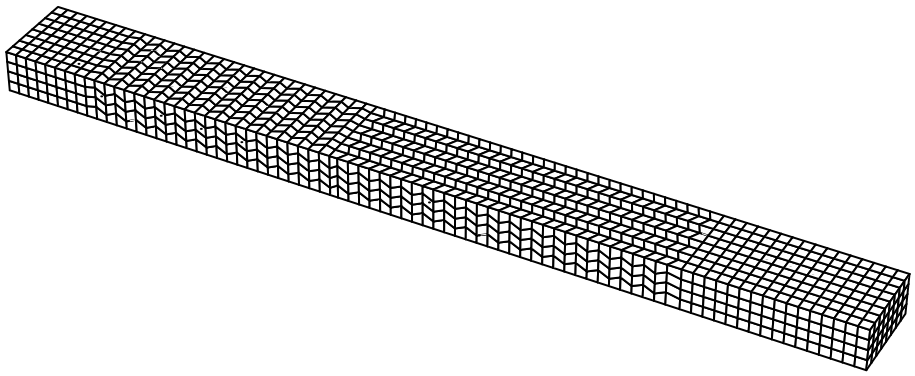


FIGURE 57. Rectangular waveguide model using 2560 chevron cells.



A TE_{10} wave is launched by forcing the boundary condition

$$E_x = 0,$$
$$E_y = \left(1 - \exp\left(-\left(\frac{t}{2T}\right)^2\right)\right)\sin(\pi x/a)\sin(\omega t), \tag{232}$$

at the left end ($z = 0$) of the waveguide, where $\omega = 5.523599$ and $T = 0.5$. The frequency of excitation is above the cutoff frequency of the TE_{10} mode, but below the cutoff of the next highest mode. The initial electric and magnetic fields in the guide are zero. The simulation was run for 20 seconds. This is enough time for the wavefront to propagate approximately 20 meters, which is twice the length of the finite guide. The wave will be attenuated by the PML at the right end of the waveguide, thus the simulation will reach a dynamic steady state condition that resembles the TE_{10} mode of an infinite waveguide.

In this simulation, a five layer PML was used to absorb the outgoing wave. Each layer is defined by the tensor material properties $\mu, \varepsilon, \sigma_E, \sigma_M$. In every layer μ and ε are simply identity matrices. The conductivity matrices are equal, $\sigma_E = \sigma_M = \sigma$, where σ is a diagonal matrix with $\sigma_{xx} = \sigma_{yy} = \sigma_\perp$ and $\sigma_{zz} = 1$. The values of $\sigma_\perp$ used are tabulated in Table 23.

TABLE 23. Perfectly Matched Layer parameters used for truncated waveguide.

	layer 1	**layer 2**	**layer 3**	**layer 4**	**layer 5**
$\sigma_\perp$	1.8	7.2	16.2	28.8	45

The simulation was run for 20 seconds, the number of time steps depending upon Δt, which of course is different for each grid. Unlike the spherical cavity problem where only the electric field was calculated using PDE II, for the waveguide problem both the electric and magnetic fields were computed using PDE I. The time step, number of steps, ICCG iterations, and CPU time are shown in Table 24. The CPU time increases for two reasons; the number of degrees of freedom increases, and the number of time steps increases. Note, however, that the number of ICCG iterations is independent of the grid spacing, hence the method requires $O(n)$ operations per time step. The same stopping criteria was the same as for the spherical cavity, i.e., equation (227).

TABLE 24. CPU time for chevron waveguide calculations.

(h/a)	**1/6**	**1/8**	**1/10**	**1/12**	**1/14**
# cells	**1080**	**2560**	**5000**	**8640**	**13720**
# edges	4425	9756	18215	30522	47397
Δt	0.016666	0.0125	0.01	0.008333	0.007142
# steps	1272	1696	2120	2544	2968
ICCG iter.	5.7	5.7	5.7	5.7	5.7
CPU sec.	352	1719	5163	12267	23161

The computed electric and magnetic fields in the waveguide are compared to the exact solution. There are several measures by which the computed solution can be compared to the exact solution. The first measure is the standard L_2 norm, given by

$$L_2 \text{ error } = \sqrt{\int_{\Omega} (\vec{E}_{exact} - \vec{E}_{computed}) \bullet (\vec{E}_{exact} - \vec{E}_{computed})\, d\Omega}. \tag{233}$$

The computed and exact vector electric field is evaluated at the center of every hexahedral cell. The difference is the error field. The L_2 norm of the error field is then divided by the L_2 norm of the exact field, where the L_2 norm is approximated using the midpoint rule. This is shown in (234) below, where the sum is over all the hexahedral cells (excluding PML cells) in the grid.

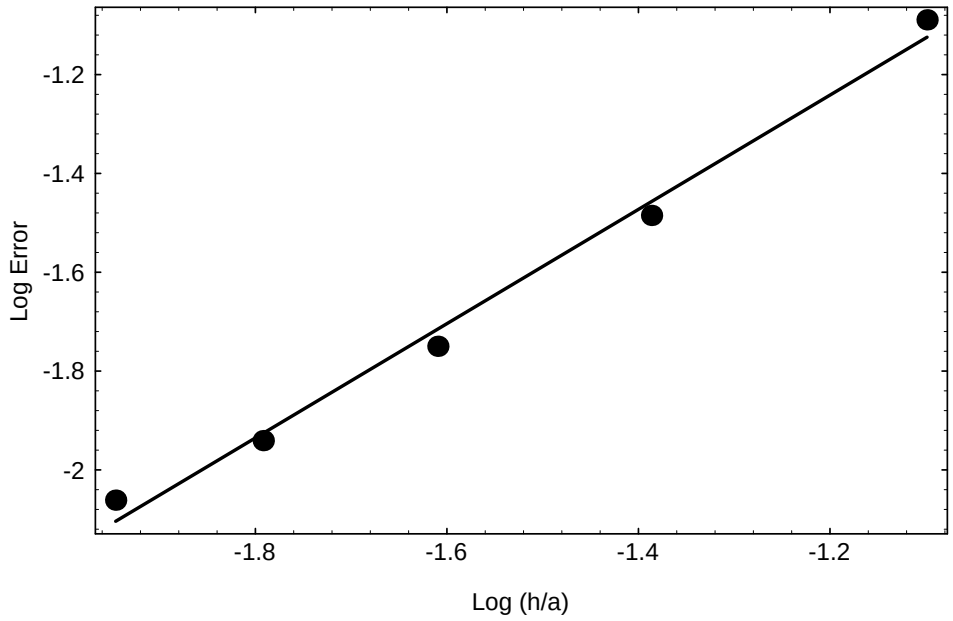
$$relative\ L_2\ error\ = \frac{\sqrt{\sum (\vec{E}_{exact} - \vec{E}_{computed}) \bullet (\vec{E}_{exact} - \vec{E}_{computed})}}{\sqrt{\sum \vec{E}_{exact} \bullet \vec{E}_{exact}}}. \tag{234}$$

The L_2 norm defined by (234) is tabulated in Table 25 as a function of grid spacing. The logarithm of the error versus the logarithm of (h/a) is shown in Figure 58 along with a linear least-square fit. The slope of the line is 1.15, indicating that the method is first order accurate.

TABLE 25. Error versus grid spacing for chevron waveguide.

(h/a)	**1/6**	**1/8**	**1/10**	**1/12**	**1/14**
# cells	1080	2560	5000	8640	13720
L_2 error	0.3365	0.2265	0.1738	0.1436	0.1273

FIGURE 58. Log error versus Log h indicating first order accuracy.



In Section 8.2 analysis of the computed resonant frequencies of a spherical cavity indicated that the DTVFEM is second order accurate. This agrees with the analytical numerical dispersion analysis of the method, Section 5.4 and [43]. However, the above results indicate first order accuracy. This is not a discrepancy, it is simply that different measures can lead to different accuracies. While the classic FDTD method [1]-[4] and related finite volume methods [6]-[9] are often considered to be second order accurate, they are only first order accurate according to the L_2 norm. This can be explained as follows.

Numerical dispersion is a measure of how the numerical phase velocity, i.e., the velocity of a wave on a finite grid, compares to the exact phase velocity. A second order accurate numerical dispersion relation means that the numerical phase velocity agrees with the exact phase velocity to second order in the Taylor series sense. Consider a simple two-

dimensional Cartesian grid with a known electric field boundary condition on the left hand side at $x = 0$. This is illustrated in Figure 59. An electromagnetic wave induced by the boundary condition propagates to the right. Since the DTVFEM is energy conserving, i.e., non-dissipative, the wave will propagate in the x with constant amplitude, but with a slightly incorrect velocity due to numerical dispersion. Thus the value of the computed electric field at some point $x = x_0$ will not agree with the exact electric field at $x = x_0$ due to the phase difference. This difference between the computed electric field and the exact electric field at $x = x_0$ is second order. Thus in this sense the DTVFEM, along with the classic FDTD and other finite volume variants, are second order accurate.

However the L_2 norm is a very different measure of the error. Consider a single cell with the electric field known along the edges of the cell as shown in Figure 60. The L_2 error is defined by (233) where the integral is over the square cell. The $\hat{x}$ component of the electric field is known at two different y positions, the $\hat{y}$ component of the electric field is known at two different x positions. Let the exact electric field be denoted by $\vec{E}$ and the approximate electric field be denoted by $\tilde{E}$. The only possible form of $\tilde{E}$ is

$$\tilde{E} = \hat{x}\,(Ex_0 + (Ex_1 - Ex_0)\,y) + \hat{y}\,(Ey_0 + (Ey_1 - Ey_0)\,x)\,, \tag{235}$$

i.e., the $\hat{x}$ component of $\tilde{E}$ is linear in y but constant in x, likewise the $\hat{y}$ component of $\tilde{E}$ is linear in x and constant in y. Since $\vec{E}$ is an arbitrary vector function, and the $\hat{x}$ component of $\tilde{E}$ is independent of x and the $\hat{y}$ component of $\tilde{E}$ is constant in y, the approxima-

tion is only first order accurate within the cell. This is true for the DTVFEM method as well as the classic FDTD method the finite volume variants. All of this analysis is equally valid for three dimensional grids. It should be noted that first order L_2 accuracy is adequate form most applications, and if not, higher order finite elements can be developed. The fact that higher order elements can be developed is an important advantage of finite element methods in general. Higher order vector finite elements have been used in the frequency domain to solve the vector Helmholtz equation [82]. While use of these elements does give a faster rate of convergence as the grid is refined, the computation cost rises dramatically. It is not clear which element gives optimal price/performance. It is interesting to note that while it is possible to develop “higher order” FDTD methods using high order finite difference schemes, the computed results are still only first order accurate in the L_2 sense. Higher order finite difference schemes give rise to a higher order dispersion relation, but the only way to increase the rate of convergence in the L_2 sense is to increase the number of degrees of freedom per cell.

FIGURE 59. Wave propagating to the right on a 2D grid.

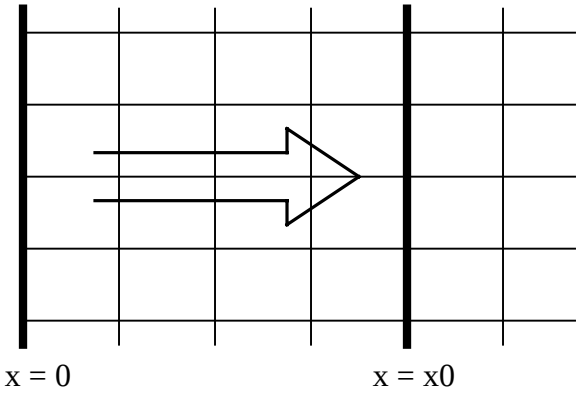
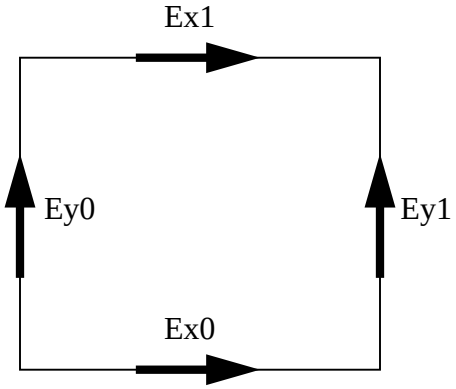


FIGURE 60. Electric field within a single cell.



There are several other measures of error that are applicable for this specific waveguide problem. For example, it is possible to compare the computed impedance to the exact impedance, where the impedance for the TE_{10} mode is defined as

$$Z = E_z / H_y . \tag{236}$$

The exact impedance is $Z = \omega\mu / \beta_z$, which is a constant in the guide. Since the computed fields are “noisy”, the computed impedance is defined to be the average impedance

over the entire guide. Another measure is to compare the computed wavelength to the exact wavelength. The wavelength is computed by fitting a sine wave to the magnitude of the electric field, the period of best fit sine wave defining the wavelength of the electric field. The exact wavelength for this problems is simply $\lambda_z = 2\pi/\beta_z$.

Finally, another measure that is applicable to this wave guide problem is the voltage standing wave ratio (VSWR). The VSWR is defined as

$$VSWR = \frac{|E_{max}|}{|E_{min}|}, \tag{237}$$

where $|E_{max}|$ is the maximum of the time average electric field in the waveguide, and $|E_{min}|$ is the minimum of the time average electric field in the waveguide. For an infinite waveguide, or a perfectly terminated finite length waveguide, the VSWR is 1.0. For a terminated waveguide, the VSWR can be expressed as a function of the reflection coefficient of the termination

$$VSWR = \frac{1 - |\rho|}{1 + |\rho|}, \tag{238}$$

where ρ is the reflection coefficient. The VSWR was computed by determining the maximum and minimum fields over one period, and the reflection coefficient is then computed from

$$|\rho| = \frac{1 - VSWR}{1 + VSWR}. \tag{239}$$

The reflection coefficient is a measure of how effective the PML is. If the PML was in fact perfect, the reflection coefficient would be zero. The error according to the L_2 norm, impedance, wavelength, VSWR, and reflection coefficient are shown in Table 26. The computed electric and magnetic fields are shown in Figure 61 and Figure 62 for the 5000 cell waveguide.

TABLE 26. Quality of computed fields for PML terminated waveguide.

Δ / a	L_2	**impedance**	**wavelength**	**VSWR**	**reflection coefficient**
1/10	17.38%	2.713%	0.453%	1.057	-31dB

FIGURE 61. Computed z component of electric field in PML terminated waveguide.

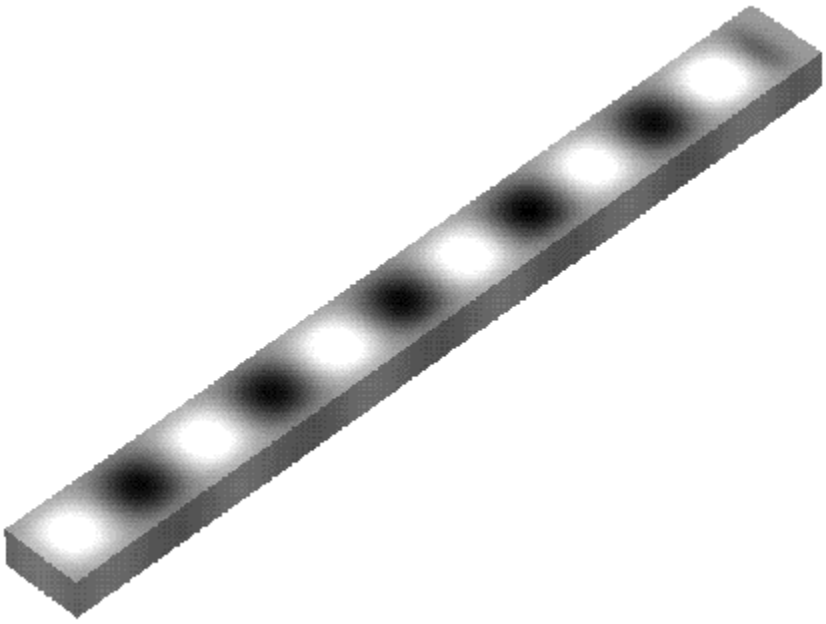
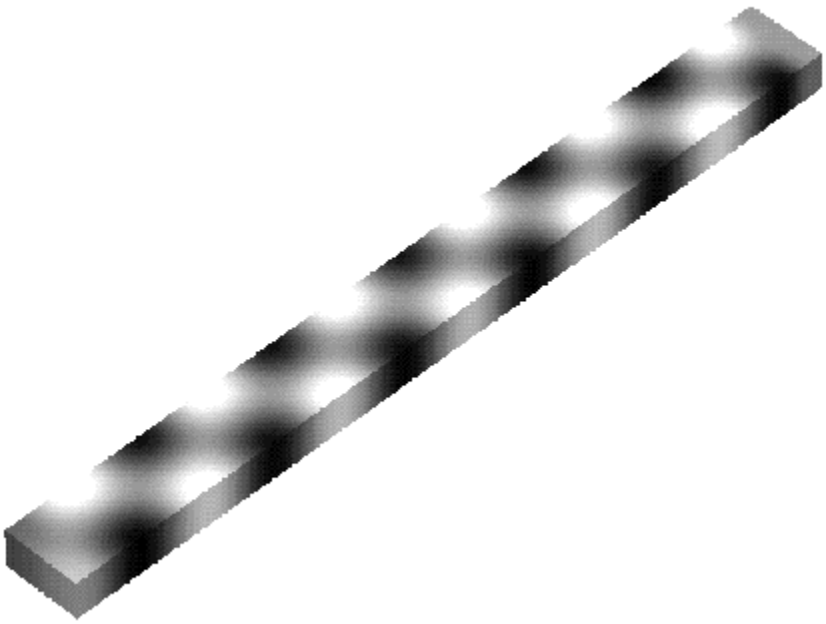


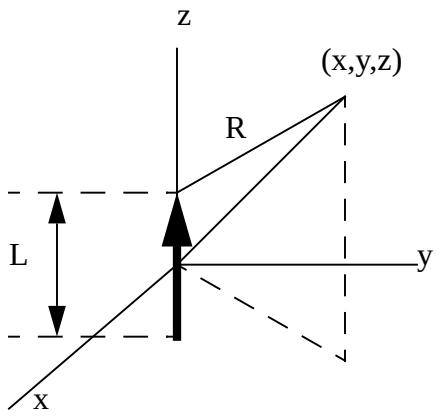
FIGURE 62. Computed x component of magnetic field in PML terminated waveguide.



8.4 Dipole Antenna

In this section, the radiated fields due to a small current source are computed using VFEM3D and compared to the exact analytical solution. Let the current be at the origin and aligned in the $\hat{z}$ direction, and let the observation point be at (x, y, z) as illustrated in Figure 63. The current is oscillating at frequency ω.

FIGURE 63. Coordinate system for dipole radiation calculation.



The exact frequency domain solution is given by

$$\vec{H} = \frac{1}{\mu}\left(\hat{x}\frac{\partial A_z}{\partial y} - \hat{y}\frac{\partial A_z}{\partial x} \right)\cos(\omega t + \theta),$$
$$\vec{E} = \frac{1}{\omega\mu\varepsilon}\left(\hat{x}\frac{\partial\,\partial A_z}{\partial x \partial z} + \hat{y}\frac{\partial\,\partial A_z}{\partial y \partial z} - \hat{z}\left(\frac{\partial\,\partial A_z}{\partial x \partial x} + \frac{\partial\,\partial A_z}{\partial y \partial x} \right) \right)\sin(\omega t + \theta), \tag{240}$$

where the z component of the complex magnetic vector potential A is

$$A_z = \frac{\mu I}{4\pi}\int_{-L/2}^{L/2} \frac{\exp(-j\beta R)}{R}\,dz, \tag{241}$$

and $\theta = \arg(A)$. The integral in (241) was evaluated numerically using Gaussian quadrature.

The parameters for this computational experiment were $\omega = 107.3132$ and $L = \lambda/12 = 0.00487916$. The problem was modeled using a hemispherical grid con-

sisting of 12032 hexahedral cells and 38005 edges. A hemisphere was used since the radiated fields are symmetric with respect to the $z = 0$ plane, thus a perfectly conducting ground plane with a $L/2$ current source generates the same fields as a current source of length L in free space. The grid had a spacing of $h = \lambda/24 = 0.00243972$ at the origin, and the grid spacing increases away from the origin. The current source is exactly two edge lengths long. The current source used in the computation is given by

$$I(t) = \left(1 - \exp\left(-\left(\frac{t}{2T}\right)^2\right)\right)\sin(\omega t), \tag{242}$$

where $T = 0.0147$. The simulation was run for 0.05855 seconds using a time step of $\Delta t = 0.0001$ seconds, which corresponds to 585 time steps.

In order to simulate free space, the same 5 layer PML used for the waveguide in Section 8.3 was used for this problem, except that the conductivity tensor is rotated such that the axial direction corresponds to the radial direction. This PML will absorb outgoing waves, or waves normal to the outer surface of the grid. The PML began at radius $a = \lambda = 0.05855$ and the grid was terminated at $b = 1.5\lambda = 0.087825$. The grid is shown in Figure 64. The relative L_2 error was computed in the same manner as for the waveguide, i.e., according to (234) where the sum is over all cells excluding PML cells. The computed electric field matched the exact electric field to within 1.6% using the L_2 error criteria. This is an excellent result since the electromagnetic field structure is quite complicated in the near field of the antenna. Snapshots of the computed electric and magnetic field are shown in Figure 65 and Figure 66 respectively.

FIGURE 64. Illustration of hexahedral grid with 5 layer PML used for dipole calculation.

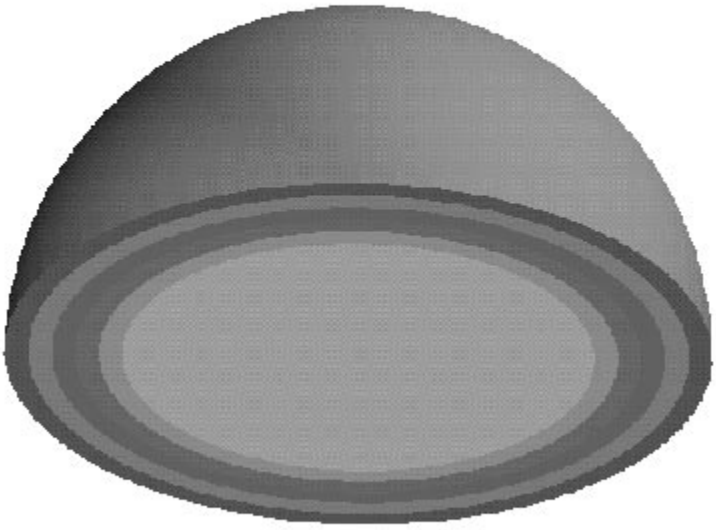


FIGURE 65. Computed electric field magnitude in vicinity of $\lambda/12$ dipole.

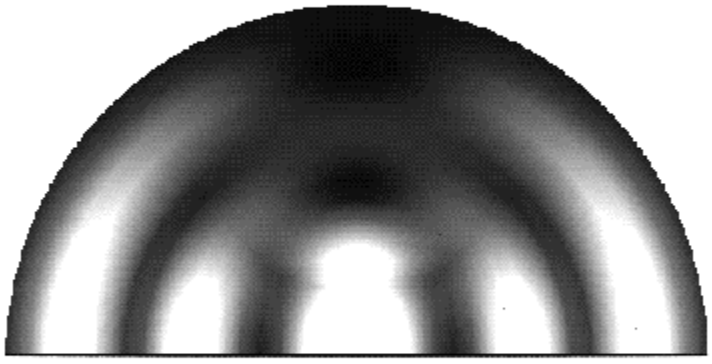
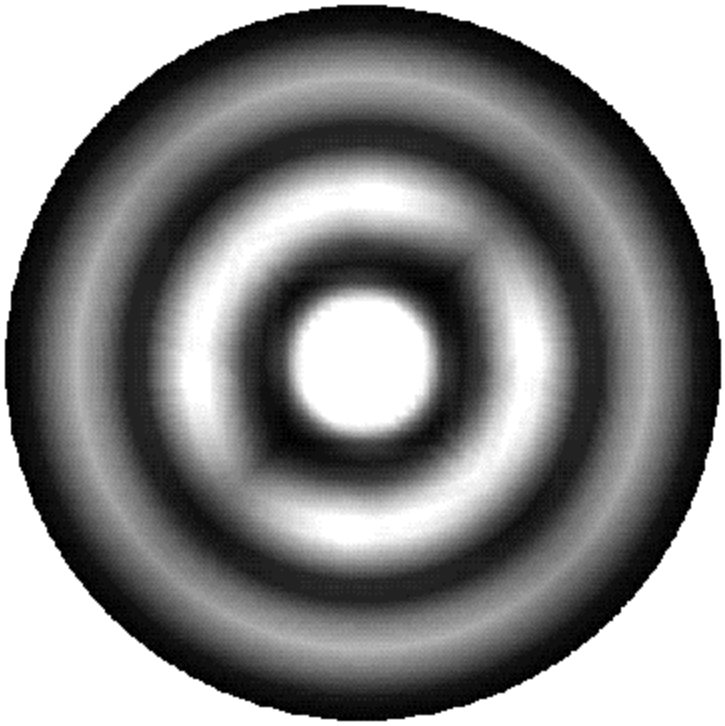


FIGURE 66. Computed magnetic field in vicinity of $\lambda/12$ dipole.



8.5 Parallel Results

Two different parallel computers were used, the Meiko CS-2 and the Cray T3D. The Meiko CS-2 consists of 48 computational nodes. Each computational node consists of dual 90 MHz SPARC microprocessors (90 MFLOPS peak theoretical), 128 MB of memory, and a one 1-GB hard disk. The computational nodes communicate with each other over a 50 MB/s peak network. No explicit effort was made to take advantage of the dual processors on each node.

The Cray T3D system consists of the 256 processor T3D and a 3 processor YMP front end. The T3D processors are 150 MHz DEC Alpha microprocessors (150 MFLOPS peak theoretical). Each processor has 8 MW of memory. The T3D processors communicate with each other over a 140 MB/s peak network. The Parallel Virtual Machine (PVM) ver-

sion 3 message passing library was used to perform all of the communication tasks on both the Meiko and the Cray. No non-standard PVM extensions were used. It should be noted that complicated data structures are required to allow for arbitrary partitioning of arbitrary grids, and the data structures used in VFEM3D may not be optimal. Thus the results shown below should not be considered the best than can be achieved.

The test problem consisted of simple sphere excited by a pulsed current source. Two different grids were used, a tetrahedral grid with 62618 cells and an hexahedral grid with 169440 cells. The grids are not shown since they were too large. As described in Section 7.2, domain decomposition is used on the grid. The actual RSB method used to decompose the grid is described in [72]. Each grid was decomposed into a power-of-two number of sub grids from 1 to 32. The computer CPU time for the Meiko CS-2 is tabulated in Table 27 and Table 28 for both grids as a function of the number of processors. This CPU time is for the time stepping part of the code only, it does not include time required for computing the matrices. The matrix calculation is trivially parallel thus there is no need to experimentally determine the parallel efficiency.

TABLE 27. Meiko performance on 62618 cell tetrahedral grid.

# processors	**1**	**2**	**4**	**8**	**16**	**32**
JCG iter	20.7	20.7	20.7	20.7	20.7	20.7
JCG cpu sec	271	121.2	54.2	27.8	18.1	11.7
ICCG iter	8.1	14.4	15.3	16.2	17.1	17.1
ICCG cpu sec	87.9	46.6	43.8	20.1	13.6	8.86

TABLE 28. Meiko performance on 169440 cell tetrahedral grid.

# processors	**1**	**2**	**4**	**8**	**16**	**32**
JCG iter	35.5	35.5	35.5	35.5	35.5	35.5
JCG cpu sec	1097	532	274	145	83.5	46.3
ICCG iter	7.2	8.1	19.09	21.8	22.5	23.1
ICCG cpu sec	453	205	190	134	77.4	45.5

The data in Table 27 and Table 28 indicate that the number of Jacobi preconditioned CG iterations is independent of the number of processors used, as expected. However, the number of block ICCG iterations grows as the number of processors is increased. As shown in Section 7.3, the form of this preconditioner is dependent upon the domain decomposition. Use of block incomplete Cholesky preconditioner reduces the number of CG iterations, and hence reduces communication, but at the expense of increased work per iteration. For four or fewer processors, block ICCG is faster than Jacobi CG, for eight or more processors the CPU time is virtually identical. The absolute CPU time and the speedup curves for the Meiko are shown in Figure 67 - Figure 70.

FIGURE 67. Meiko CPU time on 62618 tetrahedral grid: Jacobi CG vs. block ICCG.

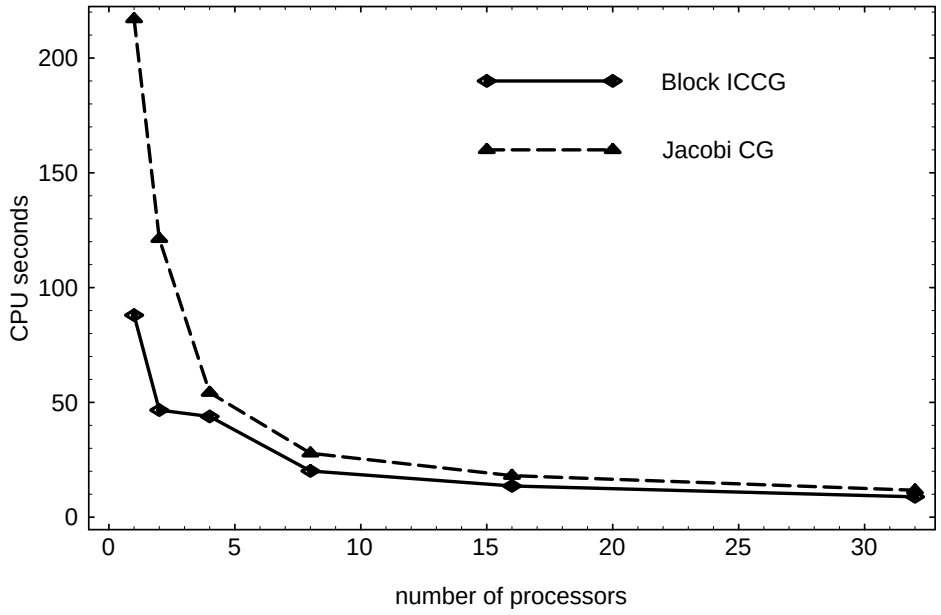


FIGURE 68. Meiko CPU time on 169440 hexahedral grid: Jacobi CG vs. block ICCG.

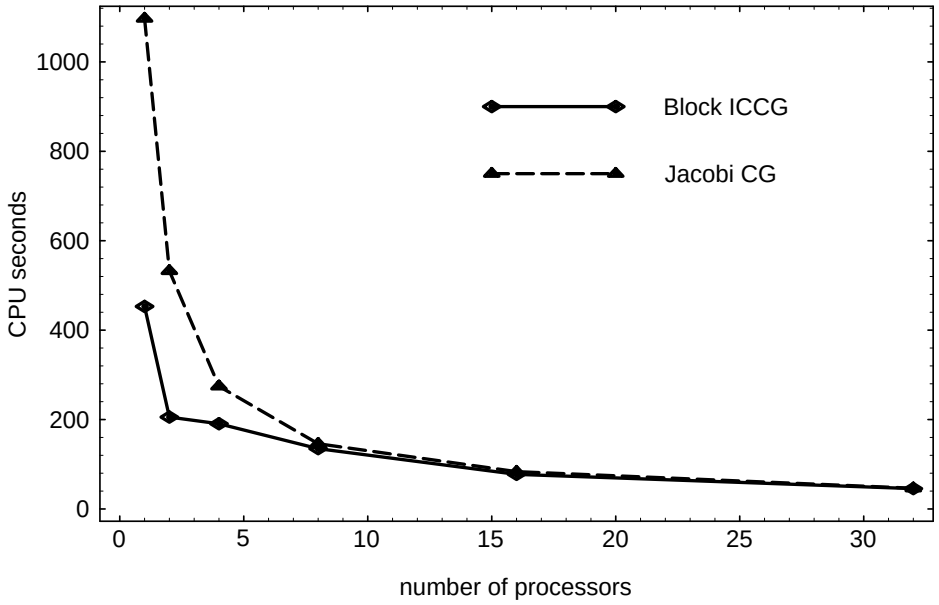


FIGURE 69. Meiko speedup: block ICCG vs. Jacobi CG on 62618 tetrahedral grid.

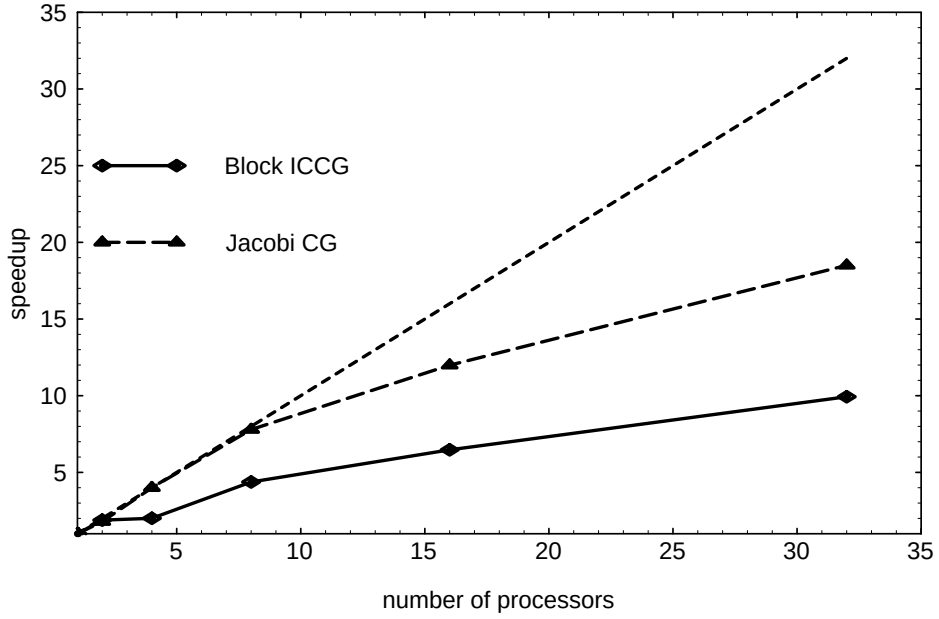
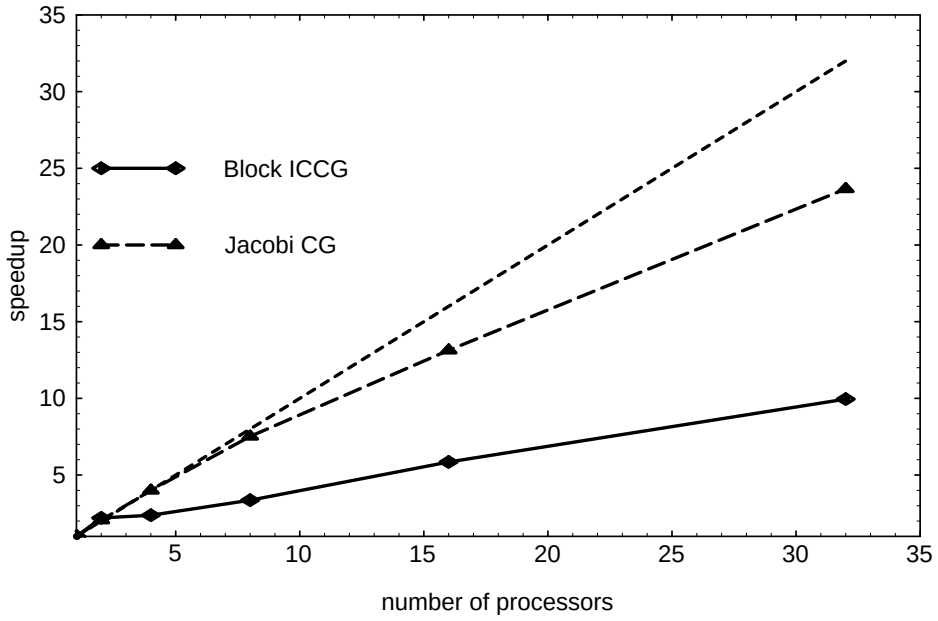


FIGURE 70. Meiko speedup: block ICCG vs. Jacobi CG on 169440 hexahedral grid.



The same computational experiments were performed on the Cray T3D, with the results shown in Table 27 and Table 28. In general, the Cray T3D was faster then the Meiko CS-2 for a given number of processors. However, block ICCG was not as effective on the Cray as it was on the Meiko. This is due to the fact that the Cray has faster communication, reduction in the number of CG iterations, with a consequent reduction in the amount of message passing, is not as significant on the Cray as on the Meiko. Another factor could be that the block IC preconditioner requires both a forward and backward substitution on an unstructured data structure. This process does not vectorize well on the Cray, hence, it costs more to perform the block IC preconditioning on the Cray than on the Meiko. In

other words, on the Cray, it is better to perform many fast iterations than fewer slow iterations.

TABLE 29. Cray performance on 62618 cell tetrahedral grid.

# processors	1	2	4	8	16	32
JCG iter	20.7	20.7	20.7	20.7	20.7	20.7
JCG cpu sec	271	121.2	54.2	27.8	18.1	11.7
ICCG iter	8.1	14.4	15.3	16.2	17.1	17.1
ICCG cpu sec	87.9	46.6	43.8	20.1	13.6	8.86

TABLE 30. Cray performance on 169440 cell tetrahedral grid.

# processors	1	2	4	8	16	32
JCG iter	35.5	35.5	35.5	35.5	35.5	35.5
JCG cpu sec	1097	532	274	145	83.5	46.3
ICCG iter	7.2	8.1	19.09	21.8	22.5	23.1
ICCG cpu sec	453	205	190	134	77.4	45.5

The absolute CPU time and the speedup curves for the Cray T3D are shown in Figure 71 - Figure 74. While the absolute CPU time for the Cray is less than for the Meiko, the speedup curves are essentially the same. As discussed in Section 7.1 the causes of non-optimal speedup include: 1) a portion of the algorithm may be inherently serial, 2) a load imbalance, 3) interprocessor communication, and 4) extra work. The portion of VFEM3D that is inherently serial is quite small and is negligible compared to the other factors. Load imbalance is an important factor in the above results. For example, on the 32 processor calculation on the 62618 cell tetrahedral grid the maximum number of degrees of freedom was 2380, while the minimum number was 2305. Also, some processors sent more messages than others, the minimum number of messages was 6 and the maximum was 12. Thus, there is a slight load imbalance in the amount of floating point calculations performed on each processors, and another imbalance in the amount of time spent sending/receiving messages. The amount of load imbalance is dependent upon the domain decom-

position. As mentioned in Section 7.2, VFEM3D allows for arbitrary partitioning of a grid, the partitioning of the grid is described by a color file. The RSB algorithm was used to generate the color file. Other algorithms may generate a better partitioning of the grid, this was not investigated. While the load imbalance is the same for the two machines the actual time spend sending/receiving messages is different since the communication hardware and software is different. The actual time spent sending/receiving messages could be significantly reduced by using a different communication library such as MPI or SCHMEM, but this was not investigated. There is also a significant amount of extra work required in order to perform the message passing. For example, it is necessary to figure out exactly which subdomains share information and which information they share. All of this message passing information is computed once and stored in a data structure, but this data structure must be accessed every time message passing is to occur. This is an example of extra work. As the number of processors increases, the amount of work done by each processor decreases, and the amount of extra work and communication increases. This is exemplified by the fact that the speedup for the 169440 cell grid is much better than for the smaller 62618 cell grid.

FIGURE 71. Cray CPU time on 62618 tetrahedral grid: Jacobi CG vs. block ICCG.

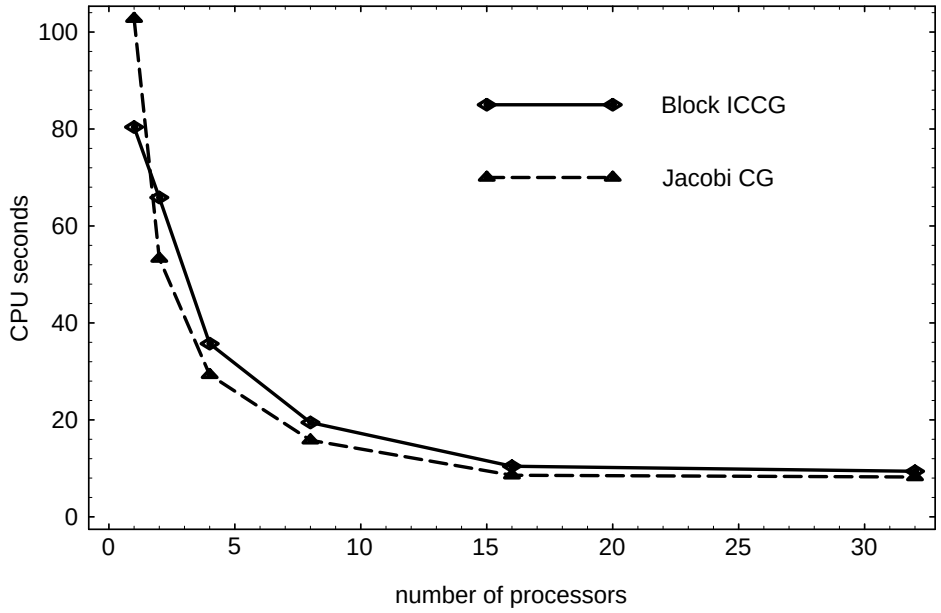


FIGURE 72. Cray CPU time on 62618 tetrahedral grid: Jacobi CG vs. block ICCG.

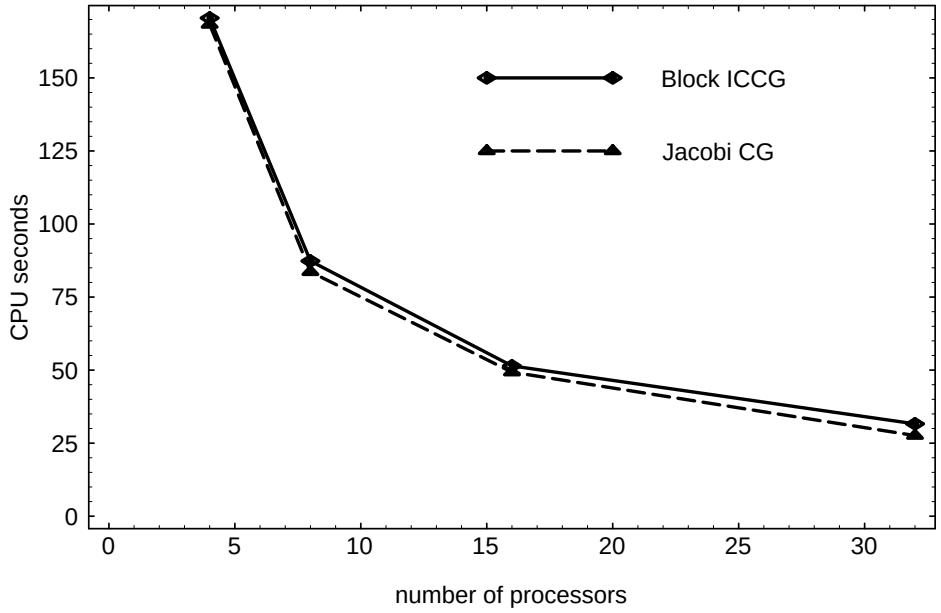


FIGURE 73. Cray speedup: block ICCG vs. Jacobi CG on 62618 tetrahedral grid.

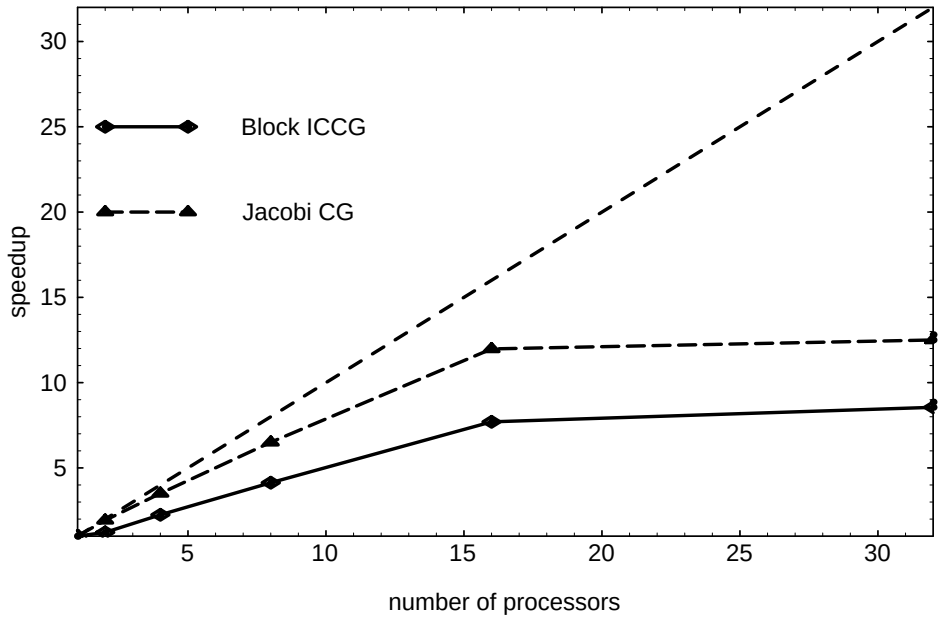
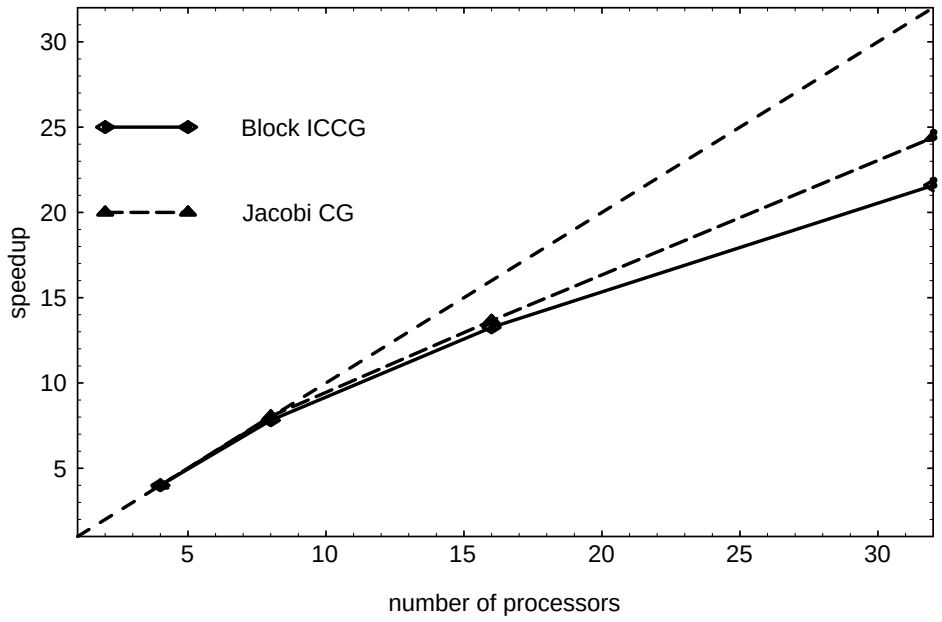


FIGURE 74. Cray speedup: block ICCG vs. Jacobi CG on 169440 hexahedral grid.



9.0 Conclusion

9.1 Summary of the method

The primary goal of this research effort was to develop a “FDTD-like” method that gives physically reasonable results for any grid. In this dissertation, physically reasonable means that the method is stable, energy is conserved, charge is conserved, and the continuity/discontinuity of the electromagnetic fields across a material interface are modeled properly. In this dissertation a method, called the Discrete Time Vector Finite Element Method (DTVFEM) is derived, analyzed, and validated. The DTVFEM uses covariant vector finite elements as a basis for the electric field and contravariant vector finite elements as a basis for the magnetic flux density. These elements are complementary in the sense that the covariant elements have tangential continuity across interfaces whereas the contravariant elements have normal continuity across interfaces. The Galerkin approximation is used to convert Ampere’s and Faraday’s law to a coupled system of ordinary differential equations (ODE). The leapfrog method is used to advance the fields in time.

By construction the DTVFEM correctly models the jump discontinuity of electromagnetic fields across material discontinuities. In Section 5.0, it is proved that the DTVFEM method is conditionally stable, and if the time step is chosen such that the method is stable, it will conserve energy and conserve charge independent of how coarse the grid is. In addition a numerical dispersion analysis indicates that the method is second order accurate, even on distorted, but regular, three dimensional hexahedral grids. However, like most finite element methods, the DTVFEM requires that a sparse linear system be solved

at every time step. This is a disadvantage compared to FDTD and FVTD methods. In Section 6.0, the solution of large, sparse, unstructured matrices that arise in the DTVFEM is discussed. It is shown that for a Cartesian grid there are two options: 1) capacitance lumping can be employed in which case the DTVFEM reduces to the classic FDTD method, or 2) the capacitance matrix can be inverted exactly, using a direct Cholesky decomposition, in $O(n)$ operations. For unstructured grids, iterative methods must be employed. Several iterative methods were investigated, the most useful being stationary iteration or conjugate gradient. The incomplete Cholesky decomposition was investigated as a preconditioner for both the stationary iteration and the conjugate gradient method. The computational effort required to solve the system depends upon how distorted the grid is. In Section 7.0, parallelization via domain decomposition is reviewed.

The DTVFEM has a combination of attributes not shared by other unstructured grid, time-domain methods for solving Maxwell's equation. Specifically, the DTVFEM

- is valid for unstructured grids
- allows for tensor permittivity, permeability, and conductivity
- correctly models field continuities/discontinuities across material interfaces
- reduces to FDTD for Cartesian grids
- is conditionally stable
- is energy conserving
- is charge preserving
- is 2nd order accurate in the dispersion relation sense

- is 1st order accurate in the L_2 sense when using linear finite elements.

9.2 Summary of the results

Only so much can be proved about the DTVFEM; at some point it is necessary to implement the method in software and perform computational experiments. The software developed under this research effort is referred to as VFEM3D. It is not really a single computer program, but actually a suite of programs. The software has been installed on a variety of computers including a Apple Macintosh IIci, HP/SUN/SGI Unix workstations, and two parallel supercomputers.

In Section 8.0, VFEM3D is validated by comparing computed solutions to analytical solutions for a simple resonant cavity, waveguide, and antenna. The accuracy and computer CPU time is tabulated for a variety of different grids. It is established that the DTVFEM is second order accurate in the dispersion relation sense and first order accurate in the L_2 norm sense, which is comparable to the classic FDTD method.

Several computational experiments are performed in order to investigate how the specific method used to solve the linear system impacts the accuracy and the efficiency of the method. It is shown that for a uniform Cartesian grid capacitance lumping works well, hence the popularity of the classic FDTD method for such grids. For general hexahedral or tetrahedral grids, iterative methods are used to solve the linear system. It is shown that as the grid is refined the number of iterations remains constant, hence the DTVFEM is of $O(n)$ complexity. The number of iterations is dependent upon how distorted the grid is

and is problem dependent; the spherical cavity using a hexahedral grid required 7.8 ICCG iterations on average whereas the chevron waveguide using a hexahedral grid required 5.7 ICCG iterations. While the DTVFEM method is $O(n)$, it is more computationaly intensive then other finite difference and finite volume methods that do not require the solution of linear systems at all.

It is also shown that the recently developed PML concept can be used to approximate an infinite space using a finite grid. Since VFEM3D allows for arbitrary tensor material properties, the PML concept is trivial to implement. While the PML concept worked well for the problems investigated in this dissertation, other approaches may be more accurate and/or more efficient, this was not investigated.

VFEM3D was tested on two MIMD distributed memory computers, the Meiko CS-2 and the Cray T3D. Domain decomposition is used to decompose the spatial domain into sub-domains. The RSB algorithm is used to perform the composition. VFEM3D follows the SPMD paradigm where each processor executes the same program, but on different grids. Each processor communicates with other processors via message passing. The PVM message passing library was used on both the Meiko and the Cray. The crux of the parallelization is the solution of the linear system. The computer CPU time versus number of processors was examined. The point Jacobi preconditioned conjugate gradient method scaled well, with the standard result that if the surface-to-volume ratio of the sub-domains was small, the parallel speedup was good. It was also shown that for small number of processors, block incomplete Cholesky preconditioned conjugate gradient was very effective. As the number of processors increases the performance reduced to that of point Jacobi

preconditioned conjugate gradient. This is because the preconditioner is dependent upon the decomposition, and the number of iterations grows as the number of processors increases.

9.3 Future research

The DTVFEM described in this dissertation is by no means the “last word” in computational electromagnetics. However, it is definitely a step in the right direction since it works as well on random unstructured grids as it does on Cartesian grids. There are many questions that remain to be answered. Like other finite element methods the DTVFEM would benefit from a faster sparse matrix solver. Perhaps approximate inverse methods [83][84] or multigrid methods [85] could be used to speed up the linear solve portion of the DTVFEM.

In this research effort, the materials were restricted to be linear and non-dispersive. However there are many interesting and important applications that require accurate modeling of nonlinear and dispersive materials. Dispersive material models have been incorporated into the FDTD framework [86][87], and nonlinear material models have been investigated also [88][89]. These same models could in theory be incorporated into the DTVFEM framework.

Finally the most intriguing future research direction is the incorporation of charged particles, either discrete or fluid, into the method. The addition of charged particles into the problem means the electric field is no longer divergence free everywhere, but only divergence free in cells with no net charge. The magnetic field, on the other hand, remains

divergence free for all time. Several researchers have proposed finite element particle-in-cell methods and finite element fluid plasma methods [90]-[92]. Methods that model the fields using traditional nodal elements will have problems similar to those encountered when solving Maxwell's equations; charge might not be conserved, energy might not be conserved, and boundary conditions might not be modeled appropriately. Perhaps the electric field should be decomposed into solenoidal (divergence free) and irrotational (curl free) components, with the solenoidal component approximated by covariant edge elements and the irrotational component approximated by contravariant face elements. The charge density would be approximated by discontinuous, i.e. three-form, volume elements. Of course, the Lorentz force law would have to be included to move the charge around, and the charge position and velocity would be updated in a leapfrog fashion just like the electric and magnetic fields are. Such a method could, in theory, conserve everything that is supposed to be conserved independent of how random or coarse the underlying unstructured grid is.

10.0 References

1. Yee K.S.,"Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," IEEE Trans. Ant. Prop. 14(3),302,1966.
2. Taflove A. and Brodwin M.E.,"Numerical solution of steady-state electromagnetic scattering problems using the time-dependent Maxwell's equations," IEEE Trans. Microwave Theory Tech.,23,623-630,1975.
3. Taflove A.,"Review of the formulation and applications of the Finite Difference Time Domain method for numerical modelling of electromagnetic wave interactions with arbitrary structures," Wave Motion,10,547-582,1988
4. Kunz K.S and Luebbers R.J,*The Finite Difference Time Domain Method for Electromagnetics, CRC* Press, Boca Raton, Florida,1993.
5. Holland R.,"The case against staircasing," *Proceedings of the 6th annual Review of Progress in Applied Computational Electromagnetics*,89-95,March 1990.
6. Madsen N. and Ziolkowski R.,"A 3 dimensional modified finite volume technique for Maxwell's equations," Electromagnetics, v. 10, n. 1, 147-161, 1990.
7. Madsen N.,"Divergence preserving discrete surface integral method for Maxwell's curl equations using non-orthogonal unstructured grids," Journal of Computational Physics, v. 119, n. 1,34-45,1995.
8. Holland R., Cable V., and Wilson L.," Finite-volume time-domain techniques for EM scattering," IEEE Trans. Electromagnetic Compatibility, v. 33, n. 4, 281-294, 1991.
9. Yee K. S. and Chen J. S., "Conformal hybrid finite difference time domain and finite volume time domain," IEEE Trans. on Antenna and Propagation, v. 42, n. 10, 1450-1454, 1994.
10. Madsen, N., personal correspondence.
11. Riley D. J. and Turner C. D., "VOLMAX: A solid model based transient volumetric Maxwell solver using hybrid grids," IEEE Antennas and Propagation Magazine, v. 39. n. 1, 20-33, 1997.
12. Shankar V.,Hall W., and Mohammadian A., "A time-domain differential solver for electromagnetic scattering problems," Proceeding of the IEEE, v. 77, n. 5, 709-721,1989.
13. Shankar V., Mohammadian A., and Hall W., "A time-domain finite-volume treatment for the Maxwell equations," Electromagnetics, v. 10, n. 1, 127-145, 1990.
14. Noack R. W. and Anderson D. A., "Time-domain solutions of Maxwell's equations using a finite-volume formulation," AIAA paper 92-0451, 1992.
15. Anderson O.W.,"Laplacian electrostatic field calculations by finite elements with automatic grid generation," IEEE Trans. Power Apparatus Syst.,92(5),1485-1492,1973.
16. Silvester P.P. and Ferrari R.L.,*Finite Elements for Electrical Engineers, Cambridge* University Press, Cambridge,1983.

17. Jin J.,*The Finite Element Method in Electromagnetics, John* Wiley & Sons, New York,1993.

18. Bossavit A., “Solving Maxwell equations in a closed cavity, and the question of spurious modes,” IEEE Trans. on Magnetics, v. 26, n. 2, 702-705, 1990.

19. Cendes Z. J., “Vector finite elements for electromagnetic field calculations,” IEEE Transactions on Magnetics, v. 27, n. 5, 3958-3966, 1991.

20. Lynch D. R., Paulsen K. D., and Boyse W. E., “Synthesis of vector parasites in finite element Maxwell solutions,” IEEE Trans. Microwave Theory and Tech., v. 41, N. 8, 1439-1447, 1993.

21. Dillon B. and Webb J. P., “A comparison of formulations for the vector finite element analysis of waveguides,” IEEE Trans. on Microwave Theory and Tech., v. 42, n. 2, 308-316, 1994.

22. Sun D., Magnes J., Yuan X., and Cendes Z., “Spurious modes in finite element methods,” IEEE Antenna and Propagation Magazine, v. 37, n. 5, 12-24, 1995.

23. Lee R. L. and Madsen N. K., “A mixed finite element formulation for Maxwell’s equations in the time domain,” Journal of Computational Physics, v. 88, 284-304, 1990.

24. Ambrosiano J. J., Brandon S. T., Lohner R., and DeVore C. R., “Electromagnetics via the Taylor-Galerkin finite element method on unstructured grids,” Journal of Computational Physics, v. 110, 310-319, 1994.

25. Ray S. L., “Grid decoupling in finite element solutions of Maxwell’s equations,” IEEE Trans. Antennas and Propagation, V. 40, N. 4, 443-445, 1992.

26. A. C. Cangellaris, C. C. Lin, and K. K. Mei, “Point-matched time domain finite element methods for electromagnetic radiation and scattering,” IEEE Trans. Antennas and Propagation, v. 35, 1160-1173, 1987.

27. D. R. Lynch and K. D. Paulsen, “Time domain integration of the Maxwell equations on finite elements,” IEEE Trans. Antennas and Propagation, v. 38, 1933-1942, 1990.

28. Nedelec J. C.,”Mixed finite elements in R3,” Numer. Math,35,315-341,1980.

29. Nedelec J. C.,”A New Family of Mixed Finite Elements in R3,” Numer. Math, 50, 57-81,1986.

30. Bossavit A.,”Whitney forms: a class of finite elements for three-dimensional computations in electromagnetism,” IEE Proceedings, v. 135, pt. A, n. 8,493-500,1988.

31. Bossavit A. and Mayergoyz I.,”Edge elements for scattering problems,” IEEE Trans. Mag. 25(4),2816-2821,1989.

32. Lee J. F., Sun D. K., and Cendes,Z.,”Tangenetial vector finite elements for electromagnetic field computation,” IEEE Trans. Mag.,27(5),4032-4035,1991.

33. Lee J.,Sun D., and Cendes Z.,”Full-wave analysis of dielectric waveguides using tangential vector finite elements,” IEEE Trans. Microwave Theory Tech.,39(8),1262-1271,1991.

34. Anderson, B., and Cendes, Z.,"Solution of ferrite loaded waveguide using vector finite elements," IEEE Trans. Mag.,31(3),1578-1581,1995.

35. Crain B. and Peterson A.,"Analysis of propagation on open microstrip lines using mixed-order covariant projection vector finite elements," Int. J. Microwave and Millimeter-Wave CAD,5(20),59-67,1995.

36. K. Mahadevan and R. Mittra, "Radar cross section computation of inhomogeneous scatterers using edge based finite element method in time and frequency domains," Radio Science, v. 28, n. 6, 1181-1193, 1993.

37. Mahadevan K., Mittra R. and Vaidya P. M., "Use of Whitney's edge and face elements for efficient finite element time domain solution of Maxwell's equations," Journal of Electromagnetic Waves and Appl., v. 8, n. 9/10, 1173-1191, 1994.

38. J. F. Lee and Z. S. Sacks, "Whitney element time domain methods," IEEE Trans. Magnetics, V. 31, 1325-1329, 1995.

39. J. F. Lee, R. Lee, and A. Cangellaris, "Time domain finite element methods," IEEE Trans. Antennas and Propagation, v. 45, n. 3, 430-442, 1997.

40. Lee J.,"WETD - A finite element time domain approach for solving Maxwell's equations," IEEE Microwave Guided Wave Letters, 4(1), 11-13, 1994.

41. Monk P. and Parrot A.,"A dispersion analysis of finite element methods for Maxwell's equations," SIAM J. Sci. Comput.,15(4),916-937,1994.

42. Warren G. and Scott W.,"Numerical dispersion in the finite elements method using triangular edge elements," Microwave Optical Tech. Letters,9(6),315-319,1995.

43. White D. and Rodrigue G. "Improved Vector FEM solutions of Maxwell's equations using grid preconditioning," International Journal for Numerical Methods in Engineering (submitted).

44.Warren G. and Scott W., "An investigation of numerical dispersion in the vector finite element method using quadrilateral elements," IEEE Trans. on Antennas and Propagation, v. 42, n. 11, 1502-1508, 1994.

45. Ramo S., Whinnery J. R., and Van Duzer T., *Fields and Waves in Communication Electronics*, John Wiley and Sons, New York, 1965.

46. Balanis C., *Advanced Engineering Electromagnetics*, John Wiley and Sons, New York, 1989.

47. Stratton J. D., *Electromagnetic Theory*, McGraw Hill, New York, 1941.

48. Landau L. D. and Lifshitz E. M., *Electrodynamics of Continuous Media*, Peragom Press, Oxford, 1960.

49. Jackson J. D., *Classical Electrodynamics*, John Wiley and Sons, New York, 1962.

50. Kong J. A., *Electromagnetic Wave Theory*, John Wiley and Sons, New York, 1986.

51. Courant R. and Hilbert D., *Methods of Mathematical Physics Volume 2*, John Wiley and Sons, New York, 1962.

52. Collatz L., *Functional Analysis and Numerical Mathematics*, Academic Press, New York, 1966.

53. Brenner S. C. and Scott L. R., *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, New York, 1994.

54. Strang G. and Fix G, *An analysis of the Finite Element Method*, Prentice-Hall, New Jersey, 1973.

55. Konrad A., "Vector variational formulations of electromagnetic fields in anisotropic media," IEEE Transactions on Microwave Theory and Tech., v. 24, 553-559, 1976.

56.Whitney, H., *Geometric Integration Theory*, Princeton University Press, 1957.

57. K. D. Paulsen and D. R. Lynch, "Elimination of vector parasites in finite element Maxwell solutions," IEEE Trans. Microwave Theory and Tech., v. 39, 395-404, 1991.

58. B. Jiang, J. Wu, and L. A. Povinelli, "The origin of spurious solutions in computational electromagnetics," Journal of Computational Physics, v. 125, 104-123, 1996.

59. Falk, R. S.,"Nonconforming finite element methods for the equations of linear elasticity," Mathematics of Computation, v. 57, n. 196, 529-550, 1991.

60. Babushka, I. and Suri M.,"Locking effects in the finite element approximation of elasticity problems," Numerical Mathematics, v. 62, 439-463, 1992.

61. Babushka, I. and Suri M.,"On locking and robustness in the finite element method," Siam J. Numer. Math, v. 29, n. 5, 1261-1293, 1992.

62. Flanders H., *Differential Forms with Applications to the Physical Sciences*, Dover Publications, New York, 1963.

63. Burke W. *Applied Differential Geometry*, Cambridge University Press, Cambridge, 1985.

64. Deschamps G. A., "Electromagnetics and differential forms," Proceedings of the IEEE, v. 69, n. 6, 676-696, 1981.

65. Chan H. C., Cai C. W., and Cheung Y. K., 'Convergence studies of dynamic analysis by using the finite element method with lumped mass matrix," Journal of Sound and Vibration, v. 165, n. 2, 193-207, 1993.

66. Golub G. H. and Van Loan C. F., *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 1989.

67. R. Richtmyer and K. Morton, *Difference Methods for Initial Value Problems*, John Wiley & Sons, New York, 1976.

68. Hoffman J. D., *Numerical Methods for Engineers and Scientists*, McGraw-Hill, New York, 1992.

69. Gear W., *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, New York, 1971.

70. Kumar V., Grama A., Gupta A., and Karypis G., *Introduction to Parallel Computing, Design and Analysis of Algorithms,* Benjamin/Cummings, Redwood City, California, 1994.

71. Amdahl G. M., “Validity of the single processor approach to achieving large scale computing capabilities,” AFIPS Conference Proceedings, 483-485, 1967.

72. Pothen A., Simon H., and Liou K. P., “Partitioning sparse matrices with eigenvectors of graphs,” SIAM J. Mat. Anal. Appl., v. 11, n. 3, 430-452, 1990.

73. Karypis G. and Kumar K., “Multilevel k-way partitioning scheme for irregular graphs,” TR 95-064, Dept. of Computer Science, University of Minnesota, 1995.

74. Karypis G. and Kumar K., “METIS: Unstructured graph partitioning and sparse matrix ordering system,” Dept. of Computer Science, University of Minnesota, 1995.

75. Berenger J. “A perfectly matched layer for the absorption of electromagnetic waves,” Journal of Computational Physics, v. 114,n. 2, 185-200,1994.

76. Berenger J. “Three-dimensional perfectly matched layer for the absorption of electromagnetic waves,” Journal of Computational Physics, v. 127,n. 2, 363-379,1996.

77. Pekel U. and Mittra R., “A finite element method frequency domain application of the PML concept,” Microwave and Optical Tech. Letters, v. 9, n.3,117-122, 1995.

78. Sacks Z., Kingsland D., Lee R., and Lee J., “A perfectly matched anisotropic absorber for use as an absorbing boundary condition,” IEEE Trans. Antennas and Propagation, v. 43, n. 12, 1460-1463, 1995.

79. Kuzuoglu M. and Mittra R., “Mesh truncation by perfectly matched anisotropic absorbers in the finite element method,” Microwave and Optical Tech. Letters, v. 12, n. 3, 136-140, 1996.

80. TrueGrid Manual, XYZ Scientific Applications Inc.

81. Joe B., “GEOMPACK - a software package for the generation of meshes using geometric algorithms,” Adv. Eng. Software, v. 13. 325-331, 1991.

82. Savage J. S. and Peterson A. F., “Higher order vector finite elements for tetrahedral cells,” IEEE Trans. Microwave Theory and Tech., v. 44, n. 6, 874-879, 1996.

83. Gravanis, G A., “The rate of convergence of explicit approximate inverse preconditioning,” Int. J. of Computer Math., v. 60, n.1, 77-89, 1996.

84. Benzi M., Meyer C. D., and Tuma M., “A sparse approximate inverse preconditioner for the conjugate gradient method,” SIAM J. on Scientific Computing, v. 17, n. 5, 1135-1149, 1996.

85. Chang Q. S., Wong Y. S., and Fu H. Q., “On the algebraic multigrid method,” J. of Computational Physics, v. 125, n.2 279-292, 1996.

86. Luebbers R. J. and Hunsberger F., “FDTD for nth-order dispersive media,” IEEE Trans. in Antennas and Propagation, v. 40, n. 11, 1297-1301, 1992.

87. Sullivan D. M., “Frequency dependent FDTD methods using Z transforms,” IEEE Trans. on Antennas and Propagation, v. 40, n. 10, 1223-1230, 1992.

88. Sullivan D. M., “Nonlinear FDTD formulations using Z transforms,” IEEE Trans. on Microwave Theory and Tech., v. 43, n. 3, 676-682, 1995.

89. Ziolkowski R. W. and Judkins J. B., “Full wave vector Maxwell equation modeling of the self focusing of ultrashort optical pulses in a nonlinear Kerr medium exhibiting a finite response time,” J. Opt. Soc. Am. B, v. 10, n. 2, 186-198, 1993.

90. Sonnendrucker E., Ambrosiano J. J., and Brandon S. T., “A finite element formulation of the Darwin PIC model for use on unstructured grids,” J. of Computational Physics, v. 121, n. 2, 281-297, 1995.

91. Cohen B. I., Barnes D. C., Dawson J. M, et. al, “The numerical tokamak project - simulation of turbulent transport,” Computer Physics Communication, v. 87, n. 1, 1-15, 1995.

92. Ramos J. I., Winowich N. S., “Finite difference and finite element methods for MHD channel flows,” Int. J. Numer. Meth. in Fluids, v. 11, n. 6, 907-934, 1990.

11.0 Appendix: Mathematica Scripts

11.1 Mathematica script for generating linear vector basis functions on a tetrahedron

```
(* generate and plot nedelec original linear elements on a tetrahedron*)
a = {a0,a1,a2,a3,a4,a5};
(* define edge polynomial space*)
pedge = {a[[1]] + a[[2]] y + a[[3]] z,
    a[[4]] - a[[2]] x + a[[5]] z,
    a[[6]] - a[[3]] x - a[[5]] y};
(*define face polynomial space*)
pface = {a[[1]] + a[[4]] x,a[[2]] + a[[4]] y,a[[3]] + a[[4]] z};
(* define normal vectors*)
n = {
{0,0,-1},
{0,-1,0},
{-1,0,0},
{1,1,1}/Sqrt[3]
};
(* define tangent vectors*)
t = {
{1,0,0},
{0,1,0},
{0,0,1},
{-1,1,0}/Sqrt[2],
{-1,0,1}/Sqrt[2],
```

```
{0,-1,1}/Sqrt[2]
};
(* construct the edge elements*)
eq = Table[0,{i,1,6}];
eq[[1]] = Integrate[x = s;y = 0;z = 0;pedge . t[[1]],{s,0,1}];
eq[[2]] = Integrate[x = 0;y = s;z = 0;pedge . t[[2]],{s,0,1}];
eq[[3]] = Integrate[x = 0;y = 0;z = s;pedge . t[[3]],{s,0,1}];
eq[[4]] = Integrate[x = 1-s/Sqrt[2];y = s/Sqrt[2];z = 0;pedge . t[[4]],{s,0,Sqrt[2]}];
eq[[5]] = Integrate[x = 1-s/Sqrt[2];y = 0;z = s/Sqrt[2];pedge . t[[5]],{s,0,Sqrt[2]}];
eq[[6]] = Integrate[x = 0;y = 1-s/Sqrt[2];z = s/Sqrt[2];pedge . t[[6]],{s,0,Sqrt[2]}];
A = Table[0,{i,1,6},{j,1,6}];
Do[
  Do[
    A[[i,j]] = Coefficient[Expand[Simplify[eq[[i]]]],a[[j]]],
  {j,1,6}],
{i,1,6}];
W = Table[0,{i,1,6}];
Do[
  rhs = Table[0,{j,1,6}];
  rhs[[i]] = 1;
  s = LinearSolve[A,rhs];
  x  = .;y = .;z = .;
  W[[i]] =  {s[[1]] + s[[2]] y + s[[3]] z,
        s[[4]] - s[[2]] x + s[[5]] z,
        s[[6]] - s[[3]] x - s[[5]] y},
```

```
{i,1,6}];
Do[Print[W[[i]]],{i,1,6}]
(* plot the edge elements*)
<< Graphics`PlotField3D`
mytet = Graphics3D[Line[{{0,0,0},{1,0,0},{0,1,0},{0,0,0},
                    {0,0,1},{0,1,0},{0,0,1},{1,0,0}}]];
myplot = Table[0,{i,1,6}];
Do[
 data = Table[0,{i,1,500}];
 n = 0;
 Do[
  x = i / 8;
  Do[
   y = j / 8;
   Do[
    z = k / 8;
    If[x + y + z <= 1,
     n = n + 1;
     data[[n]] = {{x,y,z},W[[ii]]}
    ],
   {k,0,8}],
  {j,0,8}],
 {i,0,8}];

 data = Drop[data,-(500-n)];
```

```
 myplot[[ii]] = Show[ListPlotVectorField3D[data,
                    VectorHeads->True,ScaleFactor->.15,
                    PlotLabel->StringJoin[“W”,ToString[ii]],
                    ViewPoint->{10,5,5},
                   (*  SphericalRegion->True, *)
                    BoxRatios->{1,1,1},
                    Boxed->False,
                    DefaultFont->{“Helvetica”,12}],mytet],
{ii,1,6}];
Display[“plot1.ps”,GraphicsArray[{{myplot[[1]],myplot[[2]]},
                {myplot[[3]],myplot[[4]]},
                {myplot[[5]],myplot[[6]]}}]];
(* construct the face elements*)
eq = Table[0,{i,1,4}];
eq[[1]] = Integrate[x = .;y = .;z = 0;pface . n[[1]],{x,0,1},{y,0,1-x}];
eq[[2]] = Integrate[x = .;y = 0;z = .;pface . n[[2]],{x,0,1},{z,0,1-x}];
eq[[3]] = Integrate[x = 0;y = .;z = .;pface . n[[3]],{y,0,1},{z,0,1-y}];
eq[[4]] = Integrate[x = .;y = .;z = 1-x-y;(pface . n[[4]])/n[[4,3]],{x,0,1},{y,0,1-x}];
A = Table[0,{i,1,4},{j,1,4}];
Do[
  Do[
    A[[i,j]] = Coefficient[Expand[Simplify[eq[[i]]]],a[[j]]],
  {j,1,4}],
{i,1,4}];
F = Table[0,{i,1,4}];
```

```
Do[
  rhs = Table[0,{j,1,4}];
  rhs[[i]] = 1;
  s = LinearSolve[A,rhs];
  x  = .;y = .;z = .;
  F[[i]] =  {s[[1]] + s[[4]] x,
        s[[2]] + s[[4]] y,
        s[[3]] + s[[4]] z},
{i,1,4}];
Do[Print[F[[i]]],{i,1,4}];
myplot = Table[0,{i,1,4}];
Do[
  data = Table[0,{i,1,500}];
  n = 0;
  Do[
    x = i / 8;
    Do[
      y = j / 8;
      Do[
        z = k / 8;
        If[x + y + z <= 1,
          n = n + 1;
          data[[n]] = {{x,y,z},F[[ii]]}
        ],
      {k,0,8}],
```

```
  {j,0,8}],
 {i,0,8}];
 data = Drop[data,-(500-n)];
 myplot[[ii]] = Show[ListPlotVectorField3D[data,
                  VectorHeads->True,ScaleFactor->.15,
                  PlotLabel->StringJoin[“F”,ToString[ii]],
                  ViewPoint->{10,5,5},
                 (*  SphericalRegion->True, *)
                  BoxRatios->{1,1,1},
                  Boxed->False,
                  DefaultFont->{“Helvetica”,12}],mytet],
{ii,1,4}];
Display[“plot2.ps”,GraphicsArray[{{myplot[[1]],myplot[[2]]},
            {myplot[[3]],myplot[[4]]}}]];
```

11.2 Mathematica script for generating linear vector basis functions on a hexahedron

```
(*this is a Mathematica script for generating
vector basis functions, both edge and face,
for hexahedral elements.
this script can be used to generate 3D plots
of vector basis functions, or to generate
elemental matrices.
the user must specify rr, the location of
eight nodes. see bottom of this script.
subroutine MakeLocal makes the basis functions,
subroutine MakeCurl computed the curl of the functions,
subroutine MakeMatrices computes the elemental matrices,
subroutine PlotHex plots the hexaheron,
subroutine PlotLocal Edge plots the local (undistorted)
  edge basis functions,
subroutine PlotLocal face plots the local (undistorted)
  face basis functions,
subroutine PlotEdge plots the edge basis functions,
subroutine PlotFace plots the face basis functions.
*)
<< Graphics`PlotField3D`
map = {
{1,2},
{4,3},
```

```
{5,6},
{8,7},
{1,4},
{5,8},
{2,3},
{6,7},
{1,5},
{2,6},
{4,8},
{3,7}
};
MakeLocal := (
psi = .;
eta = .;
nu = .;
Print[“making lambda”];
lambda = {
1 - eta - nu + eta nu - psi + eta psi + nu psi - eta nu psi,
psi - eta psi - nu psi + eta nu psi,
eta psi - eta nu psi,
eta - eta nu - eta psi + eta nu psi,
nu - eta nu - nu psi + eta nu psi,
nu psi - eta nu psi,
eta nu psi,
eta nu - eta nu psi
```

```
};
h1[psi_,eta_,nu_] = lambda[[1]];
h2[psi_,eta_,nu_] = lambda[[2]];
h3[psi_,eta_,nu_] = lambda[[3]];
h4[psi_,eta_,nu_] = lambda[[4]];
h5[psi_,eta_,nu_] = lambda[[5]];
h6[psi_,eta_,nu_] = lambda[[6]];
h7[psi_,eta_,nu_] = lambda[[7]];
h8[psi_,eta_,nu_] = lambda[[8]];

Print[“making jac”];
xx = Simplify[Sum[lambda[[i]]*rr[[i,1]],{i,1,8}]];
yy = Simplify[Sum[lambda[[i]]*rr[[i,2]],{i,1,8}]];
zz = Simplify[Sum[lambda[[i]]*rr[[i,3]],{i,1,8}]];
jac[psi_,eta_,nu_] = {
{D[xx,psi],D[yy,psi],D[zz,psi]},
{D[xx,eta],D[yy,eta],D[zz,eta]},
{D[xx,nu],D[yy,nu],D[zz,nu]}
};
Print[“making G”];
G[psi_,eta_,nu_] = jac[psi,eta,nu] . Transpose[jac[psi,eta,nu]];
Print[“making W”];
temp1[psi_,eta_,nu_] = {lambda[[map[[1,1]]]]+lambda[[map[[1,2]]]],0,0};
temp2[psi_,eta_,nu_] = {lambda[[map[[2,1]]]]+lambda[[map[[2,2]]]],0,0};
temp3[psi_,eta_,nu_] = {lambda[[map[[3,1]]]]+lambda[[map[[3,2]]]],0,0};
```

```
temp4[psi_,eta_,nu_] = {lambda[[map[[4,1]]]]+lambda[[map[[4,2]]]],0,0};

temp5[psi_,eta_,nu_] = {0,lambda[[map[[5,1]]]]+lambda[[map[[5,2]]]],0};

temp6[psi_,eta_,nu_] = {0,lambda[[map[[6,1]]]]+lambda[[map[[6,2]]]],0};

temp7[psi_,eta_,nu_] = {0,lambda[[map[[7,1]]]]+lambda[[map[[7,2]]]],0};

temp8[psi_,eta_,nu_] = {0,lambda[[map[[8,1]]]]+lambda[[map[[8,2]]]],0};

temp9[psi_,eta_,nu_] = {0,0,lambda[[map[[9,1]]]]+lambda[[map[[9,2]]]]};

temp10[psi_,eta_,nu_] = {0,0,lambda[[map[[10,1]]]]+lambda[[map[[10,2]]]]};

temp11[psi_,eta_,nu_] = {0,0,lambda[[map[[11,1]]]]+lambda[[map[[11,2]]]]};

temp12[psi_,eta_,nu_] = {0,0,lambda[[map[[12,1]]]]+lambda[[map[[12,2]]]]};

W = {

temp1,

temp2,

temp3,

temp4,

temp5,

temp6,

temp7,

temp8,

temp9,

temp10,

temp11,

temp12

};

Print[“making F”];

temp21[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {0,0,1-nu};
```

```
temp22[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {0,0,nu};
temp23[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {0,1-eta,0};
temp24[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {0,eta,0};
temp25[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {1-psi,0,0};
temp26[psi_,eta_,nu_] = 1.0/Det[jac[psi,eta,nu]] {psi,0,0};
F = {
temp21,
temp22,
temp23,
temp24,
temp25,
temp26
};
);
PlotLocalEdge[ii_] := (
PlotVectorField3D[W[[ii]][psi,eta,nu],{psi,0,1},{eta,0,1},{nu,0,1},
  VectorHeads->True,
  ViewPoint->{100,100,100}];
);

PlotLocalFace[ii_] := (
PlotVectorField3D[F[[ii]][psi,eta,nu],{psi,0,1},{eta,0,1},{nu,0,1},
  VectorHeads->True,
  ViewPoint->{100,100,100}];
);
```

```
mycurl[f_] := {
  D[f[[3]],eta]-D[f[[2]],nu],
  D[f[[1]],nu] - D[f[[3]],psi],
  D[f[[2]],psi] - D[f[[1]],eta]
};
MakeCurl := (
psi = .;
eta = .;
nu = .;
maggie1[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {0, -1 + eta, 1 - nu};
maggie2[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {0, -eta, -1 + nu};
maggie3[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {0, 1 - eta, nu};
maggie4[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {0, eta, -nu};
maggie5[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {1 - psi, 0, -1 + nu};
maggie6[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {-1 + psi, 0, -nu};
maggie7[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {psi, 0, 1 - nu};
maggie8[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {-psi, 0, nu};
maggie9[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {-1 + psi, 1 - eta, 0};
maggie10[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {-psi, -1 + eta, 0};
maggie11[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {1 - psi, eta, 0};
maggie12[psi_,eta_,nu_] := 1.0/Sqrt[Det[G[psi,eta,nu]]] {psi, -eta, 0};
KW = {
maggie1,
maggie2,
maggie3,
```

```
maggie4,
maggie5,
maggie6,
maggie7,
maggie8,
maggie9,
maggie10,
maggie11,
maggie12
};
);
MakeBasis := (
lisa1[psi_,eta_,nu_] := Inverse[jac[psi,eta,nu]] . {1,0,0};
lisa2[psi_,eta_,nu_] := Inverse[jac[psi,eta,nu]] . {0,1,0};
lisa3[psi_,eta_,nu_] := Inverse[jac[psi,eta,nu]] . {0,0,1};
basis = {
lisa1,
lisa2,
lisa3
};
);
PlotHex := (
lines = Table[{AbsoluteThickness[2],Line[{rr[[map[[i,1]]]],rr[[map[[i,2]]]]}]},
        {i,1,Length[map]}];
p1 = Show[Graphics3D[lines],ViewPoint->{100,100,100}];
```

```
);
PlotEdge[iplot_] := (
nvect = 7;
scale = 0.1;


data = Table[{{0,0,0},{0,0,0}},{i,1,(nvect+1)^3}];
ivect = 0;
Do[
  psi = i/nvect;
  Do[
    eta = j/nvect;
    Do[
      ivect = ivect + 1;
      nu = k/nvect;
      data[[ivect]] = {{xx,yy,zz},Inverse[jac[psi,eta,nu]] .
                    W[[iplot]][psi,eta,nu]},
    {k,0,nvect}],
  {j,0,nvect}],
{i,0,nvect}];
Show[ListPlotVectorField3D[data,VectorHeads->True,ScaleFactor->scale,
  ViewPoint->{100,100,100}],p1]
);
PlotFace[iplot_] := (
nvect = 7;
scale = 0.1;
```

```
data = Table[{{0,0,0},{0,0,0}},{i,1,(nvect+1)^3}];
ivect = 0;
Do[
  psi = i/nvect;
  Do[
    eta = j/nvect;
    Do[
      ivect = ivect + 1;
      nu = k/nvect;
      data[[ivect]] = {{xx,yy,zz},
                  Transpose[jac[psi,eta,nu]] . F[[iplot]][psi,eta,nu]},
    {k,0,nvect}],
  {j,0,nvect}],
{i,0,nvect}];
Show[ListPlotVectorField3D[data,VectorHeads->True,ScaleFactor->scale,
  ViewPoint->{100,100,100}],p1]
);
MyIntegrate1[i_,j_] := NIntegrate[
      W[[i]][psi,eta,nu] . Inverse[G[psi,eta,nu]] .
      W[[j]][psi,eta,nu] Det[jac[psi,eta,nu]],
                  {psi,0,1},{eta,0,1},{nu,0,1}];
MyIntegrate2[i_,j_] := NIntegrate[
      KW[[i]][psi,eta,nu] . G[psi,eta,nu] .
      KW[[j]][psi,eta,nu] Det[jac[psi,eta,nu]],
                  {psi,0,1},{eta,0,1},{nu,0,1}];
```

```
MyIntegrate3[i_,j_] := NIntegrate[
     F[[i]][psi,eta,nu] . G[psi,eta,nu] .
     F[[j]][psi,eta,nu] Det[jac[psi,eta,nu]],
                {psi,0,1},{eta,0,1},{nu,0,1}];
MyIntegrate4[i_,j_] := NIntegrate[
     F[[i]][psi,eta,nu] . G[psi,eta,nu] .
     KW[[j]][psi,eta,nu] 1.0/Det[jac[psi,eta,nu]],
                {psi,0,1},{eta,0,1},{nu,0,1}];
PlotBasis[iplot_] := (
nvect = 7;
scale = 0.001;
data = Table[{{0,0,0},{0,0,0}},{i,1,(nvect+1)^3}];
ivect = 0;
Do[
  psi = i/nvect;
  Do[
    eta = j/nvect;
    Do[
      ivect = ivect + 1;
      nu = k/nvect;
      data[[ivect]] = {{xx,yy,zz},basis[[iplot]][psi,eta,nu]},
    {k,0,nvect}],
  {j,0,nvect}],
{i,0,nvect}];
Show[ListPlotVectorField3D[data,VectorHeads->True,ScaleFactor->scale,
```

```
  ViewPoint->{100,100,100}],p1]
);
MakeMatrices := (
Print[“making KK”];
KK = Table[0,{i,1,6},{j,1,12}];
KK[[1, 1]] =  1;
KK[[1, 2]] = -1;
KK[[1, 5]] = -1;
KK[[1, 7]] =  1;
KK[[2, 3]] =  1;
KK[[2, 4]] = -1;
KK[[2, 6]] = -1;
KK[[2, 8]] =  1;
KK[[3, 1]] = -1;
KK[[3, 3]] =  1;
KK[[3, 9]] =  1;
KK[[3,10]] = -1;
KK[[4, 2]] = -1;
KK[[4, 4]] =  1;
KK[[4,11]] =  1;
KK[[4,12]] = -1;
KK[[5, 5]] =  1;
KK[[5, 6]] = -1;
KK[[5, 9]] = -1;
KK[[5,11]] =  1;
```

```
KK[[6, 7]] =  1;
KK[[6, 8]] = -1;
KK[[6,10]] = -1;
KK[[6,12]] =  1;
Print[“making CC”];
CC = Table[0.0,{i,1,12},{j,1,12}];
Do[
  Do[
    CC[[i,j]] = MyIntegrate2[i,j];
    Print[i,”  “,j,”  “,CC[[i,j]]];
    CC[[j,i]] = CC[[i,j]],
  {j,i,12}],
{i,1,12}];
Print[“making AA”];
AA = Table[0.0,{i,1,12},{j,1,12}];
Do[
  Do[
    AA[[i,j]] = MyIntegrate1[i,j];
    Print[i,”  “,j,”  “,AA[[i,j]]];
    AA[[j,i]] = AA[[i,j]],
  {j,i,12}],
{i,1,12}];
Print[“making DD”];
DD = Table[0.0,{i,1,6},{j,1,6}];
Do[
```

```
  Do[
    DD[[i,j]] = MyIntegrate3[i,j];
    Print[i,"  ",j,"  ",DD[[i,j]]];
    DD[[j,i]] = DD[[i,j]],
  {j,i,6}],
{i,1,6}];
);
pp = {
{0,0,0},
{1,0,0},
{1,1,0},
{0,1,0},
{0,0,1},
{1,0,1},
{1,1,1},
{0,1,1}
};
(* shift hex n x and in z*)
phi = 60;
theta = 60;
rr = {
{0,0,0},
{1,0,0},
{1 + N[Cos[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  N[Sin[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
```

```
  N[Cos[Pi / 180 * theta]]
},
{N[Cos[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  N[Sin[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  N[Cos[Pi / 180 * theta]]
},
{0,0,1},
{1,0,1},
{1 + N[Cos[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  N[Sin[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  1 + N[Cos[Pi / 180 * theta]]
},
{N[Cos[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  N[Sin[Pi / 180 * phi]] * N[Sin[Pi / 180 * theta]],
  1 + N[Cos[Pi / 180 * theta]]}};
```

11.3 Mathematica script for performing numerical dispersion analysis on distorted hexahedral grids

```
(*
this is a Mathematica script for performing
numerical dispersion analysis of the
DTVFEM.
the user must first compute the elemental matrices
AA and CC using subroutine covariant.m
this script computes the 3 by 3 homogeneous equation, the
roots of the equation determine the numerical
dispersion relation.
the user can either Taylor series the roots using
subroutine hex_dispersion3.m, or plot 2D curves
using subroutine hex_dispersion2.m, or plot 3D
surfaces using subroutine parametric_plot.m
*)
a = .;b = .;c = .;
w = .;
n = .;m = .;l = .;
(* shear grid in two directions*)
theta = 60;
phi = 60;
f[n_,m_,l_] := Exp[-I n a]  *
  Exp[-I m (a Cos[Pi / 180 * phi] Sin[Pi / 180 * theta] +
     b Sin[Pi / 180 * phi] Sin[Pi / 180 * theta] +
```

```
    c Cos[Pi / 180 * theta])] *
  Exp[-I l c];
(* make Ex equation*)
Print[“makeing Q”];
eq1 =
P * Ex *
(
  AA[[1,1]] +
  AA[[1,2]] * f[0,1,0] +
  AA[[1,3]] * f[0,0,1] +
  AA[[1,4]] * f[0,1,1] +
  AA[[2,1]] * f[0,-1,0] +
  AA[[2,2]] +
  AA[[2,3]] * f[0,-1,1] +
  AA[[2,4]] * f[0,0,1] +
  AA[[3,1]] * f[0,0,-1] +
  AA[[3,2]] * f[0,1,-1] +
  AA[[3,3]] +
  AA[[3,4]] * f[0,1,0] +
  AA[[4,1]] * f[0,-1,-1] +
  AA[[4,2]] * f[0,0,-1] +
  AA[[4,3]] * f[0,-1,0] +
  AA[[4,4]]
) +
P * Ey *
```

```
(
  AA[[1,5]] +
  AA[[1,6]] * f[0,0,1] +
  AA[[1,7]] * f[1,0,0] +
  AA[[1,8]] * f[1,0,1] +
  AA[[2,5]] * f[0,-1,0] +
  AA[[2,6]] * f[0,-1,1] +
  AA[[2,7]] * f[1,-1,0] +
  AA[[2,8]] * f[1,-1,1] +
  AA[[3,5]] * f[0,0,-1] +
  AA[[3,6]] +
  AA[[3,7]] * f[1,0,-1] +
  AA[[3,8]] * f[1,0,0] +
  AA[[4,5]] * f[0,-1,-1] +
  AA[[4,6]] * f[0,-1,0] +
  AA[[4,7]] * f[1,-1,-1] +
  AA[[4,8]] * f[1,-1,0]
) +
P * Ez *
(
  AA[[1,9]] +
  AA[[1,10]] * f[1,0,0] +
  AA[[1,11]] * f[0,1,0] +
  AA[[1,12]] * f[1,1,0] +
  AA[[2,9]] * f[0,-1,0] +
```

```
  AA[[2,10]] * f[1,-1,0] +
  AA[[2,11]] +
  AA[[2,12]] * f[1,0,0] +
  AA[[3,9]] * f[0,0,-1] +
  AA[[3,10]] * f[1,0,-1] +
  AA[[3,11]] * f[0,1,-1] +
  AA[[3,12]] * f[1,1,-1] +
  AA[[4,9]] * f[0,-1,-1] +
  AA[[4,10]] * f[1,-1,-1] +
  AA[[4,11]] * f[0,0,-1] +
  AA[[4,12]] * f[1,0,-1]
) -
Ex *
(
  CC[[1,1]] +
  CC[[1,2]] * f[0,1,0] +
  CC[[1,3]] * f[0,0,1] +
  CC[[1,4]] * f[0,1,1] +
  CC[[2,1]] * f[0,-1,0] +
  CC[[2,2]] +
  CC[[2,3]] * f[0,-1,1] +
  CC[[2,4]] * f[0,0,1] +
  CC[[3,1]] * f[0,0,-1] +
  CC[[3,2]] * f[0,1,-1] +
  CC[[3,3]] +
```

```
  CC[[3,4]] * f[0,1,0] +
  CC[[4,1]] * f[0,-1,-1] +
  CC[[4,2]] * f[0,0,-1] +
  CC[[4,3]] * f[0,-1,0] +
  CC[[4,4]]
) -
Ey *
(
  CC[[1,5]] +
  CC[[1,6]] * f[0,0,1] +
  CC[[1,7]] * f[1,0,0] +
  CC[[1,8]] * f[1,0,1] +
  CC[[2,5]] * f[0,-1,0] +
  CC[[2,6]] * f[0,-1,1] +
  CC[[2,7]] * f[1,-1,0] +
  CC[[2,8]] * f[1,-1,1] +
  CC[[3,5]] * f[0,0,-1] +
  CC[[3,6]] +
  CC[[3,7]] * f[1,0,-1] +
  CC[[3,8]] * f[1,0,0] +
  CC[[4,5]] * f[0,-1,-1] +
  CC[[4,6]] * f[0,-1,0] +
  CC[[4,7]] * f[1,-1,-1] +
  CC[[4,8]] * f[1,-1,0]
) -
```

```
Ez *
(
  CC[[1,9]] +
  CC[[1,10]] * f[1,0,0] +
  CC[[1,11]] * f[0,1,0] +
  CC[[1,12]] * f[1,1,0] +
  CC[[2,9]] * f[0,-1,0] +
  CC[[2,10]] * f[1,-1,0] +
  CC[[2,11]] +
  CC[[2,12]] * f[1,0,0] +
  CC[[3,9]] * f[0,0,-1] +
  CC[[3,10]] * f[1,0,-1] +
  CC[[3,11]] * f[0,1,-1] +
  CC[[3,12]] * f[1,1,-1] +
  CC[[4,9]] * f[0,-1,-1] +
  CC[[4,10]] * f[1,-1,-1] +
  CC[[4,11]] * f[0,0,-1] +
  CC[[4,12]] * f[1,0,-1]
);
(* make Ey equation*)
eq2 =
P * Ex *
(
  AA[[5,1]] +
  AA[[5,2]] * f[0,1,0] +
```

```
  AA[[5,3]] * f[0,0,1] +
  AA[[5,4]] * f[0,1,1] +
  AA[[6,1]] * f[0,0,-1] +
  AA[[6,2]] * f[0,1,-1] +
  AA[[6,3]] +
  AA[[6,4]] * f[0,1,0] +
  AA[[7,1]] * f[-1,0,0] +
  AA[[7,2]] * f[-1,1,0] +
  AA[[7,3]] * f[-1,0,1] +
  AA[[7,4]] * f[-1,1,1] +
  AA[[8,1]] * f[-1,0,-1] +
  AA[[8,2]] * f[-1,1,-1] +
  AA[[8,3]] * f[-1,0,0] +
  AA[[8,4]] * f[-1,1,0]
) +
P * Ey *
(
  AA[[5,5]] +
  AA[[5,6]] * f[0,0,1] +
  AA[[5,7]] * f[1,0,0] +
  AA[[5,8]] * f[1,0,1] +
  AA[[6,5]] * f[0,0,-1] +
  AA[[6,6]] +
  AA[[6,7]] * f[1,0,-1] +
  AA[[6,8]] * f[1,0,0] +
```

```
  AA[[7,5]] * f[-1,0,0] +
  AA[[7,6]] * f[-1,0,1] +
  AA[[7,7]] +
  AA[[7,8]] * f[0,0,1] +
  AA[[8,5]] * f[-1,0,-1] +
  AA[[8,6]] * f[-1,0,0] +
  AA[[8,7]] * f[0,0,-1] +
  AA[[8,8]]
) +
P * Ez *
(
  AA[[5,9]] +
  AA[[5,10]] * f[1,0,0] +
  AA[[5,11]] * f[0,1,0] +
  AA[[5,12]] * f[1,1,0] +
  AA[[6,9]] * f[0,0,-1] +
  AA[[6,10]] * f[1,0,-1] +
  AA[[6,11]] * f[0,1,-1] +
  AA[[6,12]] * f[1,1,-1] +
  AA[[7,9]] * f[-1,0,0] +
  AA[[7,10]] +
  AA[[7,11]] * f[-1,1,0] +
  AA[[7,12]] * f[0,1,0] +
  AA[[8,9]] * f[-1,0,-1] +
  AA[[8,10]] * f[0,0,-1] +
```

```
  AA[[8,11]] * f[-1,1,-1] +
  AA[[8,12]] * f[0,1,-1]
) -
Ex *
(
  CC[[5,1]] +
  CC[[5,2]] * f[0,1,0] +
  CC[[5,3]] * f[0,0,1] +
  CC[[5,4]] * f[0,1,1] +
  CC[[6,1]] * f[0,0,-1] +
  CC[[6,2]] * f[0,1,-1] +
  CC[[6,3]] +
  CC[[6,4]] * f[0,1,0] +
  CC[[7,1]] * f[-1,0,0] +
  CC[[7,2]] * f[-1,1,0] +
  CC[[7,3]] * f[-1,0,1] +
  CC[[7,4]] * f[-1,1,1] +
  CC[[8,1]] * f[-1,0,-1] +
  CC[[8,2]] * f[-1,1,-1] +
  CC[[8,3]] * f[-1,0,0] +
  CC[[8,4]] * f[-1,1,0]
) -
Ey *
(
  CC[[5,5]] +
```

```
  CC[[5,6]] * f[0,0,1] +
  CC[[5,7]] * f[1,0,0] +
  CC[[5,8]] * f[1,0,1] +
  CC[[6,5]] * f[0,0,-1] +
  CC[[6,6]] +
  CC[[6,7]] * f[1,0,-1] +
  CC[[6,8]] * f[1,0,0] +
  CC[[7,5]] * f[-1,0,0] +
  CC[[7,6]] * f[-1,0,1] +
  CC[[7,7]] +
  CC[[7,8]] * f[0,0,1] +
  CC[[8,5]] * f[-1,0,-1] +
  CC[[8,6]] * f[-1,0,0] +
  CC[[8,7]] * f[0,0,-1] +
  CC[[8,8]]
) -
Ez *
(
  CC[[5,9]] +
  CC[[5,10]] * f[1,0,0] +
  CC[[5,11]] * f[0,1,0] +
  CC[[5,12]] * f[1,1,0] +
  CC[[6,9]] * f[0,0,-1] +
  CC[[6,10]] * f[1,0,-1] +
  CC[[6,11]] * f[0,1,-1] +
```

```
  CC[[6,12]] * f[1,1,-1] +
  CC[[7,9]] * f[-1,0,0] +
  CC[[7,10]] +
  CC[[7,11]] * f[-1,1,0] +
  CC[[7,12]] * f[0,1,0] +
  CC[[8,9]] * f[-1,0,-1] +
  CC[[8,10]] * f[0,0,-1] +
  CC[[8,11]] * f[-1,1,-1] +
  CC[[8,12]] * f[0,1,-1]
);
(* make Ez equations*)
eq3 =
P * Ex *
(
  AA[[9,1]] +
  AA[[9,2]] * f[0,1,0] +
  AA[[9,3]] * f[0,0,1] +
  AA[[9,4]] * f[0,1,1] +
  AA[[10,1]] * f[-1,0,0] +
  AA[[10,2]] * f[-1,1,0] +
  AA[[10,3]] * f[-1,0,1] +
  AA[[10,4]] * f[-1,1,1] +
  AA[[11,1]] * f[0,-1,0] +
  AA[[11,2]] +
  AA[[11,3]] * f[0,-1,1] +
```

```
  AA[[11,4]] * f[0,0,1] +
  AA[[12,1]] * f[-1,-1,0] +
  AA[[12,2]] * f[-1,0,0] +
  AA[[12,3]] * f[-1,-1,1] +
  AA[[12,4]] * f[-1,0,1]
) +
P * Ey *
(
  AA[[9,5]] +
  AA[[9,6]] * f[0,0,1] +
  AA[[9,7]] * f[1,0,0] +
  AA[[9,8]] * f[1,0,1] +
  AA[[10,5]] * f[-1,0,0] +
  AA[[10,6]] * f[-1,0,1] +
  AA[[10,7]] +
  AA[[10,8]] * f[0,0,1] +
  AA[[11,5]] * f[0,-1,0] +
  AA[[11,6]] * f[0,-1,1] +
  AA[[11,7]] * f[1,-1,0] +
  AA[[11,8]] * f[1,-1,1] +
  AA[[12,5]] * f[-1,-1,0] +
  AA[[12,6]] * f[-1,-1,1] +
  AA[[12,7]] * f[0,-1,0] +
  AA[[12,8]] * f[0,-1,1]
) +
```

```
P * Ez *
(
  AA[[9,9]] +
  AA[[9,10]] * f[1,0,0] +
  AA[[9,11]] * f[0,1,0] +
  AA[[9,12]] * f[1,1,0] +
  AA[[10,9]] * f[-1,0,0] +
  AA[[10,10]] +
  AA[[10,11]] * f[-1,1,0] +
  AA[[10,12]] * f[0,1,0] +
  AA[[11,9]] * f[0,-1,0] +
  AA[[11,10]] * f[1,-1,0] +
  AA[[11,11]] +
  AA[[11,12]] * f[1,0,0] +
  AA[[12,9]] * f[-1,-1,0] +
  AA[[12,10]] * f[0,-1,0] +
  AA[[12,11]] * f[-1,0,0] +
  AA[[12,12]]
) -
Ex *
(
  CC[[9,1]] +
  CC[[9,2]] * f[0,1,0] +
  CC[[9,3]] * f[0,0,1] +
  CC[[9,4]] * f[0,1,1] +
```

```
  CC[[10,1]] * f[-1,0,0] +
  CC[[10,2]] * f[-1,1,0] +
  CC[[10,3]] * f[-1,0,1] +
  CC[[10,4]] * f[-1,1,1] +
  CC[[11,1]] * f[0,-1,0] +
  CC[[11,2]] +
  CC[[11,3]] * f[0,-1,1] +
  CC[[11,4]] * f[0,0,1] +
  CC[[12,1]] * f[-1,-1,0] +
  CC[[12,2]] * f[-1,0,0] +
  CC[[12,3]] * f[-1,-1,1] +
  CC[[12,4]] * f[-1,0,1]
) -
Ey *
(
  CC[[9,5]] +
  CC[[9,6]] * f[0,0,1] +
  CC[[9,7]] * f[1,0,0] +
  CC[[9,8]] * f[1,0,1] +
  CC[[10,5]] * f[-1,0,0] +
  CC[[10,6]] * f[-1,0,1] +
  CC[[10,7]] +
  CC[[10,8]] * f[0,0,1] +
  CC[[11,5]] * f[0,-1,0] +
  CC[[11,6]] * f[0,-1,1] +
```

```
  CC[[11,7]] * f[1,-1,0] +
  CC[[11,8]] * f[1,-1,1] +
  CC[[12,5]] * f[-1,-1,0] +
  CC[[12,6]] * f[-1,-1,1] +
  CC[[12,7]] * f[0,-1,0] +
  CC[[12,8]] * f[0,-1,1]
) -
Ez *
(
  CC[[9,9]] +
  CC[[9,10]] * f[1,0,0] +
  CC[[9,11]] * f[0,1,0] +
  CC[[9,12]] * f[1,1,0] +
  CC[[10,9]] * f[-1,0,0] +
  CC[[10,10]] +
  CC[[10,11]] * f[-1,1,0] +
  CC[[10,12]] * f[0,1,0] +
  CC[[11,9]] * f[0,-1,0] +
  CC[[11,10]] * f[1,-1,0] +
  CC[[11,11]] +
  CC[[11,12]] * f[1,0,0] +
  CC[[12,9]] * f[-1,-1,0] +
  CC[[12,10]] * f[0,-1,0] +
  CC[[12,11]] * f[-1,0,0] +
  CC[[12,12]]
```

```
);
bart = {eq1,eq2,eq3};
homer = {Ex,Ey,Ez};
Q = Table[Coefficient[Collect[bart[[i]],homer[[j]]],homer[[j]]],
        {i,1,3},{j,1,3}];
```

11.4 Mathematica script for performing Taylor series of numerical dispersion relation

```
(*

this Mathematica script performs a Taylor

series on the roots of the homogenesous equation

*)

a = .;b = .;c = .;

(* takes a long time*)

marge = Det[Q];

aaa = Coefficient[Collect[marge,P^3],P^3];

bbb = Coefficient[Collect[marge,P^2],P^2];

ccc = Coefficient[Collect[marge,P],P];

s2 = (-bbb + Sqrt[bbb^2 - 4 aaa ccc])/(2 aaa);

ans = Series[s2,{a,0,4},{b,0,4},{c,0,4}];
```

11.5 Mathematica script for plotting numerical phase velocity error curves in 2D

```
(*
this Mathematica script plots 2D curves
of numerical dispersion and also performs
least-square fit to determin the accuracy
the user must run covariant.m first, then
run hex_dispersion.m
*)
<< Graphics`Graphics`
w = .;dt = .;k = .;phi = .;theta = .;
myfunc[k_,dt_,phi_,theta_] := (
  lisa = (2 Cos[w dt] - 2)/dt^2;
  a = k Sin[phi] Sin[theta];
  b = k Cos[phi] Sin[theta];
  c = k Cos[theta];
  myroots = Solve[Det[N[Q]] == 0,P];
  rhs = myroots[[3,1,2]];
  sol = N[Solve[lisa == -rhs,w]];
  omega = Re[sol[[1,1,2]]];
  omega/k
);
k = N[2 Pi/5];
dt = 1/3;
theta = N[Pi/2];
```

```
p0 = PolarPlot[myfunc[k,dt,phi,theta] - 1.0,{phi,0,2 Pi},
PlotRange->{{-.25,.25},{-.25,.25}}];
k = N[2 Pi/10];
dt = 1/3;
theta = N[Pi/2];
p1 = PolarPlot[myfunc[k,dt,phi,theta] - 1.0,{phi,0,2 Pi},
PlotRange->{{-.25,.25},{-.25,.25}}];
k = N[2 Pi/15];
dt = 1/3;
theta = N[Pi/2];
p2 = PolarPlot[myfunc[k,dt,phi,theta] - 1.0,{phi,0,2 Pi},
PlotRange->{{-.25,.25},{-.25,.25}}];
k = N[2 Pi/20];
dt = 1/3;
theta = N[Pi/2];
p3 = PolarPlot[myfunc[k,dt,phi,theta] - 1.0,{phi,0,2 Pi},
PlotRange->{{-.25,.25},{-.25,.25}}];
(* to perform a least-square fit un-comment these lines
blake = Table[0,{i,1,4}];
k = N[2 Pi/5];
dt = 1/3;
theta = N[Pi/2];
data = Table[phi = i * Pi / 180.0;myfunc[k,dt,phi,theta],{i,1,360}];
maxdata = Max[data];
mindata = Min[data];
```

```
ratio = maxdata/mindata;
Print[“k = “,k,”  max = “,maxdata,”  min = “,mindata,”  ratio = “,ratio];
blake[[1]] = {k,maxdata-1};
k = N[2 Pi/10];
dt = 1/3;
theta = N[Pi/2];
data = Table[phi = i * Pi / 180.0;myfunc[k,dt,phi,theta],{i,1,360}];
maxdata = Max[data];
mindata = Min[data];
ratio = maxdata/mindata;
Print[“k = “,k,”  max = “,maxdata,”  min = “,mindata,”  ratio = “,ratio];
blake[[2]] = {k,maxdata-1};
k = N[2 Pi/15];
dt = 1/3;
theta = N[Pi/2];
data = Table[phi = i * Pi / 180.0;myfunc[k,dt,phi,theta],{i,1,360}];
maxdata = Max[data];
mindata = Min[data];
ratio = maxdata/mindata;
Print[“k = “,k,”  max = “,maxdata,”  min = “,mindata,”  ratio = “,ratio];
blake[[3]] = {k,maxdata-1};
k = N[2 Pi/20];
dt = 1/3;
theta = N[Pi/2];
data = Table[phi = i * Pi / 180.0;myfunc[k,dt,phi,theta],{i,1,360}];
```

```
maxdata = Max[data];

mindata = Min[data];

ratio = maxdata/mindata;

Print[“k = “,k,”  max = “,maxdata,”  min = “,mindata,”  ratio = “,ratio];

blake[[4]] = {k,maxdata-1};

Fit[Log[blake],{1,x},x]

*)
```

11.6 Mathematica script for plotting numerical phase velocity error surfaces in 3D

```
(*

this Mathematica script make 3D surface plots

of numerical dispersion.

the user must first run covariant.m and

then hex_dispersion.m

*)

<< Graphics`Graphics`

w = .;dt = .;k = .;phi = .;theta = .;

myfunc[k_,dt_,phi_,theta_] := (

  lisa = (2 Cos[w dt] - 2)/dt^2;

  a = k Sin[phi] Sin[theta];

  b = k Cos[phi] Sin[theta];

  c = k Cos[theta];

  myroots = Solve[Det[N[Q]] == 0,P];

  rhs = myroots[[3,1,2]];

  sol = N[Solve[lisa == -rhs,w]];
```

```
  omega = Re[sol[[1,1,2]]];
  omega/k
);
k = N[2 Pi / 5];
dt = 1/3;
p0 = ParametricPlot3D[{(myfunc[k,dt,t,u]-1) Sin[t] Sin[u],
    (myfunc[k,dt,t,u]-1) Cos[t] Sin[u],
    (myfunc[k,dt,t,u]-1) Cos[u]},
    {t,0,2 Pi},{u,0,Pi},
    PlotRange->{{-0.25,0.25},{-0.25,0.25},{-0.25,0.25}},
    Boxed->False,
    Axes->False
];
p1 = Show[
  Graphics3D[{
    Line[{{-.25,0,0},{.25,0,0}}],
    Line[{{0,-.25,0},{0,0.25,0}}],
    Line[{{0,0,-.25},{0,0,0.25}}]}],
  Boxed->False,Axes->False
];
Show[p0,p1]
```