Investigation of finite element - ABC methods for electromagnetic field simulation

A. Chatterjee, J.L. Volakis and J. Nguyen

National Aeronautics & Space Administration
Ames Research Center
Moffett Field, CA 94035

September 1994

Report #031155-2-T

NASA Ames Grant NAG 2-866

Grant Title:	Development and parallelization of the Finite Element Method with Mixed Termination Schemes
Report Title *:	Investigation of finite element -ABC methods for electromagnetic field simulation
Report Authors:	A. Chatterjee, J.L.Volakis and J. Nguyen
Primary University Collaborator:	John L. Volakis volakis@um.cc.umich.edu Telephone: (313) 764-0500
Primary NASA-Ames Collaborator:	Alex Woo woo@ra-next.arc.nasa.gov
University Address:	Radiation Laboratory Department of Electrical Engineering and Computer Science The University of Michigan Ann Arbor, MI 48109-2122
Date:	September 1994 (replaces report published in May 1994)

Funds for the support of this study have been allocated in part of the NASA-Ames Research Center, Moffett Field, California, under Grant No. NAG 2-866

* This report was also submitted by the first author (AC) as a dissertation in partial fulfillment of the requirements for the Ph.D. degree at the University of Michigan.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

LIST OF APPENDICES

Appendix

CHAPTER I

Introduction

The mechanics of wave propagation in the presence of obstacles is of great interest in many branches of engineering and applied mathematics like electromagnetics, fluid dynamics, geophysics, seismology, etc. Such problems can be broadly classified into two categories: the bounded domain or the closed problem and the unbounded domain or the open problem. Analytical techniques have been derived for the simpler problems; however, the need to model complicated geometrical features, complex material coatings and fillings and to adapt the model to changing design parameters have inevitably tilted the balance in favor of numerical techniques. The modeling of closed problems presents difficulties primarily in proper meshing of the interior region. However, problems in unbounded domains pose a unique challenge to computation, since the exterior region is inappropriate for direct implementation of numerical techniques. A large number of solutions have been proposed but only a few have stood the test of time and experiment.

The goal of this thesis is to develop an efficient and reliable partial differential equation technique to model large three dimensional scattering problems in electromagnetics.

1.1 Background

Ever since the method of moments (MoM) was introduced by Harrington [3] in the late 60s, numerical techniques for predicting electromagnetic field behavior have gained in popularity. With increases in computing speeds and memory and the need to simulate real-life problems, researchers have been actively trying to refine the older numerical methods as well as devise newer and more efficient solution techniques. The MoM is based on applying integral equations on the surface of the desired structure and computing the fields everywhere in space [4]. For anisotropic materials, the entire volume needs to be discretized and a volume integral equation must then be solved. However, the matrix obtained from discretizing an integral equation is full and thus requires $O(N^2)$ storage, where N is the number of unknowns. In 3D problems, this is a serious limitation since the method can scale up to large problems only at considerable computational cost. Thus we need to seek solution techniques which scale favorably with increasing problem size.

Partial differential equation (PDE) methods, like finite element and finite difference methods, offer the most attractive alternative to integral equation techniques since they lead to matrix systems which are sparse. Therefore, only the non-zero entries of the final matrix system need to be stored resulting in $O(N)$ storage requirement. Thus the increase in storage demand with increasing problem size is seen to be minimal.

PDE techniques like finite elements were, however, originally constructed for solving bounded domain problems. In recent times, finite elements are increasingly being used for modeling unbounded problems, where the desired parameter decays off to zero infinitely away from the region of interest. In electromagnetics, the desired

parameter is a field quantity like the electric or magnetic field. It is obviously impractical to extend the finite element mesh to infinity; thus the mesh must be truncated at a *suitable* distance from the region of interest. Boundary conditions should then be applied at this artificial mesh truncation surface such that the boundary appears transparent to the propagating field.

There are two types of mesh truncation conditions: exact and approximate. Exact boundary conditions can be placed very close to the region of interest; however, they suffer from potential uniqueness problems [5] and give rise to partly full systems. The loss of uniqueness associated with systems where the exact boundary condition is employed on the mesh truncation boundary is well-known and was first pointed out by Mautz and Harrington[6]. Remedies like complexification of the wave number [7] and using the combined field integral equation [8] exist, and must be used for a robust implementation. Although the problem of interior resonances can be now avoided, the finite element-boundary integral system still possesses a partly full system which affects its scalability to large problems.

Approximate boundary conditions, on the other hand, are local in nature but preserve the sparsity of the finite element system. This advantage is partly offset by the fact that the finite element mesh must be extended some distance away from the region of interest and the approximate boundary condition imposed on the mesh truncation surface. These boundary conditions work on the principle that the higher order terms of the expansion for the propagating field decay rapidly away from the target. Therefore, if the truncation boundary is placed far enough from the region of interest, the boundary condition on the mesh termination surface needs to absorb only the lowest order terms of the field expansion to accurately model the physics of the problem. In this thesis, our aim is to examine the performance of these

approximate conditions, also known as absorbing boundary conditions (ABCs), in practical three dimensional problems and to derive improved boundary conditions which will enable more efficient utilization of the available resources.

1.2 Outline of thesis

This dissertation describes the development of a finite element method for the solution of general three-dimensional scattering problems. The entire research has been geared towards a robust, state-of-the-art solution of unbounded domain problems in electromagnetics. Improvements in solution convergence, mesh termination conditions, algorithmic complexity and computational speed have all been carried out with an eye to making the methodology more efficient in terms of computer storage and time.

Chapter 1 presents a brief introduction to the problem, its possible applications in science and industry and our motivation in preferring this solution methodology over more traditional ones.

Chapter 2 gives a short review of the fundamental laws that govern electromagnetic phenomena and the modeling technique of finite elements. The wave equation for electromagnetic fields is derived and various boundary conditions satisfied on material interfaces are presented. A brief outline of the method of finite elements and its application in electromagnetics is given.

Chapter 3 provides a detailed review on the construction and implementation of scalar and vector shape functions for two- and three-dimensional finite elements. Traditional node-based shaped functions are presented for a wide variety of element shapes and their pros and cons are outlined. The problem with nodal basis for a full-scale vector formulation is explained and edge-based vector shape functions are

introduced. The development of two- and three-dimensional edge bases is presented for a wide variety of element shapes. Higher order edge basis functions are presented for triangles and tetrahedra along with other recently developed novel shape functions.

Chapter 4 describes the formulation and implementation for closed and open domain problems. In the first part of the chapter, the closed problem is solved by determining the eigenvalues of an empty or filled metallic cavity. The origin and avoidance of spurious modes is discussed. The open problem is then formulated in the second part of the chapter and schemes for terminating the finite element mesh are discussed. The code is then validated for a wide class of perfectly conducting and composite geometries having arbitrary shapes.

Since the finite element-absorbing boundary condition methodology becomes extremely attractive for large problems, it is essential that the computer code be as computationally efficient as possible. Chapter 5 details the optimization and the subsequent parallelization of the finite element code and the various numerical considerations associated with it. The strategies for sparse matrix storage as well as solution of sparse systems using preconditioned iterative methods is outlined. The inherent parallelism in various types of point and block preconditioners is examined along with performance figures on the KSR1 and the Intel iPSC/80 massively parallel architectures.

In Chapter 6, new mesh termination conditions which can be applied on termination surfaces conformal to the target are derived and applied on some benchmark geometries. Since these ABCs are enforceable on doubly curved surfaces, dramatic reductions in computer storage and solution time are obtained. The improved boundary conditions are applied on mesh truncation surfaces composed of combinations of

cylinders, spheres and flat planes and their performance examined with respect to mesh termination distance and system symmetry. Extensions to more complex mesh termination boundaries are possible.

Chapter 7 concludes the thesis by summarizing the important results obtained during the course of this work and its possible future extensions.

CHAPTER II

Basic concepts of electromagnetics and finite elements

As mentioned in the last chapter, this thesis deals with the application of finite elements to three-dimensional problems in electromagnetics. Therefore, it is important to have a grasp of both the finite element method as well as electromagnetic theory to solve these problems. This chapter is thus divided into two parts. The first part gives a brief review of the basic concepts of electromagnetics and the resulting differential and integral equations pertaining to our problem. The second part introduces the reader to the formulation of boundary-value problems with finite elements.

2.1 Electromagnetics: basic concepts

2.1.1 Maxwell's equations

Electromagnetic waves in all space are governed by a set of fundamental equations called Maxwell's equations. In differential form, they are written as

$$\nabla \times \mathbf{E} = -\frac{\partial \boldsymbol{B}}{\partial t} \quad : (\text{Faraday's law}) \tag{2.1}$$

$$\nabla \times \mathbf{H} = \boldsymbol{J} + \frac{\partial \boldsymbol{D}}{\partial t} \quad : (\text{Maxwell-Ampere's law}) \tag{2.2}$$

$$\nabla \cdot \boldsymbol{D} = \rho \quad : (\text{Gauss' law}) \tag{2.3}$$

$$\nabla \cdot \boldsymbol{B} = 0 \quad : (\text{Gauss' magnetic law}) \tag{2.4}$$

where

$\mathbf{E} \equiv$ electric field intensity in volts/m

$\boldsymbol{D} \equiv$ electric flux density in coulombs/sq. m

$\mathbf{H} \equiv$ magnetic field intensity in amperes/m

$\boldsymbol{B} \equiv$ magnetic flux density in webers/sq. m

$\boldsymbol{J} \equiv$ electric current density in amperes/sq. m

$\rho \equiv$ electric charge density in coulombs/cu. m

Another fundamental equation, frequently referred to as the equation of continuity, is given by

$$\nabla \cdot \boldsymbol{J} = -\frac{\partial \rho}{\partial t} \tag{2.5}$$

and expresses the conservation of charge.

Of the five equations stated in this chapter, only three are independent. Either the first three equations, (2.1-2.3), or the first two equations, (2.1 and 2.2) and the fifth equation (2.5), can be chosen as independent equations. The remaining two equations,(2.4 and 2.5) or (2.3 and 2.4), can be derived from the independent equations and are thus called auxiliary or dependent equations.

The three independent equations cannot be solved since the number of unknowns exceeds the number of equations. Maxwell's equations become definite only when the three constitutive relations between the field quantities are specified. These relations describe the macroscopic properties of the medium being considered. For a simple medium, they are given by

$$\boldsymbol{D} = \epsilon \mathbf{E} \tag{2.6}$$

$$\boldsymbol{B} = \mu\mathbf{H} \tag{2.7}$$

$$\boldsymbol{J} = \sigma\mathbf{E} \tag{2.8}$$

where the parameters ϵ, μ and σ denote the permittivity (farads/m), permeability (henrys/m) and conductivity (mhos/m) of the material, respectively. For anisotropic media, the constitutive relations are given as

$$\boldsymbol{D} = \bar{\bar{\epsilon}} \cdot \mathbf{E} \tag{2.9}$$

$$\boldsymbol{B} = \bar{\bar{\mu}} \cdot \mathbf{H} \tag{2.10}$$

$$\boldsymbol{J} = \sigma\mathbf{E} \tag{2.11}$$

We will be considering only isotropic media in the following sections since the generalization to anisotropy is trivial and introduces needless algebraic complexity.

It is usually sufficient to consider the steady-state solution for electromagnetic fields as produced by currents having sinusoidal time dependence. The set of Maxwell's equations, using complex phasor notation and the constitutive relations, can then be written as

$$\nabla\times\mathbf{E} = -\jmath\omega\mu\mathbf{H} \tag{2.12}$$

$$\nabla\times\mathbf{H} = \boldsymbol{J} + \jmath\omega\epsilon\mathbf{E} \tag{2.13}$$

$$\nabla\cdot(\epsilon\mathbf{E}) = \rho \tag{2.14}$$

$$\nabla\cdot(\mu\mathbf{H}) = 0 \tag{2.15}$$

$$\nabla\cdot\boldsymbol{J} = -\jmath\omega\rho \tag{2.16}$$

where ω is the angular frequency of oscillation and the time convention $e^{\jmath m\omega t}$ is used and suppressed.

2.1.2 Wave equations

The two curl equations, (2.1 and 2.2), can be combined together with the assumed constitutive relations, (2.7 and 2.8), to obtain a separate second-order differential equation for each field. By taking the curl of (2.1) or (2.2) and eliminating **H** or **E** respectively, we obtain

$$\nabla\times\frac{1}{\mu_r}\nabla\times\mathbf{E} - k_o^2\epsilon_r\mathbf{E} = -\jmath\omega\boldsymbol{J} \tag{2.17}$$

$$\nabla\times\frac{1}{\epsilon_r}\nabla\times\mathbf{H} - k_o^2\mu_r\mathbf{H} = \nabla\times\left(\frac{\boldsymbol{J}}{\epsilon_r}\right) \tag{2.18}$$

where $k_o = \omega\sqrt{\epsilon_o\mu_o}$ is the free-space wave number, ϵ_r and μ_r are the respective relative permittivity and permeability of the medium under consideration and $\boldsymbol{J}$ is an impressed or source current. The differential equations shown above are called inhomogeneous vector wave equations in three dimensions.

2.1.3 Boundary conditions

Mathematically, the solution of a partial differential equation (PDE) like the wave equation, outlined in (2.17) and (2.18), is not unique in a region unless boundary conditions are specified, i.e, the behavior of the field on the boundary of the region of interest. Boundary conditions play the same role in the solution of PDEs that initial conditions play in the solution of differential equations for electric circuits. An electromagnetic problem is thus completely defined only when it contains information about the governing differential equation and the corresponding boundary conditions at material discontinuities or inhomogeneities.

At the interface between two media, say medium 1 and medium 2, the boundary conditions can be mathematically expressed as

$$\hat{\mathbf{n}}\times(\mathbf{E}_1 - \mathbf{E}_2) = 0 \tag{2.19}$$

$$\hat{\mathbf{n}} \cdot (\boldsymbol{D}_1 - \boldsymbol{D}_2) = 0 \tag{2.20}$$

for electric fields and

$$\hat{\mathbf{n}} \times (\mathbf{H}_1 - \mathbf{H}_2) = 0 \tag{2.21}$$

$$\hat{\mathbf{n}} \cdot (\boldsymbol{B}_1 - \boldsymbol{B}_2) = 0 \tag{2.22}$$

for magnetic fields, where $\hat{\mathbf{n}}$ denotes the unit normal to the interface. It is assumed in (2.20) and (2.21) that neither surface currents nor surface charge exist on the boundary. Equations (2.19) and (2.21) state that tangential electric and magnetic fields are continuous across dielectric boundaries.

It is possible to simplify the above boundary conditions at the interface of a perfect electric conductor and free-space. Since a perfect conductor cannot sustain a field inside it and likewise since the flux lines of $\boldsymbol{B}$ are continuous, (2.19) can be rewritten as

$$\hat{\mathbf{n}} \times \mathbf{E} = 0 \tag{2.23}$$

and (2.22) reduces to

$$\hat{\mathbf{n}} \cdot \boldsymbol{B} = 0 \tag{2.24}$$

However, the conductor surface can support a surface current ($\boldsymbol{J}_s = \hat{\mathbf{n}} \times \mathbf{H}$) and a surface charge ($\rho_s = \hat{\mathbf{n}} \cdot \boldsymbol{D}$).

2.1.4 Radiation conditions

It can be shown that the electric field within a finite volume can be derived in terms of the sources within the volume and the field values on the surfaces bounding that volume. If we make this volume infinitely large, we arrive at the Sommerfeld

radiation condition [9] given by

$$\lim_{r\to\infty} r\left[\frac{\partial\psi}{\partial r} + jk_o\psi\right] = 0 \tag{2.25}$$

where ψ is regular at infinity and describes a component of the electric or magnetic field and $r = \sqrt{x^2+y^2}$. In three dimensions, the radiation becomes

$$\lim_{R\to\infty} R\left[\nabla\times\mathbf{E} + jk_o\hat{\boldsymbol{R}}\times\mathbf{E}\right] = 0 \tag{2.26}$$

where $R = \sqrt{x^2+y^2+z^2}$. A similar result exists for the magnetic field. The radiation condition requires that $\mathbf{E}$ and $\mathbf{H}$ diminish as R^{-1} when $R \to \infty$.

2.1.5 Radar cross-section

The radar cross-section (RCS) is a quantity characterizing the scattering from an obstacle. It is defined as the area intercepting that amount of power which, when scattered isotropically, produces at the receiver a power density equal to that scattered by the target under consideration. In the three-dimensional case, the RCS is defined as

$$\sigma(\theta,\phi) = \lim_{R\to\infty} 4\pi R^2 \frac{|\boldsymbol{F}^s|^2}{\left|\boldsymbol{F}^{inc}\right|^2} \tag{2.27}$$

wher $\boldsymbol{F}^s$ denotes the scattered field (either $\mathbf{E}^s$ or $\mathbf{H}^s$) at the observation point (R,θ,ϕ) and $\boldsymbol{F}^{inc}$ represents the incident field , usually a plane wave, coming from $(R,\theta^{inc},\phi^{inc})$.

2.2 Finite elements: basic ideas

The finite element method (FEM) is a numerical method for obtaining approximate solutions to boundary-value problems in physics and engineering. Very few analytical solutions are possible and thus a numerical method like the FEM provides

us with an alternative solution technique for these problems. For a particular class of such problems, there exist extremum principles by which the solution being sought makes an appropriate functional stationary, or, in certain cases, extremal. In other problems for which no genuine extremal principles can be derived, the error resulting from the substitution of the numerical approximation into the differential equation is minimized.

In the following pages, we will give a brief description of the two accepted formulation schemes - the Ritz variational method [10] and the Galerkin method of weighted residuals [11]. We will then discuss how the finite element method is used to solve PDE problems formulated by the two abovementioned schemes.

2.2.1 Boundary value problem

We basically seek an unknown function u which satisfies a differential equation

$$\mathbf{A}(u) = \mathcal{L}u - f = 0 \tag{2.28}$$

in a domain Ω and certain boundary conditions

$$\mathbf{B}(u) = 0 \tag{2.29}$$

on the domain boundary Γ. In electromagnetics, the form of the governing differential equation ranges from the simple Poisson equation in statics to the complicated vector wave equation such as (2.17) and (2.18). The boundary conditions can also vary from Dirichlet and Neumann conditions to more complicated higher-order transition and radiation conditions.

2.2.2 The Ritz method

The Ritz method, or the Rayleigh-Ritz method, is a variational formulation where the solution to the boundary value problem is obtained by searching for the stationary

point of the functional. If the operator $\mathcal{L}$ in (2.28) is self-adjoint and positive-definite, then the solution to (2.28) can be found by determining the stationary point of the functional [1]

$$F(\hat{u}) = \frac{1}{2} < \mathcal{L}\hat{u}, \hat{u} > - < \hat{u}, f > \tag{2.30}$$

with respect to $\hat{u}$, where $\hat{u}$ denotes the trial function. The inner product, denoted by angular brackets, is given by

$$< a, b >= \int_{\Omega} a\ b\ dV \tag{2.31}$$

Once the functional is found, we approximate the trial function by the expression

$$\hat{u} = \sum_{i=1}^{N} C_i w_i = \mathbf{C}^T \mathbf{w} \tag{2.32}$$

where w_i are basis functions and C_i are constant coefficients to be determined. Substituting (2.32) in the expression for the functional, we get

$$F(\hat{u}) = \frac{1}{2}\mathbf{C}^T \left(\int_{\Omega} \mathbf{w}\mathcal{L}\mathbf{w}^T\ d\Omega \right) \mathbf{C} - \mathbf{C}^T \int_{\Omega} \mathbf{w} f\ d\Omega \tag{2.33}$$

where $\mathbf{C}$ is the column vector of unknown coefficients and the superscript denotes the transpose of a vector. On differentiating $F(\hat{u})$ with respect to $\mathbf{C}$ and setting the resultant expression to zero - equivalent to finding the stationary point of $F(\hat{u})$ - we obtain a system of equations

$$[\mathcal{A}]\{\mathbf{C}\} = \{\mathbf{b}\} \tag{2.34}$$

where the elements of the matrix $\mathcal{A}$ and the vector $\mathbf{b}$ are given by

$$\mathcal{A} = \int_{\Omega} w_i \mathcal{L} w_j\ d\Omega \tag{2.35}$$

$$\mathbf{b} = \int_{\Omega} w_i f\ d\Omega \tag{2.36}$$

On solving (2.34) for the unknown coefficients $\{C\}$ of the finite element bases, we obtain a solution for the desired quantity everywhere within the computational domain.

It should be pointed out that the inner product as defined in (2.31) extends the applicability of the variational formulation to complex numbers. This is of utmost importance in electromagnetics since material fillings are often lossy. However, unlike in other branches of science, the functional does not have any physical significance in electromagnetics and is thus used sparingly. A further limitation is that the matrix $\mathcal{A}$ must be symmetric for a variational principle to exist. Problems which give rise to unsymmetric matrices need to be handled differently. The weighted residual or the Galerkin method provides an alternative, and simpler, approach of formulating the finite element equations.

2.2.3 Galerkin's method

Galerkin's method or the weighted residual method tries to minimize a residual in the mean square sense. If we assume that $\hat{u}$ is an approximate solution to (2.28), on replacing u with $\hat{u}$ in (2.28) we obtain the residual

$$\mathcal{R} = \mathcal{L}\hat{u} - f \neq 0 \tag{2.37}$$

Naturally, the best approximation for $\hat{u}$ would be one that reduces the value of the residual to a minimum at all points in Ω. The integral of the residual, weighted with some known weighting functions w_i, is then required to vanish in Ω.

$$\int_{\Omega} \mathcal{R} w_i \, d\Omega = 0 \tag{2.38}$$

In the Galerkin method, the weighting functions w_i are chosen to be identical to the basis functions used for expanding $\hat{u}$. With this choice of weighting functions,

the residual $\mathcal{R}$ is orthogonal to the subspace of functions spanned by the basis of $\hat{u}$, ensuring that the resulting approximation is in this sense the best possible from the space of approximating functions. The expression (2.38) then becomes

$$\int_{\Omega} \left(w_i \mathcal{L} \mathbf{w}^T \mathbf{c} - w_i f \right) \, d\Omega = 0, \quad i = 1, \ldots, N \tag{2.39}$$

where $\hat{u}$ has been expanded as in (2.32). This again leads to a matrix system identical to the one given in (2.34), obtained by the Ritz method. One of the advantages of this formulation is that the matrix system need not be symmetric for its validity. Also, the method can be applied to problems for which no genuine extremum principles exist.

2.2.4 Implementation scheme

The implementation scheme for FEM follows three broad outlines.

Step 1: At first, the problem is discretized by dividing the entire computational domain into simple subdomains, the elements. For two-dimensional problems, commonly used elements are triangles, parallelograms and quadrilaterals with straight or curved edges. In three dimensional implementations, the elements of choice are usually terahedra, curvilinear bricks or prisms with straight or curved surfaces [12, 13].

Step 2: Next, a suitable approximation function is chosen for the problem to be solved. The form of approximation depends on the type of element and must satisfy certain continuity conditions across inter-element boundaries. Further, the form of the polynomial function must remain unchanged under a linear transformation from one Cartesian coordinate system to another. This requirement is satisfied if the polynomials are complete to a specific order like

$$u(x, y) = c_1 + c_2 x + c_3 y + c_4 x^2 + c_5 xy + c_6 y^2 \tag{2.40}$$

or when the extra terms are symmetric with respect to one another, as in the following incomplete third-order polynomial

$$u(x,y) = c_1 + c_2x + c_3y + c_4x^2 + c_5xy + c_6y^2 + c_7x^2y + c_8xy^2 \tag{2.41}$$

Such approximation functions have the characteristic that, for fixed x or y, they are always complete ploynomials in the other variable. The two examples shown above apply in two-dimensions; the extension to three-dimensional elements is trivial.

Once the order of the polynomial is selected, we can derive an expression for the unknown solution in an element, say the eth element, having the following form:

$$u^e = \sum_{k=1}^{n} u_k^e N_k^e \tag{2.42}$$

where n is the number of bases in the eth element, u_k^e is the value of the unknown function at node, edge or facet j and N_k^e is the basis (or shape) function for the element.

Step 3: In the third step, we enforce the extremum principle by substituting (2.42) into the functional expression (2.30) or into the residual value (2.37). On imposing the stationarity of the functional as explained in Section 2.2.2, we obtain the system of linear equations (2.34). An alternative procedure of minimizing the weighted residuals (2.38) yields an identical system of linear equation (2.34). As mentioned earlier, both the variational (Ritz) and the weighted residual (Galerkin) formulations are equivalent when the matrix system is symmetric. In addition, the Galerkin method can handle unsymmetric systems. Finally, the solution of (2.34) specifies the values of the desired function everywhere within the computational domain.

In the next chapter, we will be discussing step 2 in more detail. The two subsequent chapters are devoted to the derivation of the finite element equations for our

application and the optimizations undertaken to enable rapid solution of the system.

CHAPTER III

Shape functions for scalar and vector finite elements

The finite element method is used for modeling a wide class of problems by breaking up the computational domain into elements of simple shapes. Suitable interpolation polynomials (or shape functions) are used to approximate the unknown function within each element. It is then possible to program the computer to solve complicated geometries by specifying the shape functions only. The element choice, however, needs human intervention and intelligence to ensure a reliable solution of the of the problem at hand.

In this chapter, we will discuss the derivation of node-based and edge-based shape functions for two dimensional and three dimensional finite elements. Node-based shape functions have been used extensively in civil and mechanical engineering applications as well as in scalar electromagnetic problems. However, a full three dimensional vector formulation brings out numerous deficiencies in these traditional element shape functions [14, 15]. Edge-based shape functions have thus been derived to overcome the problems associated with nodal bases and are now being applied widely for solving vector problems in electromagnetics. We will also describe a general procedure for deriving higher-order shape functions for node and edge basis.

3.1 Node-based elements

In node-based finite elements, the form of the sought function in the element is controlled by the function values at its nodes. The approximating function can then be expressed as a linear combination of basis functions weighted by the nodal coefficients. If the function values $\hat{u}_i^e$ at the nodes are taken as nodal variables, then the approximating function for a two-dimensional element e with p nodes has the form

$$\hat{u}^e(x,y) = \sum_{i=1}^{p} \hat{u}_i^e N_i^e(x,y) \tag{3.1}$$

Since the expression (3.1) must be valid for any nodal variable $\hat{u}_i^e$, the basis function $N_i^e(x,y)$ must be unity at node i and zero for all remaining nodes within the element.

Shape functions can be derived either by inspection (Serendipity family) or through simple products of appropriate polynomials (Lagrange family). It is easier and more systematic to construct higher-order bases in the Lagrange family while progression to higher orders is difficult in the Serendipity family. However, Lagrange shape functions have undesirable interior nodes and more unknowns than Serendipity shape functions of the same order. All shape functions derived in the following sections impose function continuity or C_0 continuity (not slope continuity) between elements.

3.1.1 Two dimensional elements

Rectangular elements

The simple shape of the rectangular element permits its shape functions to be written down merely by inspection. On examining the element shape given in Figure (3.1), the shape functions can be cast in the form

$$N_1^e = \frac{1}{A^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right)$$

$$
\begin{aligned}
N_2^e &= \frac{1}{A^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right) \\
N_3^e &= \frac{1}{A^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right) \\
N_4^e &= \frac{1}{A^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right)
\end{aligned}
$$

where x_c^e and y_c^e denote the coordinates of the mid-points of the edges, h_x^e and h_y^e represent the edge length and A^e denotes the area of the element. Higher order

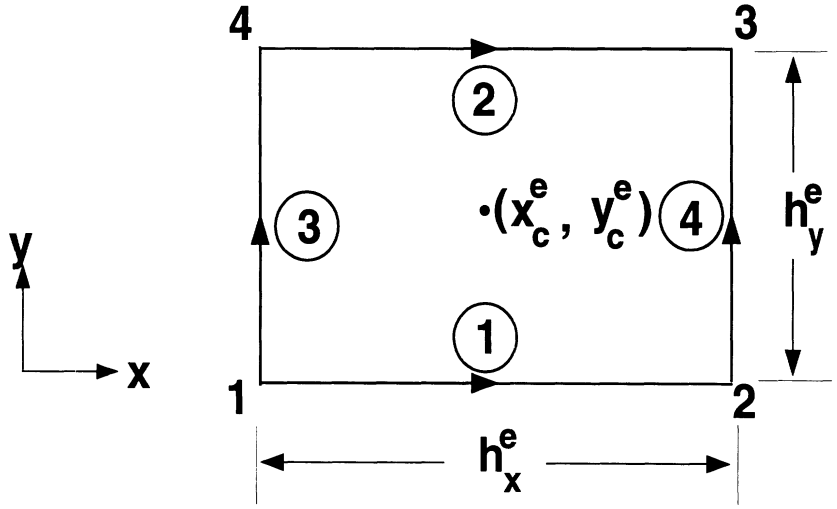

Figure 3.1: Rectangular element

rectangular elements are presented in Zienkiewicz [12]. However, these elements can model only regular geometries and are thus not very useful in practice.

Irregular geometries can be modeled by using quadrilateral elements which can also be viewed as distorted rectangles. To construct basis functions for a quadrilateral element, we need to use a transformation that maps a quadrilateral element in the xy-plane to a square element in the $\xi\eta$ plane (Figure 3.2). Such a transformation can be found by satisfying the following relation at the four nodes of the quadrilateral element:

$$x = a + b\xi + c\eta + d\xi\eta \quad y = a' + b'\xi + c'\eta + d'\xi\eta \tag{3.2}$$

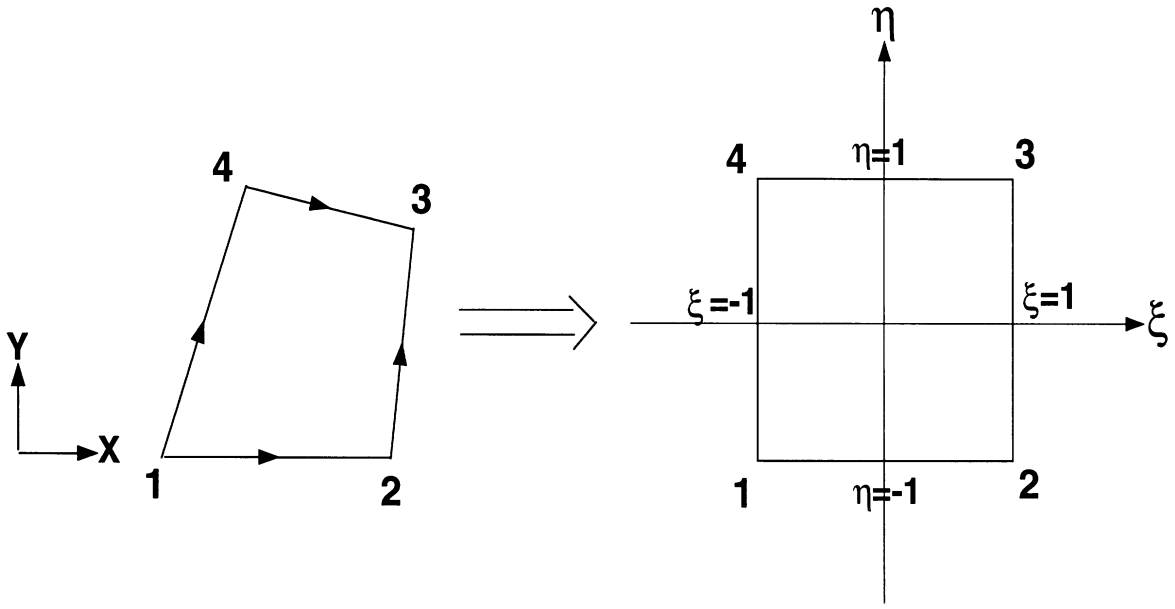

Figure 3.2: Transformation of a quadrilateral element in the xy plane to a unit square in the $\xi\eta$ plane

On solving for the unknown coefficients $a, \ldots, d$, the basis functions can be cast in the following form

$$N_i = \frac{1}{4}(1+\xi_o)(1+\eta_o), \quad i = 1, \ldots, 4 \tag{3.3}$$

where $\xi_o = \xi\xi_i$ and $\eta_o = \eta\eta_i$. The variables (ξ_i, η_i) denote the coordinates of the ith node in the (ξ, η) coordinate system.

Triangular elements

Triangular elements are popular since they can model arbitrary geometries. We will determine the shape functions of triangular elements by using Lagrange interpolation polynomials. Let us consider a point P within a triangular element (Figure 3.3). The area of the smaller triangle formed by points $p, 2$ and 3 is given by

$$\Delta_1 = \frac{1}{2}\begin{vmatrix} 1 & x & y \\ 1 & x_2^e & y_2^e \\ 1 & x_3^e & y_3^e \end{vmatrix} \tag{3.4}$$

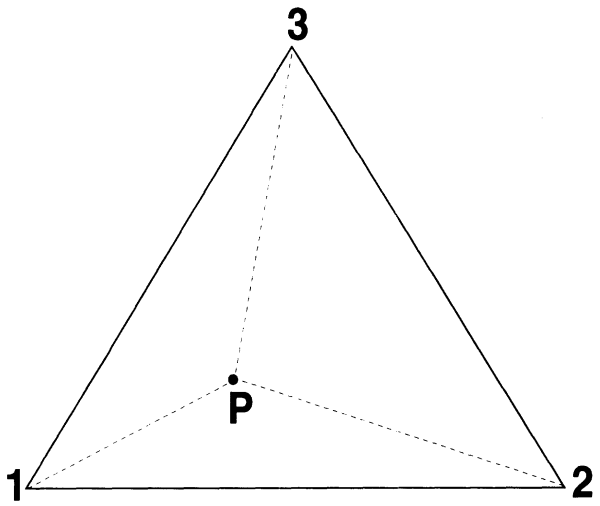

Figure 3.3: Triangular element

The area coordinate L_1^e is then given by

$$L_1^e = \frac{\Delta_1}{\Delta} \tag{3.5}$$

where Δ is the area of the whole triangle and can be found from (3.4) by replacing x and y in the first row with x_1 and y_1. Similarly, the two remaining area coordinates L_2 and L_3 are given by

$$L_2^e = \frac{\Delta_2}{\Delta} = \frac{\text{Area } P31}{\text{Area } 123} \tag{3.6}$$

$$L_3^e = \frac{\Delta_3}{\Delta} = \frac{\text{Area } P12}{\text{Area } 123} \tag{3.7}$$

The values for x and y inside the triangular element reduce to

$$x = \sum_{i=1}^{3} L_i^e x_i^e \quad y = \sum_{i=1}^{3} L_i^e y_i^e \tag{3.8}$$

The area coordinates are equal to the basis functions - $N_i^e, i = 1, 2, 3$ - when the required interpolation order is linear. Higher order basis functions for triangular and quadrilateral elements are derived in [12, 13].

3.1.2 Three dimensional elements

Shape functions for three dimensional elements can be described in a precisely analogous way to their two dimensional counterpart. However, the simple rules for inter-element continuity given previously must be modified. The nodal field values should now interpolate to give continuous fields across the face of each element.

Rectangular bricks

The simplest polynomial approximation to a rectangular brick element is the trilinear function

$$\hat{u}^e(x,y,z) = a^e + b^e x + c^e y + d^e z + e^e xy + f^e yz + g^e zx + h^e xyz \tag{3.9}$$

whose eight parameters are uniquely defined by the values of the function $\hat{u}$ at the eight corners of the brick. From the eight resulting equations, we can determine the coefficients $a^e, b^e, \ldots, h^e$ and write the final expression in the form

$$\hat{u}^e(x,y,z) = \sum_{i=1}^{8} N_i^e(x,y,z)\hat{u}_i^e \tag{3.10}$$

However, this is a cumbersome process and can be easily avoided by writing down the required basis functions by mere inspection. Since the basis function N_i^e must be unity at node i and zero at the remaining nodes, the eight interpolation functions can be written down as

$$\begin{aligned}
N_1^e &= \frac{1}{V^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right)\left(z_c^e + \frac{h_z^e}{2} - z\right) \\
N_2^e &= \frac{1}{V^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right)\left(z_c^e + \frac{h_z^e}{2} - z\right) \\
N_3^e &= \frac{1}{V^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right)\left(z_c^e + \frac{h_z^e}{2} - z\right) \\
N_4^e &= \frac{1}{V^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right)\left(z_c^e + \frac{h_z^e}{2} - z\right) \\
N_5^e &= \frac{1}{V^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right)\left(-z_c^e + \frac{h_z^e}{2} + z\right)
\end{aligned}$$

$$
\begin{aligned}
N_6^e &= \frac{1}{V^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(y_c^e + \frac{h_y^e}{2} - y\right)\left(-z_c^e + \frac{h_z^e}{2} + z\right) \\
N_7^e &= \frac{1}{V^e}\left(-x_c^e + \frac{h_x^e}{2} + x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right)\left(-z_c^e + \frac{h_z^e}{2} + z\right) \\
N_8^e &= \frac{1}{V^e}\left(x_c^e + \frac{h_x^e}{2} - x\right)\left(-y_c^e + \frac{h_y^e}{2} + y\right)\left(-z_c^e + \frac{h_z^e}{2} + z\right)
\end{aligned}
$$

where x_c^e, y_c^e, and z_c^e denote the coordinates of the center of the element, h_x^e, h_y^e, and h_z^e represent the edge lengths of the element and V^e denotes the element volume. Bricks with quadratic interpolation functions need 20 degrees of freedom and thus have node points at the corners and the mid-points of each edge.

Shape functions for hexahedral elements or distorted bricks can be derived by mapping the element in the xyz coordinate system onto a standard cube in a new $\xi\eta\zeta$ coordinate system. The required transformation yields

$$
x = \sum_{i=1}^{8} N_i^e(\xi,\eta,\zeta)x_i^e \;;\; y = \sum_{i=1}^{8} N_i^e(\xi,\eta,\zeta)y_i^e \;;\; z = \sum_{i=1}^{8} N_i^e(\xi,\eta,\zeta)z_i^e \tag{3.11}
$$

where

$$
N_i^e = \frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta) \tag{3.12}
$$

with (ξ_i, η_i, ζ_i) denoting the coordinates of the ith node.

Tetrahedral elements

The three dimensional analogue of a two-dimensional triangle is a tetrahedron (four-faced element). Once again, we can introduce special coordinates, called volume coordinates or simplex coordinates, to simplify the derivation of shape functions. If P is a point within the tetrahedron shown in Figure 3.4, the four volume coordinates are given by

$$
\begin{aligned}
L_1 &= \frac{\text{Volume } P234}{\text{Volume } 1234} \\
L_2 &= \frac{\text{Volume } P341}{\text{Volume } 1234}
\end{aligned}
$$

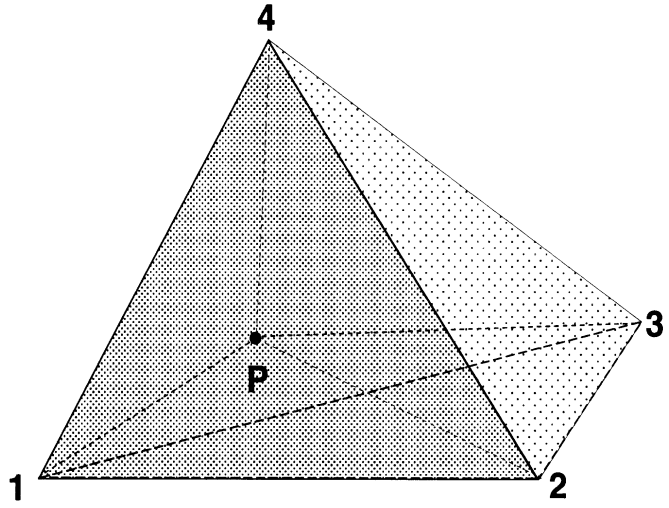

Figure 3.4: Tetrahedral element

$$\begin{aligned} L_3 &= \frac{\text{Volume } P412}{\text{Volume } 1234} \\ L_4 &= \frac{\text{Volume } P123}{\text{Volume } 1234} \end{aligned} \tag{3.13}$$

and any position within the element is specified by

$$x = \sum_{i=1}^{4} L_i x_i \; ; \; y = \sum_{i=1}^{4} L_i y_i \; ; \; z = \sum_{i=1}^{4} L_i z_i$$

Quadratic shape functions for a tetrahedron necessitates the use of ten node points - the 4 corner nodes and the remaining 6 on the mid-points of the edges.

Other elements

Other three-dimensional elements having simple shapes include the triangular prism and isoparametric elements. It is much easier to discretize a complicated structure by using parallelopipeds in combination with prism elements. To ensure that a small number of elements can model a relatively complex region, curved or isoparametric elements can be used. There is a good review of such elements in [12, 16].

3.2 Edge-based elements

In electromagnetics, we encounter several serious problems when node-based elements are employed to represent vector electric or magnetic fields. First, spurious modes are observed when modeling cavity problems using node-based elements [17]. Secondly, special care needs to be taken to impose boundary conditions at material interfaces and conducting surfaces [18]. The first limitation can also jeopardize the near-field results of a scattering problem, the far-field escapes contamination since spurious modes do not radiate.

Edge-based finite elements, whose degrees of freedom are associated with the edges of the finite element mesh, have been shown to be free of the above shortcomings. They were described by Whitney [19] over 35 years ago and have been revived by Nedelec [20] and Bossavit and Verite [14] and Hano [21]. Mur and de Hoop [22], van Welij [23], Barton and Cendes [24] and Lee,et al [25] have extended their applicability to various two- and three-dimensional shapes and even constructed higher order elements for a more accurate approximation of the field values.

3.2.1 Two dimensional elements

Rectangular elements

We first consider the rectangular element first since its shape function is usually the easiest to formulate. For the element shown in Figure 3.1, we can find its edge-based finite element basis function merely by inspection. If the edges are numbered according to Table 3.1 and considering that the basis function should be unity along one edge and zero over all others, the vector basis functions can be written as

$$\boldsymbol{W}_1^e = \frac{1}{h_y^e}\left(-y + y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{x}}$$

Edge no.	i_1	i_2
1	1	2
2	4	3
3	1	4
4	2	3

Table 3.1: Edge numbering for rectangular element

$$\begin{aligned}
\boldsymbol{W}_2^e &= \frac{1}{h_y^e}\left(y - y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{x}} \\
\boldsymbol{W}_3^e &= \frac{1}{h_x^e}\left(-x + x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}} \\
\boldsymbol{W}_4^e &= \frac{1}{h_x^e}\left(x - x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}}
\end{aligned}$$

where $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ are the unit vectors in the Cartesian coordinate system. The electric field within the finite element is then given by

$$\mathbf{E}^e = \sum_{i=1} E_i^e \boldsymbol{W}_i^e \tag{3.14}$$

where E_i^e denotes the tangential field along the ith edge. The basis functions N_i^e guarantee tangential continuity across inter-element boundaries since they have a tangential component only along the ith edge and none along the other edges. They are also divergenceless within the element and possess a constant non-zero curl. It should be noted that by taking the cross-product of $\hat{\mathbf{z}}$ with $\boldsymbol{W}_i^e$, we obtain basis functions which possess normal continuity across element boundaries, have zero curl and non-zero divergence. The latter are ideal for representing surface current densities and are known as roof-top basis functions in electromagnetics. They have found extensive use in the solution of integral equations [26].

Edge basis for quadrilateral elements can be derived by carrying out the transformation detailed in the derivation of nodal basis for quadrilaterals in the previous

section and then taking the gradient of the resulting expression for each edge. The edge-based quadrilateral element has two shortcomings. First, the integrals associated with these elements do not lend themselves to easy evaluation and secondly, the basis functions are not divergence free. However, their ability to model complicated shapes with a lesser number of unknowns than tetrahedra and property of tangential continuity across elements make them attractive for use in two-dimensional vector formulations.

Triangular elements

Since the edges of an arbitrary triangular element are not parallel to the x or y axis, it is not easy to guess the form of the vector basis function by inspection. Therefore, the edge basis for a triangular element is expressed in terms of its area coordinates, L_1, L_2 and L_3. These are the so-called *Whitney* elements. If the local edge numbers are defined according to Table 3.2, then edge bases for a triangular element are defined as

$$\boldsymbol{W}_k = \boldsymbol{N}_{ij} = L_i \nabla L_j - L_j \nabla L_i, \quad i,j = 1,\ldots,3 \tag{3.15}$$

where $\boldsymbol{W}_k$ denotes the basis function for the kth edge of the element. The vector field inside the triangular element can, therefore, be expanded as

$$\mathbf{E}^e = \sum_{k=1}^{3} E_k^e \boldsymbol{W}_k^e \tag{3.16}$$

where E_k^e denotes the tangential field along the kth edge. It can be easily shown that the edge-based functions defined in (3.15) has the following properties within the element:

$$\begin{aligned} \nabla \cdot \boldsymbol{N}_{ij} &= 0 \\ \nabla \times \boldsymbol{N}_{ij} &= 2\nabla L_i \times \nabla L_j \end{aligned}$$

Edge no.	i_1	i_2
1	1	2
2	2	3
3	3	1

Table 3.2: Edge numbering for triangular element

If $\hat{\mathbf{e}}_1$ is the unit vector pointing from node 1 to node 2 in Figure 3.3, then $\hat{\mathbf{e}}_1 \cdot \nabla L_1 = -1$ and $\hat{\mathbf{e}}_1 \cdot \nabla L_2 = 1$. Since L_1 is a linear function that varies from unity at node 1 and zero at node 2 and L_2 is unity at node 2 and zero at node 1, we have

$$\hat{\mathbf{e}}_1 \cdot \boldsymbol{N}_{12} = L_1 + L_2 = 1 \tag{3.17}$$

along the entire length of edge 1. This implies that $\boldsymbol{N}_{12}$ has a constant tangential component along edge 1. Moreover, since L_1 vanishes along edge 2 and L_2 vanishes along edge 3, $\boldsymbol{N}_{12}$ has no tangential component along these edges. Thus, tangential continuity is preserved across inter-element boundaries but normal continuity is not. A different method of constructing edge basis functions for triangular elements is given in [27].

Higher order vector basis functions include the contribution of *facet* elements to the approximating function. Unknowns in the triangular element are assigned as shown in Figure 3.5 [28]. The tangential projection of the vector field along edge $\{i, j\}$ is determined by two unknowns E_i^j and E_j^i and two facet unknowns - F_1 and F_2 - are provided to allow a quadratic approximation of the normal component along two of the three edges. Only two facet unknowns are required to make the basis functions of second order complete. Therefore, there are 8 degrees of freedom for

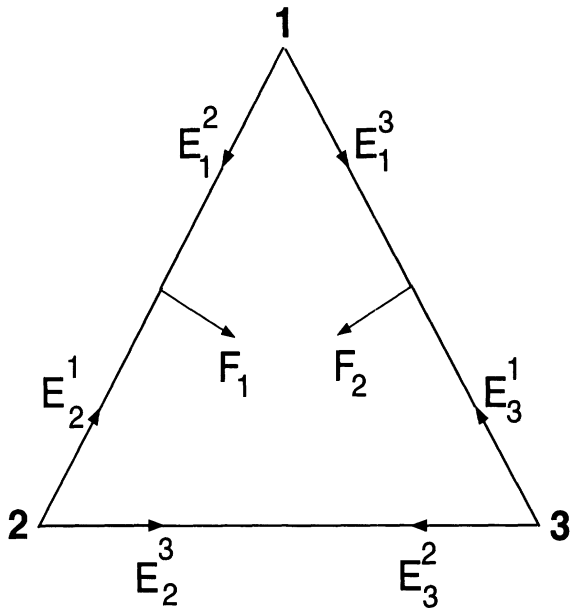

Figure 3.5: Second order triangular edge element

each triangular element. The higher order vector field within the element is given by

$$\mathbf{E}^e = \sum_{\substack{i=1 \\ i\neq j}}^{3}\sum_{j=1}^{3} E_i^{je} \mathrm{L}_i \nabla L_j + F_1^e L_1 L_2 \nabla L_3 + F_2^e L_1 L_3 \nabla L_2 \tag{3.18}$$

where we have arbitrarily chosen the facet variables to lie on edges 1 and 2. These variables are local unknowns associated with each separate triangular element and are included to provide a linear approximation for $\nabla_t \times \mathbf{E}_t$, where the subscript t denotes the tangential component. Since the edge variables provide common unknowns across element boundaries, tangential continuity of the field over the boundary is assured. However, an obvious disadvantage of these elements is that the 2 facet variables cannot be symmetrically assigned. This disadvantage can be avoided by using third order elements [29].

3.2.2 Three dimensional elements

Edge-based elements have facilitated to a great degree the finite element analysis of three dimensional structures in electromagnetics. Linear nodal basis with their problem of spurious modes and difficulty in maintaining only tangential continuity across material interfaces are not as convenient for electromagnetic field simulations in three dimensions. On the other hand, the introduction of edge based shape functions provide a robust way of treating general three dimensional problems having material inhomogeneities and structural irregularities like sharp edges and corners.

In the following section, we will consider the simple rectangular bricks first and will proceed to derive edge-based shape functions for more complicated structures like tetrahedrals and curvilinear hexahedrals. The chapter is concluded with a brief discussion on hierarchical edge elements.

Rectangular bricks and hexahedrals

As in the two dimensional case, we derive the edge-based shape function for a rectangular brick (see Figure 3.6) by simple inspection. Since a constant tangential field component must be assigned to each edge of the element, we can express the shape function along each edge of the element as [30]

$$
\begin{aligned}
\boldsymbol{W}_1^e &= \frac{1}{h_y^e h_z^e}\left(-y + y_c^e + \frac{h_y^e}{2}\right)\left(-z + z_c^e + \frac{h_z^e}{2}\right)\hat{\mathbf{x}} \\
\boldsymbol{W}_2^e &= \frac{1}{h_y^e h_z^e}\left(y - y_c^e + \frac{h_y^e}{2}\right)\left(-z + z_c^e + \frac{h_z^e}{2}\right)\hat{\mathbf{x}} \\
\boldsymbol{W}_3^e &= \frac{1}{h_y^e h_z^e}\left(-y + y_c^e + \frac{h_y^e}{2}\right)\left(z - z_c^e + \frac{h_z^e}{2}\right)\hat{\mathbf{x}} \\
\boldsymbol{W}_4^e &= \frac{1}{h_y^e h_z^e}\left(y - y_c^e + \frac{h_y^e}{2}\right)\left(z - z_c^e + \frac{h_z^e}{2}\right)\hat{\mathbf{x}} \\
\boldsymbol{W}_5^e &= \frac{1}{h_y^e h_z^e}\left(-z + z_c^e + \frac{h_z^e}{2}\right)\left(-x + x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}}
\end{aligned}
$$

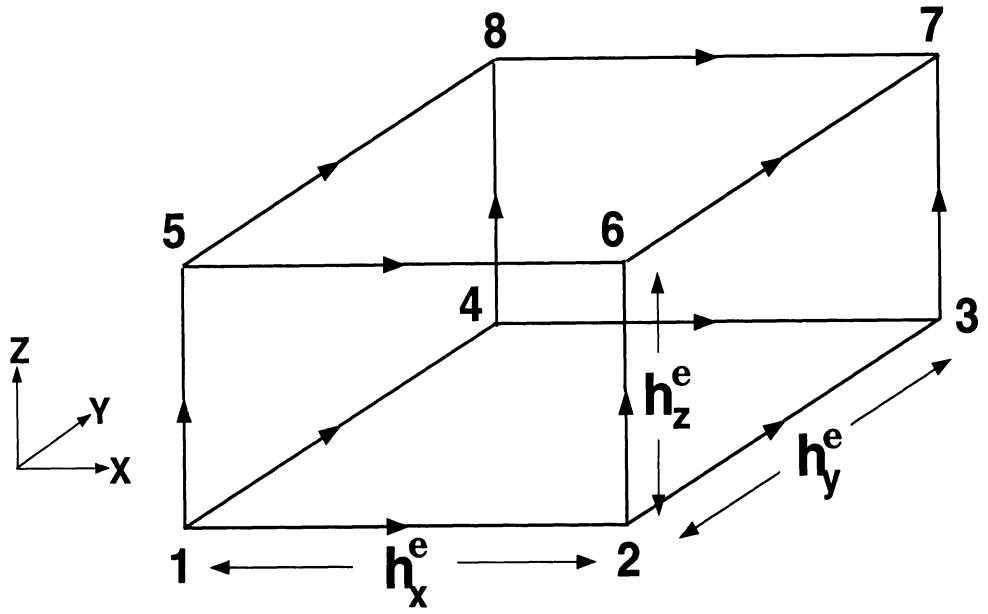

Figure 3.6: Rectangular brick element

$$
\begin{aligned}
\boldsymbol{W}_6^e &= \frac{1}{h_y^e h_z^e}\left(z - z_c^e + \frac{h_z^e}{2}\right)\left(-x + x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}} \\
\boldsymbol{W}_7^e &= \frac{1}{h_y^e h_z^e}\left(-z + z_c^e + \frac{h_z^e}{2}\right)\left(x - x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}} \\
\boldsymbol{W}_8^e &= \frac{1}{h_y^e h_z^e}\left(z - z_c^e + \frac{h_z^e}{2}\right)\left(x - x_c^e + \frac{h_x^e}{2}\right)\hat{\mathbf{y}} \\
\boldsymbol{W}_9^e &= \frac{1}{h_y^e h_z^e}\left(-x + x_c^e + \frac{h_x^e}{2}\right)\left(-y + y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{z}} \\
\boldsymbol{W}_{10}^e &= \frac{1}{h_y^e h_z^e}\left(x - x_c^e + \frac{h_x^e}{2}\right)\left(-y + y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{z}} \\
\boldsymbol{W}_{11}^e &= \frac{1}{h_y^e h_z^e}\left(-x + x_c^e + \frac{h_x^e}{2}\right)\left(y - y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{z}} \\
\boldsymbol{W}_{12}^e &= \frac{1}{h_y^e h_z^e}\left(x - x_c^e + \frac{h_x^e}{2}\right)\left(y - y_c^e + \frac{h_y^e}{2}\right)\hat{\mathbf{z}}
\end{aligned}
$$

where h_x^e, h_y^e, h_z^e denote the edge lengths in the x, y, and z directions, respectively, and the center coordinates of the brick are given by (x_c^e, y_c^e, z_c^e). If the edge numbers are defined as in Table 3.3, the expression for the vector field within the element can

be expressed as

$$\mathbf{E}^e = \sum_{i=1}^{12} E_i^e \boldsymbol{W}_i^e \tag{3.19}$$

where E_i^e represents the value of the electric field along the ith edge. The vector basis

Edge no.	Node i_1	Node i_2
1	1	2
2	4	3
3	5	6
4	8	7
5	1	4
6	5	8
7	2	3
8	6	7
9	1	5
10	2	6
11	4	8
12	3	7

Table 3.3: Edge definition for rectangular brick

N_i^e defined for the rectangular brick element have zero divergence and a nonzero curl. Furthermore, the expansion (3.19) guarantees tangential continuity of the electric field across the surfaces of the elements.

A rectangular brick element has limitations in the sense that it is unable to model irregular geometries. Due to this reason, the analog of the two dimensional quadrilateral (the hexahedral element) finds much wider use in modeling practical three dimensional problems. As in the case of the quadrilateral element in two

dimensions, a hexahedral element in Cartesian coordinates can be seen as the image of a unit cube under a trilinear mapping to the $\xi\eta\zeta$ coordinate system (see Figure 3.7).

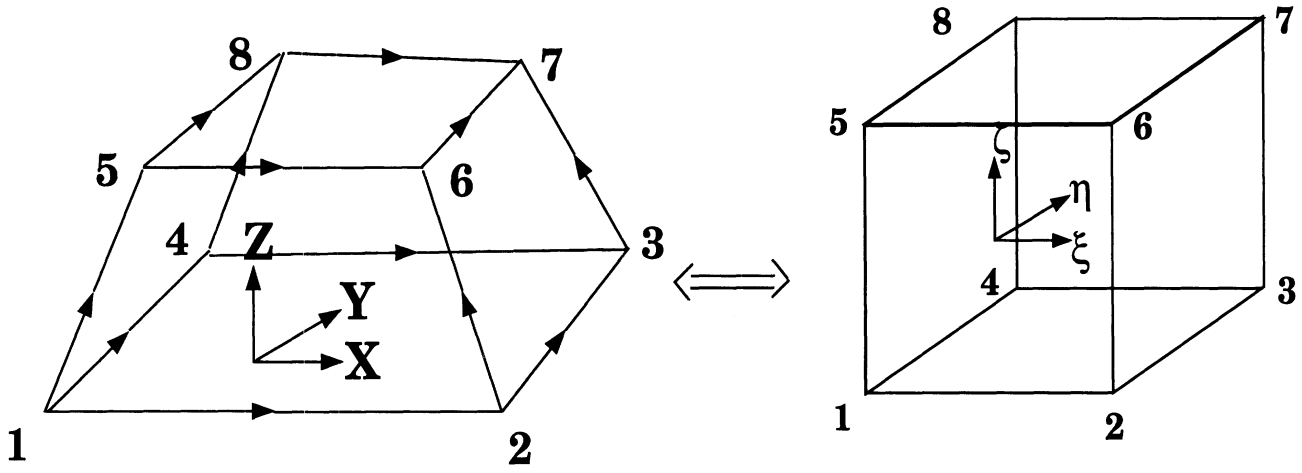

Figure 3.7: Mapping of a hexahedral element to a unit cube

Let us consider those faces for which ξ =constant. Therefore, $\nabla\xi$ must then possess only a normal component on that face. Since ξ varies linearly along the edges that are parallel to the ξ-axis, the vector function $\nabla\xi$ has nonzero tangential components only along those edges that are parallel to the ξ-axis. Using the node-based expression for the shape function in a hexahedral element given in (3.12), we may write the corresponding edge bases as

$$\boldsymbol{W}_i^e = \frac{h_i^e}{8}(1+\eta_i\eta)(1+\zeta_i\zeta)\nabla\xi, \quad \text{edges} \parallel \text{to } \xi\text{-axis} \tag{3.20}$$

$$\boldsymbol{W}_i^e = \frac{h_i^e}{8}(1+\xi_i\xi)(1+\zeta_i\zeta)\nabla\eta, \quad \text{edges} \parallel \text{to } \eta\text{-axis} \tag{3.21}$$

$$\boldsymbol{W}_i^e = \frac{h_i^e}{8}(1+\xi_i\xi)(1+\eta_i\eta)\nabla\zeta, \quad \text{edges} \parallel \text{to } \zeta\text{-axis} \tag{3.22}$$

where (ξ_i, η_i, ζ_i) denote the coordinates at edge i.

The vector bases derived above possess all the desired continuity properties of edge elements and generally result in about half the number of unknowns than that

obtained by tetrahedral gridding. However, these basis functions are not divergence-less and it is difficult to generate a finite element mesh of an arbitrary structure using hexahedral elements.

Tetrahedral elements

Tetrahedral elements are, by far, the most popular element shapes to be employed for three dimensional applications. This is because the tetrahedral element is the simplest tessellation shape capable of modeling arbitrary three dimensional geometries and is also well-suited for automatic mesh generation. The derivation of shape functions for these elements follow the same pattern as that for triangular vector basis functions. If we consider the tetrahedron shown in Figure 3.4 and define the edge numbers according to Table 3.4, we have

$$\boldsymbol{W}_k^e = \boldsymbol{N}_{ij}^e = L_i^e \nabla L_j^e - L_j^e \nabla L_i^e, \quad i,j = 1,\ldots,4 \tag{3.23}$$

and the vector field within the element can be expanded as

$$\mathbf{E} = \sum_{k=1}^{6} E_k^e \boldsymbol{W}_k^e \tag{3.24}$$

A nice explanation of the physical character of the edge-based interpolation function is given by Bossavit [31]. Let us consider edge number 1 connecting nodes 1 and 2. Since ∇L_2 is orthogonal to facet $\{134\}$ and ∇L_1 is orthogonal to facet $\{234\}$, the field turns around the axis 3-4 and is normal to planes containing 3 and 4. The field thus has only tangential continuity across element faces. Edge elements can also be described as Whitney elements of degree 1.

Whitney elements of the second degree are called *facet* elements because they are constant over the face of the tetrahedron. The vector function for the facet element can be written as

$$\boldsymbol{N}_{ijk} = 2\left(L_i \nabla L_j \times \nabla L_k + L_j \nabla L_k \times \nabla L_i + L_k \nabla L_i \times \nabla L_j\right), \quad i,j,k = 1,\ldots,4 \tag{3.25}$$

Edge no.	Node i_1	Node i_2
1	1	2
2	1	3
3	1	4
4	2	3
5	4	2
6	3	4

Table 3.4: Edge definition for tetrahedron

As explained in [31], we now have a central field (emanating as if from node 4 in Figure 3.4) on each of the two tetrahedra that share the face {1,2,3}. The field can be imagined as coming from the 'source' 4, growing, crossing the facet and vanishing into the 'well' $4'$, the fourth vertex of the other tetrahedron. This field thus has normal continuity and the flux across the facet forms the degree of freedom for the element.

Alternative expressions for linear basis inside a tetrahedron have been derived in [24]. They are given by

$$\boldsymbol{W}^e_{7-i} = \begin{cases} \mathbf{f}_{7-i} + \mathbf{g}_{7-i} \times \mathbf{r}, & \mathbf{r} \text{ in the tetrahedron} \\ 0, & \text{otherwise} \end{cases} \tag{3.26}$$

with

$$\mathbf{f}_{7-i} = \frac{h_{7-i}}{6V_e}\mathbf{r}_{i_1} \times \mathbf{r}_{i_2} \tag{3.27}$$

$$\mathbf{g}_{7-i} = \frac{h_i h_{7-i}\hat{\mathbf{e}}_i}{6V_e} \tag{3.28}$$

in which $i = 1, 2, \ldots, 6$, V_e is the volume of the tetrahedral element, $\hat{\mathbf{e}}_i = (\mathbf{r}_{i_2} - \mathbf{r}_{i_1})/h_i$ is the unit vector of the ith edge and $h_i = |\mathbf{r}_{i_2} - \mathbf{r}_{i_1}|$ is the length of the ith edge

with $\mathbf{r}_{i_1}$ and $\mathbf{r}_{i_2}$ denoting the position vector of the i_1 and i_2 nodes. It can be shown that (3.23) is identically equal to (3.26) when simplified. Therefore,

$$\mathbf{g}_{7-i} = h_{7-i}\left(\nabla L_{i_1} \times \nabla L_{i_2}\right), \quad i = 1, \ldots, 6$$

where i_1 and i_2 are given in Table 3.4. The basis functions given in (3.26) have zero divergence and constant curl ($\nabla \times \boldsymbol{W}_i^e = 2\mathbf{g}_i^e$).

The order of the polynomial approximation for the first order edge element given in (3.23) or (3.26) can be taken as 0.5. This is because the value of the basis function is constant ($O(1)$) along the edge it supports and is linear ($O(r)$) everywhere else within the element. Mur and de Hoop [22] presented edge elements which are consistently linear, yielding a linear approximation of the field both inside each tetrahedron and along its edges and faces. Since this requires two unknowns per edge, there are 12 degrees of freedom per element. The basis functions in [32] are derived by first defining the outwardly directed vectorial areas of the faces as

$$\boldsymbol{A}_i = \boldsymbol{r}_j \times \boldsymbol{r}_k + \boldsymbol{r}_k \times \boldsymbol{r}_l + \boldsymbol{r}_l \times \boldsymbol{r}_j \tag{3.29}$$

where $\boldsymbol{r}_i, i = 1, \ldots, 4$ denote the position vectors of the vertices of the tetrahedron and i, j, k, l are cyclic. Then the edge-based vectorial expansion function is defined by

$$\boldsymbol{N}_{ij}(\boldsymbol{r}) = -\frac{\phi_i(\boldsymbol{r})\boldsymbol{A}_j}{3V}, \quad i, j = 1, \ldots, 4, i \neq j \tag{3.30}$$

where V is the volume of the tetrahedron and $\phi(\boldsymbol{r})$ is a linear scalar function of position given by

$$\phi_i(\boldsymbol{r}) = \frac{1}{4} - \frac{(\boldsymbol{r} - \boldsymbol{r}_b) \cdot \boldsymbol{A}_i}{3V}$$

in which $\boldsymbol{r}_b$ is the position vector of the centroid of the tetrahedron. We observe that $\phi_i(\boldsymbol{r})$ equals unity when $\boldsymbol{r} = \boldsymbol{r}_i$ and zero for the remaining vertices of the tetrahedral

element. In that sense, they are very similar to the simplex or volume coordinates mentioned earlier. They also satisfy the following equalities:

$$\begin{aligned} \boldsymbol{r} &= \sum_{i=1}^{4} \phi_i(\boldsymbol{r}) \boldsymbol{r}_i \\ \sum_{i=1}^{4} \phi_i(\boldsymbol{r}) &= 1 \end{aligned}$$

The edge basis function $\boldsymbol{N}_{ij}$ is a linear vector function of position inside the tetrahedral element and its tangential component vanishes on all edges of the element except the one joining vertices i and j. $\boldsymbol{N}_{ij}$ varies linearly along the edge formed by nodes i and j such that $\boldsymbol{N}_{ij} \cdot \boldsymbol{r}_j = 0$ while

$$\boldsymbol{N}_{ij} \cdot (\boldsymbol{r}_i - \boldsymbol{r}_j) = 1$$

These basis functions have non-zero values of divergence and curl.

An inspection of the expressions for the vectorial areas reveals that the form is identical to that obtained by taking the gradient of one of the simplex or volume coordinates mentioned earlier. In other words, the three components of the vector $\boldsymbol{A}_1$ have the same functional dependence as that obtained by ∇L_1

$$\nabla L_1 = \frac{1}{6V} \left[\hat{\mathbf{x}} \; det \begin{vmatrix} y_2 & 1 & z_2 \\ y_3 & 1 & z_3 \\ y_4 & 1 & z_4 \end{vmatrix} - \hat{\mathbf{y}} \; det \begin{vmatrix} x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \\ x_4 & 1 & z_4 \end{vmatrix} + \hat{\mathbf{z}} \; det \begin{vmatrix} x_2 & 1 & y_2 \\ x_3 & 1 & y_3 \\ x_4 & 1 & y_4 \end{vmatrix} \right]$$

where L_1 is the volume coordinate for a tetrahedron defined in (3.13), *det* indicates the value of the determinant of the matrix and x_i, y_i, z_i denote the coordinates of the ith vertex. The basis functions with consistently linear interpolation in the tetrahedron can now be rewritten in more convenient notation as

$$\boldsymbol{W}_k = \boldsymbol{N}_{ij} = L_i \nabla L_j, \quad i,j = 1,\ldots,4, i \neq j \tag{3.31}$$

Still higher order basis functions are sometimes necessary for rapidly changing fields or for modeling extremely thin structures where linear interpolation for the highly stretched elements is not enough. The second order edge basis ($O(r^{1.5})$) for a tetrahedral element was first presented by Lee, Sun and Cendes [32]. We need 20 degrees of freedom to achieve a quadratic approximation of the vector field inside a tetrahedron (see Figure 3.8). Accordingly, the field within a tetrahedron can be written as

$$\mathbf{E} = \sum_{\substack{i=1 \\ i\neq j}}^{4}\sum_{j=1}^{4} E_i^j L_i \nabla L_j + \sum_{i=1}^{4}\left(F_1^i L_i L_j \nabla L_k + F_2^i L_i L_k \nabla L_j\right) \tag{3.32}$$

where i, j, k form cyclic indices. The facet variables F_1^i and F_2^i are common unknowns

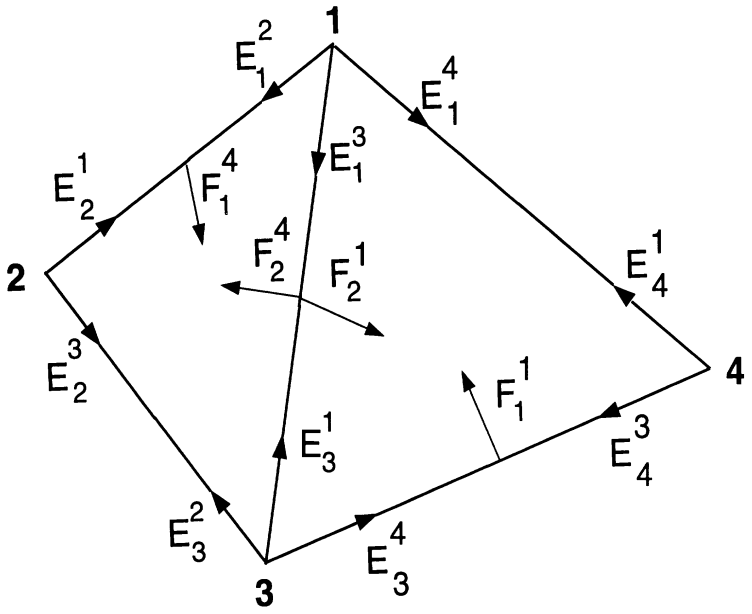

Figure 3.8: Tetrahedral element

for two tetrahedra that share the same face. Even higher order edge-based elements up to polynomial order 2 can be constructed. Each tetrahedral element now has 30 unknowns - 3 along each edge and 3 on each face.

Other elements

Recently, Wang and Ida proposed a systematic method for the construction of curvilinear elements in [33]. The vector shape function is expressed in the following form:

$$\boldsymbol{W}_i(\boldsymbol{r}) = \phi_i(\xi, \eta, \zeta)\boldsymbol{v}_i(\boldsymbol{r}), \quad i = 1, \ldots, M \tag{3.33}$$

where $\phi_i(\xi, \eta, \zeta)$ are completely defined in the local coordinate system, $\boldsymbol{v}_i$ contains the edge and facet information and M denotes the number of degrees of freedom in the element. These basis functions usually lead to a symmetric system of equations. However, it is difficult to find commercial mesh generation packages which construct curvilinear elements for a wide range of geometrical configurations.

Hierarchical vector elements

Finite elements are said to be hierarchical when the basis functions for an element are a subset of the basis functions for any elment of higher order [12]. The basis functions described in [34] are hierarchical and tangentially continuous. Vector elements complete upto polynomials of order 2 are available and basis functions of a given order are fully compatible to be used with basis functions of lower or higher orders. Thus elements of different orders could be used in the same mesh - lower order elements could be used in regions where field variation is uniform and higher order elements employed in regions where the field varies rapidly.

The implementation of hierarchical vector elements can be a bit tricky, especially at the transition boundaries where elements of one order merge into the elements of another order. If several vector elements share an edge, the field tangent to the edge can be made identical in each of the tetrahedra. This is done by setting the coefficient of the corresponding basis function for the edge in all tetrahedra to be identical. For tangential continuity across a face, the same equality must be enforced

between the coefficients of all the edge and facet functions associated with the face. Table 3.5 gives the basis functions for hierarchical vector finite elements. Higher order basis functions are constructed by systematically adding the extra terms upto the desired order. It should be noted that the bases for the tetrahedron with 6 and

Element type	Polynomial order	Unknowns per element	Basis function
Edge	0.5	6	$L_i \nabla L_j - L_j \nabla L_i$
Edge	1	12	$\nabla (L_i L_j)$
Face Face	1.5	20	$L_k L_j \nabla L_i$ $L_k L_i \nabla L_j$
Edge Face	2	30	$\nabla [L_i L_j (L_i - L_j)]$ $\nabla [L_i L_j L_k]$

Table 3.5: Hierarchical basis functions for tetrahedron

20 unknowns shown in Table 3.5 is identical to the linear and second order edge basis given in (3.23) and (3.33), respectively.

CHAPTER IV

Vector finite elements for 3D electromagnetic problems

Finite elements have been used extensively to model open and closed domain electromagnetic problems in scalar form in two and three dimensions[13, 35, 36]. But a reliable full vector formulation proved to be extremely difficult to implement. The cause of the problem was found to be the traditional nodal basis functions that were being used to discretize the unknown field variable. The reasons for the failure of node-based elements in modeling the vector wave equation will be discussed in a later section. Fortunately, a novel remedy was found by assigning the degrees of freedom to the edges rather than to the nodes of elements. These types of elements had been described by Whitney[19] in terms of geometrical forms about 35 years back and were revived by Nedelec [20] in 1980. In recent years, Bossavit [14] and others [21, 22, 23, 24] applied these edge-based finite elements successfully to model three dimensional problems. In all these works, edge elements were seen to be devoid of the shortcomings commonly experienced with node-based elements.

The goal of this thesis was to develop a general purpose code for computing the scattering pattern of three dimensional composite structures having complex shapes. Edge based functions with their robustness in modeling general three dimensional

problems were evidently our choice. Since the only simple shape to model an arbitrary three dimensional space is a tetrahedron, we settled on edge-based tetrahedra as our mesh discretization units. The mesh truncation was chosen to be done with absorbing boundary conditions (ABCs) which are local in nature, preserve the sparsity of the finite element system and permit scalability to large problems with minimal storage -$O(N)$- and computational time - $O(k * N), k \ll N$ - penalties.

In the first part of this chapter, we present the weighted residual or weak formulation for the closed domain problem and solve for the eigenvalues of a metallic cavity having arbitrary shape. In passing, we briefly describe the problem of *spurious* solutions encountered with node-based elements. We also validate our methodology by comparing the computed eigenvalues with analytically derived ones. In the latter part of the chapter (Part II), we formulate the open domain problem in terms of the variational functional, describe the enforcement of boundary conditions for perfectly conducting and composite targets and present the proof of the mesh termination condition in detail. We then validate our solution by comparison with measured or analytically derived data.

PART I : CLOSED DOMAIN PROBLEM

Solving Maxwell's equations for the resonances of a closed cavity is important in understanding and controlling the operation of many devices, including particle accelerators, microwave filters, microwave ovens and optical fibers. However, the exact eigenvalues can be obtained only for simple geometries. For arbitrarily shaped cavities, numerical techniques like the finite element method must be used, but the occurrence of *spurious* modes [17] in the node-based finite element approach has plagued the computation of their eigenvalues. This difficulty can be circumvented

with the introduction of a penalty term [37] to render the finite element vector field solutions non-divergent. However, it is difficult to satisfy continuity requirements across material interfaces and treat geometries with sharp edges [38] using classical finite-elements, obtained by interpolating the nodal values of the vector field components. As mentioned in the Introduction to this chapter, edge elements, a type of vector finite elements with their degrees of freedom associated with the edges of the mesh, have been shown to be free of these shortcomings. Generally these lead to more unknowns but the higher variable count is balanced by the greater sparsity of the finite element matrix so that the computation time required to solve such a system iteratively with a given accuracy is less than the traditional approach [24].

Here we solve for the eigenvalues of an arbitrarily shaped metallic cavity using node-based and edge-based vector finite elements. The computed data are then compared with analytical results for empty and partially filled cavities. A comparison between the storage intensity and computational accuracy for edge-based rectangular bricks and tetrahedra is also presented. Finally, we compute the eigenvalues of a metallic cavity with a ridge along one of its faces.

4.1 Formulation

4.1.1 Finite element equations

Consider a three dimensional inhomogeneous body occupying the volume V. To discretize the electric field $\mathbf{E}$ within this volume, we subdivide the volume into small tetrahedra or rectangular bricks, each occupying the volume V_e $(e = 1, 2, \ldots, M)$, where M is the total number of elements. For a numerical solution, we expand $\mathbf{E}$ within the eth volume element as

$$\mathbf{E} = \sum_{j=1}^{m} E_j^e \mathbf{W}_j^e \tag{4.1}$$

where $\mathbf{W}_j^e$ are the edge-based vector basis functions, E_j^e denote the expansion coefficients of the basis, m represents the number of edges comprising the element and the superscript stands for the element number. On substituting this into the usual vector wave equation and upon applying Galerkin's technique, some vector identities and the divergence theorem, we obtain the weak form of Maxwell's equation

$$\begin{aligned} R_i^e &= \sum_{j=1}^{m} E_j^e \int_{V_e} \left[\frac{1}{\mu_r} (\nabla \times \mathbf{W}_i^e) \cdot (\nabla \times \mathbf{W}_j^e) - k_0^2 \epsilon_r \mathbf{W}_i^e \cdot \mathbf{W}_j^e \right] dv \\ &\quad - \jmath k_0 Z_o \oint_{S_e} \mathbf{W}_i^e \cdot (\hat{\mathbf{n}} \times \mathbf{H})\, ds \end{aligned} \tag{4.2}$$

where R_i^e represents the weighted residual integral for element e, S_e denotes the surface enclosing V_e, $\hat{\mathbf{n}}$ is the outward unit vector normal to S_e, Z_o is the free-space intrinsic impedance and ϵ_r, μ_r is the material permittivity and permeability, respectively. Equation (4.2) can be conveniently written in matrix form as

$$\{R_i^e\} = [A^e]\{E^e\} - k_0^2 [B^e]\{E^e\} - \{C^e\} \tag{4.3}$$

where

$$A_{ij}^e = \int_{V_e} \frac{1}{\mu_r} (\nabla \times \mathbf{W}_i^e) \cdot (\nabla \times \mathbf{W}_j^e)\, dv \tag{4.4}$$

$$B_{ij}^e = \int_{V_e} \epsilon_r \mathbf{W}_i^e \cdot \mathbf{W}_j^e\, dv \tag{4.5}$$

$$C_i^e = \jmath k_0 Z_o \oint_{S_e} \mathbf{W}_i^e \cdot (\hat{\mathbf{n}} \times \mathbf{H})\, ds \tag{4.6}$$

and on assembling the equations from all the elements making up the geometry, we obtain the system

$$\begin{aligned} \sum_{e=1}^{M} \{R_i^e\} &= \sum_{e=1}^{M} [A^e]\{E^e\} - k_0^2 \sum_{e=1}^{M} [B^e]\{E^e\} - \sum_{e=1}^{M} \{C^e\} \\ &= \{0\} \end{aligned} \tag{4.7}$$

where all matrices and vectors following the summation sign have been augmented using global numbers.

Due to the continuity of tangential **H** at the interface between two dielectrics, an element face lying inside the body does not contribute to the last term of (4.7) in the final assembly of the element equations. As a result, the last term of (4.7) reduces to a column vector containing the surface integral of the tangential magnetic field only over the outer surface of the body. In this application, the surface enclosing the volume of the body V is perfectly conducting and, thus, the coefficients associated with the edges bordering the perfectly conducting surface can be set to zero a priori. This reduces the original unknown count and eliminates the need to generate equations for those edges/unknowns which would have otherwise involved the column vector $\{C^e\}$. Also since $\{C^e\}$ is only associated with boundary edges, the surface integral associated with it vanishes and (4.7) can be written as

$$[A]\{E\} = \lambda [B]\{E\} \tag{4.8}$$

where $[A]$ and $[B]$ are $N \times N$ symmetric, sparse matrices with N being the total number of edges resulting from the subdivision of the body excluding the edges on the boundary, $\{E\}$ is a $N \times 1$ column vector denoting the edge fields and $\lambda = k_0^2$ gives the eigenvalues of the system. A solution of (4.8) will yield the resonant field distribution $\{E\}$ and the corresponding wavenumber k_0.

4.1.2 Origin of spurious solutions

Conventional finite element basis functions give rise to spurious solutions when (4.8) is solved. As Wong and Cendes points out in [39], the origin of these spurious solutions lies in the infinitely degenerate eigenvalue $k = 0$ in the spectrum of (4.8). Given the eigenvalue system in 4.8 along with the PEC boundary condition $\hat{\mathbf{n}} \times \mathbf{E} = 0$

on the boundary, there exists an infinite number of scalar functions ψ such that $\psi = 0$ on the boundary. Then $\mathbf{E} = -\nabla\psi$ is a permitted eigenfunction corresponding to the eigenvalue $k = 0$. If the discretization scheme fails to model this infinite dimensional nullspace of the curl operator exactly, spurious solutions to the eigenvalue problem will appear.

One way to get rid of spurious modes is to formulate the eigenvalue problem such that $k = 0$ is no longer a permissible eigenvalue. This is achieved by enforcing

$$\nabla \cdot \mathbf{E} = 0 \tag{4.9}$$

exactly everywhere in the solution region. Then the only solution corresponding to the $k = 0$ eigenvalue is the trivial one $\mathbf{E} = 0$. This is also the reason why spurious solutions do not occur when the Helmholtz equation is discretized. In finite elements, solving a problem (4.8) along with a constraint (4.9) is well known [12]. Researchers have mostly tried the penalty function approach of constrained minimization [37, 38] since it is simple to implement. However, the penalty approach is a mere *fix* and not a *cure* for the problem. Since the spurious eigenmodes are now shifted far into the visible spectrum, they are not completely eliminated and are dependent on an user-defined parameter which specifies how strongly the divergenceless condition is to be imposed.

Other than the penalty method, derivative continuous finite elements (C^1 elements) have also been proposed [39] to alleviate this problem. In this method, an auxiliary vector field ζ is introduced such that

$$\mathbf{E} = \nabla \times \zeta \tag{4.10}$$

Since substitution of (4.10) for $\mathbf{E}$ into (4.8) results in second derivatives, we need to construct first derivative continuous elements or C^1 elements. As shown in [39],

discretization of $\mathbf{E}$ using node-based C^1 elements eliminates the problem of spurious solutions since the nullspace of the curl operator is modeled exactly. However, C^1 elements are not commonly found in finite elements and need to be explicitly derived for the problem at hand.

Another method of eliminating spurious modes, without getting rid of the eigenvalue $k = 0$, is by using edge elements [40]. Bossavit in [40] provides a mathematical proof as to why spurious modes do not appear with edge based elements and why they are likely to be present with node-based vectorial elements. However, special node-based elements, like the C^1 element in [39], do not present this problem.

Thus the root cause of spurious modes appears to be the improper modeling of the nullspace of the curl operator. Any basis function which approximates it correctly will be stable and free of spurious modes. As it turns out, conventional Lagrangian finite elements are unsuitable; either C^1 node-based elements or edge-based elements of any order can be used to obtain the true solutions.

4.1.3 Basis functions

Edge no.	i_1	i_2
1	1	2
2	1	3
3	1	4
4	2	3
5	4	2
6	3	4

Table 4.1: TETRAHEDRON EDGE DEFINITION

The vector edge-based expansion functions for rectangular bricks were presented in [30]. Vector fields within tetrahedral domains can be conveniently represented by expansion functions that are linear in the spatial variables and have either zero divergence or zero curl. The basis functions defined in [24] are associated with the six edges of the tetrahedron and have zero divergence and constant curl. To define them, let us assume that i_1 and i_2 are the terminal nodes of the ith edge and the six edges of a tetrahedron are numbered according to Table 4.1. The vector basis function associated with the $(7-i)$th edge of the tetrahedron is then given by

$$\mathbf{W}^e_{7-i} = \begin{cases} \mathbf{f}_{7-i} + \mathbf{g}_{7-i} \times \mathbf{r}, & \mathbf{r} \text{ in the tetrahedron} \\ 0, & \text{otherwise} \end{cases} \tag{4.11}$$

with

$$\mathbf{f}_{7-i} = \frac{b_{7-i}}{6V_e}\mathbf{r}_{i_1} \times \mathbf{r}_{i_2} \tag{4.12}$$

$$\mathbf{g}_{7-i} = \frac{b_i b_{7-i}\mathbf{e}_i}{6V_e} \tag{4.13}$$

in which $i = 1, 2, \ldots, 6$, V_e is the volume of the tetrahedral element, $\mathbf{e}_i = (\mathbf{r}_{i_2} - \mathbf{r}_{i_1})/b_i$ is the unit vector of the ith edge and $b_i = |\mathbf{r}_{i_2} - \mathbf{r}_{i_1}|$ is the length of the ith edge with $\mathbf{r}_{i_1}$ and $\mathbf{r}_{i_2}$ denoting the location of the i_1 and i_2 nodes.

In general, the implementation of the above discretization will involve two numbering systems, and thus some unique global edge direction must be defined to ensure the continuity of $\hat{\mathbf{n}} \times \mathbf{E}$ across all edges [41]. Here we choose this direction to be coincident with the edge vector pointing from the smaller to the larger global node number. Finally, since $\nabla \cdot \mathbf{W}^e_i = 0$, the electric field obtained from a solution of (4.3) satisfies the divergence equation within each element and, thus, the solution will be free from contamination due to spurious solutions.

Mode	Analytical	Computed (bricks) 270 unknowns	Computed (tetra.) 260 unknowns	Error (%) (bricks)	Error (%) (tetra.)
TE_{101}	5.236	5.307	5.213	−1.36	.44
TM_{110}	7.025	7.182	6.977	−2.23	.70
TE_{011}	7.531	7.725	7.474	−2.58	1.00
TE_{201}		7.767	7.573	−3.13	−.56
TM_{111}	8.179	8.350	7.991	−2.09	2.29
TE_{111}		8.350	8.122	−2.09	.70
TM_{210}	8.886	9.151	8.572	−2.98	3.53
TE_{102}	8.947	9.428	8.795	−5.38	1.70

Table 4.2: EIGENVALUES (k_0, CM^{-1}) FOR AN EMPTY 1CM × 0.5CM × 0.75CM RECTANGULAR CAVITY

4.2 Results

In Table 4.2, we present a comparison of the percentage error in the computation of eigenvalues for a 1cm × .5cm × .75cm rectangular cavity using edge-based rectangular bricks and tetrahedra. The edge-based approach using tetrahedral elements predicts the first six distinct non-trivial eigenvalues with less than 4 percent error and is seen to provide better accuracy than rectangular brick elements. The maximum edge length for the rectangular brick elements was .15cm whereas that for the tetrahedral elements was .2cm. To investigate this matter further, we considered a cubical metallic cavity having a side length of .5cm. A plot of the percentage error in calculating the first three degenerate resonant frequencies versus the number of unknowns is given in Figure 4.1 for both rectangular bricks and tetrahedral elements. It is clear in this example that the tetrahedral elements predict the eigenvalues with greater accuracy than the rectangular bricks.

In Tables 4.3, 4.5 and 4.4, we compare the exact eigenvalues with those computed using edge-based tetrahedral finite elements. The finite element mesh was generated

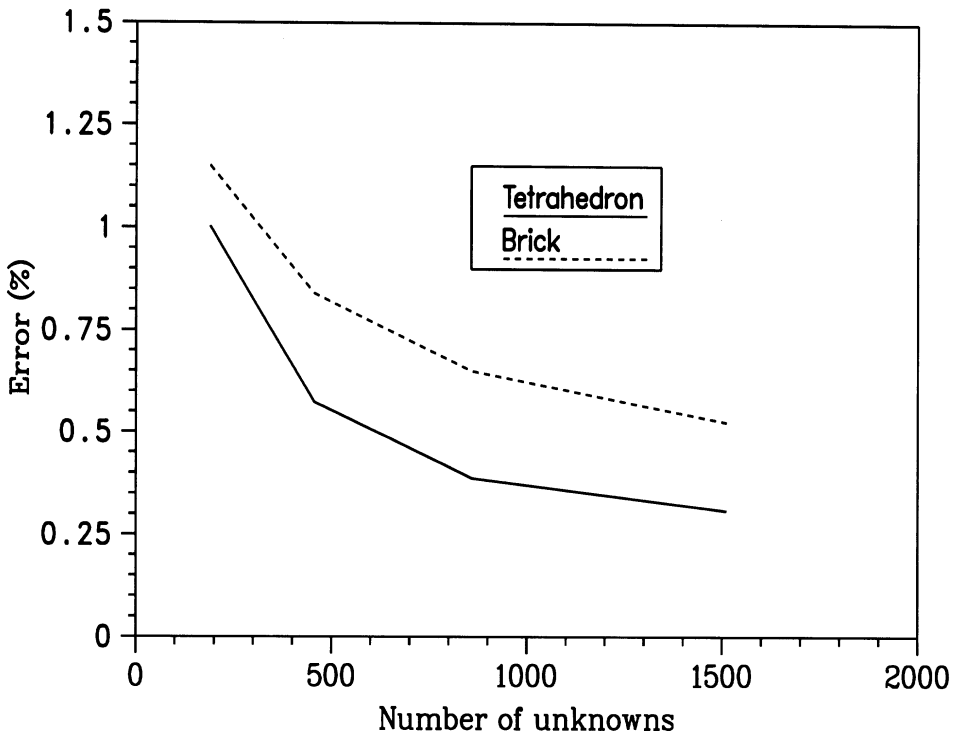

Figure 4.1: Performance comparison of rectangular bricks and tetrahedrals.

using SDRC I-DEAS, a commercial pre-processing package and it is seen that the numerical results are in good agreement with the exact values for both homogeneous and inhomogeneous cavities. The exact eigenvalues of the half-filled cavity as described in Table 4.3 are computed by solving the transcendental equation obtained upon matching the tangential electric and magnetic fields at the air-dielectric interface. As seen, these results agree with those predicted by the finite element solution to within 1 percent (no symmetry was assumed in this solution). Similar comparisons are given in Table 4.4 for a sphere having 1cm radius.

Finally, Table 4.6 presents the eigenvalues of the geometry illustrated in Figure 4.2. This is a closed metallic cavity with a ridge along one of its faces.

It is noted that as the degeneracy of the eigenvalues increases, the matrix becomes increasingly ill-conditioned and the numerical solution is correspondingly less accurate [42]. This is clearly observed from the data in Table 4.4 for the case of a

Mode	Analytical	Computed 192 unknowns	Error (%)
TEz_{101}	3.538	3.534	.11
TEz_{201}	5.445	5.440	.10
TEz_{102}	5.935	5.916	.32
TEz_{301}	7.503	7.501	.04
TEz_{202}	7.633	7.560	.97
TEz_{103}	8.096	8.056	.50

Table 4.3: Eigenvalues (k_0, cm^{-1}) for a Half-Filled 1cm × 0.1cm × 1cm Rectangular Cavity Having a Dielectric Filling of $\epsilon_r = 2$ Extending from $z = 0.5$cm to $z = 1.0$cm.

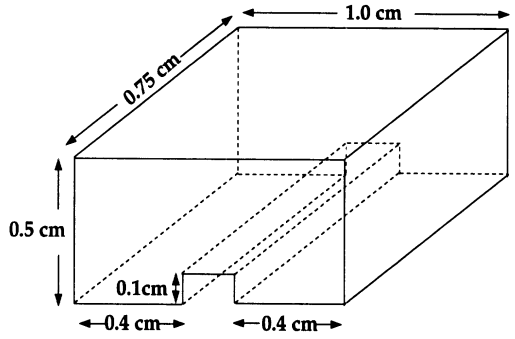

Figure 4.2: Geometry of ridged cavity

perfectly conducting hollow spherical cavity. Since the second lowest TM mode has five-fold degeneracy, the computational error is seen to be the greatest. However, for the partially filled rectangular cavity, the absence of degenerate modes gives results which are accurate to within 1 percent of the exact eigensolutions. We finally remark of the inherent presence of zero eigenvalues in our computations whose number is equal to the internal nodes. These zero eigenvalues are easily identifiable and since they do not correspond to physical modes, they were always discarded.

Mode	Analytical	Computed 300 unknowns	Error (%)
TM_{010}	2.744	2.799	−2.04
$TM_{111,even}$		2.802	−2.11
$TM_{111,odd}$		2.811	−2.44
TM_{021}	3.870	3.948	−2.02
$TM_{121,even}$		3.986	−2.99
$TM_{121,odd}$		3.994	−3.20
$TM_{221,even}$		4.038	−4.34
$TM_{221,odd}$		4.048	−4.59
TE_{011}	4.493	4.433	1.33
$TE_{111,even}$		4.472	.47
$TE_{111,odd}$		4.549	−1.25

Table 4.4: Eigenvalues (k_0, cm^{-1}) for an empty spherical cavity of radius 1cm

4.3 Conclusions

It was shown that the resonant frequencies of an arbitrarily shaped inhomogeneously filled metallic resonator can be computed very accurately via the finite element method using edge-based tetrahedral elements. The same method in conjunction with node-based elements is much less reliable and not readily applicable to regions containing discontinuous boundaries in shape and material. Edge-based rectangular bricks do not provide as good an accuracy as edge-based tetrahedral elements and their use is further limited to a special class of geometries.

PART II : OPEN DOMAIN PROBLEM

Of generic interest in electromagnetic scattering is the modeling of composite configurations comprised of metallic and non-metallic sections. In the case of man-made structures, abrupt material discontinuities and metallic corners are also encountered along with resistive sheets and thin ferrite coatings intended for controlling the scat-

Mode no.	Analytical	Computed	Error(%)
TM 010	4.810	4.809	.02
TE 111	7.283	7.202	1.1
		7.288	-.07
TM 110	7.650	7.633	.22
		7.724	-.97
TM 011	7.840	7.940	-1.28
TE 211	8.658	8.697	-.45
		8.865	-2.39

Table 4.5: Eigenvalues for an empty cylindrical cavity of base radius 0.5cm and height 0.5 cm (380 unknowns)

terer's radar cross-section (RCS). Differential equation methods, especially the finite element method (FEM), with its capability of handling arbitrary geometries and its versatility in modeling inhomogeneities and material discontinuities has been a viable solution approach for bounded domain problems. However, for unbounded problems as is the case with electromagnetic scattering, the solution is more involved since the finite element mesh needs to be truncated artificially at some distance from the object with a suitable boundary condition. These boundary conditions can be either global or local. Global boundary conditions are exact but lead to fully populated submatrices thus spoiling the sparse, banded structure of the finite element system. Further, problems due to internal resonances may arise in many cases [43]. In contrast, local conditions such as the absorbing boundary conditions (ABCs), are approximate but have the important advantage of retaining the sparsity of the matrix system. Also, they are free from the interior resonance problem that plagues boundary integral termination schemes [43]. ABCs are essentially differential equations enforced at the mesh truncation boundary and are chosen to suppress non-physical reflections from that boundary, thus ensuring the outgoing nature of the waves.

No.	(a)	(b)
1	4.941	4.999
2	7.284	7.354
3	7.691	7.832
4	7.855	7.942
5	8.016	7.959
6	8.593	8.650
7	8.906	8.916
8	9.163	9.103
9	9.679	9.757
10	9.837	9.927

Table 4.6: Ten lowest non-trivial eigenvalues (k_0, cm^{-1}) for the geometry drawn in Figure 2: (a) 267 Unknowns; (b) 671 Unknowns

A variety of ABCs have been derived and widely employed in FEM solutions of open region two-dimensional scattering problems. However, the method's implementation and performance for scattering by three dimensional geometries using edge-based finite elements has not received similar attention. The only three-dimensional implementations of the FEM for scattering has been a hybrid solution combined with the boundary element method (BEM) [41, 30, 44] and those formulations combined with ABCs [45, 46]. The boundary element method, though exact, is equivalent to employing a global boundary condition for terminating the mesh and consequently leads to a full submatrix, restricting the method's utility to small geometries. For large-scale three-dimensional applications, it is necessary to employ an ABC for terminating the mesh to retain the $O(N)$ storage requirement, characteristic of the finite element method. However, the use of traditional node-based elements suffers from various difficulties as mentioned in the Part I.

To avoid these difficulties, we consider an implementation of the FEM using vector basis functions whose degrees of freedom are associated with the fields along the six edges of a tetrahedron. Our implementation is further coupled with a mesh

termination scheme based on the vector ABCs derived in [47, 48]. In contrast to the implementation proposed in [49], the one presented here preserves the symmetry of the finite element system, thus being computationally more efficient and making it ideally suited for solution via a conjugate gradient type of algorithm. Further, the implementation discussed in [49] requires that the absorbing boundary be placed nearly a wavelength away from the scatterer, whereas in our implementation remarkably accurate results are obtained with the ABCs enforced a small fraction of a wavelength from the scattering body. This is probably due to the accuracy of the second order ABCs derived in [48].

4.4 Formulation

In the following section, the open domain problem is formulated in terms of the finite element functional. The final system is obtained by setting the first variation in the functional to zero and then a Rayleigh-Ritz minimization is performed to arrive at the final answer.

4.4.1 Derivation of finite element equations

Let us consider the problem of scattering by an inhomogeneous target associated with possible material discontinuities. To solve for the scattered fields via the FEM, it is necessary to enclose the scatterer– embedded inside the volume V– by an artificial surface S_o on which the ABC is enforced (see Figure 4.3). The ABCs to be considered in this chapter are the Sommerfeld radiation condition given by

$$\hat{\mathbf{n}} \times \nabla \times \mathbf{E}^s = -jk_o \hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \mathbf{E}^s \tag{4.14}$$

and the second-order ABC [48] which can be written as

$$\hat{\mathbf{n}} \times \nabla \times \mathbf{E}^s = \alpha \mathbf{E}_t^s + \beta \nabla \times [\hat{\mathbf{n}} (\nabla \times \mathbf{E}^s)_n] + \beta \nabla_t (\nabla \cdot \mathbf{E}_t^s) \tag{4.15}$$

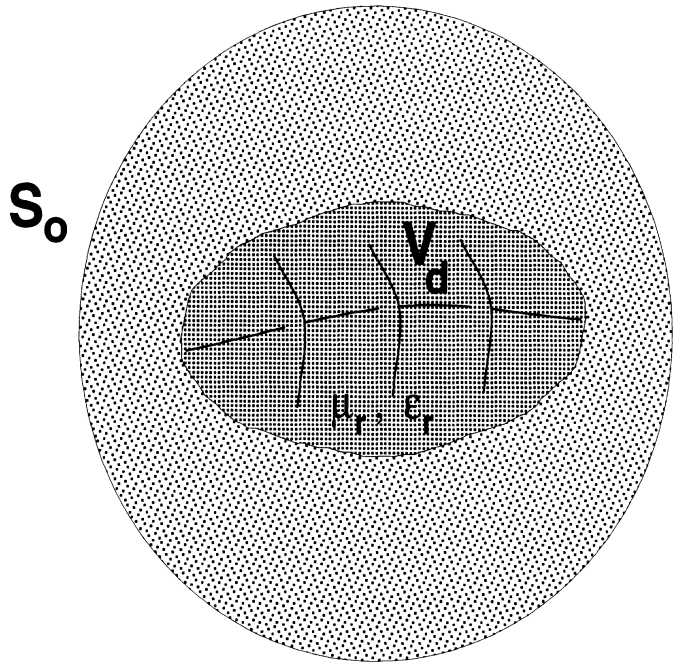

Figure 4.3: Illustration of scattering structure V_d enclosed by an artificial mesh termination surface, S_o, on which the absorbing boundary condition is imposed.

where $\alpha = jk, \beta = 1/(2jk + 2/r)$, $\mathbf{E}^s$ represents the scattered electric field, $\hat{\mathbf{n}}$ is the unit normal to the surface S_o and the subscripts t and n denote the transverse and normal component to S_o, respectively. When these ABCs are employed on the artificial boundary S_o, they annihilate all field terms of $O(r^{-(2m+1)})$ and smaller, where m denotes the order of the ABC. The ABCs outlined above were derived for spherical surfaces [48] but in this work we have extended their application to S_o which include flat sections. In this case, the local curvature is used to replace $1/r$ in (4.15). For flat sections, the $1/r$ term, therefore, reduces to zero. This permits the construction of termination boundaries conformal to the scatterer, thus reducing the size of the the computational domain. A detailed derivation of (4.15) as well as other more general ABCs are outlined in Chapter 6.

The vector ABCs (4.14) and (4.15) can be combined and more conveniently writ-

ten as

$$\hat{\mathbf{n}} \times \nabla \times \mathbf{E}^s = P(\mathbf{E}^s) \tag{4.16}$$

for the scattered field formulation in which $\mathbf{E}^s$ is the working variable and

$$\hat{\mathbf{n}} \times \nabla \times \mathbf{E} = P(\mathbf{E}) + \mathbf{U}^{inc} \tag{4.17}$$

for the total field formulation where the unknown is the total electric field. In (4.17),

$$\mathbf{U}^{inc} = \hat{\mathbf{n}} \times \nabla \times \mathbf{E}^{inc} - P\left(\mathbf{E}^{inc}\right) \tag{4.18}$$

where $\mathbf{E} = \mathbf{E}^s + \mathbf{E}^{inc}$ is the total field and $\mathbf{E}^{inc}$ is the incident electric field. Considering (4.17) to be the boundary condition employed at S_o, we can express the functional for the total electric field as

$$\begin{aligned} F(\mathbf{E}) = & \int_V \left[\frac{1}{\mu_r}(\nabla \times \mathbf{E}) \cdot (\nabla \times \mathbf{E}) - k_o^2 \epsilon_r \mathbf{E} \cdot \mathbf{E}\right] dV \\ & + \int_{S_o} \left[\mathbf{E} \cdot P(\mathbf{E}) + 2\mathbf{E} \cdot \mathbf{U}^{inc}\right] dS \end{aligned} \tag{4.19}$$

where ϵ_r and μ_r are the relative permittivity and permeability, respectively.

The above functional can be generalized to account for the presence of impedance and resistive sheets or other discontinuous boundaries. In the case of a resistive card, the transition condition [50]

$$\hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \mathbf{E}) = -R\hat{\mathbf{n}} \times \left(\mathbf{H}^+ - \mathbf{H}^-\right) \tag{4.20}$$

must be enforced, where $\mathbf{H}^{\pm}$ denotes the total magnetic field above and below the sheet, R is the resistivity in Ohms per square and $\hat{\mathbf{n}}$ is the unit normal to the sheet pointing in the upward direction (+ side). For an impenetrable impedance surface, the appropriate boundary condition on that surface is

$$\hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \mathbf{E}) = -\eta\hat{\mathbf{n}} \times \mathbf{H} \tag{4.21}$$

where $\hat{\mathbf{n}}$ is the unit normal to the surface and η is the surface impedance. Taking into consideration these boundary/transition conditions, the functional for the total electric field can be more explicitly written as

$$\begin{aligned} F(\mathbf{E}) &= \int_V \left[\frac{1}{\mu_r}(\nabla \times \mathbf{E}) \cdot (\nabla \times \mathbf{E}) - k_o^2 \epsilon_r \mathbf{E} \cdot \mathbf{E}\right] dV \\ &\quad + jk_o Z_o \int_{S_k} \frac{1}{K} (\hat{\mathbf{n}} \times \mathbf{E}) \cdot (\hat{\mathbf{n}} \times \mathbf{E})\, dS \\ &\quad + \int_{S_o} \left[\mathbf{E} \cdot P(\mathbf{E}) + 2\mathbf{E} \cdot \mathbf{U}^{inc}\right] dS \end{aligned} \tag{4.22}$$

where K is the surface resistivity (R) when integrating over a resistive card and equals the surface impedance (η) for an impedance sheet.

In order to deal with anisotropic scatterers, the functional outlined in (4.22) undergoes a slight modification since the material properties of the scatterer (permeability and permittivity) are now second rank tensors rather than scalars. Equation (4.22) can therefore be written as

$$\begin{aligned} F(\mathbf{E}) &= \int_V \left[(\nabla \times \mathbf{E}) \cdot \left\{[\overline{\overline{\mu}}]^{-1} \cdot (\nabla \times \mathbf{E})\right\} - k_o^2 \mathbf{E} \cdot \{\overline{\overline{\epsilon}} \cdot \mathbf{E}\}\right] dV \\ &\quad + jk_o Z_o \int_{S_k} \frac{1}{K} (\hat{\mathbf{n}} \times \mathbf{E}) \cdot (\hat{\mathbf{n}} \times \mathbf{E})\, dS \\ &\quad + \int_{S_o} \left[\mathbf{E} \cdot P(\mathbf{E}) + 2\mathbf{E} \cdot \mathbf{U}^{inc}\right] dS \end{aligned} \tag{4.23}$$

where

$$\overline{\overline{\mu}} = \begin{bmatrix} \mu_{xx} & \mu_{xy} & \mu_{xz} \\ \mu_{yx} & \mu_{yy} & \mu_{yz} \\ \mu_{zx} & \mu_{zy} & \mu_{zz} \end{bmatrix} \tag{4.24}$$

and

$$\overline{\overline{\epsilon}} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{xz} \\ \epsilon_{yx} & \epsilon_{yy} & \epsilon_{yz} \\ \epsilon_{zx} & \epsilon_{zy} & \epsilon_{zz} \end{bmatrix} \tag{4.25}$$

The symmetry of the final system of equations now depends on the symmetry of the permeability and the permittivity tensors.

The formulation presented above is in terms of the total field but we can easily revert to a scattered field formulation by setting $\mathbf{E}^s = \mathbf{E} - \mathbf{E}^{inc}$ and noting that the scattered field satisfies the wave equation inside the domain of interest.

Let us consider the case where the computational volume V is occupied by a dielectric structure and is bounded internally by the surface of a perfect conductor and externally by the mesh termination boundary. On examining the terms inside the volume integral in (4.19), we can define

$$\int_V \left[\frac{1}{\mu_r}(\nabla\times\mathbf{E})\cdot(\nabla\times\mathbf{E}) - k_o^2\epsilon_r\mathbf{E}\cdot\mathbf{E}\right]\, dV = G(\mathbf{E},\mathbf{E}) \tag{4.26}$$

Expressing the above relation in terms of the incident and the scattered fields, we have

$$G(\mathbf{E},\mathbf{E}) = G(\mathbf{E}^s,\mathbf{E}^s) + 2G(\mathbf{E}^s,\mathbf{E}^{inc}) + G(\mathbf{E}^{inc},\mathbf{E}^{inc}) \tag{4.27}$$

The first and the third terms on the RHS of (4.27) cannot be simplified any further. The second term will, however, lend itself to more simplification. Making use of a simple vector identity and the divergence theorem, we can rewrite $G(\mathbf{E}^s,\mathbf{E}^{inc})$ as

$$\begin{aligned} G(\mathbf{E}^s,\mathbf{E}^{inc}) \;=\; & \int_V \mathbf{E}^s\cdot\left[\nabla\times\frac{1}{\mu_r}\nabla\times\mathbf{E}^{inc} - k_o^2\epsilon_r\mathbf{E}^{inc}\right]\, dV \\ & - \int_{S_o}\mathbf{E}^s\cdot\left(\hat{\mathbf{n}}\times\nabla\times\mathbf{E}^{inc}\right)\, dS \end{aligned} \tag{4.28}$$

since

$$\begin{aligned} & \int_V \left[\frac{1}{\mu_r}(\nabla\times\mathbf{E}^s)\cdot(\nabla\times\mathbf{E}^{inc})\right]\, dV = \\ & \int_V \mathbf{E}^s\cdot\left[\nabla\times\frac{1}{\mu_r}\nabla\times\mathbf{E}^{inc}\right]\, dV - \int_S \frac{1}{\mu_r}\mathbf{E}^s\cdot\left(\hat{\mathbf{n}}\times\nabla\times\mathbf{E}^{inc}\right)\, dS \end{aligned} \tag{4.29}$$

and the surface integral cancels out everywhere inside the computational domain except on the mesh termination boundary S_o. If we define V_d to be the volume occupied by dielectric materials, then the remaining volume ($V_o = V - V_d$) is the volume occupied by free space. On incorporating this into (4.28), we have

$$\begin{aligned} G(\mathbf{E}^s, \mathbf{E}^{inc}) &= \int_{V_o} \mathbf{E}^s \cdot \left[\nabla\times\nabla\times\mathbf{E}^{inc} - k_o^2\mathbf{E}^{inc}\right]\, dV \\ &\quad + \int_{V_d} \mathbf{E}^s \cdot \left[\nabla\times\frac{1}{\mu_r}\nabla\times\mathbf{E}^{inc} - k_o^2\epsilon_r\mathbf{E}^{inc}\right]\, dV \\ &\quad - \int_{S_o} \mathbf{E}^s \cdot \left(\hat{\mathbf{n}}\times\nabla\times\mathbf{E}^{inc}\right)\, dS \end{aligned} \tag{4.30}$$

Since the incident electric field satisfies the wave equation in free space, the first term of (4.30) is identically zero. The third term cancels exactly with the cross term $\int_{S_o} \mathbf{E}\cdot\mathbf{U}^{inc}\, dS$ in the total field functional (4.19). The second term can be simplified by employing a standard vector identity and the divergence theorem to yield

$$\begin{aligned} &\int_{V_d} \mathbf{E}^s \cdot \left[\nabla\times\frac{1}{\mu_r}\nabla\times\mathbf{E}^{inc} - k_o^2\epsilon_r\mathbf{E}^{inc}\right]\, dV = \\ &\qquad \int_{V_d} \frac{1}{\mu_r}(\nabla\times\mathbf{E}^s)\cdot(\nabla\times\mathbf{E}^{inc}) - k_o^2\epsilon_r\mathbf{E}^s\cdot\mathbf{E}^{inc}\, dV \\ &\qquad + jk_oZ_o\int_{S_d}\frac{1}{\mu_r}\mathbf{E}^s\cdot(\hat{\mathbf{n}}\times\mathbf{H}^{inc})\, dS \end{aligned} \tag{4.31}$$

where the normal to S_d is directed away from V_d. The surface integral over the dielectric interface S_d occurs since the tangential component of the scattered electric field is discontinuous over the interface between two dielectrics having dissimilar permeabilities. It should be noted that (4.31) holds good even when there are multiple dielectric regions present. If the dielectric regions have the same permeability ($\mu_{r_1} = \mu_{r_2} = \ldots \mu_{r_n} = 1$, for example) and different permittivities, the surface integral contribution over the dielectric interfaces -$S_{d_1}, \ldots, S_{d_n}$- is zero. If different permeability values are also present, then the permeability values must be substituted into the element equations and the direction of the normal for the two elements on

the interface should take care of the respective signs. Therefore, $G(\mathbf{E}^s, \mathbf{E}^{inc})$ reduces to

$$\begin{aligned} G(\mathbf{E}^s, \mathbf{E}^{inc}) &= \int_{V_d} \frac{1}{\mu_r}(\nabla\times\mathbf{E}^s)\cdot(\nabla\times\mathbf{E}^{inc}) - k_o^2\epsilon_r\mathbf{E}^s\cdot\mathbf{E}^{inc}\,dV \\ &\quad + jk_oZ_o\int_{S_d}\frac{1}{\mu_r}\mathbf{E}^s\cdot(\hat{\mathbf{n}}\times\mathbf{H}^{inc})\,dS \\ &\quad - \int_{S_o}\mathbf{E}^s\cdot\left(\hat{\mathbf{n}}\times\nabla\times\mathbf{E}^{inc}\right)\,dS \end{aligned} \tag{4.32}$$

The impedance and resistive sheet boundary conditions can be incorporated in a similar way into the scattered field functional. After simplification, the functional $F(\mathbf{E}^s)$ for the scattered field is then given by

$$\begin{aligned} F(\mathbf{E}^s) &= \int_V \left[\frac{1}{\mu_r}(\nabla\times\mathbf{E}^s)\cdot(\nabla\times\mathbf{E}^s) - k_o^2\epsilon_r\mathbf{E}^s\cdot\mathbf{E}^s\right]dV \\ &\quad + jk_oZ_o\int_{S_k}\frac{1}{K}(\hat{\mathbf{n}}\times\mathbf{E}^s)\cdot(\hat{\mathbf{n}}\times\mathbf{E}^s)\,dS \\ &\quad + \int_{S_o}\mathbf{E}^s\cdot P(\mathbf{E}^s)dS \\ &\quad + 2jk_oZ_o\int_{S_d}\frac{1}{\mu_r}\mathbf{E}^s\cdot\left(\hat{\mathbf{n}}\times\mathbf{H}^{inc}\right)dS \\ &\quad + 2\int_{V_d}\left[\frac{1}{\mu_r}(\nabla\times\mathbf{E}^s)\cdot(\nabla\times\mathbf{E}^{inc}) - k_o^2\epsilon_r\mathbf{E}^s\cdot\mathbf{E}^{inc}\right]dV \\ &\quad + 2jk_oZ_o\int_{S_k}\frac{1}{K}(\hat{\mathbf{n}}\times\mathbf{E}^s)\cdot(\hat{\mathbf{n}}\times\mathbf{E}^{inc})dS \\ &\quad + f\left(\mathbf{E}^{inc}\right) \end{aligned} \tag{4.33}$$

where V_d is the volume occupied by the dielectric (portion of V where ϵ_r or μ_r are not unity), S_d encompasses all dielectric interface surfaces and

$$\int_{S_o}\mathbf{E}^s\cdot P(\mathbf{E}^s)dS = \int_{S_o}\left[\alpha\left(\mathbf{E}_t^s\right)^2 + \beta(\nabla\times\mathbf{E}^s)_n^2 - \beta(\nabla\cdot\mathbf{E}_t^s)^2\right]dS$$

when the second order ABC is employed. The function $f\left(\mathbf{E}^{inc}\right)$ is solely in terms of the incident electric field and vanishes when we take the first variation of $F(\mathbf{E}^s)$. We remark that the scattered field formulation was implemented in our code.

4.5 Finite element discretization

To discretize the functional given in (4.33), the volume V is subdivided into a number of small tetrahedra, each occupying the volume V^e ($e = 1, 2, \cdots, M$), where M denotes the total number of tetrahedral elements. Within each element, the scattered electric field is expressed as

$$\mathbf{E}^e = \sum_{j=1}^{m} E_j^e \mathbf{W}_j^e = \{\mathbf{W}^e\}^T \{E^e\} = \{E^e\}^T \{\mathbf{W}^e\} \tag{4.34}$$

where $\mathbf{W}_j^e$ are the edge-based vector basis functions [24], E_j^e denote the expansion coefficients of the basis and represent the field components tangential to the jth edge of the eth element, m is the number of edges making up the element and the superscript stands for the element number. The basis functions used in our implementation have zero divergence and constant curl.

The system of equations to be solved for E_j^e is obtained by a Rayleigh-Ritz procedure which amounts to differentiating F with respect to each edge field and then setting it to zero. On substituting (4.34) into (4.33), taking the first variation in F and assembling all M elements, we obtain the following augmented system of equations

$$\left\{\frac{\partial F}{\partial E^e}\right\} = \sum_{e=1}^{M} [A^e]\{E^e\} + \sum_{s=1}^{M_s} [B^s]\{E^s\} + \sum_{p=1}^{M_p} \{C^p\} = 0 \tag{4.35}$$

In this, M_s denotes the number of triangular surface elements on S_k and S_o and M_p is equal to the sum of the surface elements on S_k, S_d and the volume elements in V_d. The elements of the matrices $[A^e]$, $[B^s]$ and $\{C^p\}$ are given by

$$A_{ij}^e = \int_{V^e} \left[\frac{1}{\mu_r}(\nabla \times \mathbf{W}_i^e) \cdot (\nabla \times \mathbf{W}_j^e) - k_o^2 \epsilon_r \mathbf{W}_i^e \cdot \mathbf{W}_j^e\right] dV$$

$$B_{ij}^s = jk_o Z_o \left[\int_{S_k^s} \frac{1}{K}(\hat{\mathbf{n}} \times \mathbf{W}_i^s) \cdot (\hat{\mathbf{n}} \times \mathbf{W}_j^s) dS\right]$$

$$+\int_{S_o^s}\Big[\alpha\mathbf{W}_{it}^s\cdot\mathbf{W}_{jt}^s+\beta(\nabla\times\mathbf{W}_i^s)_n\cdot(\nabla\times\mathbf{W}_j^s)_n-\beta(\nabla\cdot\mathbf{W}_{it}^s)(\nabla\cdot\mathbf{W}_{jt}^s)\Big]dS$$

$$C_i^p = 2jk_oZ_o\left[\int_{S_d^p}\frac{1}{\mu_r}\mathbf{W}_i^p\cdot\Big(\hat{\mathbf{n}}\times\mathbf{H}^{inc}\Big)dS+\int_{S_k^p}\frac{1}{K}(\hat{\mathbf{n}}\times\mathbf{W}_i^p)\cdot(\hat{\mathbf{n}}\times\mathbf{E}^{inc})dS\right]$$

$$+2\int_{V_d^p}\left[\frac{1}{\mu_r}(\nabla\times\mathbf{W}_i^p)\cdot(\nabla\times\mathbf{E}^{inc})-k_o^2\epsilon_r\mathbf{W}_i^p\cdot\mathbf{E}^{inc}\right]dV$$

where V_d^p is the volume of the pth tetrahedron inside the dielectric, S^s and S^p represent the surface area of the sth and pth triangular surface element and the subscripts t and n denote the tangential and normal components of a vector, respectively. The boundary condition $\hat{\mathbf{n}}\times\mathbf{E}^s=-\hat{\mathbf{n}}\times\mathbf{E}^{inc}$ must be imposed a priori on metallic boundaries; however, no special treatment is required at material discontinuities. Only the identification of the edges on material discontinuities or inhomogeneities is required to kick in the contribution from the surface integrals in (4.33).

The biconjugate gradient algorithm with diagonal preconditioning was used to solve the sparse, symmetric system of equations. The residual norm was usually set to less than 0.1% of the solution norm as a criterion for convergence since lower tolerances did not appear to offer significant improvement on the far-field values. The data structure was constructed such that only the non-zero elements of the upper triangular part of the symmetric, sparse matrix were stored in a $N_a\times k$ complex array. In our case, N_a was typically $1.1\times N_u$, where N_u denotes the number of unknowns and k was equal to 12. The corresponding addresses were stored in a separate $N_a\times k$ integer array. The storage required in this scheme was about $25N_u$ and the number of distinct non-zero elements was typically $9N_u$.

4.6 Results

A computer program was written for implementing the proposed FE-ABC formulation. This implementation was validated by computing the scattering for several

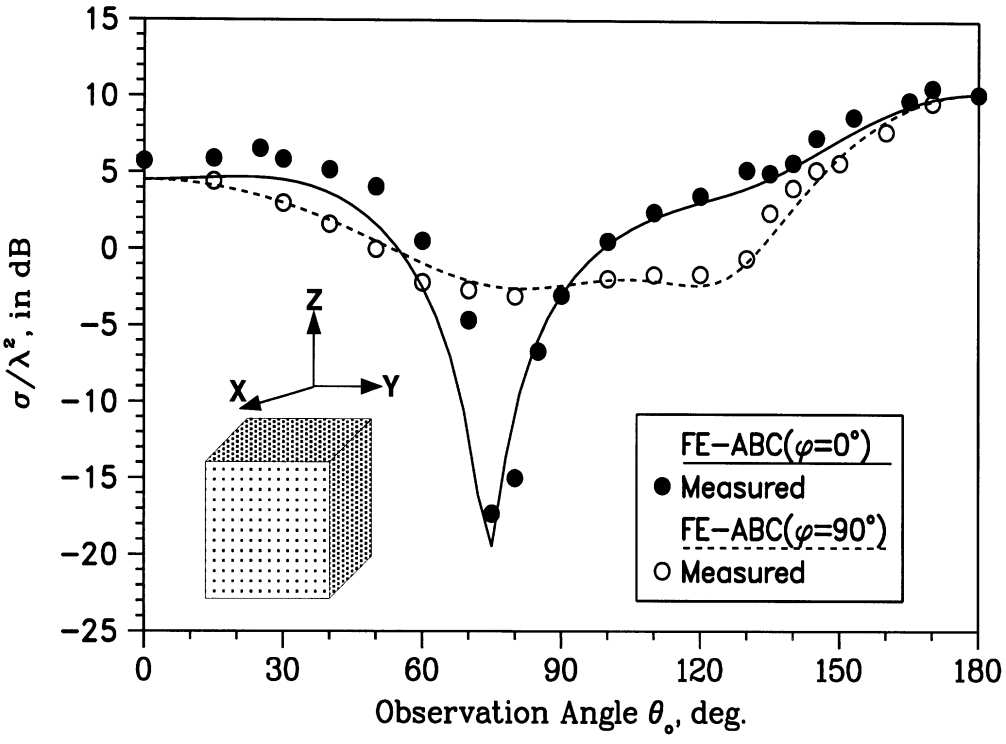

Figure 4.4: Bistatic echo-area of a perfectly conducting cube having edge length of 0.755λ. Plane wave incident from $\theta = 180^o; \phi = 90^o$.

configurations including metallic and dielectric bodies as well as structures satisfying resistive and impedance boundary conditions. Figure 4.4 compares the measured [51] bistatic cross-section ($\theta^{inc} = 180^o$, $\phi^{inc} = 90^o$) of a metallic cube having an edge length of 0.755λ with the corresponding pattern computed by the three-dimensional FE-ABC code. The second-order vector ABC was employed on a spherical mesh truncation boundary which was placed only 0.1λ from the edge of the cube. About 33,000 unknowns were used for the discretization of the computational domain and the $[A]$ matrix contained a total of 264,000 distinct non-zero entries. The storage requirement of this matrix was consequently much smaller than that of the 1400 unknown moment method system (assuming the same sampling rate as the FEM of 14 points/λ) which had 2 million non-zero entries.

In Figure 4.5, we plot the normal incidence backscatter RCS of a perfectly con-

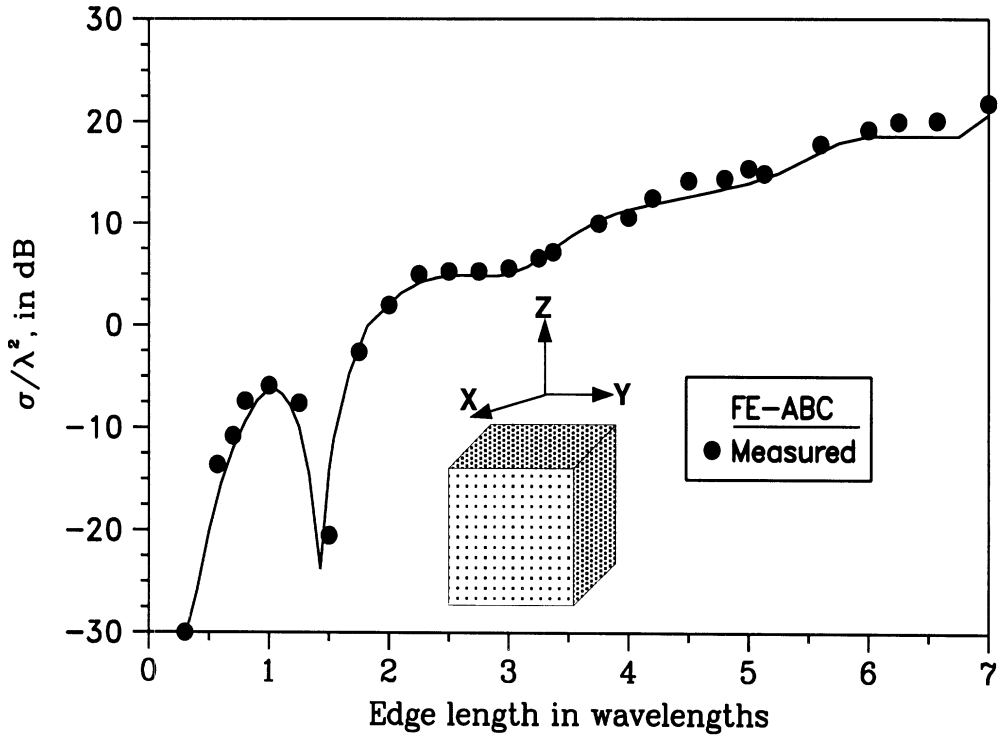

Figure 4.5: Backscatter RCS of a perfectly conducting cube at normal incidence as a function of edge length

ducting cube as a function of its edge length. The meshes constructed for this experiment were terminated on conformal boundaries, i.e, on another cube placed a small distance (more than 0.15λ) from the scatterer. As seen, the agreement with measured data [51] is remarkably good over a 50dB dynamic range.

Figure 4.6 presents backscatter data for a cylinder of radius 0.3λ and height 0.6λ. The data from the three-dimensional finite element code again compare well with that obtained from a moment method-body of revolution code. The mesh was terminated on a spherical boundary at a distance of 0.3λ from the edge of the scatterer and the system consisted of nearly 33,000 unknowns. Convergence was achieved within about 350 iterations when the Sommerfeld radiation condition was imposed on the spherical mesh termination boundary. Each iteration took approximately 0.1 seconds on a Cray YMP after vectorization and on the average it was found that for $N > 25,000$,

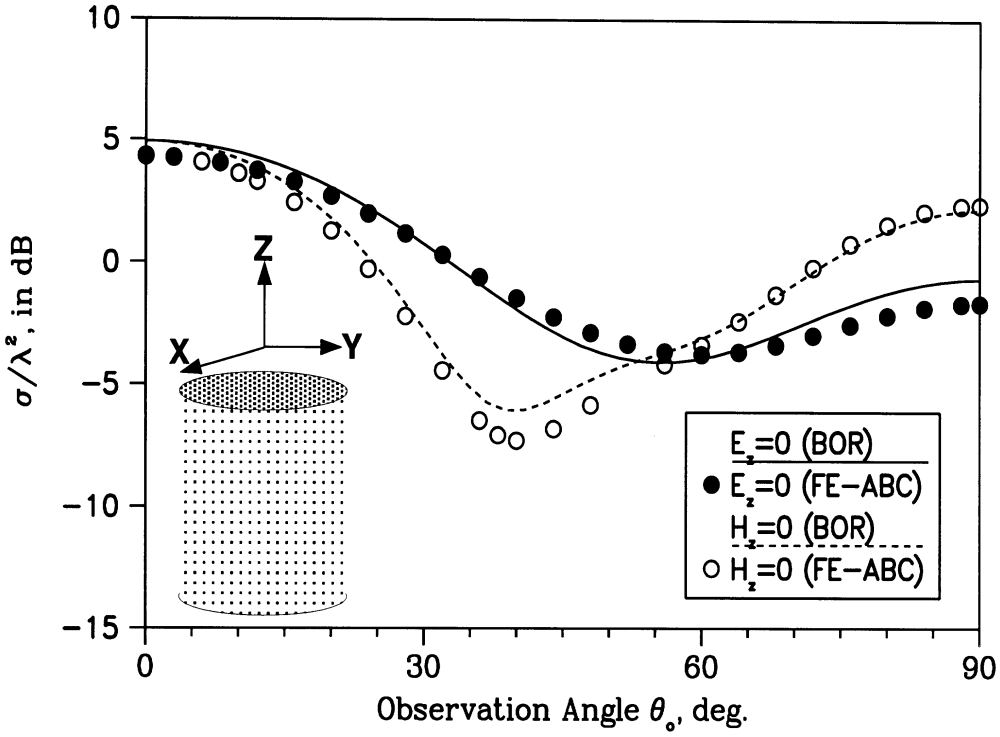

Figure 4.6: Backscatter pattern of a perfectly conducting cylinder having a radius of 0.3λ and a height of 0.6λ. The solid and the dashed lines indicate data obtained from a body of revolution code and the black and white dots indicate FE-ABC data.

the number of required iterations were approximately $N/100$. The agreement was quite good even on enclosing the metallic cylinder with a rectangular outer boundary placed 0.3λ from the edge of the scatterer.

The results presented till now have been for perfectly conducting geometries. However, the real advantage of the FEM over integral equation techniques is the ease with which the former can handle material inhomogeneities and transition conditions. With this in mind, the remaining figures show backscatter and bistatic patterns for scatterers comprised of resistive cards, dielectric material and combinations of these. The first geometry that we tested was that of a homogeneous dielectric sphere having a relative permittivity of 4 and a radius of $1/2\pi$. The bistatic pattern of the geometry is compared to that obtained using a CG-FFT formulation (Figure 4.7) and is seen

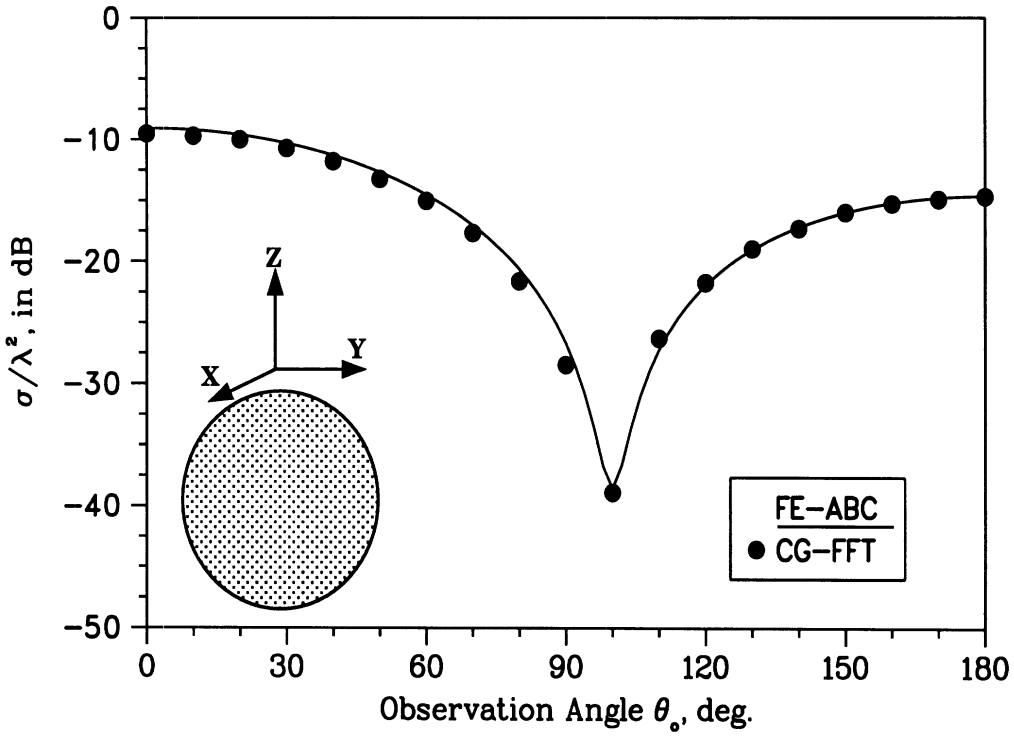

Figure 4.7: Bistatic echo-area of a homogeneous dielectric sphere ($\epsilon_r = 4; k_o a = 1$).

to agree remarkably well. The finite element mesh was terminated only 0.3λ from the dielectric body.

Another of the test cases was a prolate spheroid shown in Figure 4.8 filled with lossy dielectric having $\epsilon_r = 4 - j1$, $k_o a = \pi/2$ and $a/b = 2$, where a and b are the major and minor axes of the spheroid, respectively. The bistatic pattern ($\theta^{inc} = 180^o$; $\phi^{inc} = 90^o$) obtained from the FE-ABC solution agree reasonably well with those obtained via the hybrid finite element-boundary integral method presented in [41]. However, the corresponding convergence rate for non-metallic bodies and resistive/impedance sheets was found to be slower than that observed for metallic scatterers. A diagonal preconditioner was, therefore, used to accelerate the convergence of the biconjugate gradient algorithm with encouraging results.

For our last example, we compute the scattering from an inhomogeneous geometry with embedded resistive cards. Particularly, the scatterer shown in Figure 4.9

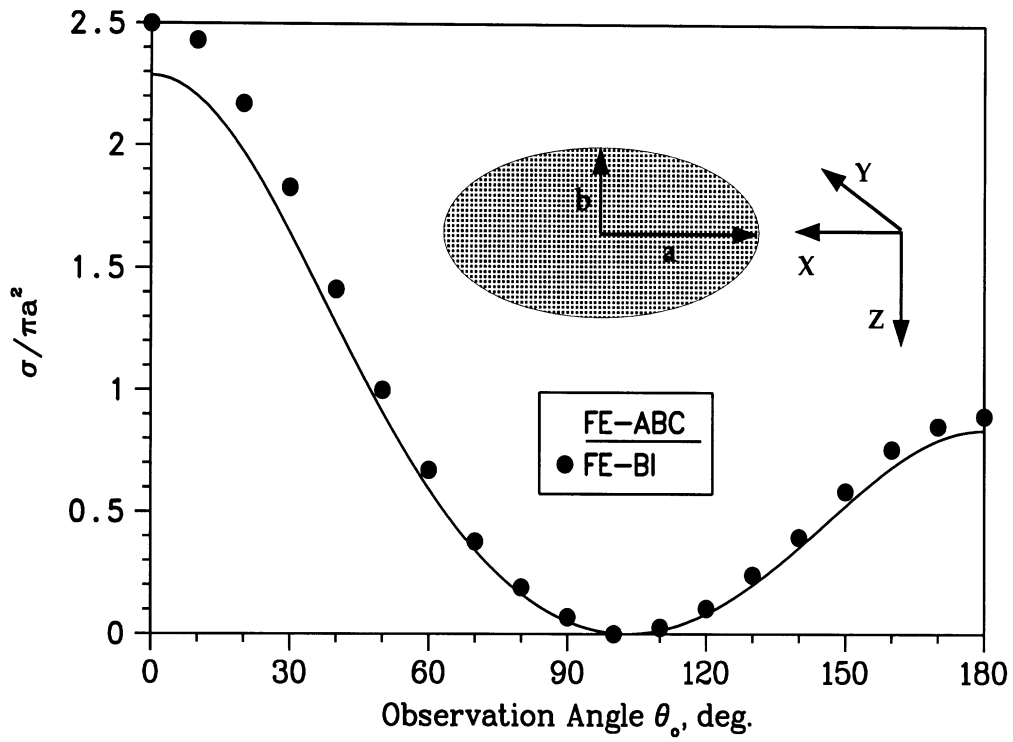

Figure 4.8: Normalized bistatic pattern of a lossy prolate spheroid ($\epsilon_r = 4 - j1; k_o a = \pi/2; a/b = 2$), where a and b are the major and minor axes of the spheroid, respectively

consists of an air-filled resistive card block ($0.5\lambda \times 0.5\lambda \times 0.25\lambda$) joined to a metallic block ($0.5\lambda \times 0.5\lambda \times 0.25\lambda$). In Figure 4.10, we compare a principal plane backscatter pattern obtained from our 3D FE-ABC implementation with data computed using a traditional moment method code [52] for both polarizations. The computed data is again seen to follow the reference data closely. For the FE-ABC solution, the scatterer was enclosed within a cubical outer boundary placed only 0.3λ away from the scatterer. This resulted in a 30,000 unknown system which converged to the solution in about 400 iterations when using the Sommerfeld radiation condition and in 1600 iterations when the second order ABC was used. For this geometry, the second order ABC did not provide a significant improvement in accuracy (only about 0.1dB) over the first order condition. The same case was run with a higher discretization resulting in a system of 50,000 unknowns; however, there was no significant difference in

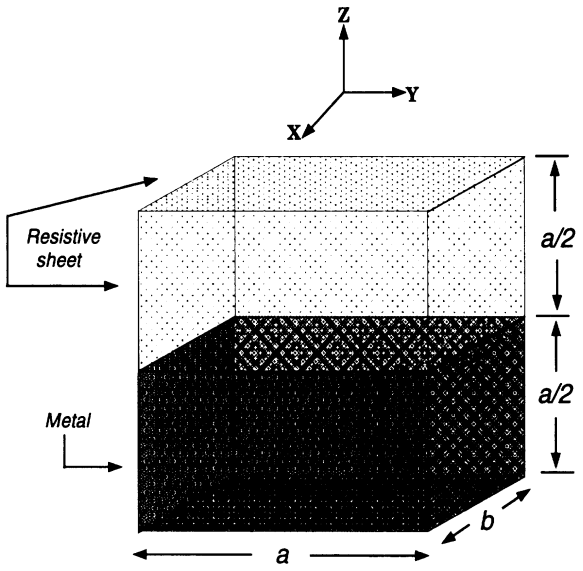

Figure 4.9: Geometry of cube ($a = b = 0.5\lambda$) consisting of a metallic section and a dielectric section ($\epsilon_r = 2 - j2$), where the latter is bounded by a resistive surface having $R = Z_o$.

the far-field values with the earlier case. The geometry for the backscatter pattern shown in Figure 4.11 is the same as in Figure 4.9 with the air-filled section now occupied by a lossy dielectric having $\epsilon_r = 2 - j2$. The backscatter echo-area pattern for the $\phi\phi$ polarization as computed by our FE-ABC code is again seen to be in good agreement with corresponding moment method data [52].

4.7 Conclusions

In this chapter, we have shown that the finite element technique with vector basis functions, when coupled with ABCs for mesh termination and the biconjugate gradient algorithm for the solution of the resulting system, is a viable procedure for computing the scattering by three-dimensional targets. We have found that these ABCs can be enforced only a small fraction of a wavelength from the scatterer's surface. This is probably due to the fast $(1/r)$ decay of the scattered fields. As a result, in addition to the sparsity of the matrix, the total number of unknowns

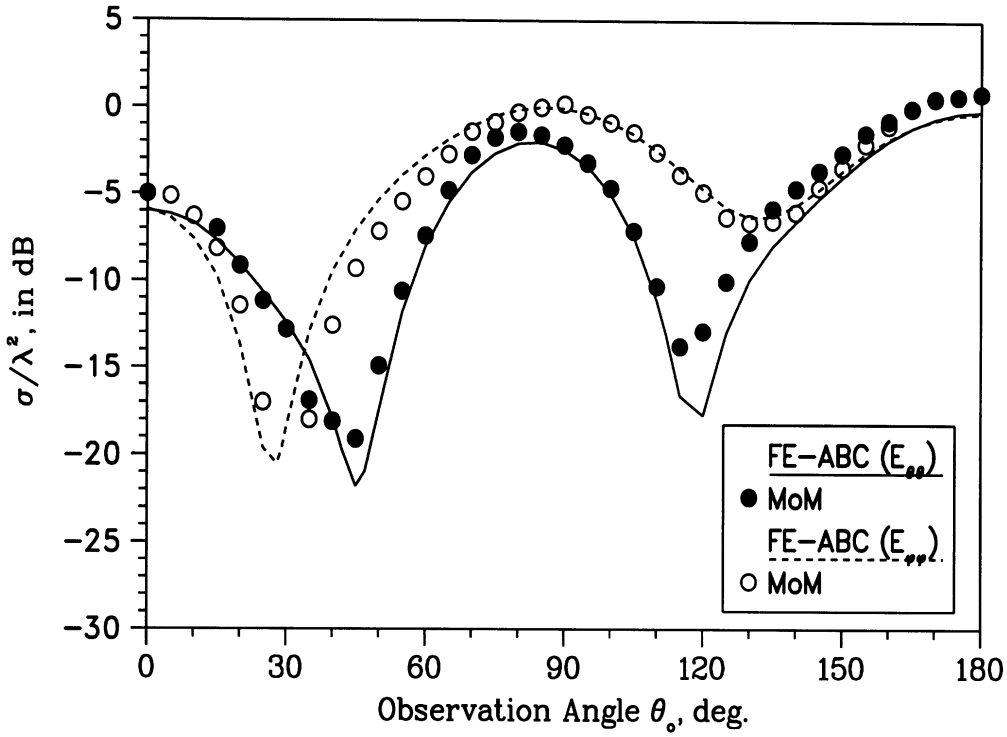

Figure 4.10: RCS pattern in the $x-z$ plane for the composite cube shown in Figure 4.9. The lower half of the cube is metallic while the upper half is air-filled with a resistive card draped over it.

is kept under control. Further, due to the use of edge elements, the program can easily handle sharp conducting edges and tips, inhomogeneous dielectric and/or magnetic materials, resistive sheets and impedance surfaces. These, in conjunction with the well-known advantages of the finite element method, result in low $O(N)$ storage requirement, making the computation of large body scattering possible. These capabilities along with the ease in modeling arbitrary geometries, makes this formulation, to the best of our knowledge, one of the first suitable for solving practical three-dimensional scattering problems.

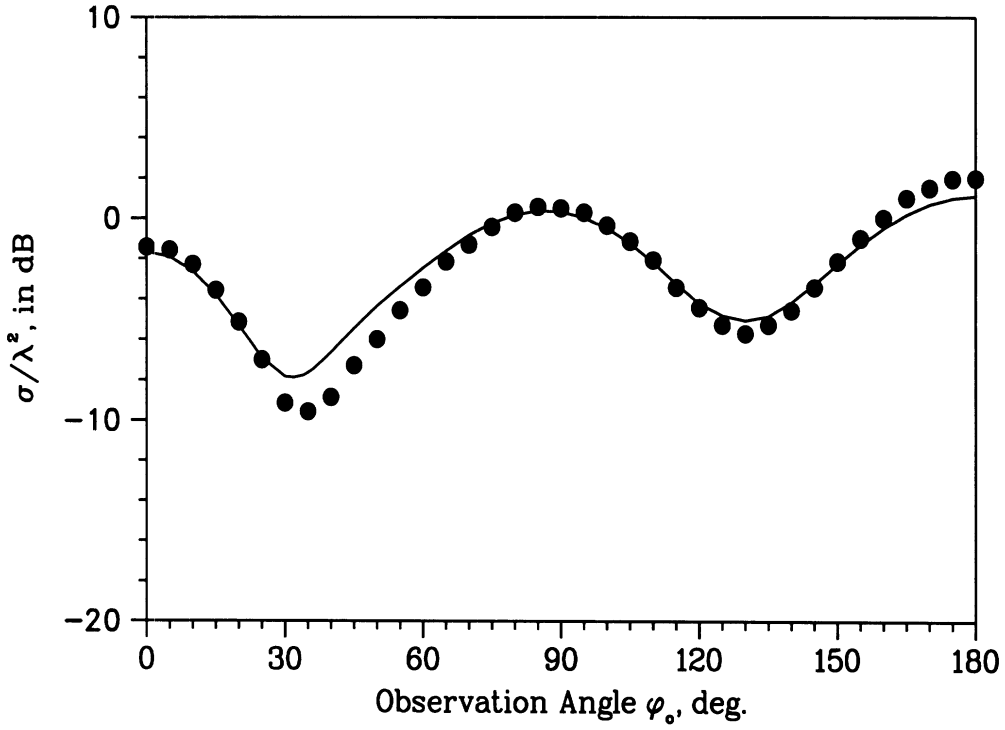

Figure 4.11: RCS pattern in the $x-z$ plane for the composite cube shown in Figure 4.9. The composition of the cube is the same as in Figure 4.10, except that the air-filled portion is filled with dielectric. The solid curve is the FEMATS pattern and the black dots are MoM data for the $E_z^{inc} = 0$ polarization.

CHAPTER V

Optimization and parallelization

In the previous chapter, we laid the foundation for our methodology by outlining the formulation of the finite elment system together with the absorbing boundary condition method of mesh termination and presenting some examples to validate our solution technique. We found that the FE-ABC technique yielded accurate far-field values for small geometries, i.e., structures whose dimensions are less than a wavelength. However, our principal motivation was to compute scattering from **large**, three dimensional structures having arbitrary material inhomogeneities and regions satisfying impedance and/or transition conditions. As mentioned earlier, the number of unknowns escalates rapidly in three dimensions as the target size increases. Therefore, the limiting factor in dealing with three dimensional problems is the unknown count and the associated demands on storage and solution time. Solution techniques which have $O(N)$ storage and feasible solution times are thus the only way that the curse of dimensionality can be avoided. This is one of the principal reasons for the popularity of partial differential equation techniques over integral equation (IE) approaches which lead to dense matrices and $O(N^2)$ storage. As the problem size increases, the IE and hybrid methods, both of which need $O(N^l), 1 < l \leq 2$, storage, quickly become unmanageable in terms of storage and solution time.

Another concern while solving problems having more than 100,000 unknowns - a scenario that can be envisioned for most practical problems - is to avoid software bottlenecks. The algorithmic complexity of any part of the program should increase at most linearly with the number of unknowns.

In this chapter, the implementation details of our finite element code are presented along with the associated numerical considerations. The various trade-offs associated with the data structures used to represent sparse matrices and their impact on vectorization and parallelization are discussed. The iterative solver, a preconditioned biconjugate gradient (BCG) algorithm, is studied along with point and block preconditioning strategies and the trade-offs between the two types of preconditioners are outlined. A modified incomplete LU (ILU) preconditioner is presented, which seems to work better than the original ILU preconditioner for our matrix systems. Iterative solvers for unsymmetric matrix systems are also mentioned to handle anisotropic geometries and situations where the mesh termination condition makes the system unsymmetric. In order to facilitate the solution of large problems, the computationally intensive portions of the finite element code have been parallelized on a variety of massively parallel architectures like the KSR1 (Kendall Square Research) and the Intel iPSC/860. A full analysis of the communication patterns is also presented for the KSR1 machine.

5.1 Numerical considerations

The finite element code implemented by the authors can be divided into four main modules:

- Input/output
- Right-hand side vector ($\mathbf{b}$) generation

- Finite element matrix (**A**) generation
- Linear equation solver

The input to the program consists of the mesh information obtained by pre-processing the mesh file generated from SDRC I-DEAS, a commercial CAD software package. The right-hand side vector,**b**, is usually a sparse vector and only a small fraction of the total CPU time is required to generate it. The finite element matrix generation consists of too many subroutine calls and highly complex loops to permit any significant speedup through vectorization. It is, however, highly amenable to parallelization as will be discussed later. The most time-consuming portion of the code is the linear equation solver, taking up approximately 90% of the CPU time. On a vector computer like the Cray YMP, it is possible to vectorize only the equation solver. However, short vector lengths and indirect addressing inhibit large vector speedups.

5.2 Matrix storage and generation

The matrix systems arising from I-DEAS were very sparse: on the average, the minimum number of non-zero elements per row was 9 and the maximum number of non-zeros per row was 30. The total number of non-zeros varied between $15N$ and $16N$, where N is the number of unknowns.

There are various storage schemes for sparse matrices. In this chapter, we will discuss the ITPACK format [53], the jagged diagonal format and the Compressed Sparse Row (CSR) format. Knowledge of the storage formats is important since the speed of computation on vector or parallel processors is directly linked to the data structure used for matrix storage.

In the ITPACK storage scheme, a sparse matrix $\mathcal{A}$ of order N is stored using two arrays $\mathcal{D}$ and $\mathcal{PC}$. For example, if we have the 5×5 unsymmetric matrix $\mathcal{A}$

$$\mathcal{A} = \begin{bmatrix} 3 & 0 & 0 & 4 & 5 \\ 7 & 0 & 4 & 0 & 2 \\ 4 & 0 & 7 & 0 & 0 \\ 0 & 0 & 8 & 0 & 0 \\ 9 & 7 & 0 & 0 & 0 \end{bmatrix}$$

Then, according to the ITPACK scheme, the rows of the array $\mathcal{D}$ will contain the non-zero elements of the corresponding rows of the original matrix. The number of columns of $\mathcal{D}$ will be equal to the maximum number of non-zeros in a row; rows containing fewer non-zero elements will be zero padded. The array $\mathcal{D}$ thus looks like

$$\mathcal{D} = \begin{bmatrix} 3 & 4 & 5 \\ 7 & 4 & 2 \\ 4 & 7 & 0 \\ 8 & 0 & 0 \\ 9 & 7 & 0 \end{bmatrix}$$

The column indices of the elements in $\mathcal{D}$ are stored in an integer array $\mathcal{PC}$ defined as

$$\mathcal{PC} = \begin{bmatrix} 1 & 4 & 5 \\ 1 & 3 & 5 \\ 1 & 3 & * \\ 3 & * & * \\ 1 & 2 & * \end{bmatrix}$$

The asterisk denotes that the corresponding elements of $\mathcal{D}$ are zeros. The ITPACK storage scheme is attractive for generating finite element matrices since the number

of comparisons required while augmenting the matrix depends only on the locality of the corresponding edge and not on the number of unknowns. Moreover, the sparse matrix-vector multiplication process can be highly vectorized because of large vector lengths when the number of non-zeros in all rows is nearly equal. However for our application, almost half the space is lost in storing zeros. As a result, a lot of storage as well as computational effort is wasted in storing and operating on zeros, respectively.

The modified ITPACK scheme [54], or the method of jagged diagonals, does alleviate this problem to a certain degree by sorting the rows of the matrix by decreasing number of non-zero elements. However, 30% of the allotted space is still lost in zero padding.

The best trade-off between storage and speed for our application is obtained by storing the non-zero matrix elements in a long complex vector, the column indices in a long integer vector and the number of non-zeros per row in another integer vector. This data structure is referred to as the Compressed Sparse Row (CSR) format. A similar data structure which stores the row indices instead of the column indices is called the Compressed Sparse Column (CSC) format. The CSC format is sometimes used when the matrix is to be accessed along the rows and not the columns, e.g., in the multiplication of the transpose of a sparse matrix with a vector. In our implementation, a map of the number of non-zeros for each row is obtained through a simple pre-processor. The main program stores the matrix in CSR format, thus minimizing storage and sacrificing a bit of speed. The required storage is $15N$ to $16N$ complex words plus integers for $\mathcal{D}$ and $\mathcal{PC}$, respectively, and N integers for the array containing the pointers to the rows' data.

5.3 Linear equation solver

In three dimensional applications, the order N of the system of linear equations may be very large. Direct solution methods usually suffer from fill-in to an extent that these large problems cannot be solved at a reasonable cost even on state-of-the-art parallel machines. It is, therefore, essential to employ solvers whose memory requirements are a small fraction of the storage demand of the coefficient matrix. This necessitates the use of iterative algorithms instead of direct solvers to preserve the sparsity pattern of the finite element matrix. Especially attractive are iterative methods that involve the coefficient matrices only in terms of matrix-vector products with $\mathbf{A}$ or $\mathbf{A^T}$. The most powerful iterative algorithm of this type is the conjugate gradient algorithm for solving positive definite linear systems [55]. In our implementation, the system of linear equations is solved by a variation of the CG algorithm, the biconjugate gradient (BCG) method. This scheme is usually used for solving unsymmetric systems; however, it performs equally well when applied to symmetric systems of linear equations. For symmetric matrices, BCG differs from CG in the way the inner product of the vectors are taken.

The conjugate gradient squared (CGS) algorithm [56] performs best when applied to unsymmetric systems of linear equations. It is usually faster than BCG but is more unstable since the residual polynomials are merely the squared BCG polynomials and hence exhibit even more erratic behavior than the BCG residuals. Moreover, there are cases where CGS diverges, while BCG still converges. Recently, Freund [57] has proposed the quasi-minimal residual (QMR) algorithm with look-ahead for complex symmetric matrices.

Based on the above, the biconjugate gradient (BCG) algorithm was found to be

most suitable for our implementation. The BCG requires 1 matrix-vector multiplication, 3 vector updates and 3 dot products per iteration. The solution scheme requires only three additional vectors of length N. The vector updates and the dot products can be carried out extremely fast on a vector Cray machine like the Cray YMP, reaching speeds of about 190 MFLOPS. However, the matrix-vector product, which involves indirect addressing and short vector lengths, runs at about 45.5 MFLOPS on 1 processor of the 8-processor Cray YMP. As a rule of thumb, the biconjugate gradient algorithm with no preconditioning consumes 4.06 microseconds per iteration per unknown on the Cray YMP.

As mentioned earlier, there are two problems which limit the vectorizability of a sparse matrix code - short vector lengths and indirect addressing. There is not much one can do about the second problem since sparse matrices must have indirect addressing to exploit its $O(N)$ storage feature. However, the first problem can be removed by storing the matrix in a different format such that the vector lengths are approximately equal to the order of the system being solved. The storage format is called the *jagged diagonal* format [61]. The rows are ordered by decreasing degree and the leftmost elements of each row are stored as a dense vector with an additional vector indicating the column numbers of each element. The matrix is thus stored as a collection of vectors of decreasing length. The inner loop of the matrix-vector multiplication routine traverses the entire length of a jagged diagonal, which can be of the order of the system being solved. The storage requirement of the above format can be made to be the same as the previously mentioned CSR format through careful programming. The altered code then runs at around $275 Mflops$ on a Cray C-90. It must be mentioned that the CRAY C-90 is a substantially faster machine than the Cray YMP but the CSR formatted matrix-vector multiplication routine runs

about 4 times slower on the C-90. Therefore, we can reliably state that the method of *jaggeddiagonals* is the best sparse matrix storage scheme in terms of computer storage and vectorizability.

5.4 Preconditioning

The condition number of the system of equations usually increases with the number of unknowns. It is then desirable to precondition the coefficient matrix such that the modified system is well-conditioned and converges in significantly fewer iterations than the original system. The equivalent preconditioned system is of the form

$$\left[\mathbf{C^{-1}}\right]\left[\mathbf{A}\right]\left\{\mathbf{x}\right\} \;=\; \left[\mathbf{C^{-1}}\right]\left\{\mathbf{b}\right\} \tag{5.1}$$

The non-singular preconditioning matrix **C** must satisfy the following conditions:

1. should be a good approximation to **A**.

2. should be easy to compute.

3. should be invertible in $O(N)$ operations.

The preconditioners that we discuss below are the diagonal and the ILU point and block preconditioners. Block preconditioners are usually preferable due to reduced data movement between memory level hierarchies as well as decreased number of iterations required for convergence. Block algorithms are also suited for high-performance computers with multiple processors since all scalar, vector and matrix operations can be performed with a high degree of parallelism.

5.4.1 Diagonal preconditioner

The simplest preconditioner that was used in our implementation was the point diagonal preconditioner. The preconditioning matrix **C** is a diagonal matrix which

is easy to invert and has a storage requirement of N complex words, where N is the number of unknowns. The entries of $\mathbf{C}$ are given by

$$C_{ij} = \delta_{ij} A_{ij}, \quad i = 1, \ldots, N; \quad j = 1, \ldots, N \tag{5.2}$$

where δ_{ij} is the Kronecker delta. The matrix $\mathbf{C^{-1}}$ contains the reciprocal of the diagonal elements of $\mathbf{A}$. The algorithm with the diagonal preconditioner converged in about 35% of the number of iterations required for the unpreconditioned case. This suggested that our finite elment matrix was diagonally dominant since the reduction in the number of iterations was rather impressive. The diagonal preconditioner is also easily vectorizable and consumes 4.1 microseconds per iteration per unknown on the Cray YMP, a marginal slowdown over the unpreconditioned system.

A more general diagonal preconditioner is the block diagonal preconditioner. The point diagonal preconditioner is a block diagonal preconditioner with block size 1. The block diagonal preconditioning matrix consists of $m \times m$ symmetric blocks as shown in Figure 5.1. The inverse of the whole matrix is simply the inverse of each individual block put together. If the preconditioning matrix $\mathbf{C}$ is broken up into n blocks of size m, the storage requirement for the preconditioner is at most $m \times N$. However, this method suffers a bit from fill-in since the inverted $m \times m$ blocks are dense even though the original blocks may have been sparse. Due to this reason, large blocks cannot be created since the inverted blocks would lead to full matrices and take a significant fraction of the total CPU time for inversion. However, since the structure of the preconditioning matrix is known a priori, this preconditioner vectorizes well and runs at 194 MFLOPS on the Cray-YMP for a block size of 8. For a test case of 20,033 unknowns, a block size of 2 caused the maximum reduction in the number of iterations(14%) and ran at 197 MFLOPS.

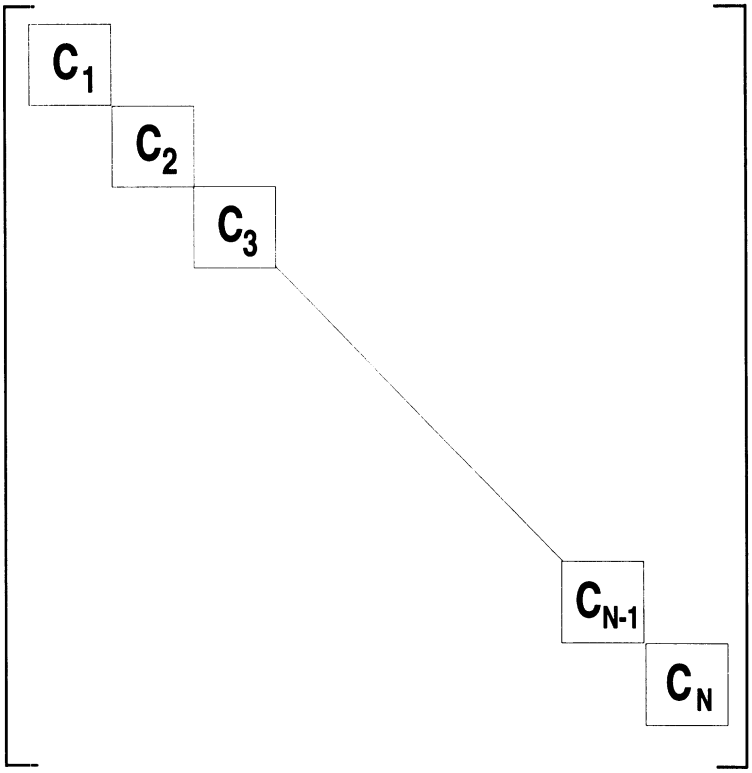

Figure 5.1: Structure of block preconditioning matrix

5.4.2 Modified ILU preconditioner

The next step was to use a better preconditioner to improve the condition number of the system resulting in faster convergence. The traditional ILU preconditioner [58] was employed with zero fill-in; however, the algorithm took a greater number of iterations than the diagonal preconditioner to converge to a specified tolerance. This was probably because the ILU preconditioned system may not have been positive definite [59]. The preconditioned conjugate gradient method usually converges faster if the preconditioner is positive definite, although this is not a necessary condition. Higher values of fill-in were not attempted since the preconditioner already occupied storage space equal to that of the coefficient matrix.

Algorithm 1 *:* **Modified ILU preconditioner with zero fill-in**

It is assumed that the data is stored in CSR format and that the column numbers

for each row are sorted in increasing order. The sparse matrix is stored in the vector $\mathcal{D}$ and the column numbers in $\mathcal{PC}$. $\mathcal{SIG}(i)$ contains the total number of non-zeros till the ith row. The locations of the diagonal entries for each row are stored in the vector $\mathcal{DIAG}$. The preconditioner is stored in another complex vector, $\mathcal{LU}$.

```
for i=1 step 1 until n-1 do
begin
  lbeg=diag(i)
  lend=sig(i)
  for j=lbeg+1 step 1 until lend do
  begin
    jj=pc(j)
    ij=srch(jj,i)
    if (ij.ne.0) then
    begin
      lu(ij)=lu(ij)/lu(lbeg)
    end
  end
end
```

A modified version of the ILU preconditioner was next employed by eliminating the inner loop of the traditional version. The algorithm basically scales the off-diagonal elements in the lower triangular portion of the matrix by the column diagonal. Since the matrix is symmetric, it retains the LDL^T form and is also positive definite if the coefficient matrix is positive definite. This preconditioner is less expensive to generate and converges in about 1/3 the number of iterations taken by the

point diagonal preconditioner. It has been tested with reliable results for $N \leq 50000$. However, the time taken by the two preconditioning strategies is approximately the same since each iteration of the ILU preconditioned system is about three times more expensive. The forward and backward substitutions carried out at each iteration runs at 26.5 MFLOPS on the Cray YMP and proves to be the bottleneck since they are inherently sequential processes and the vector lengths are approximately half that of the sparse matrix-vector multiplication process. The triangular solver is also extremely difficult to parallelize. Techniques like level scheduling and self scheduling try to exploit the fine grain parallelism in the sparse system [60].

We implemented the level scheduling algorithm to examine the potential parallelism in the forward/backward substitution step. For solving any lower triangular system $Lx = b$, the ith unknown in the forward solution is given by

$$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j<i} l_{ij} x_j \right) \tag{5.3}$$

If L is dense, all the components $x_1, \ldots, x_{i-1}$ need to be computed before x_i can be obtained. However, when L is sparse, most of the l_{ij}s are zero; hence, we may not need to compute all of the unknowns $x_1, \ldots, x_{i-1}$ before solving for x_i. Level scheduling is based on this simple observation. The dependencies between the unknowns can be modeled using a graph in which node i corresponds to the unknown x_i and an edge from node j to node i indicates that $l_{ij} \neq 0$ implying that the value of x_j is needed for solving x_i. The operation shown in (5.3) can now be rewritten as

$$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j<i : l_{i,j} \neq 0} l_{ij} x_j \right) \tag{5.4}$$

Thus x_i can be solved at the kth step if all the components x_j in (5.4) have been computed in the earlier steps.

In order to implement the level scheduling algorithm, we need to define the *depth* of a node and the *level* of the graph. The depth of a node is defined as the maximum distance from the root [61]. Therefore, we will place an imaginary root node with links to the nodes having no predecessors so that the depth of each node will be defined from the same point. The depth of each node can now be computed with one pass through the structure of the coefficient matrix L by

$$depth(i) = \left\{ \begin{array}{ll} 1, & \text{if } l_{ij} = 0 \text{ for all } j < i \\ 1 + max_{j<i}\{depth(j) : l_{ij} \neq 0\}, & \text{otherwise} \end{array} \right\} \quad (5.5)$$

The level of the graph can then be defined as the set of nodes with the same depth. The level scheduling algorithm can now be implemented without physically ordering the matrix, but solving the system in increasing order of node depth and distributing the nodes at each depth among the available processors.

Algorithm 2 *:* **Forward elimination step with level scheduling**

The number of levels of the graph , $nlev$, can be easily determined from the depth information. Two other integer vectors are also required. $ORDER(i)$ stores the ordering of the rows of L in terms of increasing node depth. $LEVEL(i)$ stores the index to the start of each level in $ORDER(i)$.

```
do k=1,...,nlev
  do j=ilevel(k),...,ilevel(k+1)-1   (parallel loop)
    i=iorder(j)
    execute Equation 5.4
  enddo
enddo
```

However, our efforts at parallelizing the ILU preconditioned system with level scheduling did not lead to significant speedup mainly due to the enormous amount of mem-

ory traffic that was generated. This observation was also noticed in [60], where the authors estimated that the parallel algorithm generated as much as 10 times more traffic than the sequential code. In order to look for an effective parallelizable ILU preconditioner, we next turned our attention to a block ILU preconditioner.

In our scheme of implementing the block ILU preconditioner, we distribute one block to each processor in a multi-processor architecture, thus achieving load balancing as well as minimizing fill-in. The modified ILU decomposition outlined earlier is then carried out on each of these individual blocks. Further, since the blocks are much larger than the block diagonal version, the preconditioner is a closer approximation to the coefficient matrix. Moreover, the triangular solver is fully parallelized since each processor solves an independent system of equations through forward and backward substitution. In our test case of 20,033 unknowns, the number of iterations was reduced by approximately half the number required by the diagonal preconditioner. Since the work done is less than twice that for the diagonal preconditioner, we achieved a marginal savings of CPU time. However, the number of iterations required for convergence is highly sensitive to block size as shown in Table 5.1 for $N = 20,033$. Table 5.1 clearly shows that a larger block size (smaller number of blocks) does not guarantee faster convergence. Nevertheless, there is an approximately 50% decrease in the number of iterations over the point diagonal preconditioner, regardless of block size. The optimum block size is dependent on the sparsity pattern of the matrix and can only be determined empirically. The savings in the number of iterations over the point diagonal preconditioner for 28 blocks is given in Table 5.2 for a system having 224,476 unknowns. From the table, it is clear that the block ILU preconditioner is very effective in reducing the iteration count; however, the CPU time required is about 10% less than that required by the point diagonal preconditioner for the best

No. of blocks	No. of iterations
1	127
2	176
4	185
8	172
12	162
16	174
24	223
28	177

Table 5.1: No. of iterations vs no. of blocks for a block ILU preconditioned biconjugate gradient solution method.

Angle of incidence	no. of iterations		Ratio (II/I)
	point diagonal (I)	block ILU (II)	
0	2943	2758	.937
10	5985	3834	.641
20	5464	3984	.729
30	6048	3651	.604
40	5770	3256	.564
50	5107	3720	.728
60	6517	4162	.639
70	5076	4108	.809
80	5305	3551	.669
90	2898	2832	.977

Table 5.2: No. of iterations required for convergence of a 224,476 unknown system using the point diagonal and block ILU preconditioning strategies.

case.

5.5 Parallelization

The different versions of the FE-ABC code were parallelized on two different types of massively parallel architectures - the KSR1 and the Intel iPSC/860. The KSR1 is a parallel machine which implements a shared virtual memory, although the memory is physically distributed for the sake of scalability. The Intel iPSC/860, on the other hand, is a distributed memory, Multiple Instruction, Multiple Data (MIMD)

system in which the nodes process information independently of one another and communicate by passing messages to each other. The conversion of sequential or vectorized code to parallel code involves two primary tasks:

- *parallelization of DO loops*

  Parallelism is introduced by allowing each processor to execute a portion of the DO loop.

- *distribution of arrays among the processor set*

  Since each processor only has a limited amount of memory, each array is divided into smaller units that reside on each node. This also allows array accesses from each processor to be serviced by different nodes, thus reducing contention for resources on any single node.

On a cache-only memory machine such as the KSR1, only the first step is necessary since the hardware cache system automatically takes care of data distribution among the processors. This makes porting codes to the KSR1 quite easy. However, the increased control of data distribution and communication on the iPSC/860 can translate into improved performance for some applications. We will first describe our port and performance figures for the KSR1 and then detail our port on the Intel iPSC/860.

1. KSR1 port

The basic strategy for the parallelization of the code is described on the biconjugate gradient solver with diagonal preconditioning. The other versions use the same parallelization scheme with slight modifications. We also comment on the parallelization of the matrix assembly phase.

	Complex		*Real*	
Operation	$*$	$+$	$*$	$+$
Matrix Multiply	nze	$nze - N$	$4nze$	$4nze - 2N$
Vector Updates	$4N$	$3N$	$16N$	$12N$
Dot Products	$3N$	$3N$	$12N$	$12N$

Table 5.3: Floating point operations per iteration.

The symmetric biconjugate gradient method iteratively refines an approximate solution of the given linear system until convergence. Figure 5.2 shows the method in terms of vector and matrix operations. For a system of equations containing N unknowns, all these vectors are of size N and the sparse matrix is of order N. The number of nonzero elements in the sparse matrix is denoted as nze. Table 5.3 shows the operation counts per iteration for each type of vector operation. In the FE-ABC code, each vector operation is implemented as a loop. The program is parallelized by tiling these loops. For P processors, the vectors are divided into P sections of N/P consecutive elements. Each processor is assigned the same section of each vector. This partitioning attempts to reduce communication while balancing load. To guarantee correctness, synchronization points are added after lines 2, 7, and 9. Lines 2 and 7 require synchronization to guarantee that the dot products are computed correctly. Note that the dot products in lines 6 and 7 require only one synchronization. The line 9 synchronization guarantees that $\boldsymbol{p}$ is completely updated before the matrix multiply for the next iteration begins.

In the sparse matrix vector multiplication, each processor computes a block of the result vector by multiplying the corresponding block of rows of the sparse matrix with the operand vector. Since the operand vector is distributed among the processors, data communication is required. The communication pattern is determined by the sparsity structure of the matrix, which in our case is derived from an unstructured

Initialization:

$\boldsymbol{x}$ given

$\boldsymbol{r} = \boldsymbol{b} - \mathrm{A}\boldsymbol{x}$; $\boldsymbol{p} = \boldsymbol{r}$; $tmp = \boldsymbol{r} \cdot \boldsymbol{r}$

Repeat until $(resd \leq tol)$

$$\left.\begin{array}{ll} \boldsymbol{q} = \mathrm{A}\boldsymbol{p} & (1) \\ \alpha = tmp/(\boldsymbol{q} \cdot \boldsymbol{p}) & (2) \end{array}\right\} Step\ 1$$

$$\left.\begin{array}{ll} \boldsymbol{x} = \boldsymbol{x} + \alpha\boldsymbol{p} & (3) \\ \boldsymbol{r} = \boldsymbol{r} - \alpha\boldsymbol{q} & (4) \\ \boldsymbol{q} = \mathcal{C}^{-1} * \boldsymbol{r} & (5) \\ resd = \sqrt{|\boldsymbol{r} \cdot \boldsymbol{r}^*|} & (6) \\ \beta = (\boldsymbol{r} \cdot \boldsymbol{q})/tmp & (7) \end{array}\right\} Step\ 2$$

$$\left.\begin{array}{ll} tmp = \beta \times tmp & (8) \\ \boldsymbol{p} = \boldsymbol{q} + \beta\boldsymbol{p} & (9) \end{array}\right\} Step\ 3$$

EndRepeat

$\mathcal{A}$ is a sparse complex symmetric matrix.

$\mathcal{C}$ is the preconditioning matrix.

$\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{x}, \boldsymbol{r}$ are complex vectors.

α, β, tmp are complex scalars; $resd, tol$ are real scalars.

Figure 5.2: Symmetric biconjugate gradient method with preconditioning.

mesh. Therefore the communication pattern is unstructured and irregular. However since the sparse matrix is not modified during the iterative process, the communication pattern is the same at each iteration. Vector updates and dot products are easily parallelized using the same block distribution as in the sparse matrix vector multiply.

Although sparse computations are known to be hard to implement efficiently on distributed memory machine, mainly because of the unstructured and irregular communication pattern, the previous scheme was easily and efficiently implemented on the KSR1 MPP thanks to the global address space [62]. Table 5.4 shows the execution time of one iteration (in seconds) and the speedup for different numbers of processors and for two problem sizes.

For both problems, the performance scales surprisingly well up to a large number of processors. For the 20,033-unknown problem, the speedup for the parallelized sparse solver varies from 1 to 38 as the number of processors is increased from 1 to 56 (Figure 5.3). The overall performance of the solver on 28 processors is more than three times that of a single processor on the Cray-YMP. The large problem (224,476 unknowns) exhibits superlinear speedup which can be attributed to a memory effect. As a matter of fact, the large data set does not entirely fit in the local cache of a single node in the KSR which results in a large number of page faults. However, as the number of processors increases, the large data set is distributed over the different processors' memories.

The global matrix assembly is the second largest computation in terms of execution time. The elemental matrices are computed for each element in the 3D mesh and assembled in a global sparse matrix. A natural way of parallelizing the global matrix assembly is to distribute the elements over the processors, have each pro-

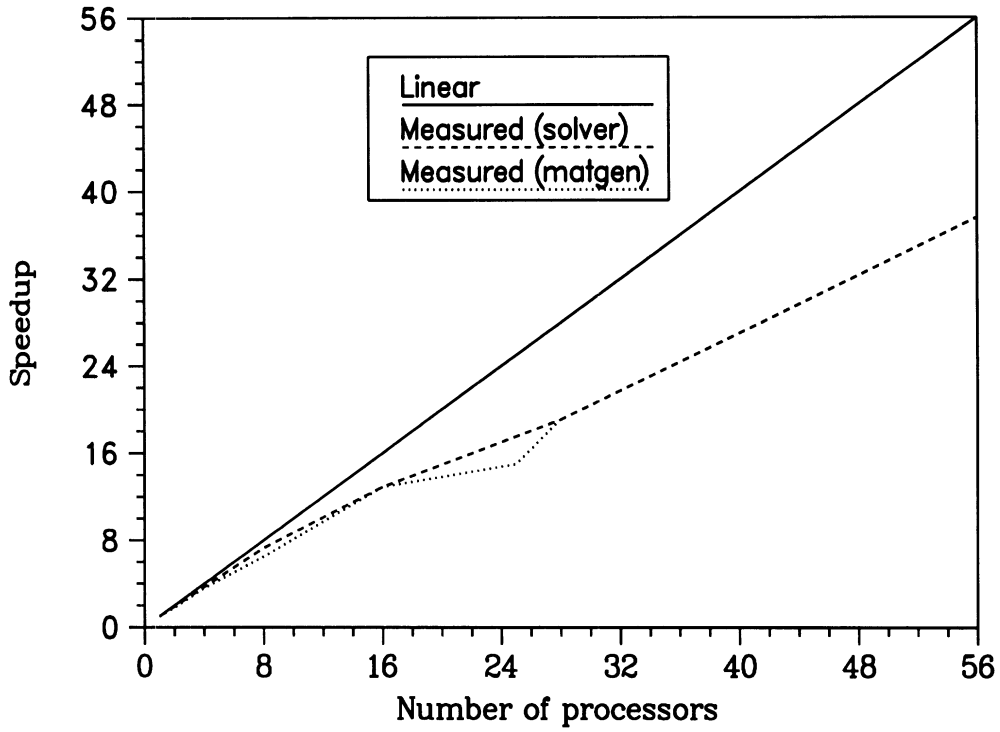

Figure 5.3: Speedup curve for the linear equation solver on the KSR1

cessor compute the elemental matrix of the elements it owns and update the global sparse matrix. Since the global sparse matrix is shared by all processors, the update needs to be done atomically. On the KSR1 this is done by using the hardware lock mechanism.

The performance for the matrix assembly is given in Table 5.5 and also in Figure 5.3.

5.5.1 Analysis of Communication

In the main loop (Figure 5.2), significant communication between processors takes place only during the sparse matrix vector multiply (line 1) and the vector update of $\boldsymbol{p}$ (line 9). The rest of the vector operations incur little or no communication at all. The distribution of the nonzero entries in the matrix affects the amount and nature

[a] For 1, 8 and 16 processors, only the first 100 iterations were run.
[b] Code run on a 64 node KSR at Cornell University

Procs	N=20,033		N=224,476	
	Execution time (secs per iter)	Speedup	Execution time (secs per iter)	Speedup
1[a]	.515	1	10.8	1
8	.071	7.3	1.4	7.7
16	.040	12.9	.671	16.1
29	.027	19.1	.304	35.6
60[b]			.149	76.2

Table 5.4: Execution time and speedup for the iterative solver

of communication. In this section, we present an analysis of the communication pattern incurred by the sparse matrix vector multiplication as derived from analysis of the sparsity structure of the matrix.

Line 1. In the matrix-vector multiply, each processor computes an N/P-sized subsection of the product $\boldsymbol{q}$. The processor needs the elements of $\boldsymbol{p}$ that correspond to the nonzero elements found in the N/P rows of A that are aligned with its subsection.

Procs	Execution time in seconds	Speedup
1	24.355	1
2	13.376	1.8
4	6.811	3.6
8	3.744	6.5
16	1.89	12.9
25	1.625	15.0
28	1.276	19.1

Table 5.5: Execution time and speedup for the matrix generation and assembly (20,033 unknowns)

Because the matrix A remains constant throughout the program, the set of elements of $\boldsymbol{p}$ that a given processor needs is the same for all iterations in the loop. However, since $\boldsymbol{p}$ is updated at the end of each iteration, all copies of its element set are invalidated in each processor's local cache except for the ones that the processor itself updates. As a result, in each iteration, processors must obtain updated copies of the required elements of $\boldsymbol{p}$ that they do not own.

These elements can be updated by a read miss to the corresponding subpage, by an automatic update, or by an explicit prefetch or poststore instruction. Figure 5.4

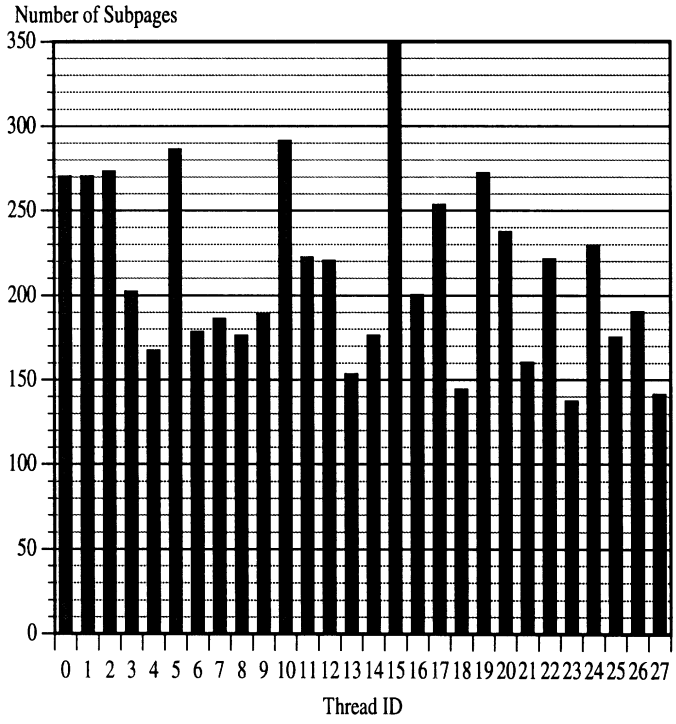

Figure 5.4: Counts of **p** subpages required by each processor for sparse matrix-vector multiply (total copies=5968)

lists the number of subpages that each of the 28 processors needs to acquire from other processors. Automatic update of an invalid copy of a subpage becomes more likely as the number of processors sharing this subpage grows. The number of processors that need a given subpage (excluding the processor that updates the subpage) is

referred to as the *degree of sharing* of that subpage. Figure 5.5 shows the degree of

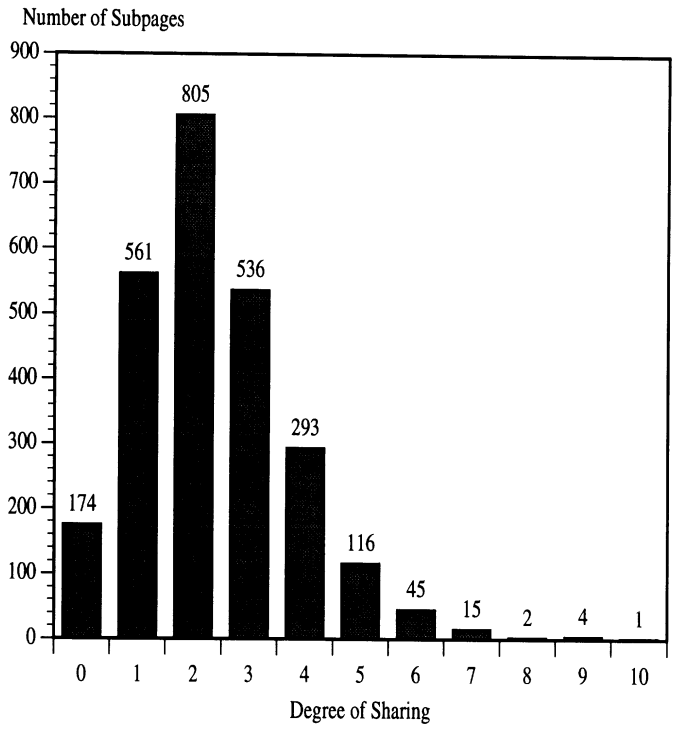

Figure 5.5: Degree of sharing histogram of **p** subpages during sparse matrix-vector multiply (28 procesors)

sharing histogram for the example problem. Since the only subpage misses occurring in Step 1 of the sparse solver are coherence misses due to the vector $\boldsymbol{p}$, the use of the poststore instruction to broadcast the updated sections of the vector $\boldsymbol{p}$ from Step 3 should eliminate the subpage misses in Step 1. However, the overhead of executing the poststore instruction in Step 3 offsets the reduction in execution time of Step 1. On a poststore, the processor typically stalls for 32 cycles while the local cache is busy for 48 cycles. As a result, the net reduction in execution time is only 3%.

Line 9. Before proceeding with the updates of the N/P elements of $\boldsymbol{p}$ for which it is responsible, each processor must acquire exclusive ownership for those elements. Because a cache line holds 8 consecutive elements, each processor will generate $N/8P$ requests for ownership (assuming all subpages are shared). In order to hide access

latencies, the request for ownership can be issued in the form of a prefetch instruction after step 1. This could lead to an eightfold decrease in the number of subpage misses. However, as with the poststore instruction, the benefit of prefetching is offset by the overhead of processing the prefetch instructions in step 2. This is because the processor stalls for at least two cycles on a prefetch and the local cache cannot satisfy any processor request until the prefetch is put on the ring. The overall execution time is reduced by only 4% in this case.

Lines 2, 6, 7. The rest of the communication is due to the three dot products. Each processor computes the dot product for the vector subsection that it owns. These are then gathered and summed up on a single processor.

2. Intel iPSC/860 port

The parallelization of the DO loops is one of the main tasks since the majority of the computer time is spent on solving the linear system of equations. The basic strategy for parallelizing the DO loops on the iPSC/860 is similar to the KSR1 - each portion executes a portion of the DO loop. This scheme works fine as long as there are no dependencies in the body of the loop, as is the case for the vector updates and the sparse matrix-vector multiply of the linear solver. However, the main loop in the matrix generation/assembly phase contains a dependency between loop iterations. As on the KSR1, this problem is solved by using a mechanism by which each processor locks a row of the matrix while performing an update. Since the locking of each row is maintained by the processor whose memory holds the particular row, processors lock and unlock rows by sending messages to the appropriate row owner.

Even though the parallelization of loops enables programs to run faster on multiprocessors, the distribution of arrays must be done for the code to run at all. Arrays

are distributed in the code by partitioning one dimension among the processors. Thus for a 1000 element array, processor holds the first 100 elements, processor 2 the next 100 and so on. The straightforward method for accessing this distributed array involves the translation of array references into subroutine calls. Thus an expression $x = a(i)$ is translated into the call $call \quad fetcha(i, x)$. The subroutine $fetcha$ then sends a message to the processor that holds element $a(i)$, which in turn sends a reply message with the value of $a(i)$. Although this scheme requires the implementation of a new subroutine for each distributed array and the replacement of each array access with a subroutine call, the process is easy and mechanical. However, such a scheme does not result in good performance.

The primary reason for this is that the overhead for sending a message is much higher than that of sending a single byte. The cost for sending 10 or even 100 bytes is usually not much higher than that of sending 1 byte. Thus, messages need to be 'bundled' for fast and efficient operation. However, the simple strategy mentioned above is in direct contrast to message bundling. Therefore, we have implemented the simple scheme for parts of the code that do not take up a significant portion of computation time like the matrix generation/assembly phase and a better scheme for accessing the distributed arrays in the equation solver phase.

The primary operation in the solver that generates communication is the sparse matrix-vector product. Since the matrix-vector product involves performing a dot product of each row with the distributed vector, each processor must obtain the values for the entire vector from the other processors. The dot product operation must be carried out in several phases as each processor may not be able to hold the entire vector in memory. Thus each processor P begins the matrix-vector multiply by sending its portion of the vector to other processors, then performs the following

tasks for every other processor P'

- Reads the portion of the vector owned by P'.
- Updates the partial dot product for each row by adding the product of the appropriate matrix element with the elements of the partial vector.

After performing the above operations for all the processors, the dot product is complete. Unfortunately, each phase requires a pass over all the sparse matrix rows owned by the processor. In the future, it may be possible to sort each row of the matrix to allow the phases to pass over the rows in order.

The speedup results on a problem with 31000 unknowns show that the problem scales reasonably well for a small number of processors. However, as the number of processors increases, much of the time is spent on communication and book-keeping than on true computation. Efforts are under way to run a larger problem.

CHAPTER VI

Conformal absorbing boundary conditions for the vector wave equation

As mentioned in the previous chapters, the focus of this thesis is the computation of scattering from large three-dimensional structures. Since we are dealing with large targets having arbitrary shape, a spherical mesh termination boundary is not as attractive in terms of storage and computational cost. This is especially true for long and thin geometries where a sphere is the least economical shape of mesh termination, in terms of the number of unknowns. The ideal situation would be to enclose the scatterer inside a mesh termination boundary which is of the same shape as the scattering body (see Figure 6.1). If boundary conditions could be derived for such conformal mesh truncation surfaces, the volume to be meshed and the corresponding computing cost would then be minimized. However, available three dimensional ABCs for the vector wave equation as derived by Peterson [47] and Webb and Kanellopoulos [48] are only suited for application on spherical mesh terminations.

Our goal, therefore, is to derive new vector ABCs for three dimensional analysis which can be applied on a surface conformal to the structure of interest. We begin with a modified Wilcox expansion whose leading order term recovers the geometrical

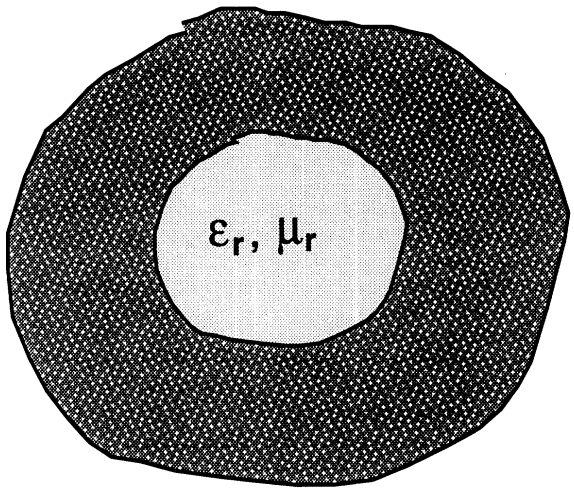

Figure 6.1: Scatterer enclosed in conformal mesh termination boundary

optics fields and thus, given the appropriate principal curvatures, the resulting ABC completely absorbs all geometrical optics fields from the surface. We then proceed to derive the first and second order absorbing boundary conditions in terms of the principal curvatures of the surface on which they are employed. We also introduce an approximation to make the absorbing boundary condition contribution symmetric. In the next step, we incorporate these boundary conditions into the finite element equations and express them in a readily implementable form. We also comment on the symmetry of the system for doubly curved surfaces. In the last section, we examine the performance of these ABCs - in terms of computational cost - when applied on mesh termination surfaces conformal to the scattering object.

6.1 Formulation

It is known that the electric field in a homogeneous region of space is governed by the vector wave equation

$$\nabla\times\nabla\times\mathbf{E} - k_o^2\mathbf{E} = 0 \tag{6.1}$$

where k_o is the free-space wave number. We assume that the field has a well-defined phase front in the region under consideration. Since we are concerned only with local behavior, we can assume that the phase fronts can be treated as parallel regions. Consequently, the surface describing the phase fronts can be specified by a net of coordinate curves denoted by t_1 and t_2 and a third variable n denotes the coordinate along the normal to the phase front. The point of observation in the Dupin coordinate system [63] can now be defined as

$$\mathbf{x} = n\hat{\mathbf{n}} + \mathbf{x}_o(t_1, t_2) \tag{6.2}$$

where $\hat{\mathbf{n}}$ is the unit normal and $\mathbf{x}_o(t_1, t_2)$ denotes the surface of the reference phase front. The curl of a vector in the above coordinate system is given by

$$\nabla \times \mathbf{E} = \nabla_T \times \mathbf{E} + \hat{\mathbf{n}} \times \frac{\partial \mathbf{E}}{\partial n} \tag{6.3}$$

where $\nabla_T \times \mathbf{E}$ is called the surface curl involving only the tangential derivatives and is defined as [64]

$$\nabla_T \times \mathbf{E} = -\hat{\mathbf{n}} \times \nabla E_n + \hat{\mathbf{t}}_2 \kappa_1 E_{t_1} - \hat{\mathbf{t}}_1 \kappa_2 E_{t_2} + \hat{\mathbf{n}} \nabla \cdot (\mathbf{E} \times \hat{\mathbf{n}}) \tag{6.4}$$

In (6.4), κ_1 and κ_2 denote the principal curvatures of the surface under consideration, E_{t_1}, E_{t_2} are the tangential components and E_n is the normal component of the electric field on the surface. The principal curvature of a surface is defined as [63]

$$\kappa_1 = \frac{1}{R_1} = -\frac{1}{h_1}\frac{\partial h_1}{\partial n} \tag{6.5}$$

$$\kappa_2 = \frac{1}{R_2} = -\frac{1}{h_2}\frac{\partial h_2}{\partial n} \tag{6.6}$$

where h_1, h_2 are the metric coefficients and R_1, R_2 are the principal radii of curvature.

Using the aforementioned coordinates, the Wilcox expansion for a vector radiating

function can now be generalized to read

$$\mathbf{E}(n,t_1,t_2) = \frac{e^{-jk_o n}}{4\pi\sqrt{R_1R_2}}\sum_{p=0}^{\infty}\frac{\mathbf{E}_p(t_1,t_2)}{\left(\sqrt{R_1R_2}\right)^p} \tag{6.7}$$

where $R_i = \rho_i + n,\ i = 1,2$ and ρ_i is the principal radius of curvature associated with the outgoing wavefront at the target. The lowest-order term in (6.7) represents the geometrical optics spread factor for a doubly curved wavefront and reduces to the standard Wilcox expansion [65] for a spherical wave. Moreover, (6.7) can be differentiated term by term any number of times and the resulting series converges absolutely and uniformly [65].

6.1.1 Unsymmetric ABCs

In the 3D finite element implementation using vector basis functions and the electric field as the working variable, we need to relate the tangential component of the magnetic field in terms of the electric field at any surface discontinuity. Therefore, our next task is to derive a relation between $\hat{\mathbf{n}}\times\nabla\times\mathbf{E}$ (i.e., $\hat{\mathbf{n}}\times\mathbf{H}$ where $\mathbf{H}$ is the magnetic field) and the tangential components of the electric field on the surface. Taking the curl of the electric field expansion given by (6.7) and crossing it with the normal vector, we have

$$\hat{\mathbf{n}}\times\nabla\times\mathbf{E} = \frac{e^{-jk_o n}}{4\pi}\sum_{p=0}^{\infty}\left[\frac{(jk_o+\kappa_m-\overline{\boldsymbol{\eta}}\cdot)\mathbf{E}_{pt}}{u^{p+1}} + \frac{\nabla_t E_{pn} + p\kappa_m\mathbf{E}_{pt}}{u^{p+1}}\right] \tag{6.8}$$

where $u = \sqrt{R_1R_2}$ and

$$\begin{aligned}\nabla_t E_n &= -(\hat{\mathbf{n}}\times\hat{\mathbf{n}}\times\nabla)E_n\\ \kappa_m &= \frac{\kappa_1+\kappa_2}{2}\\ \overline{\boldsymbol{\eta}} &= \kappa_1\hat{\mathbf{t}}_1\hat{\mathbf{t}}_1 + \kappa_2\hat{\mathbf{t}}_2\hat{\mathbf{t}}_2\end{aligned}$$

Considering that E_{0n} is zero due to the divergenceless condition [65] and simplifying, we obtain the first order absorbing boundary condition

$$\hat{\mathbf{n}} \times \nabla \times \mathbf{E} - (jk_o + \kappa_m - \overline{\boldsymbol{\eta}} \cdot) \mathbf{E}_t = \frac{e^{-jk_o n}}{4\pi} \sum_{p=1}^{\infty} \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} \tag{6.9}$$

$$\text{or, } \hat{\mathbf{n}} \times \nabla \times \mathbf{E} - (jk_o + \kappa_m - \overline{\boldsymbol{\eta}} \cdot) \mathbf{E}_t = 0 + O\left(n^{-3}\right) \tag{6.10}$$

for a conformal outer boundary. Not surprisingly, this is the impedance boundary condition for curved surfaces as derived by Rytov [66]. It should be noted that in the above equation, $\nabla_t E_n$ and κ_m are each proportional to n^{-1}. Therefore, the leading order behavior of (6.10) is $O(n^{-3})$, i.e., only the first two terms of (6.7) are exactly satisfied by (6.10). If the scattered field contains higher order terms, application of (6.10) will give rise to non-physical reflections back into the computational domain. In order to reduce these spurious reflections, we need to either shift the mesh truncation boundary farther away from the scatterer or employ higher order boundary conditions which satisfy higher order terms of (6.7).

To reduce the order of the residual error further, we consider the tangential components of the curl of (6.9). This yields

$$\begin{aligned} &\hat{\mathbf{n}} \times \nabla \times [\hat{\mathbf{n}} \times \nabla \times \mathbf{E} - (jk_o + \kappa_m - \overline{\boldsymbol{\eta}} \cdot) \mathbf{E}_t] = \\ &\quad \frac{e^{-jk_o n}}{4\pi} \sum_{p=1}^{\infty} \left[\left\{ jk_o + (p+3)\kappa_m - \frac{\kappa_g}{\kappa_m} \right\} \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} \right. \\ &\qquad \left. - \left(2\kappa_m - \frac{\kappa_g}{\kappa_m}\right) \frac{\nabla_t E_{pn}}{u^{p+1}} - \frac{p\kappa_m \overline{\boldsymbol{\eta}} \cdot \mathbf{E}_{pt}}{u^{p+1}} \right] \end{aligned} \tag{6.11}$$

where $\kappa_g = \kappa_1 \kappa_2$ is the Gaussian curvature. Using the result derived in (6.9) and simplifying, (6.11) reduces to

$$\begin{aligned} &\hat{\mathbf{n}} \times \nabla \times [\hat{\mathbf{n}} \times \nabla \times \mathbf{E} - (jk_o + \kappa_m - \overline{\boldsymbol{\eta}} \cdot) \mathbf{E}_t] = \\ &\quad \frac{e^{-jk_o n}}{4\pi} \sum_{p=1}^{\infty} \left[\left\{ jk_o + (p+3)\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}} \cdot \right\} \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} \right] \\ &\qquad - \left(2\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}} \cdot\right) \nabla_t E_n \end{aligned} \tag{6.12}$$

If we take a closer look at the term in the square brackets on the RHS of (6.12), we find that it can be written as

$$\frac{e^{-jk_on}}{4\pi}\sum_{p=1}^{\infty} p\kappa_m \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} + \left(jk_o + 3\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}}\cdot\right)\left\{\hat{\mathbf{n}} \times \nabla\times\mathbf{E} - \left(jk_o + \kappa_m - \overline{\boldsymbol{\eta}}\cdot\right)\mathbf{E}_t\right\}$$

where we have substituted

$$\frac{e^{-jk_on}}{4\pi}\sum_{p=1}^{\infty}\left[\frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}}\right] = \hat{\mathbf{n}} \times \nabla\times\mathbf{E} - \left(jk_o + \kappa_m - \overline{\boldsymbol{\eta}}\cdot\right)\mathbf{E}_t \tag{6.13}$$

using the relation derived in (6.9).

Now the dominant terms on the RHS of (6.12) can be eliminated by considering the higher-order operator

$$\left[\hat{\mathbf{n}} \times \nabla\times - \left(jk_o + 3\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}}\cdot\right)\right]\left\{\hat{\mathbf{n}} \times \nabla\times\mathbf{E} - \left(jk_o + \kappa_m - \overline{\boldsymbol{\eta}}\cdot\right)\mathbf{E}_t\right\} + \left(2\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}}\cdot\right)\nabla_t E_n = \frac{e^{-jk_on}}{4\pi}\sum_{p=1}^{\infty} p\kappa_m \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} \tag{6.14}$$

The residual of (6.14) can be reduced further to yield the absorbing boundary condition of second order which satisfies (6.7) to $O(n^{-5})$. This second order ABC is found to be

$$\left[\hat{\mathbf{n}} \times \nabla\times - \left(jk_o + 4\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}}\cdot\right)\right]\left\{\hat{\mathbf{n}} \times \nabla\times\mathbf{E} - \left(jk_o + \kappa_m - \overline{\boldsymbol{\eta}}\cdot\right)\mathbf{E}_t\right\} + \left(2\kappa_m - \frac{\kappa_g}{\kappa_m} - \overline{\boldsymbol{\eta}}\cdot\right)\nabla_t E_n = 0 \tag{6.15}$$

and the residual is equal to

$$\frac{e^{-jk_on}}{4\pi}\sum_{p=2}^{\infty}(p-1)\kappa_m \frac{\nabla_t E_{pn} + p\kappa_m \mathbf{E}_{pt}}{u^{p+1}} \tag{6.16}$$

The operator on the LHS of (6.15) can be applied repeatedly to obtain ABCs of increasing order; however, higher order basis functions are needed for their implementation.

After some algebraic manipulation, the terms on the LHS of (6.15) reduced to simpler ones. In addition to the wave equation, the following vector identities were derived to carry out the simplifications and are provided below for the reader's convenience:

$$\begin{aligned}
\hat{\mathbf{n}} \times \nabla \times \mathbf{E}_t &= \hat{\mathbf{n}} \times \nabla \times \mathbf{E} - \nabla_t E_n \\
\hat{\mathbf{n}} \times \nabla \times \nabla_t E_n &= \nabla_t \left(\nabla \cdot \mathbf{E}_t \right) + 2\kappa_m \nabla_t E_n \\
\hat{\mathbf{n}} \times \nabla \times \left(\hat{\mathbf{n}} \times \nabla \times \mathbf{E} \right) &= \nabla \times \left\{ \hat{\mathbf{n}} \left(\nabla \times \mathbf{E} \right)_n \right\} - k_o^2 \mathbf{E}_t + \Delta\kappa \, \hat{\mathbf{n}} \times \nabla \times \mathbf{E}
\end{aligned}$$

where $\Delta\kappa = \kappa_1 - \kappa_2$. The derivation of these identities is given in Appendix B. Upon simplification, the second order ABC can be compactly written as

$$\begin{aligned}
-\left(D - \Delta\kappa - 2\overline{\boldsymbol{\eta}} \cdot \right) \hat{\mathbf{n}} \times \nabla \times \mathbf{E} + \left\{ 4\kappa_m^2 - \kappa_g + D \left(jk_o - \overline{\boldsymbol{\eta}} \cdot \right) + \left(\overline{\boldsymbol{\eta}} \right)^2 \cdot \right\} \mathbf{E}_t \\
+ \nabla \times \left\{ \hat{\mathbf{n}} \left(\nabla \times \mathbf{E} \right)_n \right\} + \left(jk_o + 3\kappa_m - \frac{\kappa_g}{\kappa_m} - 2\overline{\boldsymbol{\eta}} \cdot \right) \nabla_t E_n = 0 \qquad (6.17)
\end{aligned}$$

in which

$$D = 2jk_o + 5\kappa_m - \frac{\kappa_g}{\kappa_m}$$

and

$$\left(\overline{\boldsymbol{\eta}} \right)^2 \cdot \mathbf{E}_t = \kappa_1^2 E_{t_1} \hat{\mathbf{t}}_1 + \kappa_2^2 E_{t_2} \hat{\mathbf{t}}_2$$

The derivatives of the curvatures in (6.17) have been ignored due to the reasons outlined in [64]. These derivatives essentially give rise to second-order curvature terms which add to the coefficient of $\mathbf{E}_t$ only. The second order ABC derived in [1] is recovered on setting $\kappa_1 = \kappa_2 = 1/r$.

6.1.2 Symmetric correction

It has been shown by Peterson in [47] that the LHS of (6.17) when incorporated into the finite element equations gives rise to an unsymmetric matrix system

in spherical coordinates. To alleviate this problem, Kanellopoulos and Webb [48] suggested an alternative derivation involving an arbitrary parameter which would lead to a symmetric matrix while sacrificing some accuracy. Below, we discuss a different approach which leads to a symmetric ABC without the introduction of an arbitrary parameter.

On considering the series expansion of the term $\hat{\mathbf{n}} \times \nabla \times \nabla_t E_n$, we have

$$\begin{aligned}
\hat{\mathbf{n}} \times \nabla \times \nabla_t E_n &= \frac{e^{-jk_o n}}{4\pi} \sum_{p=1}^{\infty} \{jk_o + (p+1)\kappa_m\} \frac{\nabla_t E_{pn}}{u^{p+1}} \\
&= jk_o \nabla_t E_n + 2\kappa_m \nabla_t E_n + \sum_{p=2}^{\infty} (p-1)\kappa_m \frac{\nabla_t E_{pn}}{u^{p+1}} \\
&= jk_o \nabla_t E_n + 2\kappa_m \nabla_t E_n + O\left(n^{-5}\right)
\end{aligned}$$

and on making use of the vector identity

$$\nabla_t (\nabla \cdot \mathbf{E}_t) = \hat{\mathbf{n}} \times \nabla \times \nabla_t E_n - 2\kappa_m \nabla_t E_n$$

given earlier, we arrive at the following result

$$\nabla_t (\nabla \cdot \mathbf{E}_t) = jk_o \nabla_t E_n + O\left(n^{-5}\right) \tag{6.18}$$

Since our ABC was derived to have a residual error of $O(n^{-5})$, we can replace $jk_o \nabla_t E_n$ with $\nabla_t (\nabla \cdot \mathbf{E}_t)$ without affecting the order of the approximation. Doing so, the second order ABC with a symmetric operator can be rewritten as

$$\begin{aligned}
(D - \Delta\kappa - 2\overline{\boldsymbol{\eta}}\cdot)\, \hat{\mathbf{n}} \times \nabla \times \mathbf{E} = \left\{4\kappa_m^2 - \kappa_g + D\,(jk_o - \overline{\boldsymbol{\eta}}\cdot) + (\overline{\boldsymbol{\eta}})^2 \cdot\right\} \mathbf{E}_t + \\
\nabla \times \{\hat{\mathbf{n}}\,(\nabla \times \mathbf{E})_n\} + \frac{1}{jk_o}\left(jk_o + 3\kappa_m - \frac{\kappa_g}{\kappa_m} - 2\overline{\boldsymbol{\eta}}\cdot\right) \nabla_t (\nabla \cdot \mathbf{E}_t)
\end{aligned} \tag{6.19}$$

It can be easily shown that the above boundary condition leads to a symmetric system of equations when incorporated into the finite element functional for surfaces having $\kappa_1 = \kappa_2$. Equations (6.10) and (6.19) reduce to the boundary conditions derived in [48] on setting $\kappa_1 = \kappa_2 = 1/r$ which have been found to work well for spherical and flat boundaries [67].

6.1.3 Finite element implementation

The boundary condition outlined in equation (6.19) cannot be incorporated into the finite element equations without modification. As explained in Chapter 3, the absorbing boundary condition is implemented in the finite element system through the surface integral over the mesh termination surface S_o.

$$\int_{S_o} \mathbf{E} \cdot \hat{\mathbf{n}} \times \nabla \times \mathbf{E} \, dS \;=\; \int_{S_o} \mathbf{E} \cdot P(\mathbf{E}) \, dS$$

where $P(\mathbf{E})$ denotes the boundary condition relating the tangential magnetic field to the tangential electric field on the surface.

Let $P_1(\mathbf{E})$ denote the first order absorbing boundary condition given by (6.10), where the subscript represents the order of the ABC. Therefore, the surface integral contribution for the first order ABC reduces to

$$\int_{S_o} \mathbf{E} \cdot P_1(\mathbf{E}) \, dS \;=\; (jk_o + \kappa_m) \int_{S_o} \mathbf{E} \cdot \mathbf{E}_t \, dS - \int_{S_o} \mathbf{E} \cdot (\overline{\boldsymbol{\eta}} \cdot \mathbf{E}_t) \, dS \qquad (6.20)$$

Using some basic vector identities and considering that $\mathbf{E}_t = -\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \mathbf{E}$, we deduce that

$$\int_{S_o} \mathbf{E} \cdot P_1(\mathbf{E}) \, dS \;=\; (jk_o + \kappa_m) \int_{S_o} \left(E_{t_1}^2 + E_{t_2}^2 \right) \, dS - \int_{S_o} \left(\kappa_1 E_{t_1}^2 + \kappa_2 E_{t_2}^2 \right) \, dS \qquad (6.21)$$

which is a readily implementable form of the first order ABC. However, the second order ABC does not simplify as easily. If $P_2(\mathbf{E})$ denotes the second order ABC given by (6.19), we can rewrite it in a more compact vector notation as

$$P_2(\mathbf{E}) \;=\; \overline{\boldsymbol{\alpha}} \cdot \mathbf{E}_t + \overline{\boldsymbol{\beta}} \cdot \left[\nabla \times \left\{ \hat{\mathbf{n}} \left(\nabla \times \mathbf{E} \right)_n \right\} \right] + \overline{\boldsymbol{\gamma}} \cdot \left\{ \nabla_t \left(\nabla \cdot \mathbf{E}_t \right) \right\} \qquad (6.22)$$

where the tensors $\overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\beta}}$ and $\overline{\boldsymbol{\gamma}}$ are given by

$$\overline{\boldsymbol{\alpha}} \;=\; \frac{1}{D - \Delta\kappa - 2\kappa_1} \left\{ 4\kappa_m^2 - \kappa_g + D\left(jk_o - \kappa_1 \right) + \kappa_1^2 \right\} \hat{\mathbf{t}}_1 \hat{\mathbf{t}}_1$$

$$+\frac{1}{D-\Delta\kappa-2\kappa_2}\left\{4\kappa_m^2-\kappa_g+D\left(jk_o-\kappa_2\right)+\kappa_2^2\right\}\hat{\mathbf{t}}_2\hat{\mathbf{t}}_2 \tag{6.23}$$

$$\overline{\boldsymbol{\beta}} = \frac{1}{D-\Delta\kappa-2\kappa_1}\hat{\mathbf{t}}_1\hat{\mathbf{t}}_1+\frac{1}{D-\Delta\kappa-2\kappa_2}\hat{\mathbf{t}}_2\hat{\mathbf{t}}_2 \tag{6.24}$$

$$\begin{aligned}\overline{\boldsymbol{\gamma}} &= \frac{1}{jk_o(D-\Delta\kappa-2\kappa_1)}\left(jk_o+3\kappa_m-\frac{\kappa_g}{\kappa_m}-2\kappa_1\right)\hat{\mathbf{t}}_1\hat{\mathbf{t}}_1 \\ &\quad+\frac{1}{jk_o(D-\Delta\kappa-2\kappa_2)}\left(jk_o+3\kappa_m-\frac{\kappa_g}{\kappa_m}-2\kappa_2\right)\hat{\mathbf{t}}_2\hat{\mathbf{t}}_2\end{aligned} \tag{6.25}$$

Substituting the second-order absorbing boundary condition in the surface integral given in (6.22), we have

$$\begin{aligned}\int_{S_o}\mathbf{E}\cdot P_2(\mathbf{E})\,dS &= \int_{S_o}\mathbf{E}\cdot\left(\overline{\boldsymbol{\alpha}}\cdot\mathbf{E}_t\right)\,dS+\int_{S_o}\mathbf{E}\cdot\left(\overline{\boldsymbol{\beta}}\cdot\left\{\hat{\mathbf{n}}\left(\nabla\times\mathbf{E}\right)_n\right\}\right)\,dS \\ &\quad+\int_{S_o}\mathbf{E}\cdot\left\{\overline{\boldsymbol{\gamma}}\cdot\nabla_t\left(\nabla\cdot\mathbf{E}_t\right)\right\}\,dS \\ &= I_1+I_2+I_3\end{aligned}$$

Let us examine the integral I_1. Since $\mathbf{E}_t=-\hat{\mathbf{n}}\times\hat{\mathbf{n}}\times\mathbf{E}$, we have

$$I_1 = \int_{S_o}\alpha_1E_{t_1}^2+\alpha_2E_{t_2}^2\,dS \tag{6.26}$$

after employing some simple vector identities.

The other two integrals (I_2 and I_3) do not reduce as easily to simple, implementable forms. They are first simplified using basic vector and tensor identities and then the divergence theorem is employed to eliminate one of the terms. Considering the integrand of the second integral I_2, we note that

$$\mathbf{E}\cdot\left[\overline{\boldsymbol{\beta}}\cdot\left(\nabla\times\hat{\mathbf{n}}\phi\right)\right] = \left(\overline{\boldsymbol{\beta}}\cdot\mathbf{E}\right)\cdot\left(\nabla\times\hat{\mathbf{n}}\phi\right)$$

where we have set $\phi=\left(\nabla\times\mathbf{E}\right)_n$. Using some additional vector identities and letting $\overline{\boldsymbol{\beta}}\cdot\mathbf{E}=\boldsymbol{F}$, we get

$$\begin{aligned}\boldsymbol{F}\cdot\nabla\times\hat{\mathbf{n}}\phi &= \nabla\cdot\left(\phi\hat{\mathbf{n}}\times\boldsymbol{F}\right)+\phi\hat{\mathbf{n}}\cdot\nabla\times\boldsymbol{F} \\ &= \nabla\cdot\left(\phi\hat{\mathbf{n}}\times\boldsymbol{F}\right)+\phi\left(\nabla\times\boldsymbol{F}\right)_n\end{aligned}$$

Using the results from [63], the first term in the above identity can be further simplified to read

$$\begin{aligned}\nabla\cdot(\phi\hat{\mathbf{n}}\times\boldsymbol{F}) &= \nabla_s\cdot(\phi\hat{\mathbf{n}}\times\boldsymbol{F})+\frac{\partial}{\partial n}\{\phi\hat{\mathbf{n}}\cdot(\hat{\mathbf{n}}\times\boldsymbol{F})\}-J\{\phi\hat{\mathbf{n}}\cdot(\hat{\mathbf{n}}\times\boldsymbol{F})\}\\ &= \nabla_s\cdot(\phi\hat{\mathbf{n}}\times\boldsymbol{F})\end{aligned} \tag{6.27}$$

where ∇_s denotes the surface gradient operator and $J=\kappa_1+\kappa_2$. The integral I_2 can now be written as

$$I_2 = \int_{S_o}\nabla_s\cdot(\phi\hat{\mathbf{n}}\times\boldsymbol{F})\;dS+\int_{S_o}\phi(\nabla\times\boldsymbol{F})_n\;dS$$

We can now apply the surface divergence theorem to the first term on the RHS of the this expression to yield

$$\int_{S_o}\nabla_s\cdot(\phi\hat{\mathbf{n}}\times\boldsymbol{F})\;dS = \int_C\phi\hat{\mathbf{m}}\cdot(\hat{\mathbf{n}}\times\boldsymbol{F})\;dl=0 \tag{6.28}$$

since the surface S_o is closed. We note that $\hat{\mathbf{m}}=\hat{\mathbf{l}}\times\hat{\mathbf{n}}$ and $\hat{\mathbf{l}}$ is the unit vector along the edge of the surface element and C denotes the contour of integration. On the basis of (6.28), I_2 reduces to

$$I_2 = \int_{S_o}(\nabla\times\mathbf{E})_n\left\{\nabla\times(\overline{\boldsymbol{\beta}}\cdot\mathbf{E})\right\}_n\;dS \tag{6.29}$$

We now turn our attention to simplifying I_3 for implementation in the finite element equations. Considering the integrand of I_3, we have

$$\begin{aligned}\mathbf{E}\cdot\{\overline{\boldsymbol{\gamma}}\cdot\nabla_t(\nabla\cdot\mathbf{E}_t)\} &= (\overline{\boldsymbol{\gamma}}\cdot\mathbf{E})\cdot\{\nabla_t(\nabla\cdot\mathbf{E}_t)\}\\ &= (\overline{\boldsymbol{\gamma}}\cdot\mathbf{E})\cdot\left\{\nabla\psi-\hat{\mathbf{n}}\frac{\partial\psi}{\partial n}\right\}\end{aligned}$$

where $\psi=\nabla\cdot\mathbf{E}_t$. Next, setting $\boldsymbol{G}=\overline{\boldsymbol{\gamma}}\cdot\mathbf{E}$, we obtain

$$\boldsymbol{G}\cdot\left\{\nabla\psi-\hat{\mathbf{n}}\frac{\partial\psi}{\partial n}\right\}=\nabla\cdot(\psi\boldsymbol{G})-\psi\nabla\cdot\boldsymbol{G}-G_n\frac{\partial\psi}{\partial n} \tag{6.30}$$

The first term in the above identity can be written as

$$\nabla \cdot (\psi \boldsymbol{G}) \;=\; \nabla_s \cdot (\psi \boldsymbol{G}) + \frac{\partial}{\partial n}(\psi G_n) - J\,(\psi G_n)$$

and since $\partial G_n/\partial n = \nabla \cdot \boldsymbol{G} - \nabla \cdot \boldsymbol{G}_t + JG_n$, the LHS of (6.30) reduces to

$$\boldsymbol{G} \cdot \left\{ \nabla\psi - \hat{\mathbf{n}} \frac{\partial \psi}{\partial n} \right\} \;=\; \nabla_s \cdot (\psi \boldsymbol{G}) - \psi \nabla \cdot \boldsymbol{G}_t \tag{6.31}$$

We now replace the integrand of I_3 with the expression in (6.31) and use the divergence theorem to eliminate the first term of (6.31). Specifically,

$$\begin{aligned} I_3 \;&=\; \int_{S_o} \nabla_s \cdot (\psi \boldsymbol{G})\, dS - \int_{S_o} \psi \nabla \cdot \boldsymbol{G}_t\, dS \\ &=\; \int_C \hat{\mathbf{m}} \cdot (\psi \boldsymbol{G})\, dS - \int_{S_o} \psi \nabla \cdot \boldsymbol{G}_t\, dS = -\int_{S_o} \psi \nabla \cdot \boldsymbol{G}_t\, dS \end{aligned}$$

where $\hat{\mathbf{m}}$ has been defined earlier and the contour integral vanishes since the surface is closed. The integral I_3 can finally be rewritten as

$$I_3 \;=\; -\int_{S_o} (\nabla \cdot \mathbf{E}_t)\,(\nabla \cdot \boldsymbol{G}_t)\; dS \tag{6.32}$$

Using (6.26), (6.29) and (6.32), the complete surface integral term incorporating the conformal second order ABC reduces to

$$\begin{aligned} \int_{S_o} \mathbf{E} \cdot P_2(\mathbf{E})\, dS \;&=\; \int_{S_o} \left(\alpha_1 E_{t_1}^2 + \alpha_2 E_{t_2}^2 \right)\, dS + \int_{S_o} (\nabla \times \mathbf{E})_n \left\{ \nabla \times (\overline{\boldsymbol{\beta}} \cdot \mathbf{E}) \right\}_n\, dS \\ &\quad - \int_{S_o} (\nabla \cdot \mathbf{E}_t)\, \{ \nabla \cdot (\overline{\gamma} \cdot \mathbf{E})_t \}\; dS \end{aligned} \tag{6.33}$$

It remains to be seen whether the integrals in (6.33) lead to a symmetric system when incorporated into the finite element equations. With this in mind, we will examine three simple shapes and check whether they preserve symmetry of the finite element system. It will then be possible to generalize our findings to a more general mesh truncation boundary.

Let us consider the case of a sphere of radius r. Since the two principal curvatures of the sphere are identical ($\kappa_1 = \kappa_2 = 1/r$), the first order boundary condition reduces to the simple Sommerfeld radiation condition

$$\int_{S_o} \mathbf{E} \cdot P_1(\mathbf{E})\, dS \;=\; jk_o \int_{S_o} \left(E_\theta^2 + E_\phi^2\right)\, dS \tag{6.34}$$

On a spherical boundary, the second order ABC also reduces to the comparatively simple form:

$$\int_{S_o} \mathbf{E} \cdot P_2(\mathbf{E})\, dS \;=\; \int_{S_o} \left[jk_o \mathbf{E}_t^2 + \frac{1}{2jk_o + 2/r} (\nabla \times \mathbf{E})_n^2 - \frac{1}{2jk_o + 2/r} (\nabla \cdot \mathbf{E}_t)^2 \right] dS \tag{6.35}$$

The ABC given in (6.35) is identical to the boundary condition derived in [48] for a spherical mesh termination surface and leads to a symmetric system of equations.

Next, we consider the case of a planar termination boundary in which case $\kappa_1 = \kappa_2 = 0$. The first order ABC then reduces to the Sommerfeld radiation condition and the second order ABC for a planar boundary simplifies to

$$\int_{S_o} \mathbf{E} \cdot P_2(\mathbf{E})\, dS \;=\; \int_{S_o} \left[jk_o \mathbf{E}_t^2 + \frac{1}{2jk_o} (\nabla \times \mathbf{E})_n^2 - \frac{1}{2jk_o} (\nabla \cdot \mathbf{E}_t)^2 \right] dS \tag{6.36}$$

Since the planar boundary is a special case of a spherical boundary, (6.36) again reduces to asymmetric system of equations.

Now we examine the situation when the mesh termination boundary is cylindrical in shape and of radius ρ. The principal curvatures of the cylindrical surface are then $\kappa_1 = 1/\rho$ and $\kappa_2 = 0$. Since the principal curvatures are no longer identical, the tensors $\overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\beta}}$ and $\overline{\boldsymbol{\gamma}}$ do not reduce to simple scalars. The first order ABC for a cylindrical outer boundary is given by

$$\int_{S_o} \mathbf{E} \cdot P_1(\mathbf{E})\, dS \;=\; \left(jk_o + \frac{1}{\rho}\right) \int_{S_o} \left(jk_o + \frac{1}{\rho}\right) \mathbf{E}_t^2\, dS - \int_{S_o} \frac{1}{\rho} E_\phi^2\, dS \tag{6.37}$$

and the second order ABC gives by

$$\int_{S_o} \mathbf{E} \cdot P_2(\mathbf{E})\, dS = \int_{S_o} \left(\alpha_{c1} E_\phi^2 + \alpha_{c2} E_z^2\right)\, dS + \int_{S_o} (\nabla \times \mathbf{E})_\rho \left\{\nabla \times (\overline{\boldsymbol{\beta}}_c \cdot \mathbf{E})\right\}_\rho\, dS - \int_{S_o} (\nabla \cdot \mathbf{E}_t) \left\{\nabla \cdot (\overline{\boldsymbol{\gamma}}_c \cdot \mathbf{E})_t\right\}\, dS \tag{6.38}$$

where $\alpha_{c1}, \alpha_{c2}, \overline{\boldsymbol{\beta}}_c$ and $\overline{\boldsymbol{\gamma}}_c$ are obtained by substituting $\kappa_1 = 1/\rho$ and $\kappa_2 = 0$ in the original expressions for $\overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\beta}}$ and $\overline{\boldsymbol{\gamma}}$. It is seen that the first order ABC given by (6.37) leads to a symmetric matrix for a cylindrical boundary. On the other hand, the second order ABC does not yield a symmetric matrix for an arbitrary choice of basis functions. However, the boundary condition outlined in (6.38) preserves symmetry on using linear edge-based elements for discretization.

The above discussion enables us to conclude that the first order boundary condition leads to a symmetric system for surfaces having arbitrary principal curvatures. However, symmetry is guaranteed for the second order ABC only when the two principal curvatures of the mesh termination boundary are identical, i.e., only when the outer boundary is limited to a planar or a spherical surface. Thus if we want to enclose a scatterer having arbitrary shape within a conformal outer boundary, an unsymmetric system of equations will have to be solved. It should, however, be noted that the resulting unsymmetric system will, in general, have a lesser number of unknowns than its symmetric counterpart.

6.2 Applications

In the previous section, we have discussed the derivation of absorbing boundary conditions which can be employed on surfaces conformal to the scattering or radiating structure. As a result, the mesh termination boundary can be made to enclose the scattering object more snugly. Consequently for arbitrary targets, we achieve a

substantial saving in the amount of volume to be meshed between the ABC surface and that of the scatterer. This is particularly critical when the target is cylindrical in shape or a combination of cylindrical, doubly curved and planar surfaces as is the case with any real-life structure.

In this section, we examine the performance of these boundary conditions when applied on conformal mesh termination surfaces.

A. Composite cube

For our first example, we compute the backscatter pattern of the half metal-half

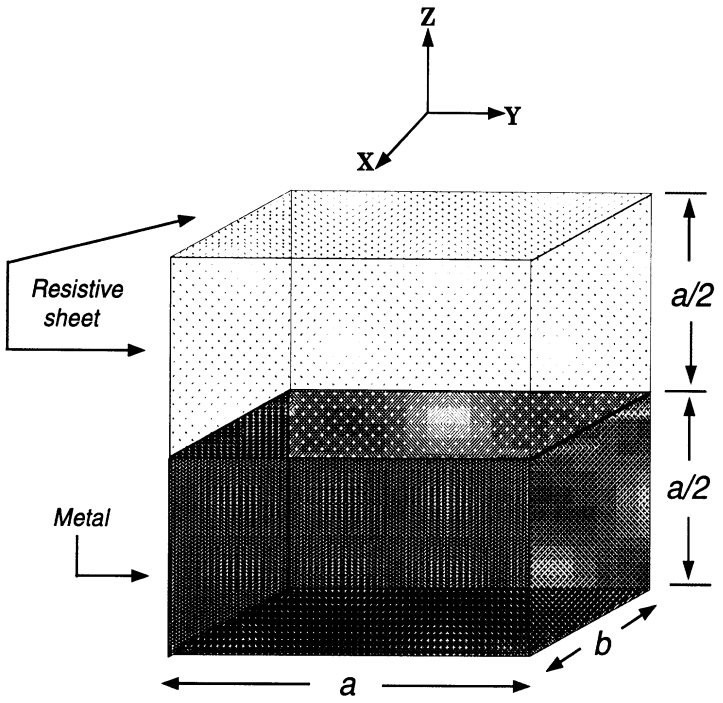

Figure 6.2: Geometry of cube ($a = b = 0.5\lambda$) consisting of a metallic section and a dielectric section ($\epsilon_r = 2 - j2$), where the latter is bounded by a resistive surface having $R = Z_o$.

dielectric cube geometry shown in Chapter 3. However, instead of using a spherical surface to terminate the mesh, we employ the absorbing boundary condition on a piecewise planar surface, i.e., a cubical box placed only 0.3λ from the face of the scatterer. The geometry is shown in Figure 6.2 and needed only 30,000 unknowns

for discretization. This is in stark contrast to the 40,000 unknown system which resulted when the geometry was enclosed in a spherical termination boundary. The decrease in the unknown count is even more dramatic as we go to larger scatterers. In Figure 6.3, we plot the backscatter pattern in the $x-z$ plane ($E_z^{inc} = 0$ polar-

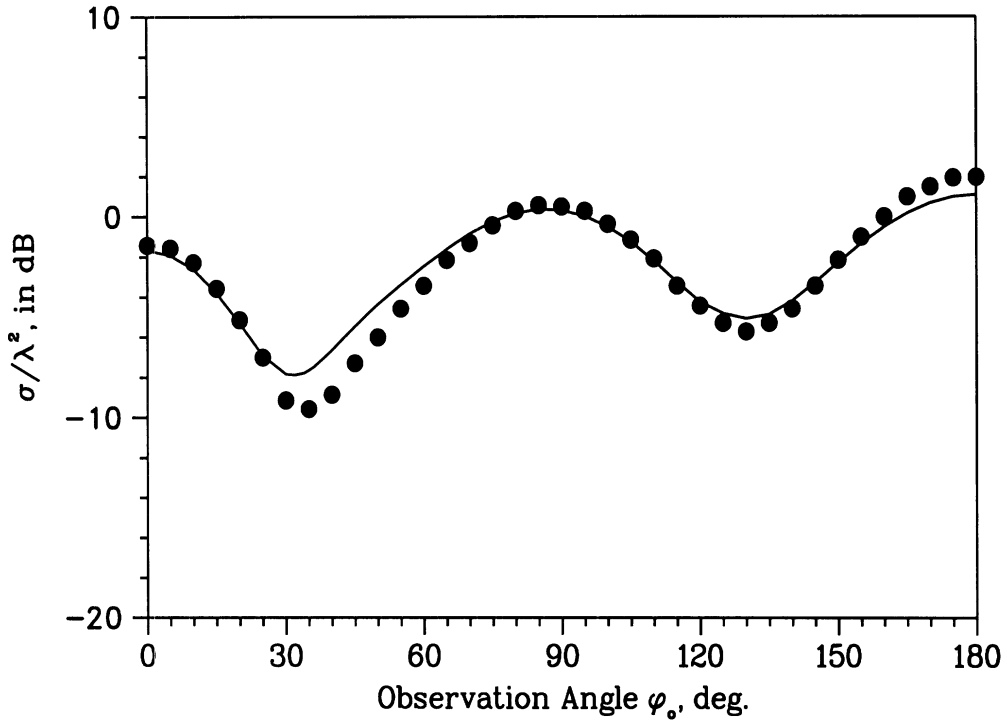

Figure 6.3: RCS pattern in the $x-z$ plane for the composite cube shown earlier. The solid curve is the FEMATS pattern and the black dots are MoM data for the $E_z^{inc} = 0$ polarization. Mesh termination is piecewise planar.

ization) for the metal-dielectric cube geometry given in Figure 6.2 and compare the computed values with data obtained from a traditional method of moments (MoM) code [52]. The dielectric-filled section has unit permeability and a relative permittivity of $2 - j2$. The agreement with reference data is seen to be excellent; it can therefore be concluded that accuracy of the far-field values has not been affected by a different mesh termination scheme. In fact, we have obtained results of comparable accuracy with only 75% of the computing resources than were necessary before. This observation will be made by the reader again and again in the following pages as the

full capability of these conformal ABCs is demonstrated.

B. Inlets

In our next example, we compute the scattering from perfectly conducting inlets. The aperture of an inlet usually has a large radar cross-section around normal incidence: therefore, a good understanding of its scattering characteristics is critical if measures need to be taken for reducing its echo-area. An accurate computer simulation of such a geometry provides a cost-effective and ready way of allowing the designer to experiment with complex material fillings to achieve satisfactory results. All our validations are carried out for empty inlets due to lack of reference data for more complicated structures.

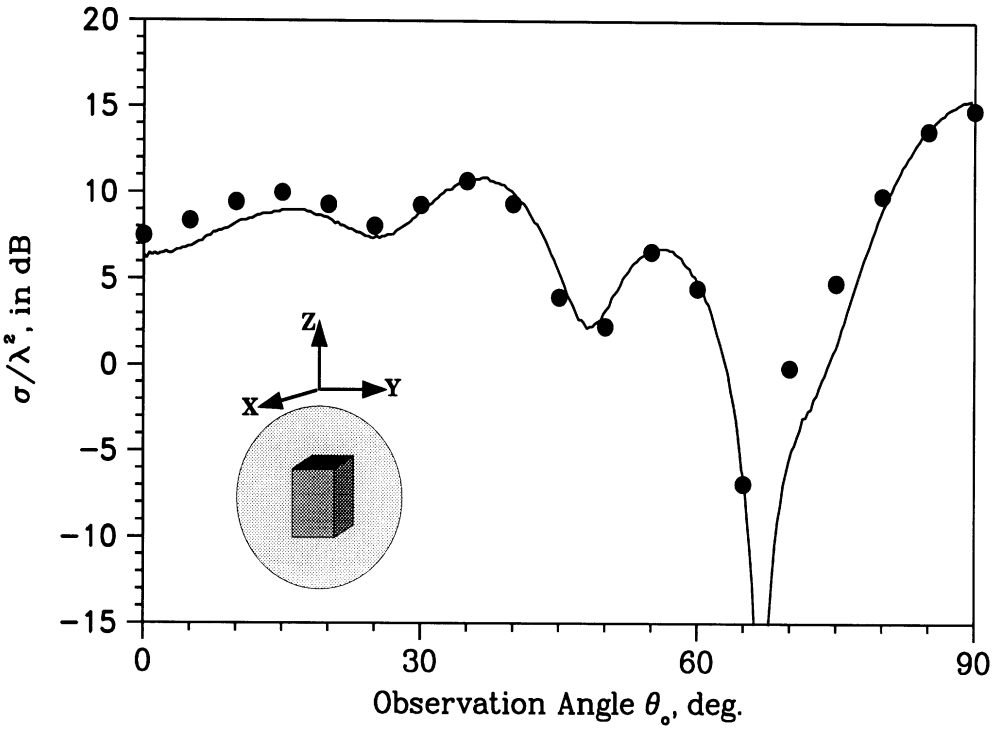

Figure 6.4: Backscatter pattern of a metallic rectangular inlet ($1\lambda \times 1\lambda \times 1.5\lambda$) for HH polarization. Black dots indicate computed values and the solid line represents measured data [1]. Mesh termination surface is spherical.

The geometry of interest is a perfectly conducting rectangular inlet, with dimensions $1\lambda \times 1\lambda \times 1.5\lambda$. For the plots shown in Figures 6.4 and 6.5, we have enclosed

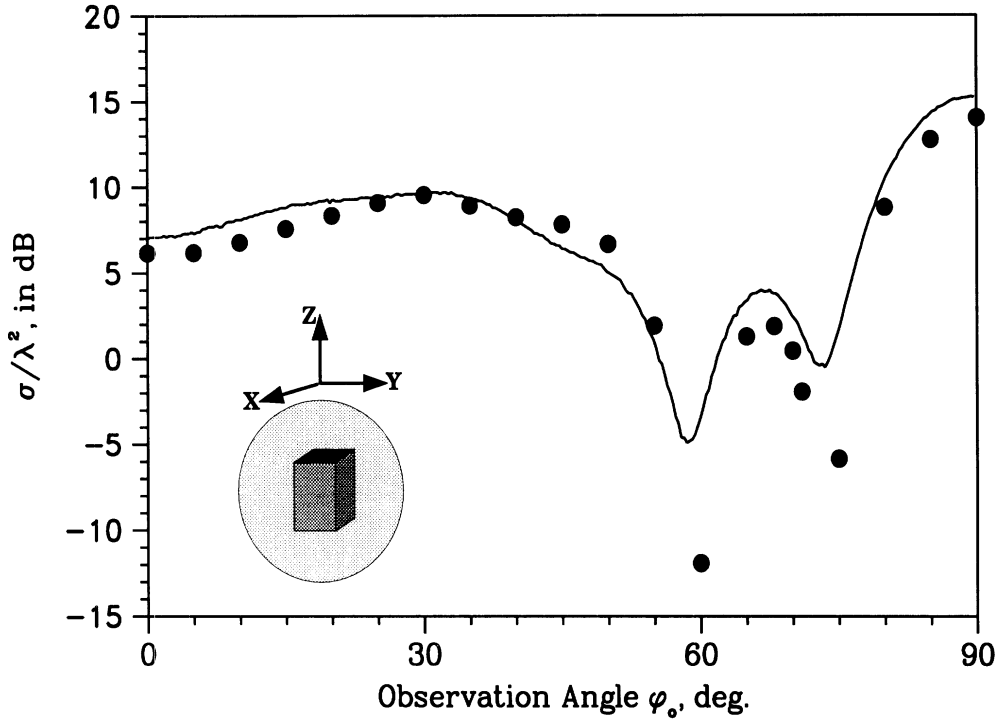

Figure 6.5: Backscatter pattern of a metallic rectangular inlet ($1\lambda \times 1\lambda \times 1.5\lambda$) for VV polarization. Black dots indicate computed values and the solid line represents measured data [1]. Mesh termination surface is spherical.

the target within a sphere of radius 1.35λ, which is only about $.35\lambda$ from the farthest edge of the scatterer. This resulted in a system of 224,476 unknowns and converged in an average of 785 seconds per incidence angle on the 56 processor KSR1. The computed values from our code agrees very well with measured data for both HH and VV polarizations. As can be seen from the above discussion, we have obtained our solution using significant computing resources and time.

Our next step is to use the conformal mesh termination scheme formulated in the previous section and utilized in Example A. Therefore, instead of using a spherical mesh truncation surface, we terminate the mesh with a rectangular box placed only 0.35λ away from the scatterer (see inset of Figure 6.6). The problem size reduces dramatically to 145,000 unknowns, a 35% reduction over the spherical mesh termination scheme. The convergence time for each excitation vector is about 220 seconds, less

than 4 minutes, when run on all 56 processors of the KSR1. The computed values are again compared with measured data for both polarizations in Figures 6.6 and 6.7; the agreement is excellent, albeit a bit worse than the spherical case. However, this fact is overshadowed by the fact that we have reduced the problem size by more than a third and computing time by about a fourth.

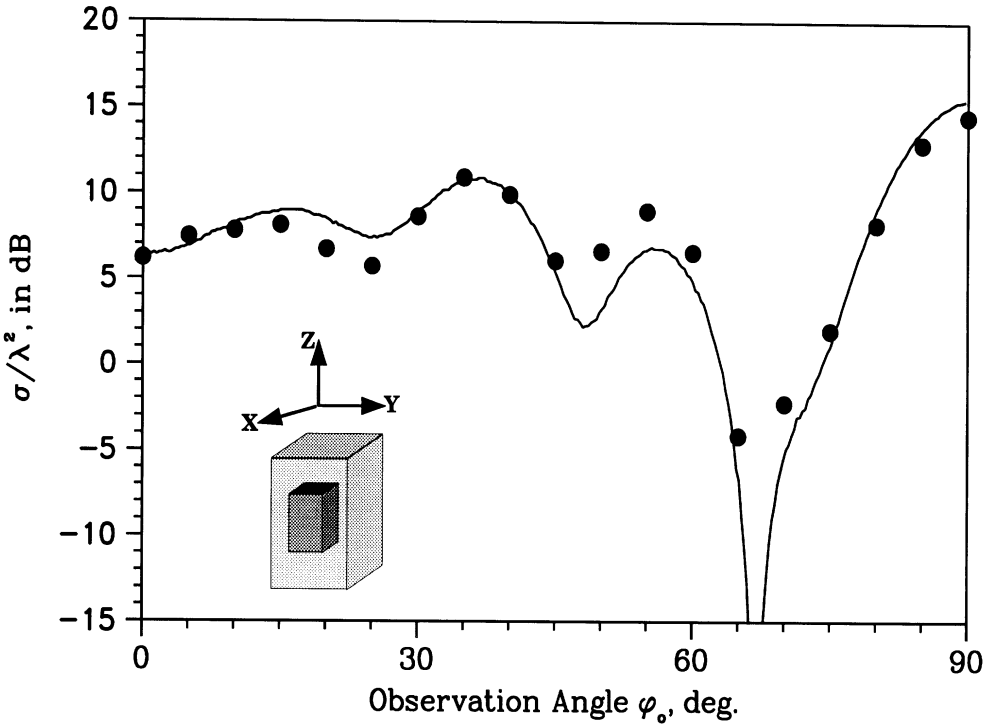

Figure 6.6: Backscatter pattern of a metallic rectangular inlet ($1\lambda \times 1\lambda \times 1.5\lambda$) for HH polarization. Black dots indicate computed values and the solid line represents measured data [1]. Mesh termination surface is piecewise planar.

We then considered the problem of scattering from a perfectly conducting cylindrical inlet. Even though integral equation codes are more efficient for such bodies of revolution, our primary concern in this test was to examine the performance of the conformal absorbing boundary conditions that we derived earlier. The target is a perfectly conducting cylindrical inlet having a diameter of 1.25λ and a height of 1.875λ. We first used a rectangular outer boundary, placed $.45\lambda$ from the farthest

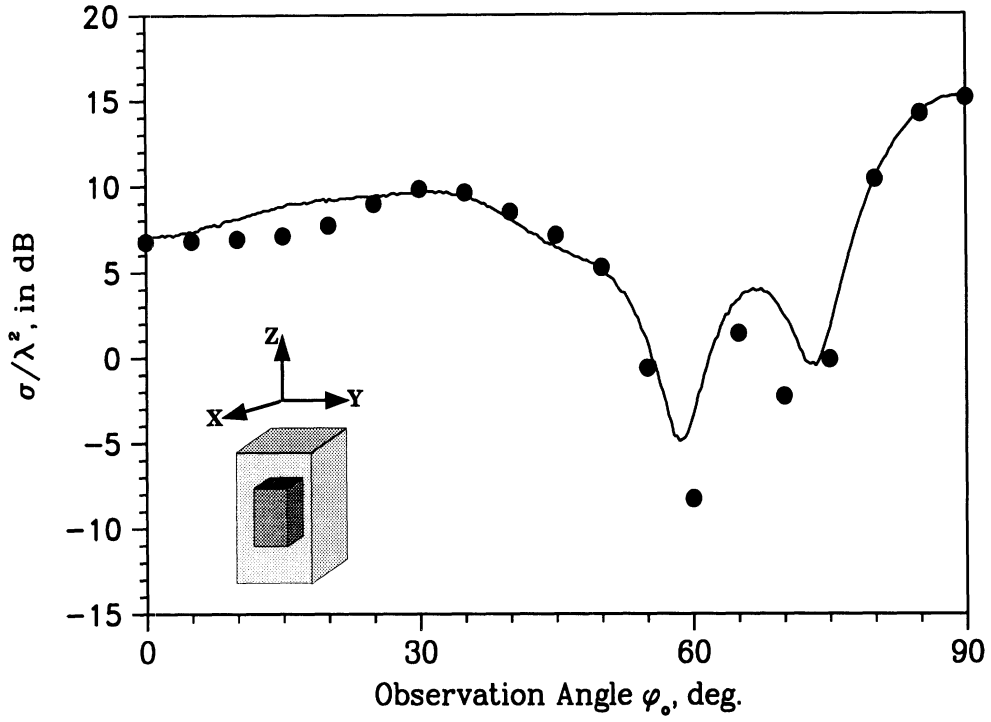

Figure 6.7: Backscatter pattern of a metallic rectangular inlet ($1\lambda \times 1\lambda \times 1.5\lambda$) for VV polarization. Black dots indicate computed values and the solid line represents measured data [1]. Mesh termination surface is piecewise planar.

edge of the target, to enclose the scatterer. The radar cross-section was then computed for a ϕ-polarized incident wave in the yz-plane and compared with measured data (Figure 6.8). The agreement was found to be quite good for all lobes except the third. We expect the results to improve on moving the outer boundary farther away. The backscatter echo-area computed for the same geometry by Shankar [68] using the finite difference-time domain method with the absorbing boundary placed a few wavelengths from the scattering structure, agrees with the computed results remarkably well for all incidence angles.

Next, we used a truly conformal termination scheme by using a cylindrical surface for mesh truncation. It should be noted that this is the first instance of a non-spherical surface (i.e., a surface having different principal curvatures) being ap-

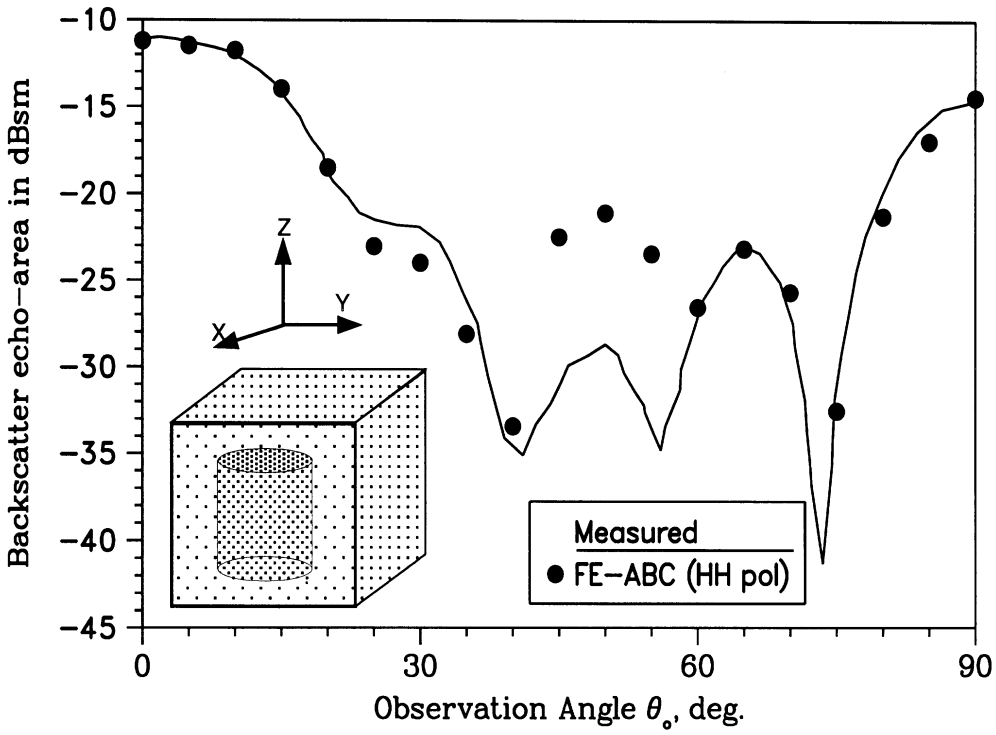

Figure 6.8: Backscatter pattern of a perfectly conducting cylindrical inlet (diameter= 1.25λ, height=1.875λ) for HH polarization. The solid line indicates measured data [69] and the black dots indicate computed data. Mesh termination surface is a rectangular box

plied to terminate a finite element mesh for solving open problems. The cylindrical outer boundary was placed about 0.45λ from the target and RCS computations were carried out for a ϕ polarized incident wave and compared with measured data [69] and with a body of revolution code [2] (Figure 6.9). As can be observed from Figures 6.8 and 6.9, the far-field results for a cylindrical termination and a rectangular termination do not differ significantly. However, the savings in computational cost is quite impressive. The cylindrical mesh termination has only 144,392 unknowns compared to the 191,788 unknowns for a rectangular truncation scheme. A spherical mesh termination would have swelled to about 265,000 unknowns, sampling density and outer boundary distance remaining the same. Thus we have reduced the problem size by about 45% and computation time by a similar, if not greater, amount by us-

ing a conformal mesh termination scheme. The savings in computational resources is quite significant even when we compare the rectangular and cylindrical termination schemes - a 25% reduction in problem size and a similar decrease in computation time. In Figure 6.10, we plot the backscatter pattern for the same cylindrical inlet

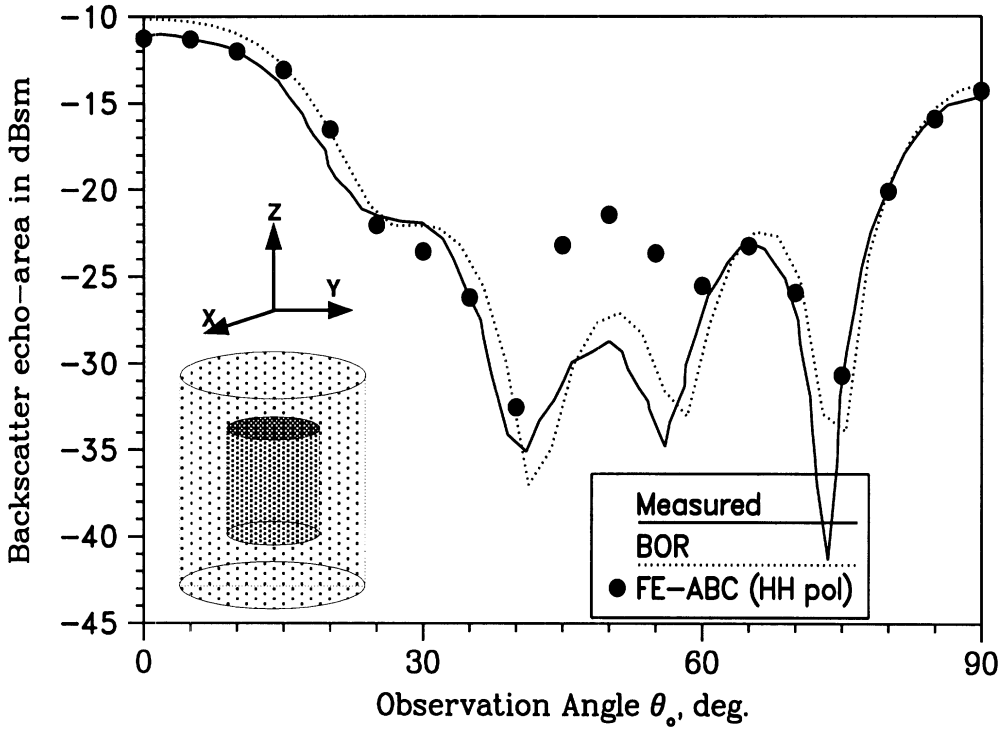

Figure 6.9: Backscatter pattern of a perfectly conducting cylindrical inlet (diameter= 1.25λ, height=1.875λ) for HH polarization. Black dots indicate computed values, the solid line represents measured data [69] and the dotted line is body of revolution data [2]. Mesh termination surface is a circular cylinder.

in which the incident wave is θ polarized. The agreement is seen to be decent for the entire range of incident angles.

C. Lossy foam cylinder with embedded wires

The next geometry to be considered was a lossy foam ($\epsilon_r = 1.05 - j0.2$) cylinder having a radius of 1λ and a height of 3.5λ with 0.5λ long perfectly conducting wires embedded in it. The wires are spaced 0.5λ apart from each other and have a diameter of 0.01λ. This is an interesting problem for two reasons: first, the orientation of the

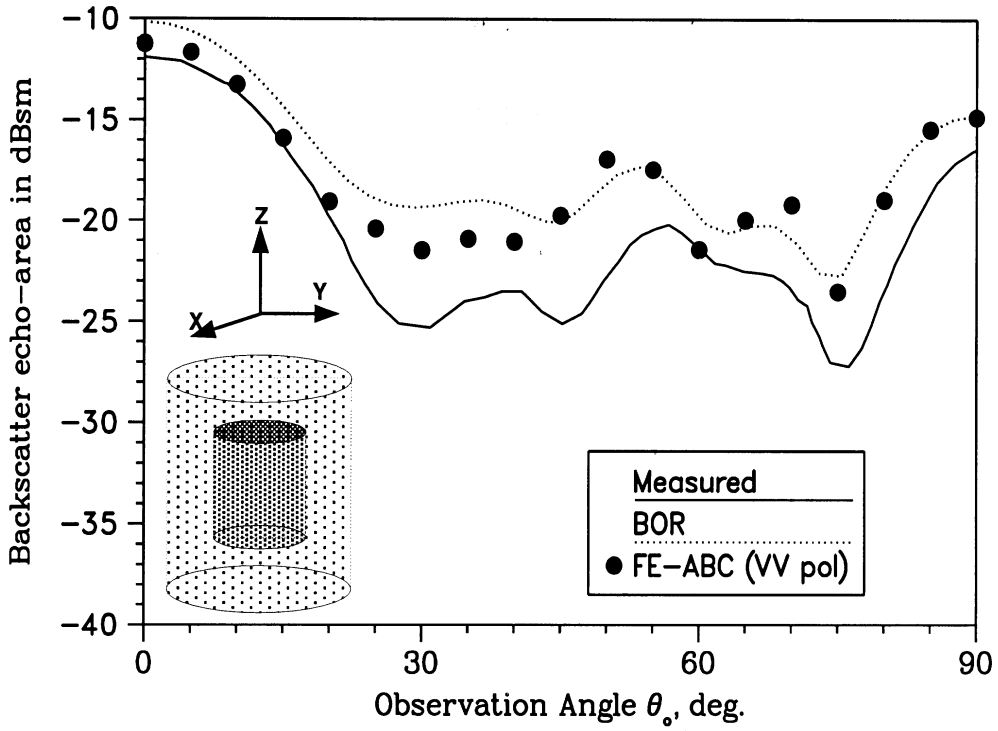

Figure 6.10: Backscatter pattern of a perfectly conducting cylindrical inlet (diameter= 1.25λ, height=1.875λ) for VV polarization. Black dots indicate computed values, the solid line is measured data [69] and the dotted line represents reference data from a body of revolution code [2]. Mesh termination surface is a circular cylinder.

embedded wires makes it a difficult geometry for other methodologies to handle; second, the problem is very large since the cylinder has a volume of $11\lambda^3$ and a surface area of $28.27\lambda^2$.

As a first case, we consider the wires to be aligned along the axis of the cylinder (the Z axis in this case) and compute the resulting backscatter pattern in the (Figure 6.11). The cylindrical mesh termination boundary was placed 0.45λ from the flat and curved surfaces of the foam cylinder. The resulting system of equations had 437,064 unknowns but needed only an average of 3050 iterations to converge to the desired answer. Thus each angle of incidence took a little more than 12.5 minutes to compute on a 56-processor KSR1 massively parallel architecture. The impressive convergence rate is due to the low contrast and the loss in the dielectric medium as

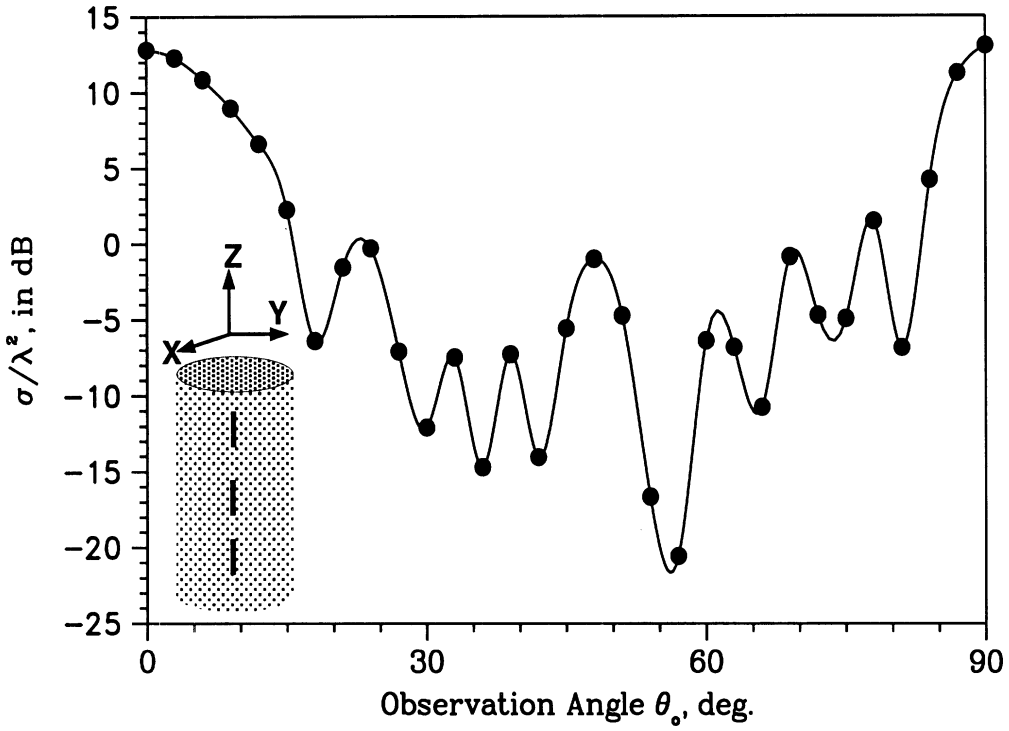

Figure 6.11: Backscatter pattern of a lossy foam cylinder with three perfectly conducting wires embedded along the axis. The incident electric field is oriented parallel to the axis of the cylinder. Mesh termination surface is a circular cylinder.

well as due to the presence of the metallic wires.

In the second case, the middle wire is offset 0.25λ in the negative X-direction from the cylinder axis. The number of unknowns and the time taken for convergence is comparable to the earlier case. The effect of the offset wire on the backscatter pattern can be studied in Figure 6.12. There is a slight asymmetry in the side lobes of the resulting backscatter pattern but, as can be expected, the effect is very small.

D. Perfectly conducting plate

The motivation for testing the FEMATS code on the perfectly conducting plate was two-fold. It is usually very difficult to model the scattering from the edges of the plate even using integral equation methods. Therefore, we wanted to carry out some tests to see how the code would behave at edge-on incidence. Secondly, we wanted

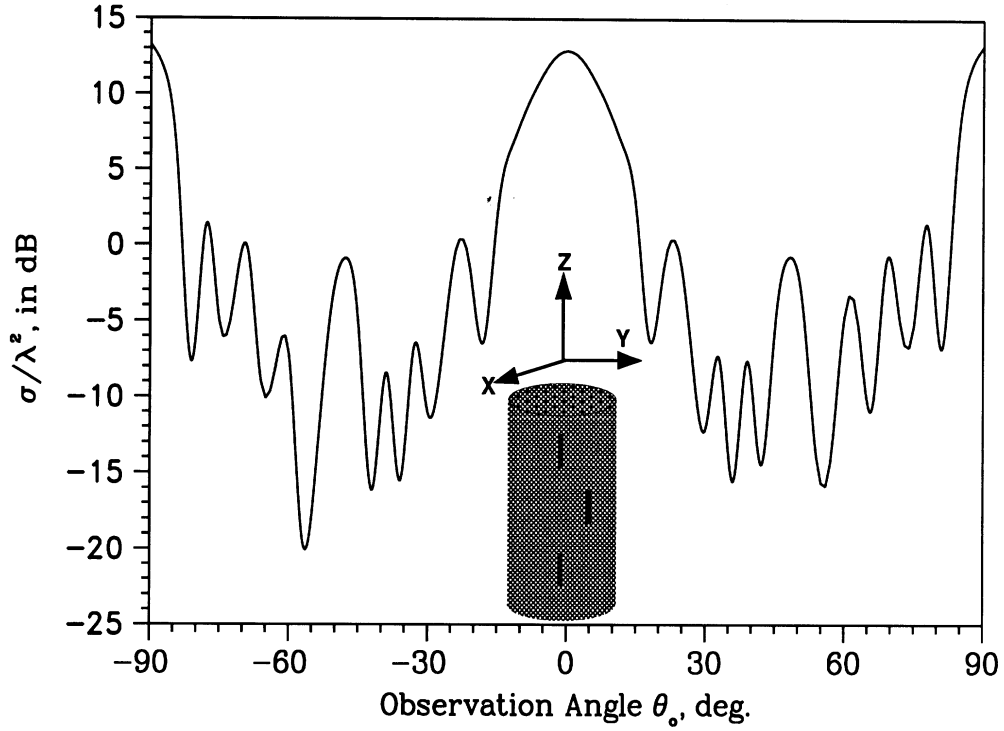

Figure 6.12: Backscatter pattern of a lossy foam cylinder with the middle wire offset 0.25λ from the cylinder axis. The incident electric field is oriented parallel to the axis of the cylinder. Mesh termination surface is a circular cylinder.

to examine the performance of termination boundaries of esoteric shapes. The first choice was to enclose the plate in a rectangular box. The second choice was to use a box with half cylinders attached to the faces normal to the plane of the plate - the reasoning being that since the edge of the plate behaves like a line source and scatters cylindrical waves, a cylindrical mesh termination was most suitable for wave absorption. It should be noted that both mesh termination schemes require approximately the same number of unknowns; the superiority of one over the other was thus decided only on the basis of computed backscatter values.

Our test case is a $3.5\lambda \times 2\lambda$ perfectly conducting rectangular plate. In Figure 6.13, we plot the backscatter pattern for the $\theta\theta$ polarization in the xz plane, i.e., over the long side of the plate. The computed values compare very well with reference

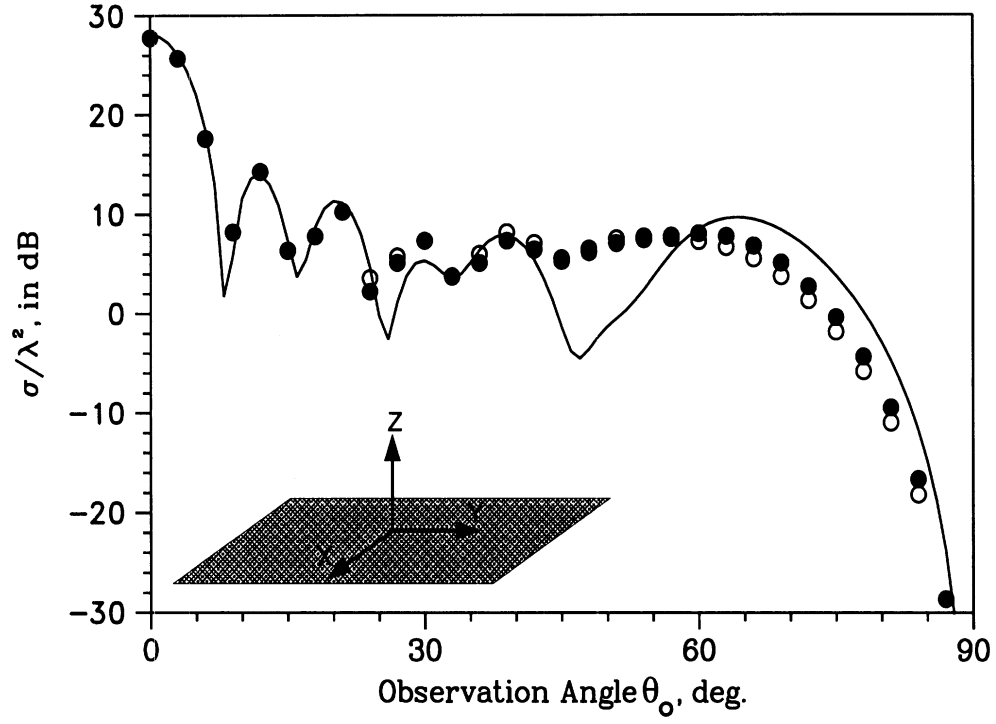

Figure 6.13: Backscatter pattern ($\sigma_{\theta\theta}$) of a $3.5\lambda \times 2\lambda$ perfectly conducting plate in the xz plane. The white dots indicate box termination; the black dots represent a combined box-cylinder termination.

data; however, the code does not pick up the sharp null at $\theta_o = 45^0$ and the two mesh termination schemes perform as well, although a slight improvement is noticeable for the box-cylinder termination.

Next, we plot the backscatter pattern of the same geometry for the $\phi\phi$ polarization over the long side of the plate in Figure 6.14. Again, the agreement with reference data is quite good. However, the backscatter echo-area at edge-on incidence is not calculated accurately.

In the next figure, we compute the RCS of the conducting plate in the yz plane, i.e., over its short side, for the $\phi\phi$ polarization. The backscatter echo-area for edge-on incidence is picked up very well for a rectangular-cylindrical termination whereas a rectangular truncation scheme gives completely incorrect results. These two schemes have approximately the same storage requirement; in fact, the box-cylinder combi-

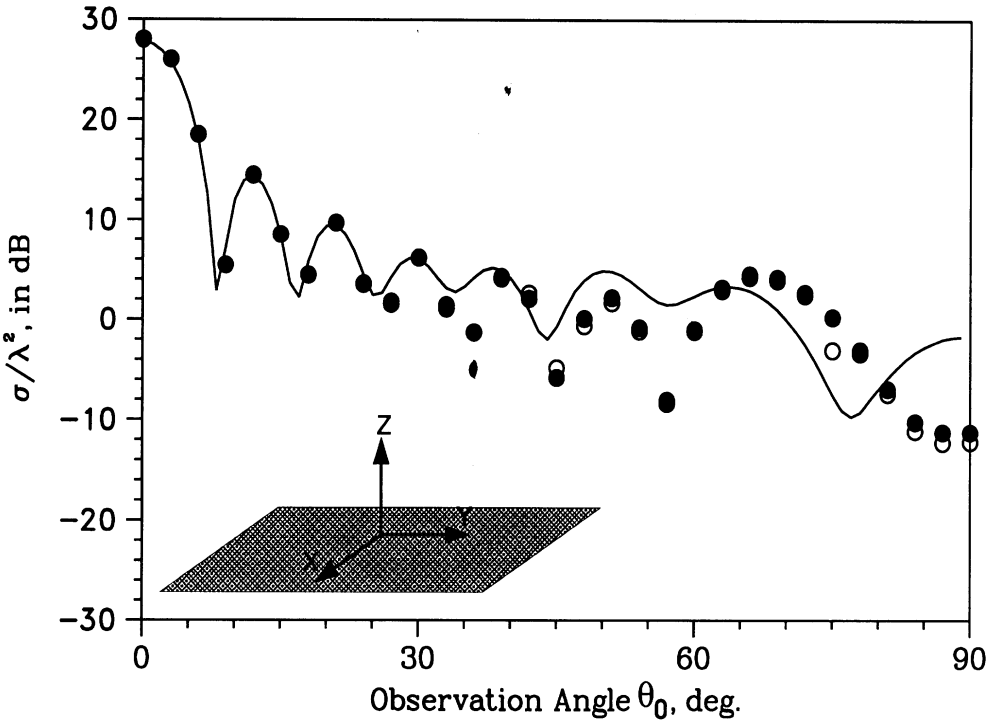

Figure 6.14: Backscatter pattern ($\sigma_{\phi\phi}$) of a $3.5\lambda \times 2\lambda$ perfectly conducting plate in the xz plane. The white dots indicate box termination; the black dots represent a combined box-cylinder termination.

nation yields a slightly smaller system of equation. This example truly illustrates the power of a conformal truncation scheme composed of simple shapes: not only are the results far more accurate but even the storage requirement is slightly less.

In all the above simulations, the boundary was terminated at 0.35λ from the flat face of the plate and 0.5λ from the edges of the plate. In order to test the accuracy of the ABC method as a function of mesh termination distance, we computed the backscatter patterns from the edges of the plate with increasing mesh termination distance. Figure 6.16 shows that the backscatter values from the plate edges slowly take the shape of the reference data as the mesh truncation distance is increased.

E. Glass plate

In the final example, we present the results for one of the most challenging problems solved by the FEMATS method. The target is a simple rectangular glass slab 1.75λ

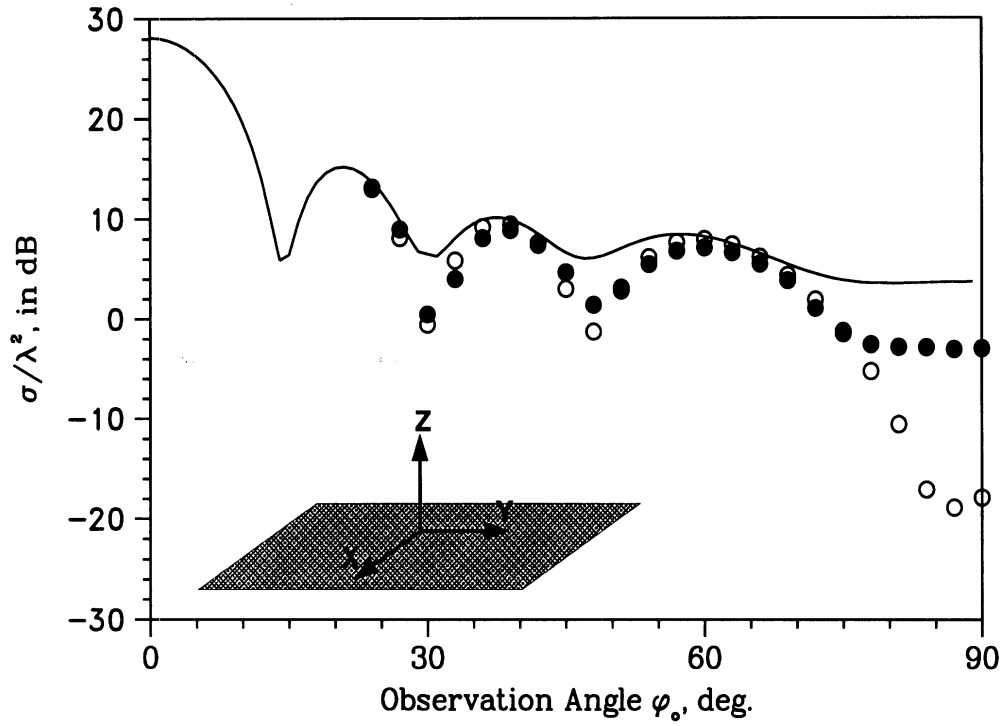

Figure 6.15: Backscatter pattern ($\sigma_{\phi\phi}$) of a $3.5\lambda \times 2\lambda$ perfectly conducting plate in the yz plane. The white dots indicate box termination; the black dots represent a combined box-cylinder termination.

long, 1λ wide and $.125\lambda$ thick. The relative permeability of glass is taken to be $3 - j.09$. The backscatter pattern ($\sigma_{\theta\theta}$) is sought for the $\theta = 80^o$ cut. This is an extremely difficult problem for any numerical method to handle since the incident field is almost edge-on to the dielectric slab, causing the scattered field to have strong higher order components which decay appreciably only at large distances from the scatterer.

In our first attempt, we enclosed the glass plate in a flat box placed $.45\lambda$ from it. Though the computed results were accurate for some angles, they departed significantly from the reference data [68] for other incidences. The results did not show a significant improvement on shifting the outer boundary 0.5λ away from the scatterer while maintaining its shape. The next step was to modify the shape of the outer boundary such that the flat box had half cylinders attached to the faces normal to

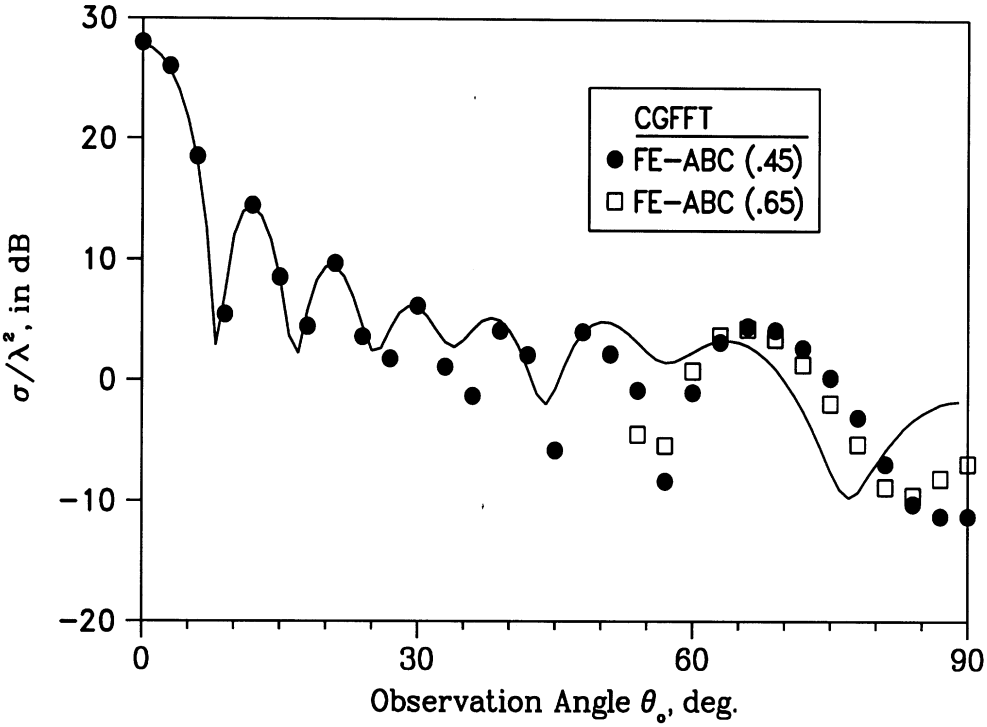

Figure 6.16: Backscatter pattern ($\sigma_{\phi\phi}$) of a $3.5\lambda \times 2\lambda$ perfectly conducting plate in the xz plane. The numbers in the legend indicate mesh termination distance from the plate edges.

the plane of the slab - the reasoning was the same as that for the perfectly conducting rectangular plates mentioned in an earlier section. Further, this termination scheme results in an outer boundary with no sharp edges. The free space between two cylinders with their axes perpendicular to each other is filled with a quarter sphere having the same radius as the cylinders (see Figure 6.17). As can be observed in Figure 6.18, the computed results agree quite closely with the reference data for most of the incident angles. The problem was solved with only 155,000 unknowns and needed an average of 2700 iterations to converge to the specified tolerance. The slow convergence is typical of geometries that are composed of dielectrics since the linear system of equations becomes indefinite due to non-unit values of the relative permittivity.

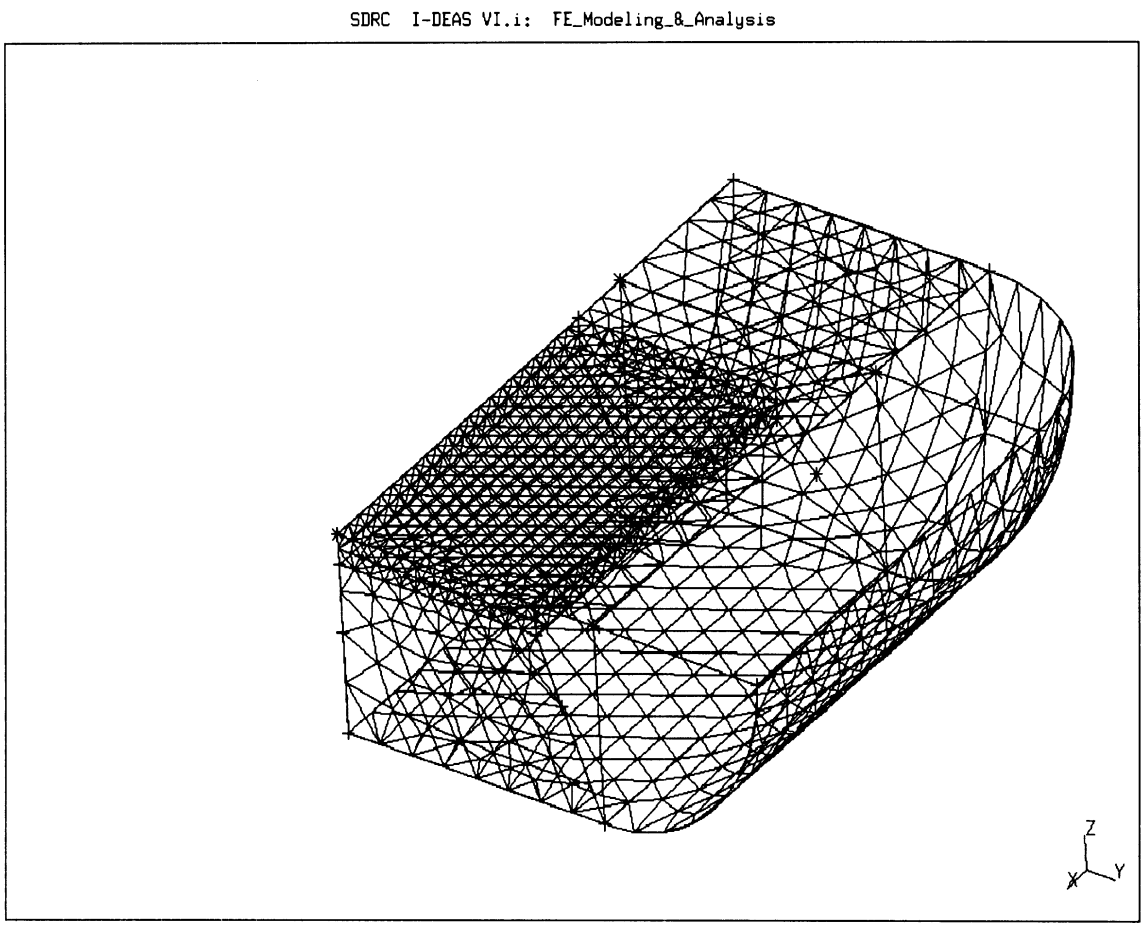

Figure 6.17: An eighth of a glass plate enclosed inside a smooth mesh termination boundary composed of flat planes and cylindrical and spherical sections.

6.3 Conclusion

In the previous section, we have used our FEMATS methodology to compute scattering patterns from large, three dimensional geometries having arbitrary shapes and complicated configurations and with material inhomogeneities. It has been shown that the solution technique does indeed live upto its promise of delivering accurate results with the expenditure of minimal computer resources. Very large problems can be solved in real time with a fraction of the resources that existing numerical electromagnetics codes require.

There are basically two parameters that govern the accuracy of the final result.

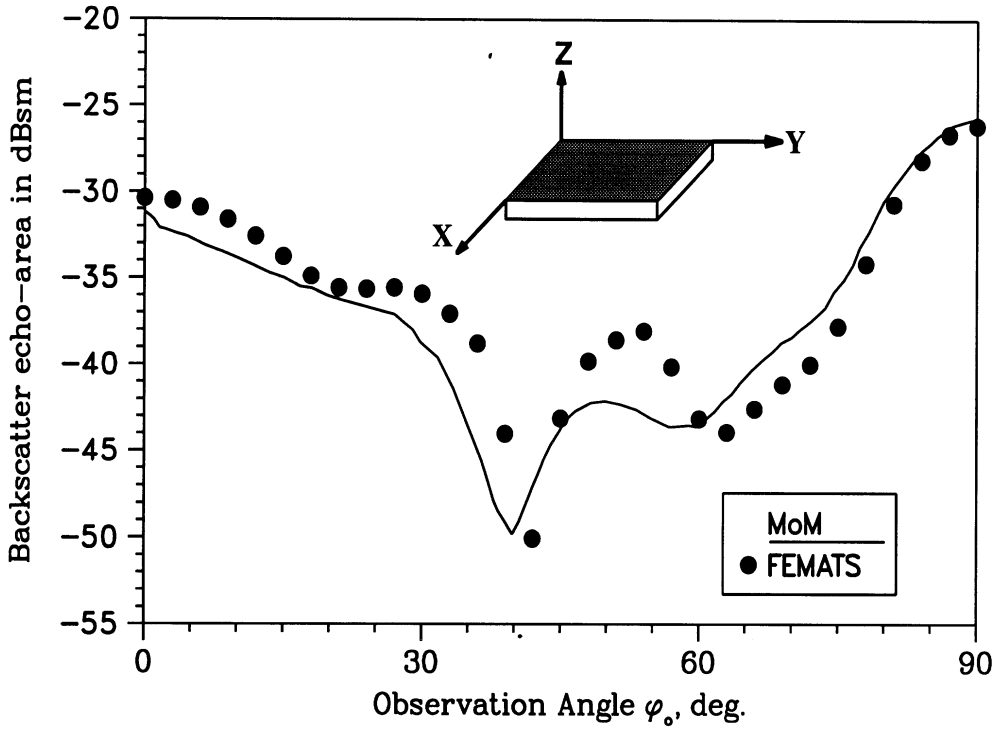

Figure 6.18: Backscatter pattern ($\sigma_{\theta\theta}$) of a glass plate ($1.75\lambda \times 1\lambda \times .125\lambda$) having a relative permittivity of $(3 - j.09)$ for the $\theta = 80^o$ cut.

The first one is the mesh termination technique. Though lower order absorbing boundary conditions are simpler to implement, a significant penalty in computational resources has to be paid for getting the same accuracy as the higher order ABCs. The reliability and the order of the ABC is thus crucial to the computation process. The second parameter is the shape of the mesh termination boundary and its distance from the scatterer. As shown in the previous sections, significantly better results can be obtained by using smooth termination boundaries in place of flat boxes with sharp corners. The distance of the termination boundary from the scattering body is also important in obtaining accurate results. We have determined that a distance of $.45\lambda$ usually gives reliable results for large three dimensional geometries. For small problems, distances of $.3\lambda - .35\lambda$ are sufficient to give accurate far-field values. In order to obtain reliable near-field values such as the input impedance of an antenna,

the mesh must be terminated farther away from the target.

CHAPTER VII

Conclusions

This thesis is one of the first works known to the author to tackle the problem of three dimensional scattering using finite elements and ABCs. It started out as an investigation into the methodology mentioned above, given its extremely attractive features for solving large problems. The $O(N)$ storage requirement and the iterative equation solver are essentially the keys to this technique, since they allow the solution of extremely large, realistic problems with minimal computational overhead. As the technology progresses to higher frequencies and longer electrical dimensions, the superiority of this solution methodology over integral equation or hybrid techniques will become even more apparent.

The opening chapter outlined the motivation in choosing this solution technique over other traditional methods. In Chapter 2, we described the basic concepts of electromagnetics and finite elements. They were by no means complete or exhaustive and the interested reader is referred to several excellent texts for reference [9, 12, 13]. Chapter 3 provided the rationale for employing edge basis functions for discretizing electromagnetic field variables in three dimensional computation. A full review of nodal and edge bases of various orders for two and three dimensions was also carried out. In Chapter 4, the basic formulation of the solution technique was presented along

with code validation for a large number of small and composite geometries. Since the methodology was found to perform extremely well for small complex geometries with inhomogeneities, we considered extending it to compute scattering from very large problems. In order to achieve this goal, the first step was to make the finite element code computationally efficient. Chapter 5 discussed the various optimization techniques that were carried out to make the code as computationally efficient as possible on a wide class of vector and parallel machines. Since the mesh termination condition used till now were valid only for spherical or flat terminations, the next step was to derive more efficient boundary conditions that would yield accurate results even with reduced computational resources. In Chapter 6, new ABCs were derived which are enforceable on mesh termination boundaries conformal to the surface of the target and result in drastic reductions in the number of unknowns and hence solution time. Thus, it can be stated that this thesis has achieved its objective of showing the efficacy and the accuracy of computing three dimensional scattering from large, composite and complex-shaped structures using finite elements in conjunction with the ABC method of mesh termination.

Any research, however, by its very nature generates more questions than it answers. The work outlined in the previous chapters is no exception. The author has sought to provide answers to the more immediate questions but many more need to be examined for making this methodology robust and viable in commercial applications. Tests on a large variety of complicated problems have produced encouraging results and exhibited the versatility of this method. Future research into this method must inevitably focus on the following aspects:

- iterative refinement of the solution through *a posteriori* error estimation and the use of hierarchical basis functions. Hierarchical basis functions use p refinement

instead of h refinement to adaptively correct the solution, maintaining the coarse mesh throughout.

- further reduction in the number of unknowns through the use of

  1. mixed node-based and edge-based elements. Since node-based elements usually have half as many unknowns as edge-based elements, nodal basis can be employed in free-space regions to reduce unknown count.

  2. derivation of higher order boundary conditions. This would allow the mesh termination boundary to be placed even closer to the target, thus reducing the number of unknowns dramatically - the possible asymmetry of the resulting equation system is an acceptable tradeoff.

- more robust, geometry-dependent, iteratively refined numerical mesh termination conditions that would guarantee the accuracy of the final solution. In most cases, it is the mesh termination condition and its distance from the scatterer which determine the accuracy of the computation. Numerical, iteratively refined ABCs could theoretically be applied very close to the scattering or radiating target with excellent accuracy.

- extension of the formulation to antenna radiation and electromagnetic interference(EMI) problems. The formulation for the antenna problem must include efficient source modeling and reliable simulation of the near-field for impedance calculations. EMI phenomena critical to electronic packaging can be predicted efficiently by the technique through proper modeling of the circuit or the device to be shielded.

- more efficient mesh generation packages dedicated to electromagnetics.

Most of the extensions mentioned above are not trivial and each could form the core of another doctoral dissertation. However, the efficient realization of even a few of these topics would result in a very powerful technique for computing electromagnetic radiation or scattering from arbitrary three dimensional structures reliably and efficiently.

APPENDICES

APPENDIX A

Derivation of matrix elements

The derivation of the matrix elements in (4.7) amounts to evaluating the integrals in (4.4) and (4.5). Therefore, from (4.4), we have

$$\int_{V_e} \frac{1}{\mu_r}(\nabla \times \mathbf{W}_i^e) \,.\, (\nabla \times \mathbf{W}_j^e) \;=\; \frac{4}{\mu_r}\mathbf{g}_i.\mathbf{g}_j V_e \tag{A.1}$$

since $\nabla \times \mathbf{W}_i^e = 2\mathbf{g}_i$. The evaluation of the integral in (4.5) is more cumbersome. Substituting into (4.5) the basis functions defined in (4.11), we obtain

$$\begin{aligned} \epsilon_r \int_{V_e} \mathbf{W}_i^e.\mathbf{W}_j^e dv &= \epsilon_r \int_{V_e} \{(\mathbf{f}_i.\mathbf{f}_j) + (\mathbf{r}.\mathbf{D}) + (\mathbf{g}_i \times \mathbf{r}).(\mathbf{g}_j \times \mathbf{r})\}\, dv \qquad (A.2) \\ &= \epsilon_r \,(I_1 + I_2 + I_3) \end{aligned}$$

where

$$\mathbf{D} \;=\; (\mathbf{f}_i \times \mathbf{g}_j) + (\mathbf{f}_j \times \mathbf{g}_i)$$

and

$$I_1 \;=\; \int_{V_e} \mathbf{f}_i.\mathbf{f}_j dv \tag{A.3}$$

$$I_2 \;=\; \int_{V_e} \mathbf{r}.\mathbf{D} dv \tag{A.4}$$

$$I_3 \;=\; \int_{V_e} (\mathbf{g}_i \times \mathbf{r}).(\mathbf{g}_j \times \mathbf{r}) dv \tag{A.5}$$

Since $\mathbf{f}$ is a constant vector, I_1 reduces to

$$I_1 = \mathbf{f}_i.\mathbf{f}_j \, V_e \tag{A.6}$$

Since

$$x = \sum_{i=1}^{4} L_i x_i$$
$$y = \sum_{i=1}^{4} L_i y_i$$
$$z = \sum_{i=1}^{4} L_i z_i$$

where L_i are the shape functions for the tetrahedral element and $x_i, y_i, z_i (i = 1, \cdots, 4)$ denote the x, y and z co-ordinates of the vertices of the tetrahedral element. Using the standard formula for volume integration within a tetrahedral element and simplifying, we have

$$I_2 = \frac{V_e}{4}\left[D_x \sum_{i=1}^{4} x_i + D_y \sum_{i=1}^{4} y_i + D_z \sum_{i=1}^{4} z_i\right] \tag{A.7}$$

where D_m is the mth component of $\mathbf{D}$. The evaluation of I_3 can be simplified by the use of basic vector identities. Therefore,

$$\begin{aligned} I_3 &= \mathbf{g}_i.\mathbf{g}_j \int_{V_e} |\mathbf{r}|^2 \, dv - \int_{V_e} (\mathbf{g}_i.\mathbf{r})\,(\mathbf{g}_j.\mathbf{r})\, dv \\ &= (g_{iy}g_{jy} + g_{iz}g_{jz}) \int_{V_e} x^2 dv + (g_{ix}g_{jx} + g_{iz}g_{jz}) \int_{V_e} y^2 dv + (g_{ix}g_{jx} + g_{iy}g_{jy}) \int_{V_e} z^2 dv \\ &\quad - (g_{ix}g_{jy} + g_{jx}g_{iy}) \int_{V_e} xy dv - (g_{ix}g_{jz} + g_{jx}g_{iz}) \int_{V_e} zx dv - (g_{iy}g_{jz} + g_{jy}g_{iz}) \int_{V_e} yz dv \end{aligned} \tag{A.8}$$

where g_{im} represents the mth component of the vector $\mathbf{g}_i$. Each of the volume integrals in the above equation can be easily evaluated analytically and the result expressed in the following general form:

$$\int_{V_e} a_l a_m dv = \frac{V_e}{20}\left[\sum_{i=1}^{4} a_{li}a_{mi} + \sum_{i=1}^{4} a_{li}\sum_{i=1}^{4} a_{mi}\right] \tag{A.9}$$

where $l, m = 1, \cdots, 3$ and a_1 represents the variable x, a_2 stands for the variable y and a_3 denotes the variable z.

APPENDIX B

Derivation of some vector identities

The curl of a vector in the Dupin coordinate system is given by

$$\nabla\times\mathbf{E} = \nabla_T \times \mathbf{E} + \hat{\mathbf{n}} \times \frac{\partial \mathbf{E}}{\partial n} \tag{B.1}$$

where $\nabla_T \times \mathbf{E}$ is called the surface curl involving only the tangential derivatives and is defined as

$$\nabla_T \times \mathbf{E} = -\hat{\mathbf{n}} \times \nabla E_n + \hat{\mathbf{t}}_\mathbf{2}\kappa_1 E_{t_1} - \hat{\mathbf{t}}_\mathbf{1}\kappa_2 E_{t_2} + \hat{\mathbf{n}}\nabla \cdot (\mathbf{E} \times \hat{\mathbf{n}}) \tag{B.2}$$

In the above equation, κ_1 and κ_2 denote the principal curvatures of the surface under consideration, E_{t_1}, E_{t_2} are the tangential components and E_n is the normal component of the vector $\mathbf{E}$ on the surface.

We are interested in the evaluation of the three vector identities given in Chapter 5. Let us consider simplifying the tangential components of the curl of a vector, $\mathbf{E}$ in this case. Using the definition of the curl given above, we have

$$\begin{aligned}
\hat{\mathbf{n}} \times \nabla\times\mathbf{E} &= -(\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \nabla)E_n - \kappa_1 E_{t1}\hat{\mathbf{t}}_\mathbf{1} - \kappa_2 E_{t2}\hat{\mathbf{t}}_\mathbf{2} + \hat{\mathbf{n}} \times \left(\hat{\mathbf{n}} \times \frac{\partial \mathbf{E}}{\partial n}\right) \\
&= \nabla_t E_n - \left[\left(\frac{\partial E_{t1}}{\partial n} + \kappa_1 E_{t1}\right)\hat{\mathbf{t}}_\mathbf{1} + \left(\frac{\partial E_{t2}}{\partial n} + \kappa_2 E_{t2}\right)\hat{\mathbf{t}}_\mathbf{2}\right] \\
&= \nabla_t E_n + \hat{\mathbf{n}} \times \nabla\times\mathbf{E}_t
\end{aligned} \tag{B.3}$$

where $-(\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \nabla) = \nabla_t$. The first vector identity is, therefore, easily proved.

Next, we will prove the second of the three identities. We start with the term $\hat{\mathbf{n}} \times \nabla \times \nabla_t E_n$ and simplify it using the definition of the curl of a vector given above.

$$\begin{aligned} \hat{\mathbf{n}} \times \nabla \times \nabla_t E_n &= \hat{\mathbf{n}} \times \nabla \times \left[\nabla E_n - \hat{\mathbf{n}} \frac{\partial E_n}{\partial n} \right] \\ &= -\hat{\mathbf{n}} \times \nabla \times \hat{\mathbf{n}} \frac{\partial E_n}{\partial n} \\ &= (\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \nabla) \frac{\partial E_n}{\partial n} \\ &= -\nabla_t \left(\frac{\partial E_n}{\partial n} \right) \end{aligned} \tag{B.4}$$

Since $\nabla \cdot \mathbf{E} = \nabla \cdot \mathbf{E}_t + (\nabla \cdot \hat{\mathbf{n}}) E_n + \frac{\partial E_n}{\partial n}$, we can simplify the above relation even further by substituting the appropriate expression for the normal derivative of the normal component of the electric field and using the fact that the electric field is divergence-free in a source-free region.

$$\begin{aligned} \hat{\mathbf{n}} \times \nabla \times \nabla_t E_n &= \nabla_t (\nabla \cdot \mathbf{E}_t) + (\nabla \cdot \hat{\mathbf{n}}) \nabla_t E_n \\ &= \nabla_t (\nabla \cdot \mathbf{E}_t) + 2\kappa_m \nabla_t E_n \end{aligned} \tag{B.5}$$

where $\kappa_m = (\kappa_1 + \kappa_2)/2$ is the mean curvature.

The proof of the third identity is more complicated since it involves two curl operations on the electric field. We first need to switch the positions of the outermost $\hat{\mathbf{n}}\times$ and the $\nabla\times$ operators to arrive at a simplified form of the rather complex expression. Therefore,

$$\begin{aligned} \hat{\mathbf{n}} \times \nabla \times (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) &= \nabla \times \{ \hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \nabla \times \mathbf{E} \} - \hat{\mathbf{n}} \nabla_t \cdot (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) + \Delta\kappa (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) \\ &= -\nabla \times \{ \nabla \times \mathbf{E} - \hat{\mathbf{n}} (\nabla \times \mathbf{E})_n \} - \hat{\mathbf{n}} \nabla_t \cdot (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) \\ &\quad + \Delta\kappa (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) \end{aligned} \tag{B.6}$$

Now we use the fact that the electric field satisfies the wave equation to reduce the expression even further.

$$\hat{\mathbf{n}} \times \nabla \times (\hat{\mathbf{n}} \times \nabla \times \mathbf{E}) = -k_o^2 \mathbf{E} + \nabla \times \{ \hat{\mathbf{n}} (\nabla \times \mathbf{E})_n \} + k_o^2 \hat{\mathbf{n}} E_n + \Delta\kappa (\hat{\mathbf{n}} \times \nabla \times \mathbf{E})$$

$$= \nabla\times\{\hat{\mathbf{n}}(\nabla\times\mathbf{E})_n\} - k_o^2\mathbf{E}_t + \Delta\kappa(\hat{\mathbf{n}}\times\nabla\times\mathbf{E}) \qquad \text{(B.7)}$$

where $\Delta\kappa = \kappa_1 - \kappa_2$.

Thus, we have shown that all three identities hold as long as the vector, $\mathbf{E}$ in this case, is divergenceless and satisfies the vector wave equation.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] A. Woo, M. Schuh, M. Simon, H.T.G. Wang, and M.L. Sanders. Radar cross-section measurement data of a simple rectangular cavity. *NWC TM 7132*, 1991.

[2] A. Glisson and D.R. Wilton. Simple and efficient numerical techniques for treating bodies of revolution. *Univ. Mississippi Tech. Rep. 105*, 1982.

[3] R.F. Harrington. *Field computation by moment method.* Macmillan: New York, 1968.

[4] E.K. Miller, L. Medgyesi-Mitschang, and E.H. Newman: eds. *Computational Electromagnetics: Frequency-domain method of moments.* IEEE Press, 1992.

[5] A.F. Peterson. The *interior resonance* problem associated with surface integral equations of electromagnetics: numerical consequences and a survey of remedies. *Electromagnetics*, vol. 10, pp. 293-312, 1990.

[6] J.R. Mautz and R.F. Harrington. H-field, E-field and combined-field solutions for conducting bodies of revolution. *AEU*, vol. 32, pp. 157-163, 1978.

[7] J.D. Collins, J.M. Jin, and J.L. Volakis. Eliminating interior resonances in FE-BI methods for scattering. *IEEE Trans. Antennas Propagat.*, vol. 40, pp. 1583-1585, Dec 1992.

[8] B. Stupfel, R. Le Martret, P. Bonnemason, and B. Scheurer. Solution of the scattering problem by axisymmetrical penetrable objects with a mixed boundary-element and finite-element method. *Proc. JINA*, pp. 116-120, 1990.

[9] J.A. Stratton. *Electromagnetic theory.* McGraw Hill: New York, 1941.

[10] W. Ritz. Über eine neue Method zur Lösung gewissen Variations - Probleme der mathematischen Physik. *J. Reine Angew. Math.*, vol. 135, pp. 1-61, 1909.

[11] S.G. Mikhlin. *Variational methods in mathematical physics.* Macmillan, New York, 1964.

[12] O.C. Zienkiewicz. *The finite element method.* McGraw Hill, New York, 3rd edition, 1979.

[13] P.P. Silvester and R.L. Ferrari. *Finite elements for electrical engineers.* Cambridge University Press, 2nd edition, 1990.

[14] A. Bossavit and J.C. Verite. A mixed FEM-BIEM method to solve 3D eddy current problems. *IEEE Trans. Magnetics*, vol. 18, pp. 431-5, Mar 1982.

[15] J.P. Webb. Edge elements and what they can do for you. *IEEE Trans. Magnetics*, vol. 29, pp. 1460-1465, 1993.

[16] H.R. Schwarz. *Finite element methods.* Academic Press, 1988.

[17] Z.J. Cendes and P. Silvester. Numerical solution of dielectric loaded waveguides: I - Finite element analysis. *IEEE Trans. Microwave Theory Tech.*, vol. 118, pp. 1124-1131, 1970.

[18] X. Yuan, D.R. Lynch, and K. Paulsen. Importance of normal field continuity in inhomogeneous scattering calculations. *IEEE Trans. Microwave Theory Tech.*, vol. 39, pp. 638-642, Apr 1991.

[19] H. Whitney. *Geometric Integration Theory.* Princeton University Press, 1957.

[20] J.C. Nedelec. Mixed finite elements in R^3. *Numer. Math.*, vol. 35, pp. 315-41, 1980.

[21] M. Hano. Finite element analysis of dielectric-loaded waveguides. *IEEE Trans. Microwave Theory Tech.*, vol. 32, pp. 1275-9, Oct. 1984.

[22] G. Mur and A.T. deHoop. A finite element method for computing three-dimensional electromagnetic fields in inhomogeneous media. *IEEE Trans. Magnetics*, vol. 21, pp. 2188-91, Nov 1985.

[23] J.S. van Welij. Calculation of eddy currents in terms of H on hexahedra. *IEEE Trans. Magnetics*, vol. 21, pp. 2239-41, Nov 1985.

[24] M.L. Barton and Z.J. Cendes. New vector finite elements for three-dimensional magnetic field computation. *J. Appl. Phys.*, vol. 61, no. 8, pp. 3919–21, Apr 1987.

[25] J.F. Lee, D.K. Sun, and Z.J. Cendes. Full-wave analysis of dielectric waveguides using tangential vector finite elements. *IEEE Trans. Microwave Theory Tech.*, vol. 39, no. 8, pp. 1262-71, Aug 1991.

[26] D.H. Schaubert, D.R. Wilton, and A.W. Glisson. A tetrahedral modeling method for electromagnetic scattering by arbitrarily shaped inhomogeneous dielectric bodies. *IEEE Trans. Antennas Propagat.*, pp. 77–85, Jan 1984.

[27] D.R. Tanner and A.F. Peterson. Vector expansion functions for the numerical solution of maxwell's equations. *Microwave Opt. Tech. Lett.*, vol. 2, no. 2, pp. 331-334, 1989.

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

[29] Z.J. Cendes. Vector finite elements for electromagnetic field computation. *IEEE Trans. Magnetics*, vol. 27, no. 5, Sep 1991.

[30] J.M. Jin and J.L. Volakis. Electromagnetic scattering by and transmission through a three-dimensional slot in a thick conducting plane. *IEEE Trans. Antennas Propagat.*, vol. 39, pp. 543-50, Apr 1991.

[31] A. Bossavit. Whitney forms: a class of finite elements for three-dimensional computations in electromagnetism. *IEE Proceedings*, vol. 135, pt. A, no.8, Nov 1988.

[32] J.F. Lee, D.K. Sun, and Z.J. Cendes. Tangential vector finite elements for electromagnetic field computation. *IEEE Trans. Magnetics*, vol. 27, no. 5, pp. 4032-5, Sep 1991.

[33] J.S. Wang and N. Ida. Curvilinear and higher order 'edge' finite elements in electromagnetic field computation. *IEEE Trans. Magnetics*, vol. 29, no. 2, pp. 1491-4, Mar 1993.

[34] J.P. Webb and B. Forghani. Hierarchal scalar and vector tetrahedra. *IEEE Trans. Magnetics*, vol. 29, no. 2, pp. 1495-8, Mar 1993.

[35] A.F. Peterson and S.P. Castillo. A frequency-domain differential equation formulation for electromagnetic scattering from inhomogeneous cylinders. *IEEE Trans. Antennas Propagat.*, vol. 37, no. 5, pp. 601-607, May 1989.

[36] R. Mittra and O. Ramahi. *Absorbing boundary conditions for the direct solution of partial differential equations arising in electromagnetic scattering problems.* Elsevier:New York, 1990. Chapter 4 in *Finite Element and Finite Difference Method in Electromagnetic Scattering* ed. M.A. Morgan.

[37] B.M.A. Rahman and J.B. Davies. Penalty function improvement of waveguide solution by finite elements. *IEEE Trans. Microwave Theory Tech.*, vol. 32, pp. 922-28, Aug 1984.

[38] J.P. Webb. Finite element analysis of dispersion in waveguides with sharp metal edges. *IEEE Trans. Microwave Theory Tech.*, vol. 36, no. 12, pp. 1819-1824, Dec 1988.

[39] Steven H. Wong and Z.J. Cendes. Combined finite element-modal solution of three-dimensional eddy current problems. *IEEE Trans. Magnetics*, vol. 24, no. 6, Nov 1988.

[40] A. Bossavit. Solving maxwell's equations in a closed cavity, and the question of spurious modes. *IEEE Trans. Magnetics*, vol. 26, no. 2, pp. 702-705, Mar 1990.

[41] X. Yuan. Three dimensional electromagnetic scattering from inhomogeneous objects by the hybrid moment and finite element method. *IEEE Trans. Antennas Propagat.*, vol. 38, pp. 1053-1058, 1990.

[42] G.H. Golub and C.F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, 1985.

[43] L.W. Pearson, A.F. Peterson, L.J. Behrmasel, and R.A. Whitaker. Inward-looking and outward-looking formulations for scattering from penetrable objects. *IEEE Trans. Antennas Propagat.*, vol. 40, pp. 714-720, 1992.

[44] J. Angelini, C. Soize, and P. Soudais. Hybrid numerical method for harmonic 3D Maxwell equations: Scattering by a mixed conducting and inhomogeneous anisotropic dielectric medium. *IEEE Trans. Antennas Propagat.*, vol. 41, no. 1, pp. 66-76, Jan 1993.

[45] J. D'Angelo and I.D. Mayergoyz. Three-dimensional RF scattering by the finite element method. *IEEE Trans. Magnetics*, vol. 27, no. 5, pp. 3827-3832, Sep 1991.

[46] I.D. Mayergoyz and J. D'Angelo. New finite element formulation for 3D scattering problem. *IEEE Trans. Magnetics*, vol. 27, no. 5, pp. 3967-3970, Sep 1991.

[47] A.F. Peterson. Absorbing boundary conditions for the vector wave equation. *Microwave and Opt. Techn. Letters.*, vol. 1, pp. 62-64, Apr 1988.

[48] J.P. Webb and V.N. Kanellopoulos. Absorbing boundary conditions for finite element solution of the vector wave equation. *Microwave and Opt. Techn. Letters*, Vol. 2, No. 10, pp. 370–372, Oct 1989.

[49] J. D'Angelo and I.D. Mayergoyz. Finite element methods for the solution of RF radiation and scattering problems. *Electromagnetics*, Vol. 10,pp. 177–199, 1990.

[50] T.B.A. Senior. Combined resistive and conductive sheets. *IEEE Trans. Antennas Propagat.*, Vol. AP–33, pp. 577–579, 1985.

[51] A.D. Yaghjian and R.V. McGahan. Broadside radar cross-section of a perfectly conducting cube. *IEEE Trans. Antennas Propagat.*, vol. 33, no. 3, pp. 321-329, Mar 1985.

[52] Courtesy of Northrop Corp., B2 Division, Pico Rivera, CA.

[53] D.R. Kincaid and T.C. Oppe. ITPACK on supercomputers. *Numerical Methods, Lecture Notes in Mathematics*, vol. 1005, pp. 151-161, 1982.

[54] G.V. Paolini and G. Radicati di Brozolo. Data structures to vectorize CG algorithms for general sparsity patterns. *BIT*, vol. 29, pp. 703-718, 1989.

[55] M.R. Hestenes and E.Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, vol. 49, pp. 409-436, 1952.

[56] P. Sonneveld. CGS, a fast solver for nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, vol. 10, pp. 35-52, 1989.

[57] R. Freund. Conjugate-gradient type methods for linear systems with complex symmetric coefficient matrices. *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 425-448, 1992.

[58] H.P. Langtangen. Conjugate gradient methods and ILU preconditioning of non-symmetric matrix systems with arbitrary sparsity patterns. *Int. J. Numer. Meth. Fluids*, vol. 9, pp. 213-233, 1989.

[59] J.R. Lovell. Hierarchical basis functions for 3D finite element methods. *ACES Digest*, pp. 657-63, 1993.

[60] E. Rothberg and A. Gupta. Parallel ICCG on a hierarchical memory multiprocessor - Addressing the triangular solve bottleneck. *Parallel Computing*, vol. 18, pp. 719-741, 1992.

[61] E. Anderson and Y. Saad. Solving sparse triangular linear systems on parallel computers. *International Journal of High Speed Computing*, vol. 1, no. 1, pp. 73-95, 1989.

[62] D. Windheiser, E. Boyd, E.Hao, S.G. Abraham, and E.S. Davidson. KSR1 multiprocessor: Analysis of latency hiding techniques in a sparse solver. In *Proc. of the 7th International Parallel Processing Symposium*, Apr 1993. Newport Beach.

[63] C.T. Tai. *Generalized vector and dyadic analysis.* IEEE Press, New York, 1992.

[64] D.S. Jones. An improved surface radiation condition. *IMA J. Appl. Math.*, vol. 48, pp. 163-193, 1992.

[65] C.H. Wilcox. An expansion theorem for electromagnetic fields. *Comm. Pure Appl. Math.*, vol. 9, pp. 115-134, May 1956.

[66] S.M. Rytov. Computation of the skin effect by the perturbation method. *J. Exp. Theor. Phys.*, vol. 10, pp. 180, 1940. Translation by V. Kerdemelidis and K.M. Mitzner.

[67] A. Chatterjee, J.M. Jin, and J.L. Volakis. Edge-based finite elements and vector ABCs applied to 3D scattering. *IEEE Trans. Antennas Propagat.*, vol. 41, no.2, pp. 221-226, Feb 1993.

[68] V. Shankar, William F. Hall, Alireza Mohammedian, and Chris Rowell. Development of a finite-volume, time-domain solver for maxwell's equations. *Rockwell Technical Report prepared for NASA/NDC under contract N62269-90-C-0257*, May 1993.

[69] M. Schuh, A. Woo, M. Sanders, and H.T.G. Wang. Radar cross-section measurement data of four small cavities. *NASA Tech. Memo. 108782*, Nov 1993.