ADVANCES IN TIME-DOMAIN ELECTROMAGNETIC SIMULATION CAPABILITIES THROUGH THE USE OF OVERSET GRIDS AND MASSIVELY PARALLEL COMPUTING

DISSERTATION

Douglas C. Blake, Captain, USAF

AFIT/DS/ENY/97-2

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/DS/ENY/97-2

ADVANCES IN TIME-DOMAIN ELECTROMAGNETIC SIMULATION CAPABILITIES THROUGH THE USE OF OVERSET GRIDS AND MASSIVELY PARALLEL COMPUTING

DISSERTATION

Douglas C. Blake, Captain, USAF
AFIT/DS/ENY/97-2

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1997	3. REPORT TYPE AND DATES COVERED Dissertation

4. TITLE AND SUBTITLE ADVANCES IN TIME-DOMAIN ELECTROMAGNETIC SIMULATION CAPABILITIES THROUGH THE USE OF OVERSET GRIDS AND MASSIVELY PARALLEL COMPUTING	5. FUNDING NUMBERS
6. AUTHOR(S) Douglas C. Blake	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583	8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/DS/ENY/97-2
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Dr Joseph Shang WL/FIM Wright-Patterson AFB, OH 45433	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Distribution Unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A new methodology is presented for conducting numerical simulations of electromagnetic scattering and wave-propagation phenomena. Technologies from several scientific disciplines, including computational fluid dynamics, computational electromagnetics, and parallel computing, are uniquely combined to form a simulation capability that is both versatile and practical. In the process of creating this capability, work is accomplished to conduct the first study designed to quantify the effects of domain decomposition on the performance of a class of explicit hyperbolic partial differential equations solvers; to develop a new method of partitioning computational domains comprised of overset grids; and to provide the first detailed assessment of the applicability of overset grids to the field of computational electromagnetics. Furthermore, the first Finite-Volume Time-Domain (FVTD) algorithm capable of utilizing overset grids on massively parallel computing platforms is developed and implemented. Results are presented for a number of scattering and wave-propagation simulations conducted using this algorithm, including two spheres in close proximity and a finned missile.

14. SUBJECT TERMS Parallel Computing, FVTD, CEM, Domain Decomposition, Overset Grids			15. NUMBER OF PAGES 187
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

AFIT/DS/ENY/97-2

ADVANCES IN TIME-DOMAIN ELECTROMAGNETIC SIMULATION CAPABILITIES THROUGH THE USE OF OVERSET GRIDS AND MASSIVELY PARALLEL COMPUTING

DISSERTATION

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Douglas C. Blake

Captain, USAF

Sponsored by the Flight Dynamics Directorate, Wright Laboratory, WPAFB, OH.

March 1997

AFIT/DS/ENY/97-2

ADVANCES IN TIME-DOMAIN ELECTROMAGNETIC SIMULATION CAPABILITIES THROUGH THE USE OF OVERSET GRIDS AND MASSIVELY PARALLEL COMPUTING

Douglas C. Blake
Captain, USAF

Approved:

[illegible]	3/7/97
THOMAS A. BUTER, Research Advisor Major, USAF Department of Aeronautical and Astronautical Engineering	
[illegible]	3/7/97
RODNEY D.W. BOWERSOX, Committee Member Assistant Professor Department of Aeronautical and Astronautical Engineering	
Gary B. Lamont	3/7/97
GARY B. LAMONT, Committee Member Professor Department of Electrical and Computer Engineering	
[illegible]	3/7/97
JOSEPH J.S. SHANG, Committee Member Adjunct Professor Department of Aeronautical and Astronautical Engineering	
[illegible]	3/7/97
MICHAEL G. STOECKER, Committee Member Adjunct Professor Department of Mathematics and Statistics	
Wm. F. Bailey	3/7/97
WILLIAM F. BAILEY, Dean's Representative Associate Professor Department of Engineering Physics	

[illegible]
ROBERT A. CALICO, Jr.
Dean, Graduate School of Engineering

Acknowledgments

The past three years have been extremely rewarding for me. I have worked in a field of study that I genuinely enjoy and with people that I greatly respect. Most prominent among those people has been Major Tom Buter, who was an absolutely first-rate advisor. Major Buter was always accessible and helpful, and no one championed my work more than he. I will always be grateful for his guidance, and more importantly, for the friendship that he and his wife Kimberlee extended to my family and me.

I would also like to thank the members of my doctoral committee: Dr. Joseph Shang, Dr. Gary Lamont, Dr. Mike Stoecker, and Dr. Rodney Bowersox. They were always willing to offer help whenever needed. A special thanks goes to Dr. Shang. Although the demands on his time were great, he never failed to sit down and discuss my work from both scientific and personal perspectives. It was Dr. Shang who first piqued my interest in the field of computational electromagnetics.

Although not a member of my doctoral committee, a heartfelt thanks goes to Dr. Kueichien Hill of Wright Labs/XPN for generously providing all of the method of moment results appearing in this document. Others who provided significant input to this work include Dr. Datta Gaitonde and Dr. Yvette Weber of the Flight Dynamics Directorate at Wright Labs. In addition to these individuals, I would especially like to thank my friend and colleague, Captain Scott Morton, with whom I have had countless enjoyable conversations, both technical and personal, over the past several years.

Finally, and most importantly, I wish to thank my wife, Terri. During the past three years, she has largely filled the roles of both mother and father to our two children and has done so with understanding, patience, and generosity. I will always be in her debt, and to her I dedicate this work.

Douglas C. Blake

Table of Contents

List of Figures

List of Tables

List of Symbols

Symbol	Definition	Chapter
Bx, By, Bz	Cartesian components of magnetic flux density vector	2, 3, A
Dx, Dy, Dz	Cartesian components of electric flux density vector	2, 3, A
k	wavenumber	3, 7, 8, A
$\vec{B}$	magnetic flux density vector (Cartesian coordinate frame)	3, A
$\vec{D}$	electric flux density vector (Cartesian coordinate frame)	3, A
$\vec{E}$	electric field intensity vector (Cartesian coordinate frame)	3, A
$\tilde{F}$	flux vector	3, A
$\vec{H}$	magnetic field intensity vector (Cartesian coordinate frame)	3, A
J	Jacobian of coordinate transformation	3, A
$\vec{J}$	surface current density (Cartesian coordinate frame)	3, A
$\tilde{J}$	surface current density (general curvilinear coordinate frame)	3, A
Jx, Jy, Jz	Cartesian components of surface current density vector	3, A
$\hat{n}$	unit normal vector	3, A
$\vec{Q}$	dependent variable vector (Cartesian coordinate frame)	3, A
$\tilde{Q}$	dependent variable vector (general curvilinear coordinate frame)	3, A
S	integration surface	3, A
Δt	time step	3, A
$\vec{U}, \vec{V}, \vec{W}$	flux vectors (Cartesian coordinate frame)	3, A
$\tilde{U}, \tilde{V}, \tilde{W}$	flux vectors (general curvilinear coordinate frame)	3, A
V	cell volume	3, A
D	bytes of storage required per grid cell	4
E	relative efficiency	4

E_a	absolute efficiency	4
l	number of message hops	4
m	message length	4
n	number of cells in the computational domain	4
p	processor count	4
R	number of grid cells along a coordinate direction	4
S	relative speedup	4
S_a	absolute speedup	4
T_1	single-processor algorithm-run time	4
T_{calc}	calculation time	4
T_{comm}	total per-processor communication time	4
T_{par}	parallel run time	4
$T_{overhead}$	parallel overhead time	4
t_c	single-processor computation time per grid cell per time step	4
t_{comm}	per-message communication time	4
t_h	per-hop time	4
t_s	message start-up time	4
t_w	per-word transfer time	4
i	dimensionality of domain decomposition	4, 5
k	dimensionality of problem space	4, 5
n_p	number of cells on a processor	4, 5
C_l	number of grid cells along each coordinate direction in grid l	5
$c_{d,I}$	cell donating interpolation information	5
c_I	cell belonging to set I	5
c_l	number of cutting planes used to partition grid l	5

c_Q	cell belonging to set Q	5
c_R	cell belonging to set R	5
D	set of cells comprising the computational domain	5
$\mathbf{D}$	cardinality of set D	5
$\mathbf{E}_s$	total number of structured communication graph edges	5
$\mathbf{E}_u$	total number of unstructured communication graph edges	5
$e(c_1,c_2)$	communication graph edge	5
$G(D,E)$	communication graph for the computational domain	5
I	set of cells updated via interpolation	5
$\mathbf{I}$	cardinality of set I	5
l	grid identifier	5
m	number of structured grids comprising the computational domain	5
$\mathbf{N}_d$	total number of donor cells	5
p_l	number of processors to which grid l is assigned	5
Q	set of cells updated via governing equation	5
$\mathbf{Q}$	cardinality of set Q	5
a	body radius (sphere, missile, etc.)	7, 8
A, B, C	flux Jacobian matrices	A
$\vec{D}_{ij}$	cell face diagonal vector from vertex i to vertex j	A
$d\vec{A}_k$	surface vector for cell face k	A
G	free-space Green's function	A
$\hat{i}, \hat{j}, \hat{k}$	unit vectors in the Cartesian coordinate system	A
j	$\sqrt{-1}$	A
P, P^{-1}	diagonalization matrices for flux Jacobians	A
$\tilde{R}$	total flux vector	A

V	cell volume	A

Greek Symbols

σ	radar cross section	3, 6, 7, 8, A
μ	magnetic permeability	3, A
ε	electric permittivity	3, A
ρ	surface change density	3, A
κ	van Leer MUSCL-extrapolation parameter	3, A
ϕ	van Leer MUSCL-extrapolation parameter	3, A
ω	angular frequency	3, A
(ξ, η, ζ)	general curvilinear coordinate system	3, A
τ	number of planes of data transferred via message	4
η	number of messages required per iteration	4
Θ	tight upper bound	4, 5
ζ	computational stencil half-width	4, 5
λ	load imbalance factor	5
ϕ	ratio of unstructured to structured cut communication graph edges	5
α	fraction of total unstructured edges cut in a decomposition	5
ψ	number of planes of grid cells having cut structured edges	5
ϕ	RCS look angle	6, A
Λ	eigenvalue matrix	A

superscripts

~	curvilinear coordinate system value	3, A
^	unit vector	3, A

+	positive component	3, A
-	negative component	3, A
0, 1, 2	Runge-Kutta stage identifier	3, A
L, R	left or right state	3, A
n	current time level	3, A
$n+1$	updated time level	3, A
k	dimensionality of problem space	4, 5

subscripts

1 - 6	cell face identifiers	3, A
ξ, η, ζ	derivatives with respect to ξ, η, or ζ	3, A
$i \pm 1/2$	cell face identifier	3, A
$i \pm 1, etc$	cell center identifier	3, A
i	incident field	3, A
k	cell face identifier	3, A
s	scattered field	3, A
t	derivative with respect to time	3, A
t	total field	3, A
l	grid identifier	5
~	tensor	A
0	free-space value	A
A, B, C	association with A, B, or C flux Jacobian matrices	A
x, y, z	derivatives with respect to x, y, or z	A

Abstract

A new methodology is presented for conducting numerical simulations of electromagnetic scattering and wave-propagation phenomena on massively parallel computing platforms. A process is constructed which is rooted in the Finite-Volume Time-Domain (FVTD) technique to create a simulation capability that is both versatile and practical. In terms of versatility, the method is platform independent, is easily modifiable, and is capable of solving a large number of problems with no alterations. In terms of practicality, the method is sophisticated enough to solve problems of engineering significance and is not limited to mere academic exercises.

In order to achieve this capability, techniques are integrated from several scientific disciplines including computational fluid dynamics, computational electromagnetics, and parallel computing. The end result is the first FVTD solver capable of utilizing the highly flexible overset-gridding process in a distributed-memory computing environment. In the process of creating this capability, work is accomplished to conduct the first study designed to quantify the effects of domain-decomposition dimensionality on the parallel performance of hyperbolic partial differential equations solvers; to develop a new method of partitioning a computational domain comprised of overset grids; and to provide the first detailed assessment of the applicability of overset grids to the field of computational electromagnetics.

Using these new methods and capabilities, results from a large number of wave propagation and scattering simulations are presented. The overset-grid FVTD algorithm is demonstrated to produce results of comparable accuracy to single-grid simulations while simultaneously shortening the grid-generation process and increasing the flexibility and utility of the FVTD technique. Furthermore, the new domain-decomposition approaches developed for overset grids are shown to be capable of producing partitions that are better load balanced and require less interprocessor communication than did previously used overset-grid decomposition methods. This results in parallel efficiencies routinely in excess of 90 percent, even for relatively small problems and large numbers of processors.

I. Introduction and Problem Motivation

This study is dedicated to advancing the scientific community's ability to conduct accurate electromagnetic wave-propagation and scattering simulations for complex geometries. It is grounded in a philosophy which emphasizes commonality between various design disciplines and generality in its applicability to a wide range of problem types. This first chapter discusses the motivation behind the research and formulates the problem statement and work objectives that carry throughout the document.

1.1 Historical Perspective on Aircraft Design

Before the use of radar became commonplace during World War II, little or no consideration was paid on the part of the airframe designer to the electromagnetic signature properties of an aircraft. Indeed, even after the viability of radar had long been established, aircraft were typically not constructed with radar cross section (RCS) reduction in mind. Instead, the threat of detection was met with efforts that included "electronic jamming, window, 'spoofs', and where possible, avoidance" [31]. This philosophy continued nearly unchanged from World War II through the development of such high-performance aircraft as the F-15 and F-16 which were designed to be highly maneuverable and aerodynamically capable rather than electromagnetically "stealthy." Although this design philosophy produced some of the most successful military aircraft in history, continued improvements to radar and missile technology have forced the realization that superior aerodynamic performance alone does not provide sufficient survivability enhancement in a modern combat environment. Thus, aircraft designers have been motivated to consider the electromagnetic-signature issue early in the design process. The reason for doing so is eloquently expressed by Fuhs [41] who states, "The advantage of a certain ECM (electronic countermeasures) technique can be nullified in a week or two. The advantage of a built-in low RCS requires a decade for the opponent to nullify." (parentheses added)

Although the importance of considering signature issues early in the aircraft-design phase is now well understood, these issues often conflict with aerodynamic design requirements. The same rounded wing leading edge that reduces electromagnetic signature can dramatically increase aerodynamic wave drag at supersonic speeds; the wing sweep angle designed for optimum aerodynamic performance may result in an unacceptable RCS spike; or the radar-absorbent material (RAM) that diminishes radar returns may exact a substantial weight penalty [41]. Clearly, the considerations are different when designing for aerodynamic or electromagnetic performance, and in today's environment, it can be dangerous to promote one exclusively at the expense of the other. Given the potentially conflicting design requirements, an efficient design environment becomes quite critical. Even in the absence of electromagnetic signature considerations, the design-optimization portion of the production process can be extremely labor intensive. The configuration development of the Lockheed L-1011 required over two-million man hours solely for the determination of the optimum aircraft design [119]. Simultaneously considering signature and aerodynamic issues would seem to exacerbate the already labor-intensive process. Thus, a high degree of cooperation between electromagneticists and aerodynamicists is necessary during all design phases. As noted by Chawla [29], such a design philosophy can be expected to yield superior results to one in which the optimized design of a single discipline serves as the primary driver to the remaining disciplines of a system.

1.2 New Design Approaches

Recognizing the need to develop new design methodologies, researchers from the computational fluid dynamics (CFD) community have recently begun to apply CFD-based numerical techniques to the field of computational electromagnetics (CEM) [97,98,100,101,123]. Their ability to do so stems from the fact that the governing equations for the inviscid-limiting behavior of fluid flow–the Euler equations–and for the propagation of electromagnetic waves–the Maxwell equations–are both hyperbolic sets of partial differential equations (PDEs). The efforts of the CFD researchers have produced the Finite-Volume Time-Domain (FVTD) technique which utilizes direct time integration of the Maxwell Equations, a dramatic departure from the frequency-domain solution methods of the past. The FVTD approach has several advantages: first, it brings to bear on the Maxwell equations the large arsenal of hyperbolic PDE solution techniques developed for the Euler equations, and second, (and perhaps more importantly) it offers the potential for an integrated algorithmic approach to solving both the aerodynamic and electromagnetic problems.

Despite the potential advantages of the FVTD technique, acceptance of the method by the CEM community has been limited because of several shortcomings. To begin, because the method solves the Maxwell equations instead of an approximation to them, it is computationally intensive. At present, generating a single monostatic RCS profile for a complicated, three-dimensional geometry can easily consume days or even weeks of supercomputer time. Furthermore, the method requires the generation of a volumetric grid surrounding the body of interest. Generating an acceptable grid is often a difficult procedure and is typically one of the most time-consuming aspects of the entire solution process. Finally, the FVTD codes produced to date are often highly specialized and lack user-friendly features, thus making it difficult to apply the codes to a wide range of geometries.

1.3 Problem Statement

Although the viability of the FVTD approach has been firmly established[1], evolving the methodology into a product that can be effectively utilized as a multi-disciplinary design tool requires overcoming the limitations identified in the previous section. The shortfall thus addressed by the present study can be succinctly stated as follows:

> *Current time-domain electromagnetic simulation implementations do not fully capitalize on technologies which could offer the promise of significant capability enhancements.*

In order to address this problem, innovations having roots in other disciplines such as computer science and fluid dynamics should be incorporated into the FVTD methodology. Two of the more promising technologies are discussed in the following section.

1.4 Promising Technological Areas

1.4.1 Parallel Computing

The evolution of supercomputer design and capability has been nothing short of astounding. During the 1970s and 1980s, supercomputing was dominated by vector machines from companies such as CDC and Cray. As the name "vector supercomputer" implied, these machines operated most efficiently when computer codes contained a large number of array- or vector-based operations. To this end, scientists and engineers quickly learned how to construct their computer codes (written largely in FORTRAN) to exploit this type of architecture. Towards the end of the 1980s, however, vector machines began to reach physical limitations in terms of clock speeds, and thus designers shifted paradigms to one of quantity by applying more processors to solve a given problem. The recent dramatic advances in commodity processors and high-speed networks have enabled designers to construct computing machines comprised of several hundred to several

1. A more detailed discussion of the viability of the FVTD approach appears in Chapter 2.

thousand very powerful microprocessors and thereby produce machines capable of performance exceeding that of the vector supercomputer [7,36,136].

One particular feature which makes parallel machines such a potential benefit to CFD-based CEM algorithms is their expandability, both in terms of memory and processor numbers. This feature makes these machines viable platforms for attacking problems of very large size and complexity. This advantage is potentially offset, however, because the programming environment for these machines is much more complex than that of the more traditional serial or shared-memory vector computers in that the user cannot simply rely on compiler optimizations to achieve efficient utilization of the machine. Furthermore, the memory structure of these machines does not necessarily lend itself to the same vector-programming paradigm that served so well with the Cray-class computers. Thus, although the machines offer great promise, considerable effort is required in order to achieve that promise. At this time, the scientific community is only beginning to develop the algorithms and programming techniques necessary to realize the full potential of these platforms.

1.4.2 Advanced Grid-Generation Techniques

Although parallel computing has the potential to shorten computer run times, the problem-solving process is often dominated by aspects other than the computation phase. As mentioned previously, grid generation for a complex geometry can be one of the most labor-intensive portions of a numerical investigation. Consequently, researchers have devoted significant effort to developing new methods for simplifying the grid-generation process. One technique originally developed for the CFD community that is especially noteworthy is the concept of overset or Chimera grids [11-13]. In this method, a complex geometry is divided into a set of component objects, and a structured, body-conformal grid is generated around each component. The structured grids are allowed to arbitrarily overlap and are thus combined to form the computational domain. Cells comprising the grid are updated either via application of the governing equations or by interpolation of information from other grids. To date, a large number of CFD simulations have been conducted with the aid of Chimera grids; however, the method has gone largely unnoticed by the CEM community.

1.5 Project Objectives

Using the technologies described in the previous section, the present study seeks to provide a solution to the problem discussed in Section 1.3. The overarching objective of this work can thus be stated as follows:

> *To develop a versatile and practical capability for conducting time-domain numerical simulations of electromagnetic scattering and wave-propagation phenomena.*

In the above-stated objective, the key words are *versatile* and *practical.* A versatile simulation capability is deemed to be one which is platform independent and capable of easily and rapidly addressing a large number of problems. A practical capability is one which can used to address problems of realistic engineering significance. In order to achieve these desirable features and thereby satisfy the primary objective, several sub-objectives must be met. They are

- to develop new methods for exploiting parallel architectures using both single and overset grids in conjunction with typical grid-based PDE solvers in general and FVTD solvers in specific,
- to ascertain the applicability and limitations of overset grids to the FVTD procedure, and
- to develop and validate an overset-grid-capable FVTD solution *process* for distributed-memory parallel computing platforms.

The interrelation of these objectives is depicted in Figure 1.1. This "objective triad" forms the basis and motivation for the research outline presented in the next section and is used throughout the remainder of this document to guide the reader through the material presentation.

1.6 Research Outline and Project Scope

In accordance with the objectives stated in the previous section, the research presented in this document proceeds along the following lines:

1. A single-grid, parallel FVTD solver is developed and validated.
2. A communication model for hyperbolic PDE solvers in parallel environments is developed in conjunction with the single-grid algorithm. Using this model and the parallel solver, a study is conducted to determine how parallel performance is affected by the method used to partition the computational domain.
3. The capability of the single-grid FVTD solver is extended to allow for overset grids and more general domain decompositions.
4. An algorithm for partitioning a computational domain comprised of overset grids is conceived and implemented.
5. A theoretical parallel-performance model for overset grids is developed, and the new FVTD

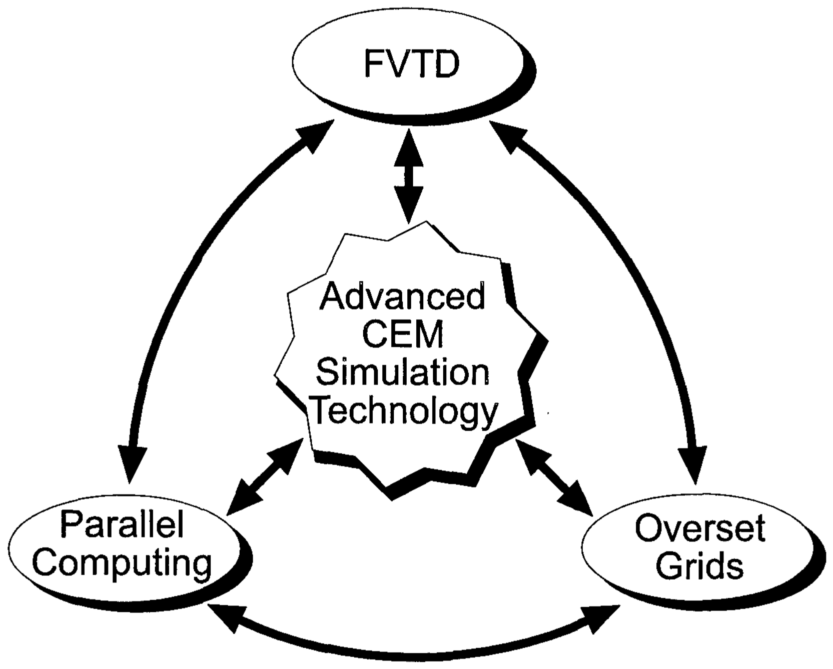


Figure 1.1: Objective Relation Diagram

solver is used in conjunction with the grid-partitioning algorithm to test the model.

6. The accuracy and applicability of the overset grid FVTD process is assessed using several test cases.

7. The resultant CEM simulation capability (i.e., a parallel, overset-grid FVTD solution process) is demonstrated for a complex geometry.

Note that the work description follows an incremental approach for attacking the problem presented in Section 1.3. Each phase of the work builds upon the previous phase thereby leading to a final product that achieves the desired characteristics of versatility and practicality.

1.7 Document Organization

The organization of this document revolves around the objective triad with each chapter dedicated either to one aspect exclusively, or to the interplay between two or more of the triad elements. A survey of the relevant literature is provided in Chapter 2 while the governing equations and numerical methods that ultimately form the foundation of the finished product are discussed in Chapter 3. Using this introductory material as a basis, a presentation of the theoretical development and numerical results begins in Chapter 4 with a study of single-grid domain-decomposition methods for parallel environments. Chapter 5 expands this discussion to include overset grids. Together, Chapters 4 and 5 form the crux of the parallel-computing-investigative portion of this work. The analysis and results presented in these chapters is intended to be general and applicable to a large class of hyperbolic PDE solvers and thus only discusses the FVTD scheme from a parallel-implementation standpoint. Electromagnetic-specific issues are addressed beginning in Chapter 6 wherein several test and validation cases for the FVTD algorithm are presented. The chapter is designed to identify important algorithm-behavioral issues and to provide a reference so that a meaningful comparison can be conducted with the overset-grid results presented in Chapter 7. Chapters 4-7 thus cumulatively treat all aspects of the objective triad. A discussion of the capabilities realized by combining the work of the four chapters appears in Chapter 8. Finally, Chapter 9 draws pertinent conclusions and recommends areas for further study. In addition to the work presented in the main body of this document, two appendices are provided as supplementary material: Appendix A augments the numerical discussion Chapter 3 while Appendix B provides details on the computational resources used throughout the course of this study.

II. Background

The bolded portions of Figure 2.1 represent the areas of emphasis of this chapter. As the chapter title indicates, the discussion presented here focuses on background and familiarity issues with each of the three main objective areas. To begin, a brief introduction of the various methods used to solve the Maxwell equations is presented. This is intended to provide a backdrop against which the benefits of a time-domain solution can be compared. The discussion then proceeds into a presentation of previous research pursuant to FVTD algorithm development and notes the areas in which the overall methodology can be improved, namely by incorporating parallel-computing and overset-grid technologies into the process. Discussion relating to parallel computing is limited to work previously accomplished in the field of parallel FVTD implementations and on the primary method used to achieve parallelism for grid-based scientific applications–that of domain decomposition. Finally, the overset-grid process is explained and the notion of domain decomposition is extended to overset-grid-based solvers.

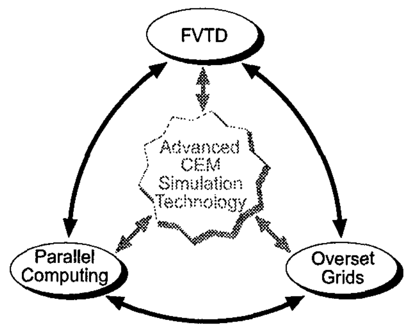


Figure 2.1: Chapter 2 Emphasis

2.1 Computational Electromagnetics

2.1.1 Frequency-Domain Methods

Before beginning the discussion of time-domain CEM, it is enlightening to briefly examine other computational methods that generally preceded time-domain techniques. These techniques for solving electromagnetic wave-propagation and scattering problems have been historically grouped into the categories of high- or low-frequency methods. High-frequency methods are generally applicable when the scattering object is large in comparison to the wavelength of the incident electromagnetic field and include such techniques as geometrical optics, physical optics, geometrical theory of diffraction, uniform theory of diffraction, and physical theory of diffraction [8,61,79,108]. These techniques usually do not involve spatial discretization of the domain of interest. Since spatial discretizations typically require at least ten grid cells per wavelength in order to accurately capture the physical phenomena of interest, they quickly become cumbersome as frequency increases. High-frequency methods are designed to avoid such problems; however, they are approximate methods and their range of applicability is strictly limited.

In contrast to high-frequency techniques, low-frequency methods are typically characterized by the discretization of the surface of the scatterer and/or the space surrounding the scatterer and the subsequent solution of a system of algebraic equations. Possibly the most widely used low-frequency technique is the Method of Moments (MoM) developed by Harrington in the 1960s [50]. The MoM solves for the surface currents on the body by discretizing the body itself, ultimately yielding a system of equation of the type $[Z]\{I\} = \{F\}$ where Z is termed the *impedance matrix* and I and F are the *current* and *excitation* vectors, respectively. The MoM has proven to be very accurate and has been applied to a large number of problems. Because of its accuracy, results obtained via other numerical techniques are most often compared to MoM results for validation purposes. Although the MoM has many desirable characteristics, the impedance matrix is fully populated and contains $O(N^2)$ entries where N is the number of field unknowns. Furthermore, computation of the elements of Z involves the calculation of relatively complex integrals. Such systems can place very heavy demands on computer storage and processing-rate capabilities [117]. Furthermore, because the MoM is a frequency-domain technique, it cannot model time-varying phenomena nor can it extract multiple frequency responses from a single excitation source. An excellent discussion of the basics of the MoM is given by Volakis and Kempel [125] while Umashankar and Taflove [120] provide a comprehensive survey into the development and applicability of a wide range of CEM techniques.

2.1.2 Development of the Time-Domain Technique

All of the methods discussed in the previous section involve at least some degree of approximation or limitation on their range of applicability. A direct solution of the governing equations of electromagnetics–the Maxwell equations–overcomes these limitations since these equations accurately model the behavior of the entire electromagnetic spectrum [46]. Recognizing this fact, in 1966 Yee [140] published his pioneering Finite-Difference Time-Domain (FDTD) algorithm which utilized direct time integration of the Maxwell equations to compute the electric and magnetic fields over the region of interest. In Yee's scheme, the temporal and spatial derivatives appearing in Maxwell's equations are discretized using standard finite-difference stencils on a staggered grid as shown in Figure 2.2*a*. This staggered-grid approach has several advantages [67] over a collocated grid as shown in Figure 2.2*b*. To begin, it eliminates ambiguities in the specification of

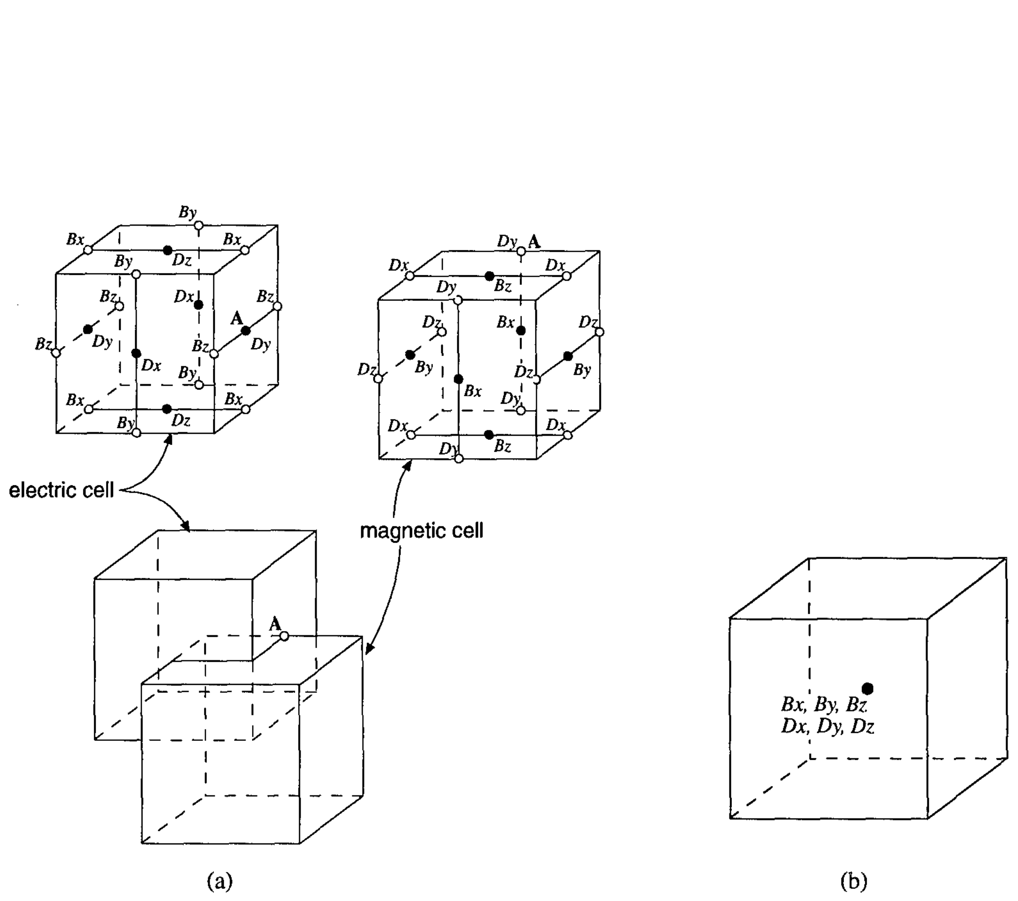


Figure 2.2: Time-Domain Grid Examples: a) Staggered Grid, b) Collocated, Cell-Centered Grid

the field variables on the surface of a scattering object[1]. Furthermore, it naturally exploits the means by which the electric and magnetic fields are related through the curl operators. Unfortunately, although staggered-mesh techniques exist in the CFD community [38,52], the locations where the unknowns are defined are not directly analogous to the FDTD algorithm. Thus, despite significant advantages, it is difficult to directly adapt Yee's technique to a fluid-mechanics problem, and thus its use in an integrated design environment is somewhat limited. However, in the CEM arena, the FDTD technique has proven to be highly successful. Since Yee's original work, a myriad of problems have been solved using FDTD, and Taflove [117,118] has emerged as the preeminent researcher in the field. One of the primary limitations of the original algorithm–that of the inability of the grid to conform to the scatterer surface–has been overcome by the work several researchers including Holland [56], Fusco [42], and Jurgens and Taflove [59].

In contrast to finite-difference techniques which are characterized by the application of the governing equations at discrete points of the physical domain, finite-volume methods typically use the integral form of

1. Surface boundary conditions are discussed in Chapter 3 and Appendix A.

conservation laws applied to discrete volumetric elements [124]. Because the governing equations of fluid mechanics are statements of conservations laws, finite-volume techniques have lately become very popular in the CFD community. Furthermore, over the past several years, researchers from the CFD community have begun to apply techniques developed for CFD to CEM problems. Specifically, they have drawn on the vast body of knowledge developed for solving hyperbolic systems of PDEs in an attempt to advance the state-of-the-art in time-domain simulations of electromagnetic phenomena. Their efforts have resulted in the genesis of the FVTD methodology.

Arguably the most active group of researchers using the FVTD technique is the group at Rockwell International Science Center headed by Dr. Vijaya Shankar. During the past eight years, they have developed and applied a temporally and spatially second-order-accurate Lax-Wendroff finite-volume method to a large number of one-, two-, and three-dimensional problems with both perfectly conducting and layered-dielectric media [70,97-107].

Another group of researchers who have contributed significantly to FVTD algorithm development is the group at Wright Laboratories headed by Dr. Joseph Shang. Under the guidance of Dr. Shang, the group has adapted a flux-vector-splitting algorithm in conjunction with a Runge-Kutta time-integration technique to yield a temporally fourth and spatially third-order-accurate method [84-96,132,133]. Recently, members of the group–most notably Dr. Datta Gaitonde–have investigated compact-differencing techniques in an attempt to extend the spatial accuracy of the method to higher order [43,95].

In addition to the aforementioned works of Shankar and Shang, several other researchers have investigated FVTD techniques. Harmon [48] has used Shang's FVTD code to analyze the scattering from several rather simple three-dimensional objects. Aftosmis [2] has compared Lax-Wendroff and Runge-Kutta methods using a cell-vertex FVTD scheme with special consideration for the effect of various higher-order absorbing boundary conditions and found that Lax-Wendroff yielded superior results for both one- and two-dimensional wave-propagation and scattering problems. Bishop and Anderson [15,16] have successfully demonstrated the use of material-based limiters in an upwind-based predictor-corrector scheme and a staggered-mesh central-difference scheme. Such limiters are designed to control oscillations in the field amplitudes as the wave propagates between media of different intrinsic impedances. Huh, Shu, and Agarwal [58] have developed a finite-volume algorithm capable of working in either the time or frequency domain and have applied it to successfully solve several two-dimensional scattering problems. Finally, emphasizing the advantages of staggered grids for implementing surface boundary conditions, Noack and Anderson [72] have used the integral form of Maxwell's equations to construct a finite-volume scheme to compute the radar cross section of a number of three-dimensional objects including a sphere and a finned projectile.

The examples cited above provide evidence that the FVTD method has begun to demonstrate its viability as a method for solving the Maxwell equations. The present study seeks to leverage the advances of previous works by incorporating two technologies which have either been completely omitted or else only partially implemented in other FVTD solvers. The remainder of this chapter is devoted to discussing those technologies.

2.2 *Parallel Computing*

Numerical simulations of electromagnetic or fluid-flow phenomena are most often constrained in their complexity by available computational processing capability. As mentioned in Chapter 1, massively parallel machines have recently begun to displace vector computers at the pinnacle of computing capability. Unfortunately, efficient utilization of these machine requires much more effort on the part of the algorithm designer since, at the time of this study, compilers are unable to fully extract the parallelism inherent in a computer code and properly map that parallelism to a distributed-memory environment. One technique that has demonstrated potential in realizing parallelism for typical grid-based scientific applications is that of domain decomposition.

2.2.1 Achieving Parallelism Through Domain Decomposition

Domain decomposition is simply the act of partitioning the computational domain into a set of sub-domains and assigning the sub-domains to the available computational processors. For a computational domain consisting of a single structured grid, the decomposition is usually performed by dividing the grid along constant-coordinate lines. Using this technique, the issue of domain decomposition has been examined from an efficiency standpoint by Blosch and Shyy [23] and by Wong, et al. [137]. Both of these works utilized schemes for solving the Navier-Stokes equations. The work of Blosch and Shyy used a semi-implicit, pressure-based algorithm and was conducted on the CM-2 and MP-1 machines. Because the CM-2 and MP-1 are older single-instruction, multiple-data (SIMD) machines, it is difficult to extend their findings to more modern multiple-instruction, multiple-data (MIMD) machines. The work of Wong, et al. investigated strip and patch-type decompositions on the Intel Gamma machine and found that patch-type decompositions resulted in performance superior to that of strip decompositions.

Unlike the structured-grid decomposition techniques described above, unstructured-grid decomposition techniques are inherently more complicated since unstructured grids lack definable coordinate directions. Nevertheless, unstructured-grid decomposition techniques have been analyzed more heavily than structured-grid methods [22,47,60,121], perhaps precisely because of their inherently greater complexity. This increased degree of complexity extends to overset grids as well. In fact, because of the need to interpolate information between the various sub-grids, overset grids possess data dependencies not found in single-grid problems–either structured or unstructured. Despite these unique data dependencies, overset-grid domain-decomposition techniques have been left almost completely unexplored.

2.2.2 Parallel FVTD Solver Development

Although the literature is replete with examples of parallel implementations of CFD solvers–a few of the better examples represented by the works of Yadlin and Caughey [139], Scherr [81], Stagg, et al. [112], and Drikakis, et al. [37]–substantially fewer examples exist for FVTD codes. Within the past two years, Shankar has devoted considerable effort to developing a parallel version of his Lax-Wendroff scheme [106,107]. His implementation is largely a port of his vector-machine code to a parallel environment rather than a ground-up parallel design [106]. Although he has achieved a degree of success, his published claims of only 1 to 2.5 percent communication overhead using realistic geometries are difficult to verify. Shang too has worked with several other researchers and published works which have made limited use of parallel machines [87,88,93,94]. Additionally, Ahuja and Long have implemented a staggered-grid FVTD algorithm on the IBM SP2 and the Thinking Machines CM-5 [3]. All of the works cited utilize domain decomposition to achieve parallelism. Shankar's implementation capitalizes on his use of multi-zone structured grids in that each processor operates on a single zone or a subset of zones. On the other hand, Shang, et al., and Ahuja and Long utilize a single-zone grid which is generally partitioned one dimensionally. Although these works provide a demonstration of the possibilities inherent in using massively parallel machines with FVTD solvers, there remains much work to do in the area of parallel FVTD algorithm development.

2.3 Overset Grids

2.3.1 Overset Grid Overview

Overset grids (also known as Chimera grids) were originally developed by Benek, et al. [11,12,13] to simplify the grid-generation process for complex geometries. In this approach, a complex configuration is divided into a series of component geometries, and a body-conformal structured grid is generated around each component. The resulting grids are allowed to arbitrarily overlap and thereby combined to form a globally unstructured grid with the requirement that all regions in the domain be included in at least one of the grids. Cells within each computational grid are typically classified as either *field*, *hole*, or *boundary* (sometimes called *fringe*) cells. Field cells are those cells which are updated via application of the governing equation or from physical boundary conditions; hole cells are those cells that are excluded from computation on a grid due to the fact that their physical location places them within a region that is either computed on another grid or excluded from the computational domain; and boundary cells are those cells that must be updated via interpolation from data contained on other grids. An example of the cell classifications is provided in Figure 2.3 for a simple two-grid problem consisting of a cylindrical grid embedded within a Cartesian background grid. For a finite-volume scheme, the cell centers are given by the intersections of the grid lines in the figure. Those cells which are updated via interpolation (the boundary cells) are clearly depicted. Furthermore, the hole cells of the background Cartesian grid are those cells that fall inside the cylinder body. If the grids are fixed in time, classification of the hole and boundary cells and calculation of the interpolation stencils can be accomplished as a preprocessing step prior to execution of the solver using the widely available program known as PEGSUS [116]. PEGSUS, like Chimera, was originally developed for the CFD community but is equally well suited for other grid-based methods such as the FVTD technique.

Because of the unique grid structure and interpolation requirement, the Chimera method poses unique design considerations when developing an algorithm for a distributed environment. Those considerations are discussed in the next section.

2.3.2 Parallel Overset-Grid Implementations

Because of the increase in popularity and availability of massively parallel computing machines and the prevalence of overset-grid implementations for CFD, it is no surprise that several researchers have devel-

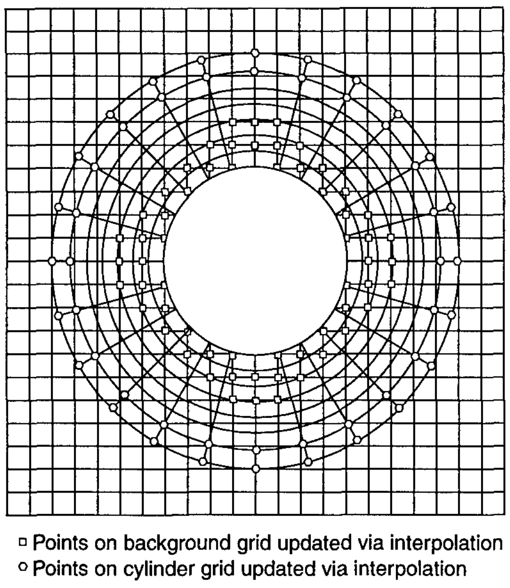

□ Points on background grid updated via interpolation
○ Points on cylinder grid updated via interpolation

Figure 2.3: Overset Grid Example

oped parallel overset-grid implementations for CFD solvers. To date, implementations have centered around partitioning each of the structured grids individually, and assigning the individual partitions to the processors in some fashion. Clearly, there are several possible levels of complexity with such an approach. From a more simplistic standpoint, Smith and Pallis simply assign each structured grid in its entirety to a processor [109]. Their approach is tailored to a distributed workstation environment. Progressing up a level of complexity, Weeratunga, Chawla, Barszcz, Ryan, and Meakin have performed high-quality work on an implicit Navier-Stokes overset-grid implementation [9,80,134,135]. In their approach, each structured grid is partitioned independently and assigned to a unique subset of the available processors. Although they report communication overhead of only 3 to 4 percent, it is believed that this choice of decomposition poses a limitation to parallel efficiency and scalability. Still more complex is the set of tools which are collectively known as DSK and were developed by Chessire and Naik [30]. DSK is designed to manage an overset-grid implementation in distributed parallel environments. Unfortunately, the tools are proprietary and not available to the public [71].

In all the cases described, partitioning of the structured grids is performed by dividing the grids along constant-coordinate lines thereby creating computational blocks which are hexahedral in shape. Although this is the simplest decomposition approach for this type of grid, it is not necessarily the most efficient[1].

1. See Chapter 5.

2.3.3 *Overset Grids with FVTD*

Most of the FVTD research thus-far accomplished has been performed using a computational domain comprised of a single structured grid (either staggered or collocated). As previously mentioned, Shankar's method differs slightly in that his group utilizes a multi-zone structured gridding approach which has certain flexibility advantages. The Rockwell group has also initiated an effort to develop an FVTD algorithm for unstructured grids [104,106,107]; however, results from that algorithm have been plagued by accuracy concerns [107].

Since certain electromagnetic problems are characterized by structures having radically different length scales (e.g. thin wires, slots, gaps, etc.), it has been necessary to develop modifications to structured grid algorithms so that sufficient mesh resolution is obtained while simultaneously maintaining computer requirements at acceptable levels. Several researchers [55,64,142] have consequently introduced modifications to the structured FDTD algorithm aimed at addressing these issues. In particular, the work of Zivanovic, et al. [142] utilizes a sub-gridding technique in which the structured grid is locally refined in areas where field variables exhibit rapid variations. Not surprisingly, the abrupt change in grid resolution causes some degradation of the solution accuracy [14].

All of the above-mentioned techniques use a single grid to solve the problem of interest. In 1992, Yee, et al. departed from this philosophy and applied the concept of overset grids to a staggered-mesh FDTD algorithm [141]. At the outset of the present study, this represented the only known published work using overset grids in a CEM environment. Although demonstrating the possibility of using overset grids with FDTD, the method of Yee, et al. is highly specialized and of limited value in a more generalized, integrated design environment.

2.4 *Chapter Summary*

A survey of the work previously accomplished by researchers in the fields of FVTD algorithm development, parallel algorithm development for CFD and CEM, and overset grids has been presented. Although a great deal of progress has been made in each of these disciplines, prior to this study there was no single work which incorporated all of these technologies into a unified approach for solving problems in CEM. The next chapter serves to discuss the governing equations and numerical techniques used in this work and provides the bridge between the background material presented to this point and the results-oriented portion of this document.

III. Governing Equations and Numerical Methodology

This chapter serves to introduce the equations and numerics used in the present study. To begin, the governing equations of electromagnetics–the Maxwell equations–are presented and cast into conservative form for a general curvilinear coordinate system. A description of the finite-volume procedure used to solve the conservative form of the equations follows. Because one of the focuses of this work involves the application of overset grids to the FVTD methodology, the numerical issues particular to Chimera-grid implementations are then discussed. The presentation contained in this chapter is intended only to introduce the terminology and methods used throughout the remainder of this document; it is not intended as an in-depth presentation. A more thorough description of the FVTD process is provided in Appendix A.

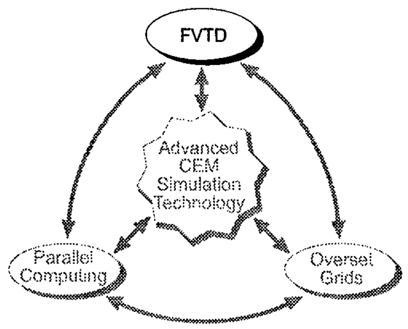


Figure 3.1: Chapter 3 Emphasis

3.1 The Maxwell Equations

The phenomena of electromagnetic wave propagation is modeled by the set of four equations collectively referred to as the Maxwell Equations. In differential vector form, these equations can be written as [8]

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{3.1}$$

$$\nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \tag{3.2}$$

$$\nabla \cdot \vec{D} = \rho \tag{3.3}$$

$$\nabla \cdot \vec{B} = 0 \tag{3.4}$$

where:

$\vec{B}$ = magnetic flux density (webers/square meter)

$\vec{D}$ = electric flux density (coulombs/square meter)

$\vec{H}$ = magnetic field intensity (amperes/meter)

$\vec{E}$ = electric field intensity (volts/meter)

$\vec{J}$ = conduction electric current density (amperes/square meter)

ρ = electric charge density (coulombs/cubic meter)

The flux densities can be related to the field intensities through the constitutive relations given by

$$\vec{D} = \varepsilon\vec{E} \text{ and } \vec{B} = \mu\vec{H} \tag{3.5}$$

where

ε = electric permittivity (farad/meter)

μ = magnetic permeability (henry/meter)

These constitutive relations, when applied in conjunction with equations (3.1) and (3.2), yield a system of six scalar equations for the unknown electromagnetic field components. By using the chain rule of differentiation, equations (3.1) and (3.2) can be transformed to a general (ξ, η, ζ) curvilinear coordinate space and written in conservative form as

$$\tilde{Q}_t + \tilde{U}_\xi + \tilde{V}_\eta + \tilde{W}_\zeta = \tilde{J} \tag{3.6}$$

where

$$\begin{aligned}
\tilde{Q} &= \vec{Q}/J \\
\tilde{U} &= (\xi_x\vec{U} + \xi_y\vec{V} + \xi_z\vec{W})/J \\
\tilde{V} &= (\eta_x\vec{U} + \eta_y\vec{V} + \eta_z\vec{W})/J, \\
\tilde{W} &= (\zeta_x\vec{U} + \zeta_y\vec{V} + \zeta_z\vec{W})/J \\
\tilde{J} &= \vec{J}/J
\end{aligned} \tag{3.7}$$

$$\vec{Q} = \begin{bmatrix} Bx \\ By \\ Bz \\ Dx \\ Dy \\ Dz \end{bmatrix} \quad \vec{U} = \begin{bmatrix} 0 \\ -Dz/\varepsilon \\ Dy/\varepsilon \\ 0 \\ Bz/\mu \\ -By/\mu \end{bmatrix} \quad \vec{V} = \begin{bmatrix} Dz/\varepsilon \\ 0 \\ -Dx/\varepsilon \\ -Bz/\mu \\ 0 \\ Bx/\mu \end{bmatrix} \quad \vec{W} = \begin{bmatrix} -Dy/\varepsilon \\ Dx/\varepsilon \\ 0 \\ -By/\mu \\ -Bx/\mu \\ 0 \end{bmatrix} \quad \vec{J} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -Jx \\ -Jy \\ -Jz \end{bmatrix}, \tag{3.8}$$

and

$$J = \begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} \tag{3.9}$$

In the above expressions, terms of the form ξ_x are known as the *metrics* of the coordinate transformation while J represents the *Jacobian* of the coordinate transformation.

Although equation (3.6) was derived for the *total fields*, it can be shown[1] that the *scattered-field formulation*–a form more suitable for solution via numerical means–is identical in form. Thus, equation (3.6) represents the final form of the governing equations which are to be solved numerically.

3.2 Finite-Volume Procedure

Equation (3.6) is solved in the present work via a collocated, cell-centered finite-volume formulation which, for a general time-invariant hexahedral cell, can be written as

$$\tilde{Q}_t V + \sum_{k=1}^{6} \tilde{F}_k \cdot d\vec{A}_k - \tilde{J} V = 0 \tag{3.10}$$

where $\tilde{F} = \tilde{U}\hat{\xi} + \tilde{V}\hat{\eta} + \tilde{W}\hat{\zeta}$, $d\vec{A}_k$ is a vector normal to face k with a magnitude equal to the area of the face, and V is the volume of the cell.

Because the focus of this work did not include developing a new FVTD core solver, a numerical scheme developed by Shang, et al. [86,90] was selected for implementation. That scheme treats the flux and time integration terms appearing in equation (3.10) using a flux-vector-splitting scheme in conjunction with a Runge-Kutta time integration procedure. Each of these numerical aspects is treated briefly in the following sections.

3.3 Flux-Calculation Procedure

Calculation of the flux term appearing in equation (3.10) is accomplished via a flux-vector-splitting approach after Steger and Warming [113]. As an example of this procedure, for an arbitrary cell face of con-

1. See Appendix A.

stant ξ (denoted as face $i + 1/2$), the flux vector is written as

$$\tilde{F}_{i+1/2} = \tilde{U}(\tilde{Q}_{i+1/2}) = \tilde{U}^{+}(\tilde{Q}^{L}_{i+1/2}) + \tilde{U}^{-}(\tilde{Q}^{R}_{i+1/2}) \tag{3.11}$$

where the superscripts L and R refer to the left and right states of the dependent variables at cell face $i + 1/2$. Because the dependent variables are defined with respect to cell centers rather than at cell faces, information is extrapolated via a *Monotone Upstream Centered Schemes for Conservation Laws* (MUSCL) approach after van Leer [122] to the cell faces. This extrapolation approach can be written as

$$\begin{aligned} \tilde{Q}^{L}_{i+1/2} &= \tilde{Q}_i + \frac{\phi}{4}[(1-\kappa)(\tilde{Q}_i - \tilde{Q}_{i-1}) + (1+\kappa)(\tilde{Q}_{i+1} - \tilde{Q}_i)] \\ \tilde{Q}^{R}_{i+1/2} &= \tilde{Q}_i - \frac{\phi}{4}[(1+\kappa)(\tilde{Q}_{i+1} - \tilde{Q}_i) + (1-\kappa)(\tilde{Q}_{i+2} - \tilde{Q}_{i+1})] \end{aligned} \tag{3.12}$$

For this work, κ and ϕ were set to one-third and one, respectively, resulting in a spatially third-order-accurate scheme.

Although the exact flux-splitting procedure is not discussed here, it is possible to generate a locally orthogonal coordinate system at each cell face and split the fluxes in that coordinate system rather than in the general (ξ, η, ζ) system. This approach has been examined by Anderson, et al. [5] and by Shang and Gaitonde [86] but was not adopted here after timing studies conducted on RISC-based parallel computers revealed that such a splitting approach required 40% greater run times than simply splitting the fluxes in the general curvilinear coordinate system. A detailed explanation of the flux-vector-splitting scheme and the entire FVTD methodology is contained in Appendix A.

3.4 Time-Integration Procedure

Once the numerical fluxes at cell faces are determined via the procedure described in the preceding section, the solution is numerically advanced in time via a single-step, two-stage, second-order-accurate Runge-Kutta procedure given by

$$\begin{aligned} \tilde{Q}^{0} &= \tilde{Q}^{n} \\ \tilde{Q}^{1} &= \tilde{Q}^{0} - \frac{\Delta t}{V}(R(\tilde{Q}^{0})) \\ \tilde{Q}^{2} &= \tilde{Q}^{0} - \frac{\Delta t}{2V}(R(\tilde{Q}^{0}) + R(\tilde{Q}^{1})) \\ \tilde{Q}^{n+1} &= \tilde{Q}^{2} \end{aligned} \tag{3.13}$$

where the superscripts n and $n + 1$ denote the solution time level, Δt represents the time step, and R represents the residual which is formed from the cell fluxes and can be written as

$$R(\tilde{Q}^k) = (\tilde{U}_2(\tilde{Q}^k) - \tilde{U}_1(\tilde{Q}^k) + \tilde{V}_4(\tilde{Q}^k) - \tilde{V}_3(\tilde{Q}^k) + \tilde{W}_6(\tilde{Q}^k) - \tilde{W}_5(\tilde{Q}^k)) \tag{3.14}$$

where the numerical subscripts denoted the appropriate cell face ($1 = \xi_{min}$, $2 = \xi_{max}$, etc.).

For the present study, all simulations were performed in source-free regions (i.e., $\vec{J} = 0$). Thus, the two time-integration stages are given by

Stage 1:

$$\tilde{Q}^1 = \tilde{Q}^0 - \frac{\Delta t}{V}(\tilde{U}_2(\tilde{Q}^0) - \tilde{U}_1(\tilde{Q}^0) + \tilde{V}_4(\tilde{Q}^0) - \tilde{V}_3(\tilde{Q}^0) + \tilde{W}_6(\tilde{Q}^0) - \tilde{W}_5(\tilde{Q}^0)) \tag{3.15}$$

Stage 2:

$$\tilde{Q}^2 = \frac{1}{2}\left(\tilde{Q}^0 + \tilde{Q}^1 - \frac{\Delta t}{V}(\tilde{U}_2(\tilde{Q}^1) - \tilde{U}_1(\tilde{Q}^1) + \tilde{V}_4(\tilde{Q}^1) - \tilde{V}_3(\tilde{Q}^1) + \tilde{W}_6(\tilde{Q}^1) - \tilde{W}_5(\tilde{Q}^1))\right) \tag{3.16}$$

3.5 Boundary Conditions

With the flux and time-integration terms suitably treated, only the surface and far-field boundary conditions need be specified in order to solve the discretized system. These conditions are described in the following sections.

3.5.1 Surface Condition

All surfaces in this project are modeled as perfectly electrically conducting (PEC) material. Since no time-varying fields can exist within a PEC material [74], the wall or surface boundary conditions can be written as

$$\hat{n} \times \vec{E}_t = 0 \tag{3.17}$$

$$\hat{n} \cdot \vec{B}_t = 0 \tag{3.18}$$

$$\hat{n} \cdot \nabla(\hat{n} \times \vec{B}_t/\mu) = 0 \tag{3.19}$$

$$\hat{n} \cdot \nabla(\hat{n} \cdot \vec{E}_t/\varepsilon) = 0 \tag{3.20}$$

where the subscript t refers to the total field values and it is understood that all evaluations are performed at the scattering surface. Equations (3.17) and (3.18) are derived from control-surface arguments using the Maxwell equations. Equations (3.19) and (3.20), on the other hand, are extrapolation conditions developed by Shang and Gaitonde [89] in order to provide the requisite number of conditions at the scattering surface.

3.5.2 Far-Field Condition

The far-field boundary condition is simply a statement that the scattered field must be outwardly-propagating at the edge of the computational domain. This condition is numerically implemented by setting the incoming flux component to zero at the cell faces which form the periphery of the computational space.

3.6 RCS-Calculation Procedure

3.6.1 RCS Integral

The ultimate objective of most scattering simulations is the determination of the RCS of the body. For a three-dimensional target, the RCS, σ, is defined as [61]

$$\sigma = \lim_{R \to \infty} 4\pi R^2 \left| \frac{\vec{E}_s^2}{\vec{E}_i^2} \right| \tag{3.21}$$

where $\vec{E}_s$ and $\vec{E}_i$ are the scattered and incident fields and R is the magnitude of a vector which originates on or near the scattering object and terminates at the observation point. In the far zone (where R is assumed very large), the Stratton-Chu integral equation [35] for the scattered electric field becomes

$$\vec{E}_s\Big|_{R \to \infty} = \frac{jke^{jkR}}{4\pi R} \iint_S [\sqrt{\mu/\varepsilon}(\hat{n} \times \vec{H}_t) - (\hat{n} \times \vec{E}_t) \times \hat{r} - (\hat{n} \cdot \vec{E}_t)\hat{r}] e^{-jk(\hat{r} \cdot \vec{R}')} dS \tag{3.22}$$

where

$$k = \omega\sqrt{\mu\varepsilon} \tag{3.23}$$

and ω and k are the angular frequency and wavenumber of the incident field, respectively. In equation (3.22), S is an appropriate surface over which the integration takes place. Unless otherwise noted, for the present work, S is always defined as the scattering surface. A pictorial description of the vectors $\vec{R}$, $\hat{r}$, $\vec{R}'$, and $\hat{n}$ is given in Figure A.5.

3.6.2 Fourier Transform

Because RCS is a frequency-domain response, the electromagnetic field variables for the cells comprising the surface S described in the previous section must be transformed to the frequency domain using a Fourier transform defined by [131]

$$Q(\omega) = \frac{1}{N}\sum_{n=0}^{N}(Q(t))e^{-j\omega n\Delta t} \qquad (3.24)$$

where N is the number of time steps over which the Fourier sampling takes place. Note that N must equate to an integral number of excitation periods. Because the total fields appear in equations (3.22) and the dependent variables in the FVTD scheme are scattered field quantities, the incident field must be added to the scattered field during the sampling. Fourier sampling should not begin until surface current start-up transients have diminished. Experience gained throughout this work showed that waiting approximately two periods of wave motion was sufficient to produce accurate results. Further details of the Fourier sampling requirements are contained in Chapter 6.

3.7 Interpolation-Stencil-Calculation Procedure

As mentioned in Section 2.3.1, cells which are updated via interpolation are identified prior to execution of the FVTD algorithm by the program known as PEGSUS[1] which is maintained by NASA Ames Research Center. Given an overset computational grid, PEGSUS computes the interpolation distances *dx*, *dy*, and *dz* (as shown in Figure 3.2) by using an isoparametric mapping of the computational space. In the figure, the center of the cell which is updated via interpolation is represented by the dark spot. In addition, the vertices of the cube which are numbered 1 though 8 represent the centers of the cells which must supply interpolation data. Using the interpolation distances computed by PEGSUS, a boundary cell is updated via [13]

$$Q_i = a1 + a2dx + a3dy + a4dz + a5dxdy + a6dxdz + a7dydz + a8dxdydz \qquad (3.25)$$

where Q_i is the dependent variable vector associated with a given boundary cell, and

1. PEGSUS version 4.1_28 was used exclusively in this work

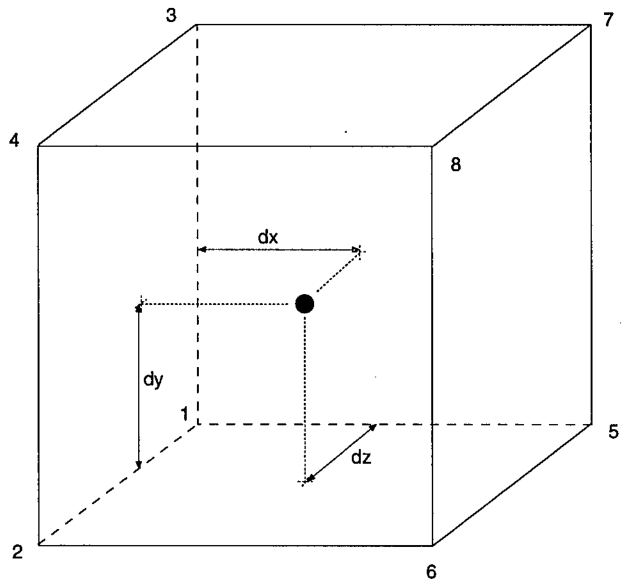


Figure 3.2: Interpolation Stencil Determination

$$
\begin{aligned}
a1 &= Q1 \\
a2 &= Q1 + Q5 \\
a3 &= -Q1 + Q3 \\
a4 &= -Q1 + Q2 \\
a5 &= Q1 - Q2 + Q6 - Q5 \\
a6 &= Q1 - Q3 - Q5 + Q7 \\
a7 &= Q1 - Q2 - Q3 + Q4 \\
a8 &= -Q1 + Q5 - Q6 + Q2 + Q3 - Q7 + Q8 - Q4
\end{aligned}
\tag{3.26}
$$

In equation (3.26), terms of the form QX represent the dependent-variable vectors associated with the cell which is denoted as vertex X in the interpolation cube shown in Figure 3.2.

3.8 Numerical-Solution Procedure

The numerical procedure of the overset grid/FVTD algorithm described above can be summarized as follows:

1. Solve the discretized form of the Maxwell equations (embodied in equation (3.10)) over all the interior cells of the computational domain using the flux-vector-splitting/Runge-Kutta

methodology given in equations (3.11) - (3.16).

2. Update the surface boundary cells using equations (3.17) - (3.20).
3. Update the boundary cells (not to be confused with cells updated via the surface boundary condition) using equations (3.25).
4. Perform the Fourier transform on the computed fields over an appropriate surface, S, (after a two-period start-up interval) using equation (3.24).

Repeat steps 1-4 over one period of wave motion. After each period,

5. Compute the RCS for the desired look angles using equations (3.21) and (3.22) applied to a suitable surface in the computational domain.

Repeat steps 1-5 until the simulation is complete.

3.9 Chapter Summary

This chapter served as an overview of the governing equations of electromagnetics and the overset grid/FVTD numerical procedure. Parallel computing issues are deferred to the next two chapters where they are presented in the context of studies centering on parallel algorithm development for both single- and overset-grid-based FVTD techniques.

IV. Single-Grid Domain-Decomposition Study

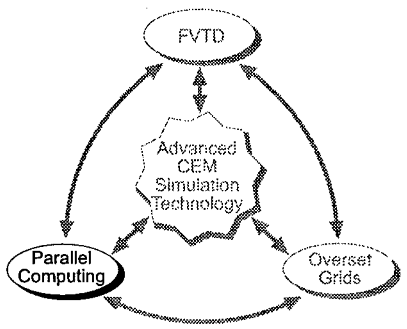


Figure 4.1: Chapter 4 Emphasis

This chapter begins the focus on the parallel-computing aspects of this work. Specifically, domain-decomposition strategies for solving hyperbolic systems of PDEs on distributed-memory parallel computing platforms are presented. To begin, several domain-decomposition approaches for single structured grids are developed, and the communication requirements associated with each decomposition method for a generic, explicit hyperbolic PDE solver are discussed. Based on these communication requirements, theoretical parallel performance is predicted using a widely accepted communication model. The performance model is then applied to a single-grid FVTD solver used to compute the electromagnetic fields within a rectangular waveguide. Predictions are compared to timing measurements made on three different parallel architectures, and the relation between processor-connection topology and message-passing performance are identified.

4.1 Domain Decomposition of Structured Grids

As mentioned in Chapter 1, modern distributed-memory parallel architectures are capable of performance exceeding that of vector supercomputers. Unfortunately, efficient utilization of these parallel machines requires much more effort on the part of the algorithm designer since compilers are not yet able to fully extract the parallelism inherent in a computer code and properly map that parallelism to a distributed-memory environment.

One method of potentially achieving a high degree of concurrency in typical grid-based scientific computations such as those found in CFD or time-domain CEM algorithms is to divide the computational domain into sub-domains or blocks and then assign the blocks to the available processors. Since determining an optimal decomposition and assignment is in problem space NP-Complete [39], no such attempt is made here. Instead, the computational domain is partitioned along constant coordinate lines, either one, two, or three dimensionally as shown in Figure 4.2. This approach has two primary advantages: first, it is easy to

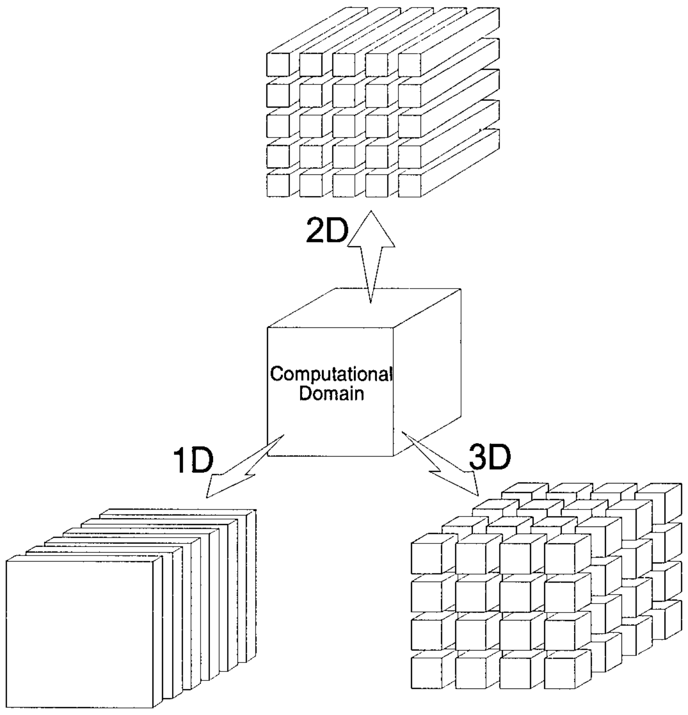


Figure 4.2: Single-Grid Domain-Decomposition Approaches

implement such a decomposition, and second, the resulting partitions are logically hexahedral. The second advantage facilitates a ready incorporation within the existing framework of many legacy engineering computer codes that are heavily loop based and thus assume block-type data structures. For these reasons, this type of decomposition is often employed since it facilitates a relatively straightforward conversion path from a traditional vector serial code to a parallel implementation.

Despite these advantages, this type of decomposition has its drawbacks, primarily with respect to *load balancing*. A computational decomposition is said to be load balanced if all processors perform the same amount of computational work. Unfortunately, a block-type decomposition can lead to an unequal distribution of the computational workload since it is often difficult to partition the computational domain so that all processors operate on the same number of computational cells. This is easily illustrated by assuming the computational domain consists of 1000 cells with 10 cells in each of the three coordinate directions. The domain can be partitioned into eight 5 x 5 x 5 blocks for eight processors, but cannot be partitioned into sixteen blocks so that all processors receive the same number of grid cells. Unless the computational grid is

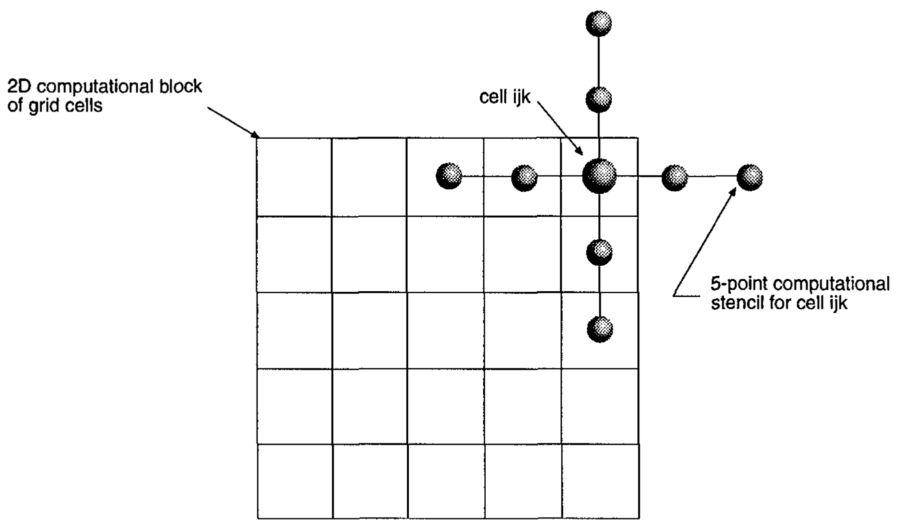

2D computational block of grid cells
cell ijk
5-point computational stencil for cell ijk

Figure 4.3: Computational Stencil on Computational Block Boundary

specifically designed for the decomposition (an approach which is not advisable since the grid design should be dictated by problem physics rather than by decomposition considerations), then load imbalance can become a serious drawback with this type of partitioning scheme. In the academic exercise presented throughout the remainder of this chapter, however, the computational domain is specifically constructed so that all processors receive the same number of grid cells. This greatly simplifies the analysis for the theoretical parallel performance of the general hyperbolic PDE solver presented in the next section.

4.2 A Generalized Communication Model for Single-Grid, Explicit, Hyperbolic PDE Solvers

4.2.1 Data Dependencies for an Explicit Hyperbolic PDE Solver

Given an arbitrary cell (denoted by *ijk*) in the computational domain, that cell's data dependencies are contained in its *computational stencil.* The computational stencil for cell *ijk* is simply the set of cells which affect the solution of cell *ijk* at the next time level. A two-dimensional example of a 5-point computational stencil (in each coordinate direction) for cell *ijk* appears in Figure 4.3. The size and arrangement of a computational stencil varies depending on the desired accuracy and traits of the numerical scheme, but all explicit schemes share the trait that all cells in the computational domain can be updated simultaneously to the new time level using only previous-time-level information from the cells in the computational stencils. In this figure, cell *ijk* lies on the border of a block of grid cells formed from the domain decomposition. Clearly, certain cells in the computational stencil reside in blocks different than that of cell *ijk*. If these blocks are assigned to different processors, then the data must be transferred via interprocessor communication. It is the transfer of information that forms the crux of the parallel-performance model presented in the following section.

4.2.2 The Communication Model

Based on the data dependencies described in the previous section, the theoretical parallel performance of a general hyperbolic PDE solver can be determined. Parallel run time, T_{par}, can be assumed to consist of calculations, T_{calc}, and parallel overhead, $T_{overhead}$, i.e.

$$T_{par} = T_{calc} + T_{overhead} \tag{4.1}$$

The overhead consists of several factors including communication and any extra calculations necessary to implement the code on a parallel machine. For this analysis, the parallel overhead is assumed to be dominated by the communication time, T_{comm}. If the classical cut-through-routing communication-cost model of Kumar, et al. [63] is used, then the communication time for a single message to travel between processors *a* and *b*, t_{comm}, is given by

$$t_{comm} = t_s + mt_w + lt_h \tag{4.2}$$

where t_s is the message start-up time, t_w is the per-byte transfer time, t_h is the latency associated with a hop between two processors, *m* represents the number of bytes transferred, and *l* represents the number of switch hops the message must make in order to travel from processor *a* to processor *b*. In most modern parallel architectures, the per-hop time is extremely small and all processors can be considered computationally close. This allows the communication cost model to be simplified [39], viz.

$$t_{comm} = t_s + mt_w \tag{4.3}$$

Although other communication models have been proposed and utilized [1,39,53], equation (4.3) is widely accepted within parallel-computing circles. The remainder of this section is therefore devoted to determining the theoretical parallel efficiency of a general hyperbolic PDE solver based on the communication model stated in equation (4.3).

For this general analysis, the computational domain is assumed to be k-dimensional and to consist of n cells with $n^{1/k} \equiv R$ cells distributed along each of the k coordinate directions. Furthermore, the domain is assumed to be evenly divided among the number of available processors, p. Thus, the number of grid cells residing on each processor, n_p, is given by

$$n_p = R^k/p \tag{4.4}$$

Referring again to Figure 4.3, the length of any interprocessor messages is simply the product of the number of cells on the face of a cut, some number of grid planes, τ (which is algorithm specific), and the number of bytes of information per cell. From Figure 4.3, τ is usually bounded above by the computational stencil half-width, ζ, which can be defined as

$$\zeta \equiv \lfloor (sw - 1)/2 \rfloor \tag{4.5}$$

where sw is the computational stencil width. Thus, the message length is given by

$$m = \tau D R^{k-1} p^{(1-i)/i} \tag{4.6}$$

where D represents the number of storage bytes required per grid point for the dependent variable data, τ represents the number of grid planes to transfer, and i represents the dimensionality of the decomposition, $0 \le i \le k$. Note that for $i = 0$, the computational domain is not decomposed, and thus $i = 0 \Leftrightarrow p = 1$. In all cases, the term $p^{(1-i)/i}$ is restricted to integral values.

With the message length determined, the total communication time can be calculated as the product of the communication time per message and the number of messages required per iteration, viz.

$$T_{comm} = \eta(t_s + (\tau D R^{k-1} p^{(1-i)/i}) t_w) \tag{4.7}$$

where η is the number of messages required by the chosen numerical scheme per time step. Substituting the communication time into equation (4.1) yields a general expression for the theoretical parallel run time, namely

$$T_{par} = t_c(R^k/p) + \eta(t_s + (\tau D R^{k-1} p^{(1-i)/i}) t_w) \tag{4.8}$$

where t_c is the single-processor computation time per grid cell per time step.

With parallel run time determined, theoretical absolute speedup, S_a, and absolute efficiency, E_a–two of the most common metrics for parallel performance–can be determined using the standard definitions [63]

$$S_a = T_1/T_{par} \tag{4.9}$$

$$E_a = S_a/p \tag{4.10}$$

where T_1 is the run time for the best-known serial algorithm to solve the problem in question. It is often not practical to compare the chosen parallel algorithm against the best-known serial algorithm, and consequently, relative speedup, S, and efficiency, E, are often used whereby the parallel algorithm run time is compared against the run time of the algorithm on a single processor. Setting $T_{comm} = 0$ and $p = 1$ in equation (4.8) to determine T_1, the theoretical relative performance is thus given by

$$S = \frac{1}{\frac{1}{p} + \frac{\eta}{R}[R^{1-k}(t_s/t_c) + \tau D R^{k-1} p^{(1-i)/i}(t_w/t_c)]} \tag{4.11}$$

$$E = \frac{1}{1 + \frac{\eta}{R}[pR^{1-k}(t_s/t_c) + \tau D p^{1/i}(t_w/t_c)]} \tag{4.12}$$

Because parallel machines allow one to solve large problems using a large number of processors, it is useful to conduct an asymptotic analysis of equation (4.12) in which p and R are assumed to be arbitrarily large. To assess the dominant term appearing in the denominator of equation (4.12), it is necessary to compare the terms p/R^k and $p^{1/i}/R$. To do so, it can be noted that

$$p^{1/i} \geq p^{1/(i+1)} \quad \forall p, i \geq 1 \tag{4.13}$$

Since $i \leq k$, then from equation (4.13) the term $p^{1/i}/R$ must always dominate the term p/R^k if $p^{1/k}/R \geq p/R^k$ for any allowable values of p, k, and R. However, from equation (4.4),

$$p/R^k = 1/n_p \tag{4.14}$$

and since

$$p^{1/k}/R = (1/n_p)^{1/k} \tag{4.15}$$

then $p^{1/k}/R \geq p/R^k$. Of course, this assumes that $n_p \geq 1$ which follows naturally from the definition of n_p. Thus, provided the constants appearing with each of the terms can be assumed to be on the same order of magnitude, then the asymptotic analysis yields

$$E = \frac{1}{1 + \Theta(p^{1/i}/R)} \tag{4.16}$$

where the Θ notation indicates a tight upper bound.

It is important to note that the asymptotic analysis encapsulated in (4.16) is taken for p and R appropriately large when in fact, for a practical implementation, either variable may be restricted by memory or machine architecture. Consequently, a consideration of the constants appearing in (4.11) and (4.12) may be necessary when assessing true performance of the algorithm on the machine of implementation.

It is often desirable to know how the parallel performance of an algorithm scales as the problem size increases. This can be measured by examining the amount by which the problem size must be increased as the number of processors is increased in order to keep a constant efficiency. This is termed *isoefficiency* [45]. Clearly, in order to keep a constant efficiency in equation (4.16), then

$$R = n^{1/k} = \Theta(p^{1/i}) \Rightarrow n = \Theta(p^{k/i}) \tag{4.17}$$

Thus, in order to maintain a constant parallel efficiency, a computational domain that is decomposed with the same dimensionality as the problem space need only be increased in size by an amount linearly proportional to increasing processor number. This represents optimal isoefficiency [45]. Lower dimensional decompositions require larger increases in the size of the computational domain as increasing numbers of processors are applied to the problem if the efficiency is to be maintained at a constant value.

4.3 Parallel Performance Analysis of the Single-Grid FVTD Algorithm

4.3.1 Implementation Details

4.3.1.1 Model Problem

The problem selected for examining the various domain-decomposition approaches was the computation of the electromagnetic fields inside a rectangular waveguide as depicted in Figure 4.4. The computational domain for the problem was uniform and Cartesian, thereby facilitating a relatively straightforward partitioning. The physical dimensions a, b, and L of the waveguide were scaled to π, π, and 1 respectively, and the waveguide was excited in a TMZ_{11} mode. This problem has been extensively studied by Shang ([87], [91]), and the existence of an analytical solution [111] allows for a ready verification of parallel program correctness.

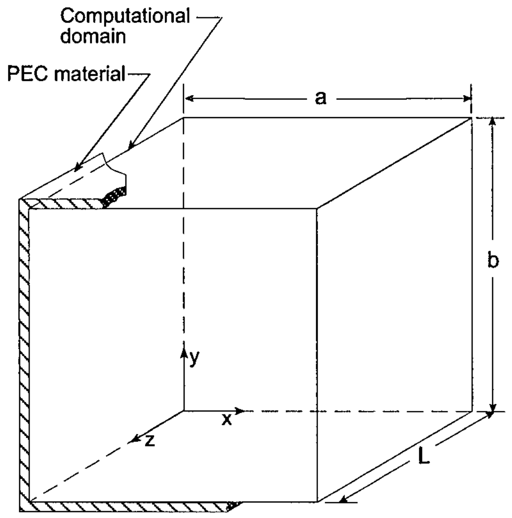


Figure 4.4: Rectangular Waveguide Geometry

4.3.1.2 Flux Message Details

The MUSCL scheme described in Section 3.3 forms the basis for the data dependencies of the FVTD algorithm. Referring to Figure 4.5, in order to compute the total flux at cell face $i + 1/2$, $\tilde{U}_{i+1/2}$, the positive flux component requires dependent variable information from cells $i-1$, i, and $i+1$. Similarly, the negative flux component requires information from cells i, $i+1$, and $i+2$. In a parallel environment, one or more of these cells may reside on different processors, and thus a means must exist to transfer necessary information between processors. The transfer of information is facilitated through the use of buffer storage locations. These locations serve to hold dependent variable or other information that is computed on one processor but necessary for other computations on another. A sample of the buffer locations is shown as the shaded cells in the lower portion of Figure 4.5. Using this approach for the simple two-processor, one-dimensional example shown in the figure, the flux calculation for a cell which falls on a boundary created by the domain decomposition begins as processors 1 and 2 independently compute the positive and negative flux component, respectively, for cell face $i + 1/2$. This calculation requires dependent-variable data stored in the buffer locations on each processor. Processor 1 passes the positive flux component to processor 2 which then computes the total flux for the cell face and returns this information to processor 1. Assuming the fluxes at the remaining cell faces have been calculated, the processors subsequently update cells i and $i+1$, respectively, to the new time level. For a higher-dimensional problem, the message-passing scenario is

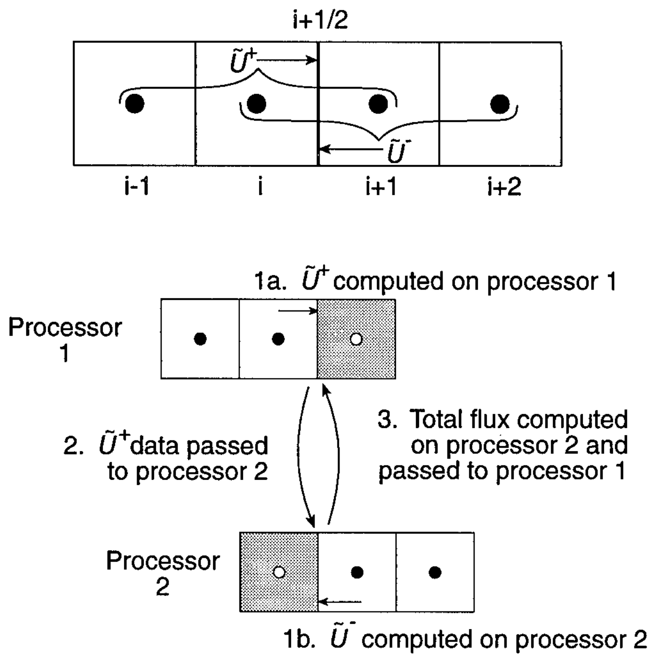

i+1/2
$\tilde{U}^+$
$\tilde{U}^-$
i-1
i
i+1
i+2
1a. $\tilde{U}^+$ computed on processor 1
Processor 1
2. $\tilde{U}^+$ data passed to processor 2
3. Total flux computed on processor 2 and passed to processor 1
Processor 2
1b. $\tilde{U}^-$ computed on processor 2

Figure 4.5: Flux Message Detail

repeated for each of the decomposition directions. No messages are required along a coordinate direction in which the computational domain is not decomposed. Once all cells have been updated to the new time level, dependent-variable information is exchanged so the buffer storage locations contain new-time-level data. This procedure thus requires a total of eight message sends and eight message receives along each coordinate direction that is decomposed. Furthermore, the number of grid planes transferred in each message is one. Using this algorithm, the theoretical speedup and efficiency given by equations (4.11) and (4.12) become

$$S = \frac{1}{\frac{1}{p} + \frac{16i}{R}[R^{-2}(t_s/t_c) + Dp^{(1-i)/i}(t_w/t_c)]} \tag{4.18}$$

$$E = \frac{1}{1 + \frac{16i}{R}[pR^{-2}(t_s/t_c) + Dp^{1/i}(t_w/t_c)]} \tag{4.19}$$

It is possible to double the size of the buffers and thereby obviate the need for either processor to exchange any flux data; however, this approach was examined and discarded since the storage penalty exacted was not offset by any significant performance gain.

4.3.1.3 Computer Code Development

In order to test the theoretical results obtained in Sections 4.2.2 and 4.3.1.2, a single-grid, parallel, 3-D, FVTD computer code (designated "SIGMA"[1]) was constructed. At the heart of the code is the flux-vector-splitting/Runge-Kutta (FVSRK2) algorithm discussed in Chapter 3 and Appendix A. SIGMA allows for any of the decompositions illustrated in Figure 4.2 and is written in C to take advantage of that language's dynamic-memory-allocation capabilities. This allows the number of processors and the decomposition approach to be selected at run time, thus providing a very flexible environment in which to conduct a domain-decomposition study.

4.3.1.4 Target Architectures and Message-Passing Implementation

The parallel machines used to complete the single-grid domain-decomposition study were the Maui High Performance Computing Center (MHPCC) IBM SP2, the Eglin Air Force Base (AFB) Cray T3D, and the Wright Patterson AFB (WPAFB) Intel Paragon. Although several message-passing libraries were available for each machine, the libraries used in this study were chosen based on vendor support, portability, and performance. At the inception of the work, PVM for the T3D was actively supported by Cray Research [75]. Similarly, support and documentation for the Intel message-passing library, NX [73], was readily available. Finally, Message Passing Interface (MPI) [110] was selected for the SP2 due to its portability and because its performance has been shown to be nearly equal to that of IBM's native Message Passing Library (MPL) for point-to-point communications [138]. By abstracting the library-specific programming calls through the use of the macro feature of the C programming language, a single computer code was used to collect performance data on all three machines. A more detailed discussion of the machine architectures and the message-passing libraries used appears in Appendix B.

Because the computational domain was decomposed as shown in Figure 4.2, the communication model assumed that no block was allowed to have more than a single neighboring block along a given face. This leads to a relatively straightforward message-passing implementation. In the flux-calculation routines, all processors post non-blocking receives and pass flux information to left-hand neighbors as appropriate. After the left-hand messages are received, the process is repeated for the right-hand neighbors. This process is depicted in Figure 4.6.

1. *SI*ngle *G*rid for *M*IMD *A*rchitectures

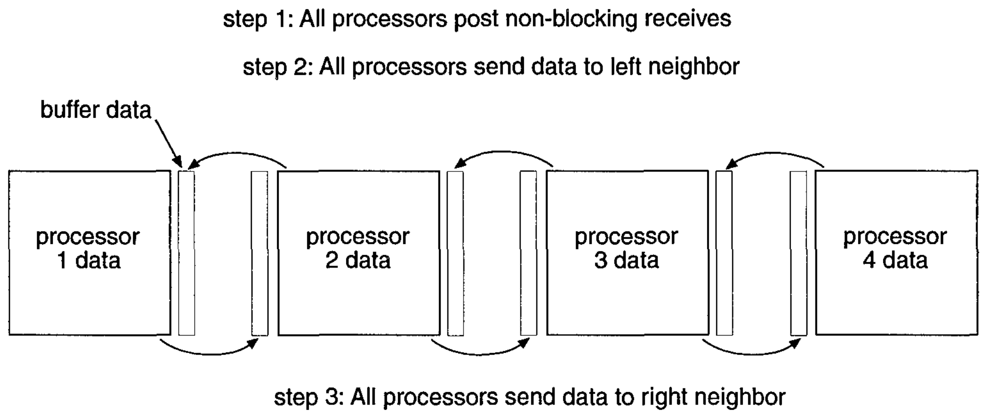

step 1: All processors post non-blocking receives
step 2: All processors send data to left neighbor
buffer data
processor 1 data
processor 2 data
processor 3 data
processor 4 data
step 3: All processors send data to right neighbor

Figure 4.6: Message-Passing Implementation Detail

4.3.2 Test Procedure

As is evident in the theoretical derivations, parallel speedup and efficiency are dependent on a variety of parameters including the problem size and the number of processors on which the algorithm is executed. In order to test these parameters, SIGMA was run for 100 iterations on grid sizes ranging from $R = 32$ to $R = 128$ using up to 128 processors. One-hundred iterations was deemed an acceptable number so that run times would not be excessive yet any transient parallel-environment effects such as inter-process message contentions would be suitably minimized. In every decomposition, the number of finite-volume cells residing on each processor was identical. This reduced, but did not completely eliminate, load imbalance since boundary condition cells required less computational effort than interior cells. The choice of domain decompositions and grid sizes was constrained primarily by the amount of memory available on each processor. This was especially true in the case of the Paragon which, with the exception of 16 MP nodes, possesses only 32 megabytes per processor. In addition to the memory constraint, additional restrictions were imposed by the processor allocation scheme of the T3D. The current version of the operating system on this machine allows processors to be allocated only in powers of two. This required that the computational domain be partitioned in powers of two along each decomposition direction.

In addition to the dimensionality of the decomposition, the directional dependence of the decompositions was assessed by permuting the decompositions for each coordinate direction. The permutations correspond to a re-orientation of the planes or lines of grid cells as depicted in Figure 4.2. Examination of the

Dimension of Partition	Grid Size (R)	Processors (P)	Decomposition*
1D	32, 64, 128	4, 8, 16, 32, 64, 128	$P \times 1 \times 1$ such that $P \leq R$
2D	32, 64, 96	4, 16, 64	$\sqrt{P} \times \sqrt{P} \times 1$
2D	32, 64, 96	8, 32, 128	$\sqrt{\frac{P}{2}} \times 2\sqrt{\frac{P}{2}} \times 1$
3D	32, 64, 96	8, 64	$\sqrt[3]{P} \times \sqrt[3]{P} \times \sqrt[3]{P}$
3D	32, 64, 96	16, 128	$\sqrt[3]{\frac{P}{2}} \times \sqrt[3]{\frac{P}{2}} \times 2\left(\sqrt[3]{\frac{P}{2}}\right)$
3D	32, 64, 96	32	$2 \times 4 \times 4$

*All permutations of the tabulated decompositions were performed. For example, in addition to the stated $P \times 1 \times 1$ one-dimensional partitioning, $1 \times P \times 1$ and $1 \times 1 \times P$ partitions were also examined. The decomposition nomenclature refers to the number of blocks by which the computational domain was partitioned in each coordinate direction.

Table 4.1: Summary of Examined Decompositions

decomposition along each direction allows for an assessment of machine memory-accessing performance and uncovers potential message-bus contentions which are the inevitable result of mapping a higher-dimensional physical problem onto a lower-dimensional interconnection network. Table 4.1 contains a summary of the decompositions examined. Rather than list each grid size, processor count, and decomposition separately, they are combined into a single listing whenever possible with the understanding that all permutations on a given row were examined.

As timing data was collected, several runs were re-accomplished to assess the repeatability of the timing data. In the case of the T3D and the Paragon, results were found to be repeatable to well within one percent. However, such was not the case for the SP2 where timing data varied much more dramatically. In order to ensure accurate results, all runs on the SP2 were accomplished at least five times and the minimum time observed was used as the reported execution time. This procedure is similar to that recommended by Think-

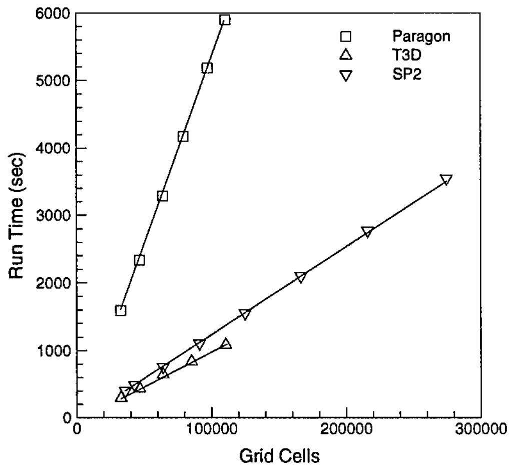


Figure 4.7: Single-Node Run Times

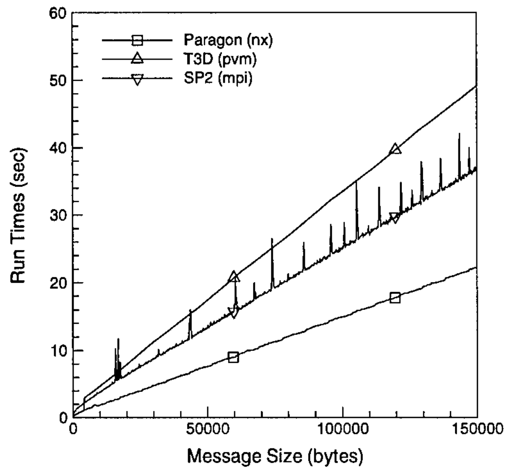


Figure 4.8: Ring Message-Passing Times
(1000 rings on 8 processors)

ing Machines in the timing studies conducted by Blosch and Shyy [23].

4.3.3 Machine Performance Characterization

Determination of theoretical performance as embodied in equations (4.18) and (4.19) requires that the values of t_c, t_s, and t_w be ascertained. In general, t_s and t_w are machine specific, while t_c depends both on the target architecture as well as the algorithm. In order to determine t_c, SIGMA was run for 100 time steps on each of the three machines on a single processor using a variety of grid sizes up to the maximum size containable in the machine's core memory. The times for these execution runs are contained in Figure 4.7. Although not presented here, a formal analysis of the FVTD algorithm run-time complexity reveals the single iteration run-time complexity is in time space $\Theta(n)$. This analysis is corroborated by the run times exhibited in the figure which show the execution times to be a linearly increasing function of n. Any slight deviations from linearity can be explained by noting that boundary condition cells require less computational work, and as the grid size decreases, the boundary condition cells have an increased effect on the algorithm run time. Using a simple least-squares fit of the data, t_c was determined from the slope of each plot.

The message-passing performance of each of the three machines was determined by configuring eight processors as a logical ring and circulating messages of varying size 1000 times around the ring. Run-time

	t_c (μsec)	t_s (μsec)	t_w (μsec)
T3D	102	25	.030
Paragon	546	36	.018
SP2	132	76	.040

Table 4.2: Machine and SIGMA Performance Parameters

data for this experiment appears in Figure 4.8 on the previous page. By performing a linear curve fit of the data, the message start-up time and per-word transfer time can be computed directly from the y-intercept and line slope. Although the linear curve-fitting process generated excellent results (goodness of fit > 0.99 in all cases), the SP2 is characterized by marked and somewhat-random variations in execution time. These variations are similar to those observed by Bokhari [24] and are most likely due to message contention over the omega switching network. Table 4.2 contains a summary of the findings of this portion of the study. The tabulated values agree in general with those of Foster [39]. Any discrepancies can be explained by differences in operating systems and message-passing libraries used.

4.3.4 Results

4.3.4.1 Electromagnetic Field Computations

Figure 4.9 contains a comparison of the magnitudes of the computed and exact magnetic fields inside the waveguide. The computed solution was obtained from a 32-node Paragon run using approximately 110,000 grid points with the computational domain partitioned three-dimensionally in a $4 \times 2 \times 4$ configuration. In the figure, the y and z axes have been scaled so that the cutting planes located at $y = 0.6$, 1.6, and 2.5 are unobstructed. The computed and exact solutions are in excellent agreement, and in fact, the plots are indistinguishable. While not a formal proof of correctness, it does indicate that the parallel algorithm functioned as intended.

4.3.4.2 Parallel Scalability

The parallel scalability results of the domain-decomposition studies appear in Figures 4.10 to 4.15. Figures 4.10 to 4.12 contain parallel speedup results for the one-, two-, and three-dimensional decomposi-

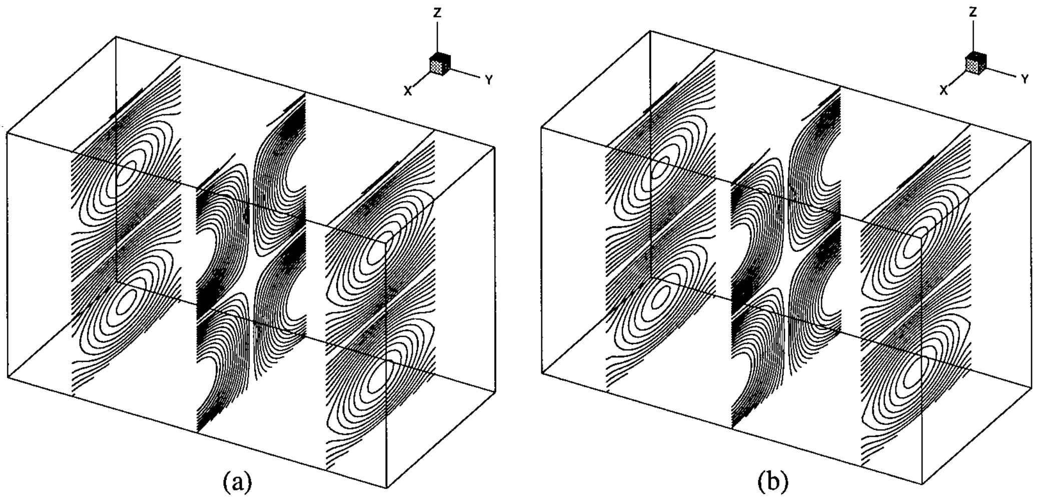
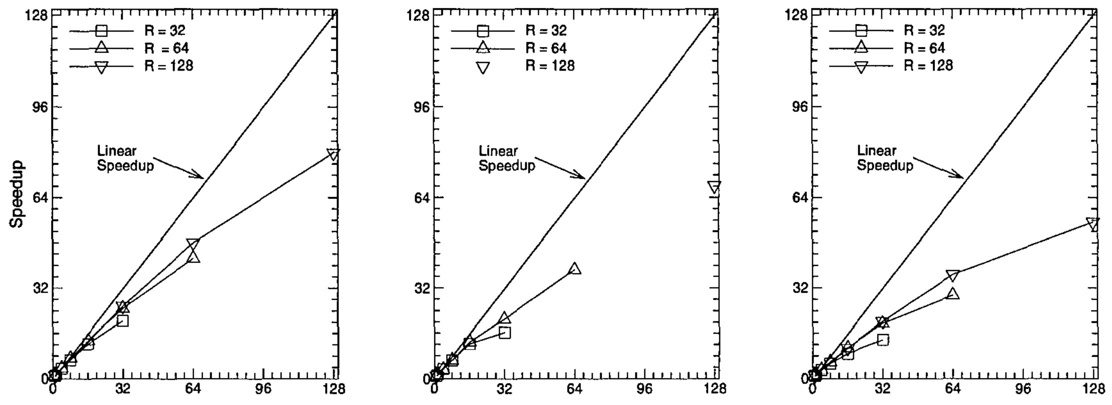


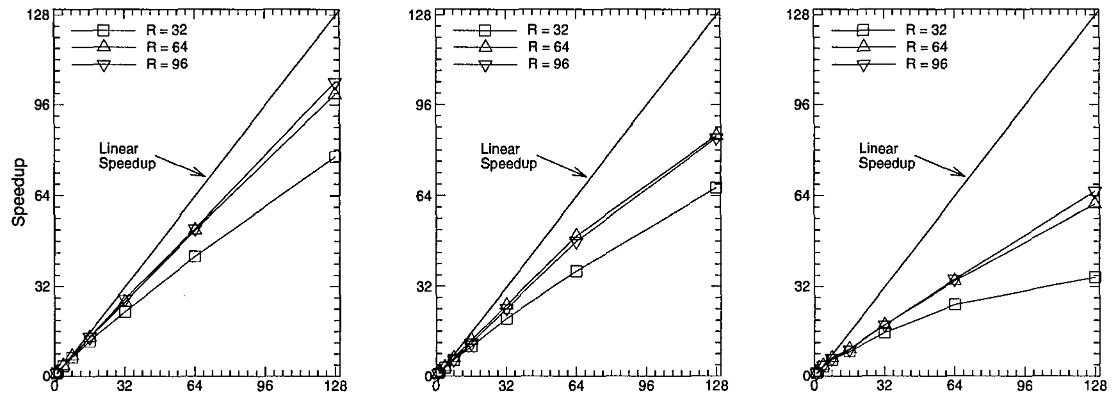
Figure 4.9: Waveguide Total-Magnetic-Field Contours: a) Exact, b) Computed

tions, respectively, while Figures 4.13 to 4.15 contain efficiency results. A comparison of the relative performance of the SIGMA code for a given decomposition dimensionality on each of the three machines can be conducted by examining sub-figures *a*, *b*, and *c* of a single figure. On the other hand, a comparison of the sub-figures in a given column facilitates an examination of the effects of the dimensionality of the decomposition for a given platform. It is apparent from the figures that in nearly every case, the two- and three-dimensional decompositions produced superior performance relative to the one-dimensional decompositions. Furthermore, three-dimensional decompositions exhibited slightly better performance in several instances on the T3D while two-dimensional decompositions tended to be slightly superior on the Paragon and noticeably superior on the SP2. This is especially evident in an examination of the efficiency curves of Figures 4.13 to 4.15. These trends in parallel performance correspond quite closely to the topologies of the three machines. The three-dimensional torus structure of the T3D allows each processor six neighboring processors while the two-dimensional mesh structure of the Paragon translates to at most four neighbors for a given computational node. In contrast, the omega network of the SP2 requires that any communication between two processors traverse at least one switch hop and two communication lines. It appears that superior parallel performance is achievable when there is a close agreement between the physical dimensionality of the domain decomposition and the physical topology of the architecture onto which it is mapped. It should be remembered that the speedup and efficiency curves of Figures 4.10 to 4.15 only provide one measure of the



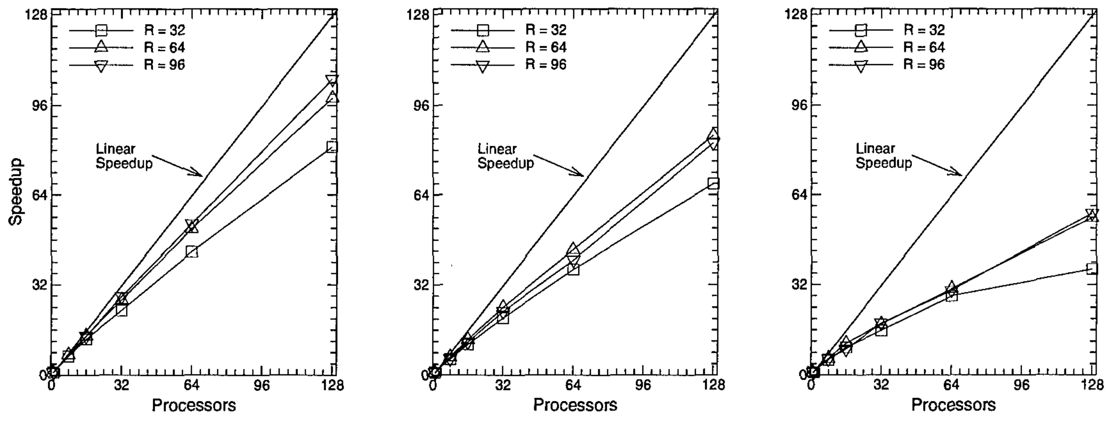
(a) (b) (c)

Figure 4.10: One-Dimensional Speedups: a) T3D, b) Paragon, c) SP2



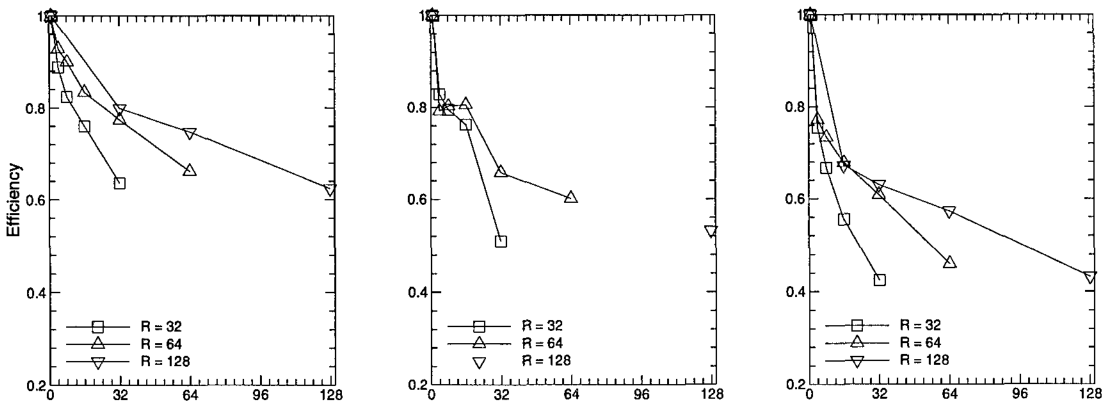
(a) (b) (c)

Figure 4.11: Two-Dimensional Speedups: a) T3D, b) Paragon, c) SP2



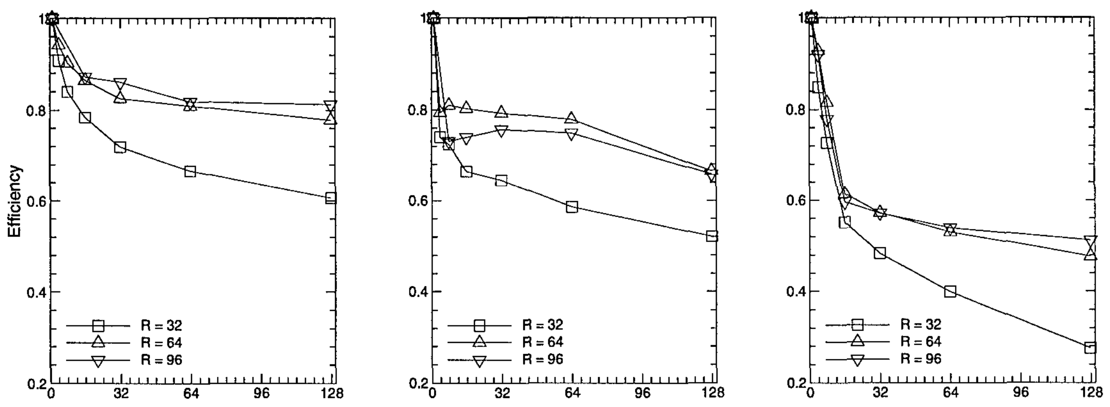
(a) (b) (c)

Figure 4.12: Three-Dimensional Speedups: a) T3D, b) Paragon, c) SP2



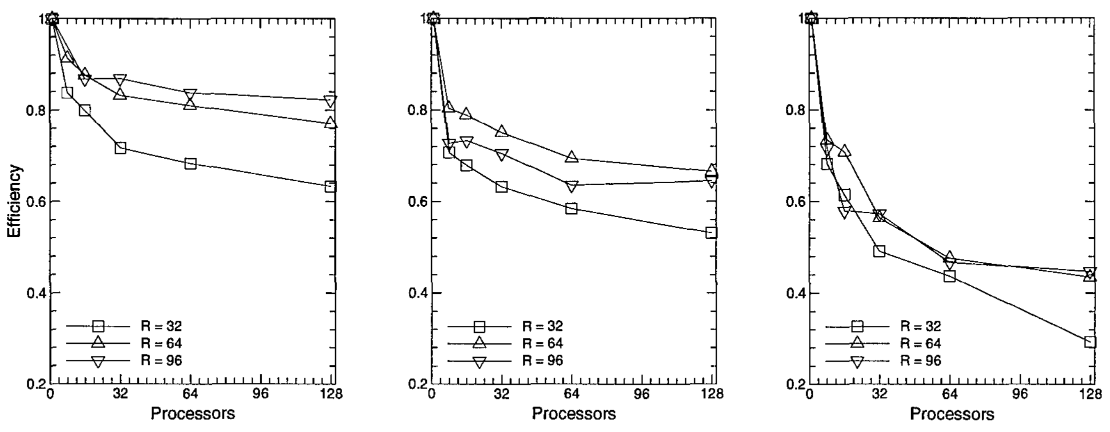
(a) (b) (c)

Figure 4.13: One-Dimensional Efficiencies: a) T3D, b) Paragon, c) SP2



(a) (b) (c)

Figure 4.14: Two-Dimensional Efficiencies: a) T3D, b) Paragon, c) SP2



(a) (b) (c)

Figure 4.15: Three-Dimensional Efficiencies: a) T3D, b) Paragon, c) SP2

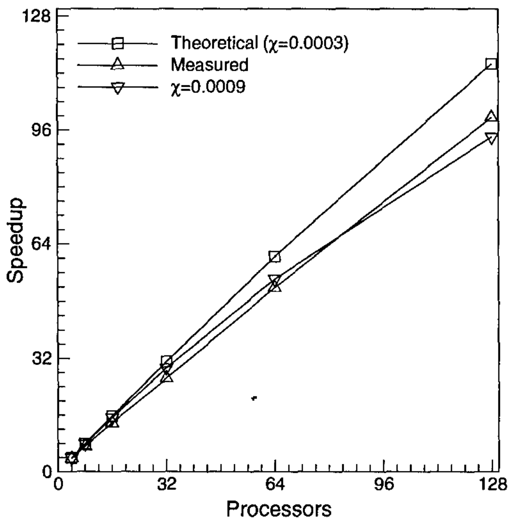


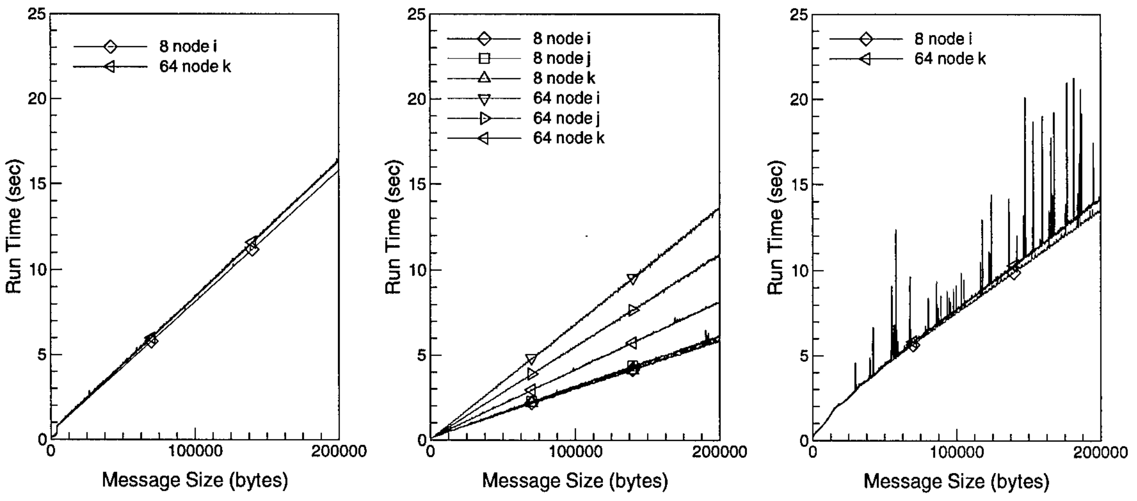
Figure 4.16: Theoretical and Measured Performance of the T3D
($R = 64$, 2D decomposition)

scalability of the algorithm and do not reflect actual program execution times. For example, although the curves show the scalability of the computer code to be decidedly less on the SP2 than on the Paragon, the superior performance of the RS/6000 as compared to the i/860XP[1] yielded substantially faster run times. In other cross-platform comparisons, the T3D was able to significantly outperform the other two machines both in terms of parallel scalability and in terms of absolute execution speed. This is despite the comparatively poor performance of the message passing (as shown in Figure 4.8) which is most likely due to the use of PVM.

4.3.4.3 Comparison with Theoretical Model

A comparison of the theoretical and measured parallel speedups for a two-dimensional decomposition ($R = 64$) appears in Figure 4.16. The theoretical curve was generated by substituting the measured values contained in Table 4.2 into equation (4.18). The measured speedup is somewhat over predicted, but this is not surprising since the theoretical model neglects such issues as load imbalance and message contention. It is therefore expected that this model provides a best-case performance prediction. The behavior of the model with respect to variations in the ratio $t_w/t_c \equiv \chi$ is also shown in the figure. Using the value for t_w and t_c found in Table 4.2 yields the ratio $\chi = 0.0003$. A value of $\chi = 0.0009$ is also shown for reference pur-

1. Machine processor characteristics are discussed in Appendix B.



(a) (b) (c)

Figure 4.17: Nearest-Neighbor Message-Passing Performance a) T3D, b) Paragon, c) SP2

poses. It is not unreasonable to expect fairly large variations in χ for different decompositions due to differences in memory-accessing patterns. In fact, these variations are demonstrated in section 4.3.4.4.

Although the theoretical model was able to predict the performance of certain decompositions on the T3D to an acceptable manner, such was not the case for the Paragon and the SP2. In these cases, the model drastically over predicted parallel performance. Since it is known that χ plays a primary role in parallel performance, a more realistic test problem was conceived to measure t_w. Instead of utilizing a ring structure in which a single message is in transit at any given instant, processors were configured as a logical three-dimensional mesh of dimension $2 \times 2 \times 2$ and $4 \times 4 \times 4$. Nearest neighbors in the mesh simultaneously exchanged messages of varying lengths along each coordinate direction and the times for 1000 exchanges in each direction were recorded. The results of this experiment are contained in Figure 4.17.

In the absence of message contention, the message-passing times are expected to be identical regardless of processor number or direction of message exchange since no two messages simultaneously transit the same logical connection between processors. In reality, however, the mapping of the logical three-dimensional structure to a lower-dimensionality architecture results in the mapping of more than one logical connection to the same physical connection. Although all three machines exhibit some degree of variation in message-passing times as the number of processors is increased, the effect is much less dramatic on the

T3D. The deviation in the general trend is also small for the SP2, but the variations exhibited in Figure 4.8 become much more pronounced as the number of processors is increased. In addition to the variation with processor number, the Paragon also exhibits a directional bias in message-transfer times which becomes more pronounced as the number of processors increases. The magnitude of the variations in message-passing times relates directly to the quality of the theoretical parallel-performance model, and large variations in message-passing times are expected (and observed) to adversely impact the theoretical predictions.

4.3.4.4 Variations in Decomposition Times

The results contained in Figures 4.10 to 4.15 represent the best observed times for a given dimensionality of decomposition, processor number, and grid size. As noted in Table 4.1, all permutations of a given decomposition were performed for each case. Although each permutation yielded the same number of grid points on a given processor and the same amount of message traffic, certain decompositions were observed to yield substantially better performance than others. The performance differences were quantified by constructing the ratio

$$\nu \equiv T_{max}/T_{min}$$

where T_{min} and T_{max} represent the minimum and maximum observed run times for each combination of processor number, grid size, and decomposition dimensionality, Figure 4.18 depicts this ratio for the one- and two-dimensional decompositions on the SP2. Although the trends differed slightly depending on the machine and decomposition approach, similar variations were observed for the Paragon and T3D. The sometimes-marked variations indicate that memory accessing issues are as important as the dimensionality of the decomposition in achieving good performance. This is to be expected since the RISC processors employed in all machines require a high degree of data locality in order to achieve near-advertised megaflop ratings. Should the domains be decomposed in a manner that precludes locality-of-reference, then performance suffers dramatically.

4.3.5 Study Conclusions

The relation between parallel performance and the dimensionality of the domain decomposition was assessed on three modern distributed-memory parallel computing platforms using the SIGMA code and a rectangular waveguide geometry. Higher-dimensionality (two- and three-dimensional) decompositions were found to nearly always outperform one-dimensional decompositions. The performance of the decomposi-

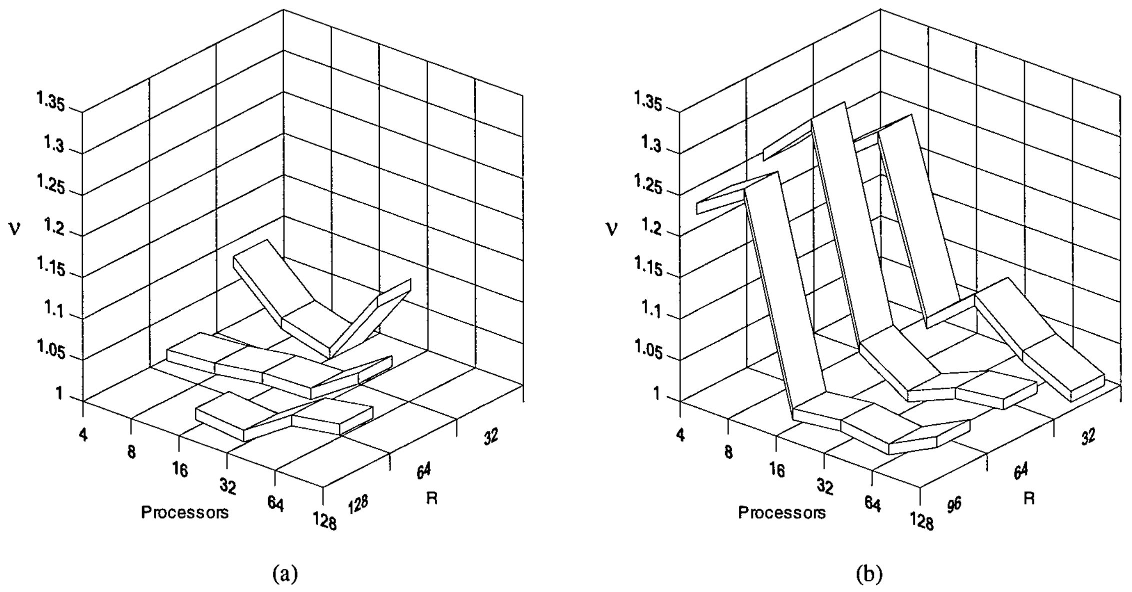


Figure 4.18: SP2 Execution-Time Ratios, a) One-Dimensional Decompositions, b) Two-Dimensional Decompositions

tions was found to relate very closely to the topology of the machine on which the algorithm was implemented. In general, machines with higher processor connectivity favored higher-dimensional decompositions.

Despite the fact the classical parallel performance model used in this study accurately predicted performance trends, it occasionally dramatically over predicted the actual performance of the algorithm. This is due to the fact that the model does not account for issues such as message contention which occurs when more than one logical message path is mapped to the same physical connection. The adverse affect of message contention was observed to increase with increasing processor count on all architectures but most dramatically on the Paragon.

Although caution must be exercised when attempting to extend program performance characteristics to other algorithms, because many algorithms designed for solving systems of partial differential equations possess similar data dependencies, the results presented here can be generalized to a large number of explicit schemes for solving hyperbolic, parabolic, and elliptic PDEs.

4.4 Chapter Summary

An analysis of domain-decomposition techniques for single, structured-grid problems was presented. The purpose of this presentation was two-fold: first, to provide analysis and data not currently found in the literature, and second, to serve as a basis for examining overset-grid problems. Because an overset grid is simply a collection of structured grids, the analysis presented here is applicable in no small extent to the parallel overset-grid problem. This is precisely the focus of the next chapter.

V. Chimera-Grid Domain-Decomposition Study

This chapter expands on the domain-decomposition analysis presented in Chapter 4 by incorporating analysis and experimental results for Chimera grids. An overset-grid decomposition method is presented which produces domains requiring less interprocessor data exchange than the typical block-type structured-grid decompositions discussed in Chapter 4. The technique first constructs a single globally unstructured grid and then applies an unstructured- grid decomposition method which capitalizes on spatial locality–recursive coordinate bisection (RCB) [40,63]–to globally partition the computational domain. A model is developed which identifies the unique communication requirements of a parallel overset-grid implementation, and parallel performance improvement is demonstrated using an overset-grid test problem which is partitioned using both RCB and block-type decompositions.

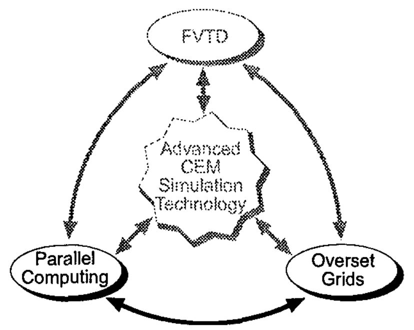
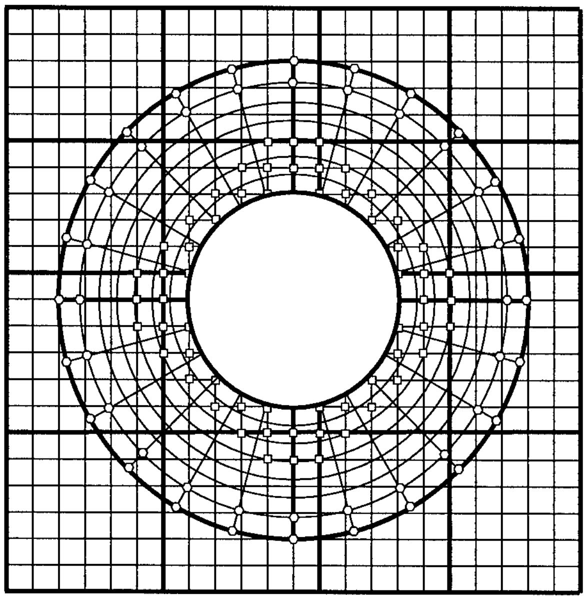


Figure 5.1: Chapter 5 Emphasis

5.1 Domain Decomposition for Overset Grids

As discussed in Section 2.3, the overset-grid decomposition approaches used prior to the present study partitioned each of the structured grids independently along constant-coordinate lines and then assigned the resulting grid blocks to the available processors in some fashion. This type of decomposition is pictured in Figure 5.2 using the same cylindrical/Cartesian grid arrangement presented in Chapter 2. In this figure, however, the darkened lines represent the constant-coordinate lines along which the computational domain is partitioned. Here, the Cartesian and cylindrical grids are partitioned for 16 and 4 processors, respectively. An examination of the figure reveals the problems associated with load imbalance which can occur for this decomposition approach when applied to overset grids. Considering for a moment only the Cartesian grid in the absence of the cylindrical grid, it is apparent that certain processors receive 6×6 blocks of grid cells and thus must perform 44 percent more computational work than those processors which receive 5×5 blocks of cells. In this example, the addition of the cylindrical grid exacerbates the load imbalance problem. Because cells in the Cartesian grid which fall inside the body of the cylinder are excluded



□ Points on background grid updated via interpolation
○ Points on cylinder grid updated via interpolation

Figure 5.2: Overset Grid Domain Decomposition Example

from the computational domain, the number of cells in certain blocks falls from 25 to approximately 10. Defining the load imbalance, λ, as

$$\lambda \equiv 1 - \frac{n_{min}}{n_{max}} \tag{5.1}$$

where n_{min} and n_{max} are the minimum and maximum number of cells assigned to a processor, then considering only the disparities in cell counts for the Cartesian grid, the load imbalance for this simple example has already grown to roughly 0.6. Since a load imbalance of 1.0 represents the worst possible case, it is clear that the illustrated decomposition is less than desirable from a processor-loading standpoint. Unfortunately, the situation grows worse when differences between cell counts on the two grids are considered. Noting that certain processors assigned to the cylindrical grid possess approximately 56 grid cells, the load imbalance for the global grid becomes 0.82! This simple example clearly illustrates the conventional block-type decomposition approach can result in completely unscalable parallel implementations.

In addition to load-balance concerns, this notional decomposition results in a more complex interprocessor communication requirement than for single-grid problems. Recall that the single-grid problem of

Chapter 4 required interprocessor communication only between block faces, since for the case of a single-grid explicit numerical scheme, a given cell is always updated to the new time level using current-time-level information from the cell's neighbors. Conversely, for an overset grid, a cell which is updated via interpolation requires information *at the new time level* from surrounding cells which are part of a different grid. From a data-dependency standpoint, overset grids thus possess an additional trait not found in single-grid problems–either structured or unstructured. It is convenient to classify overset-grid data dependencies as either *structured* or *unstructured* depending on the nature of the cell updates. Structured dependencies occur between cells of the same grid while unstructured dependencies are a result of the interpolation stencils and occur between cells on different grids. Along similar lines, any interprocessor communications that are required can be divided into structured and unstructured communications. Using this classification system, the computational block in the upper-left-hand quadrant of the cylindrical grid must engage in unstructured communications with six different processors associated with the Cartesian grid.

Because of these hindrances to performance which are associated with the overset-grid domain-decomposition strategies used in the past, the remainder of this chapter is devoted to proposing, developing, and testing a new decomposition approach designed to produce partitions which are better load balanced and require less interprocessor data exchange.

5.2 Overset Grid Communication Model

For a computational domain comprised of overset grids, the solution in any cell of the domain is advanced in time either by application of the discretized governing equation or by interpolation. Therefore, denoting the set of cells which form the computational domain as D, let D be divided into subsets Q and I where Q is the set of cells updated via application of the governing equation and I is the set of cells updated via interpolation. The number of cells in each set is denoted by

$$\mathbf{D} = |D|,\ \mathbf{Q} = |Q|,\ \text{and}\ \mathbf{I} = |I| \tag{5.2}$$

where $|A|$ denotes the cardinality of a set A.

In addition, let the communication graph for the computational domain be given by the set of cells D and directed edges, $E(c_i,c_j)$, $0 \le i, j \le \mathbf{D}$, where an edge $e(c_1,c_2)$ exists iff cell c_2 appears in the computational stencil of cell c_1. An edge in E is classified as a *structured edge* if c_1 and c_2 are within the same structured grid and as an *unstructured edge* if c_1 and c_2 reside on different grids. Given sets *D*, *Q*, and *I*, the task is to determine the number of structured and unstructured edges in the communication graph *G(D,E)*.

Because the numerical scheme used for this work is explicit, any cell $c_Q \in Q$ can be updated to time level $n + 1$ using only time level n data from the cells contained in the computational stencil of c_Q. On the other hand, as mentioned in section 5.1, any cell $c_I \in I$ can be updated using only time level $n + 1$ data from the cells in its computational stencil. Denoting a cell in the computational stencil of c_I as a donor cell, $c_{d,I}$, then

$$c_{d,I} \in Q \quad \forall c_I \in I \tag{5.3}$$

Furthermore, because a cell updated via linear interpolation requires information from its nearest surrounding cells, the total number of donor cells, $\mathbf{N}_d$ can be bounded above by

$$\mathbf{N}_d = \Theta(2^k \mathbf{I}) \tag{5.4}$$

where Θ denotes a tight upper bound and k is the dimension of the problem space. Equation (5.4) is an upper bound since any donor cell can appear in the computational stencil of one or more c_I ; however, in terms of unstructured communication edges, the number is exact, namely

$$\mathbf{E}_u = 2^k \mathbf{I} \tag{5.5}$$

where $\mathbf{E}_u$ is the number of unstructured edges in the communication graph *G(D,E)*.

With the number of unstructured edges determined, the number of structured edges in *G(D,E)* can be found directly from the computational stencil of a typical cell contained in *Q*. The number of structured edges associated with a cell is simply the number of cells in its computational stencil, excluding the cell itself. Assuming that all cells are updated via the same scheme (which neglects any differences in stencils due to boundary conditions), then the number of structured edges, $\mathbf{E}_s$, is given by

$$\mathbf{E}_s = k(sw - 1)\mathbf{Q} \tag{5.6}$$

where sw is the computational stencil width.

The total number of edges in the communication graph, $\mathbf{E}_{tot}$, is simply the sum of the structured and unstructured edges. Thus

$$\mathbf{E}_{tot} = 2k\zeta\mathbf{Q} + 2^k\mathbf{I} \tag{5.7}$$

where ζ is the computational stencil half-width defined in Chapter 4.

When the computational domain is partitioned for a distributed implementation, certain edges of the communication graph are said to be *cut.* An edge *$e(c_1,c_2)$* is cut if cells c_1 and c_2 reside on different processors after the decomposition. The number of cut edges in *G(D,E)* relates directly to the communication overhead associated with the domain decomposition. It is therefore important to determine the relationship between the decomposition approach and the number of cut edges. To this end, the remainder of this section is devoted to assessing the ratio of structured to unstructured communications given a particular domain decomposition. In order to simplify the analysis, the following assumptions are made:

- The computational domain consists of *m* structured grids, $m \geq 1$.
- Any grid partitioning takes place along a constant-coordinate line through an entire grid.
- The cells of each grid are distributed equally along each of the *k* coordinate directions.
- Grid *l* is assigned to p_l processors so that all processors possess an identical number of grid cells.

Under these assumptions, the number of cells from grid *l* assigned to a processor, n_l, is given by

$$n_l = R_l^k / p_l \tag{5.8}$$

where R_l is the number of grid cells along each coordinate direction in grid *l.*

As stated in the assumptions, grid *l* is partitioned by passing a cutting plane through the entire grid along a constant-coordinate line. The number of cutting planes that are used to partition grid *l*, c_l, is thus given by

$$c_l = i_l(p_l^{1/i_l} - 1) \tag{5.9}$$

where the term p_l^{1/i_l} is restricted to integral values and i_l is the dimensionality of the decomposition used for grid *l* ($0 \leq i_l \leq k$ and $i_l = 0 \Leftrightarrow p_l = 1$).

For each cutting plane passing through grid *l*, a number of structured edges of *G(D,E)* are cut. That number is a function of the number of cells on the face of the cut which are elements of *Q* and the computational stencil half-width. Consequently, the number of cut structured edges for grid *l* is given by

$$\mathbf{E}_{cut,\,l}\Big|_{str} = \Theta\left[i_l(p_l^{1/i_l} - 1)\psi_l R_l^{k-1}\right] \tag{5.10}$$

where the bounding notation is used since some cells on a cut face may not be elements of *Q*, and

$$\psi_l = 2\left(\sum_{j=1}^{min(\zeta,\,R_l)} j\right) \tag{5.11}$$

Summing over all grids yields

$$\mathbf{E}_{cut}\Big|_{str} = \Theta\left[\sum_{l=1}^{m} i_l(p_l^{1/i_l} - 1)\psi_l R_l^{k-1}\right] \tag{5.12}$$

Although the number of cut structured edges can be expressed analytically, the number of cut unstructured edges is highly geometry and decomposition specific, and it is therefore difficult to develop a general analytical expression for this number. Consequently, it is convenient to assume that a given decomposition produces a number of cut unstructured edges which can be expressed as some fraction, α, of the total number of unstructured edges, namely

$$\mathbf{E}_{cut}\Big|_{unstr} = \alpha 2^k \mathbf{I} \tag{5.13}$$

With this definition, the total number of cut edges, $\mathbf{E}_{cut}$, can be expressed as

$$\mathbf{E}_{cut} = \Theta\left[\sum_{l=1}^{m} i_l(p_l^{1/i_l} - 1)\psi_l R_l^{k-1} + \alpha 2^k \mathbf{I}\right] \tag{5.14}$$

Equation (5.14) is a general expression for the number of cut edges in *G(D,E)* under the stated initial assumptions. For illustrative purposes, the expression can be greatly simplified by setting the dimensionalities of the problem space and the grid decompositions equal (i.e., let $i_l = k$), and by assuming that all grids have the same number of grid cells. Under these assumptions, the grid-identifier subscript can be dropped, and equation (5.14) reduces to

$$\mathbf{E}_{cut} = \Theta\left[\psi k \mathbf{D}^{(k-1)/k}(p^{1/k} - m^{1/k}) + \alpha 2^k \mathbf{I}\right] \tag{5.15}$$

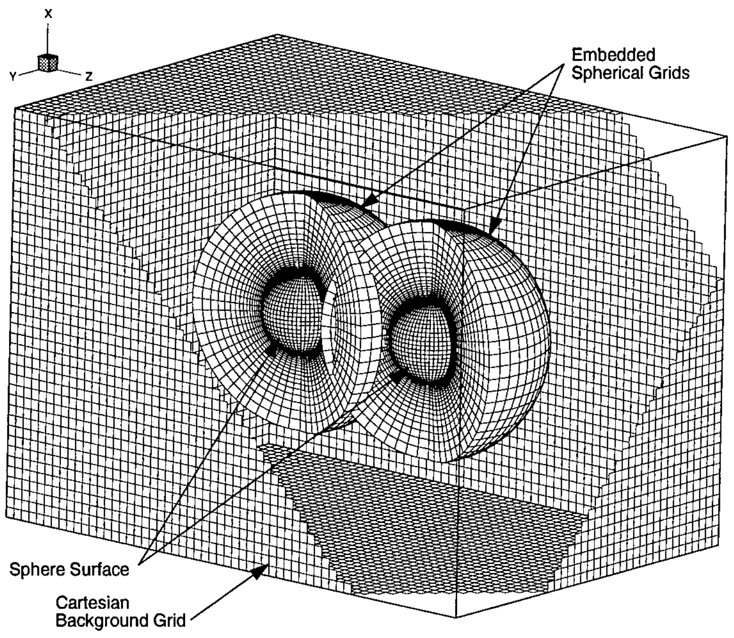


Figure 5.3: Dual-Sphere Overset-Grid Geometry

where p is the total number of processors used for the decomposition.

The first and second terms of equation (5.15) represent the structured and unstructured edges of the communications graph that are cut due to the domain decomposition. In order to assess the relative contributions of these two terms, the decomposition dimensionality, problem dimensionality, and number of grids were fixed at three, the stencil half-width at two, and **D** and **I** were set to 150,000 and 7,500 respectively. Note that the chosen values closely approximate the geometry depicted in Figure 5.3 and were determined by the grid necessary to resolve the problem physics rather than by the choice of decomposition. With these values set, the variables p and α were be allowed to vary and the ratio

$$\phi \equiv \frac{\mathbf{E}_{cut}\Big|_{unstr}}{\mathbf{E}_{cut}\Big|_{str}} \tag{5.16}$$

was computed. The results of this computation are shown in Figure 5.4. Note that the case $p/m = 1$ is not contained in the figure since for that scenario there are no structured cut edges and ϕ is undefined. From the figure, it is apparent that the second term in equation (5.15) comprises a significant portion of the total number of cut communication graph edges for larger values of α. For a typical structured-grid-based decomposi-

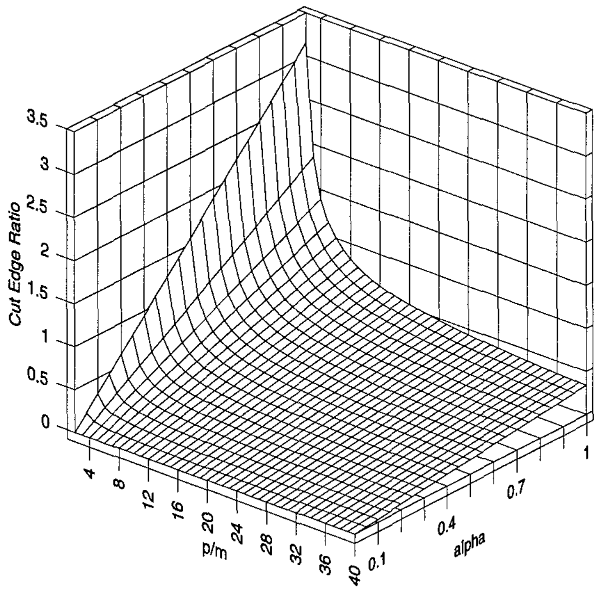


Figure 5.4: Theoretical Unstructured/Structured Cut-Edge Ratio

tion in which each structured grid is partitioned individually and assigned to a unique subset of the available processors, all unstructured communication graph edges are cut and α is exactly one. In the scenario shown in Figure 5.4, a value of $\alpha = 1$ results in the unstructured communications accounting for at least 25 percent of the total communication requirements for the depicted range of processor count. Note that for a smaller number of processors, the unstructured communications dominate the decomposition. It is therefore beneficial to consider alternative decomposition approaches for overset grids which address this portion of the communication overhead.

5.3 Numerical Investigation

5.3.1 Methodology

As discussed in the previous section, the unstructured communications can comprise a significant portion of the total parallel overhead and are a result of interpolation which occurs between cells which are physically close yet reside on different grids. In order to reduce this overhead, it follows naturally to utilize a decomposition approach that partitions the computational domain such that cells in close proximity reside on the same processor. This can be accomplished using RCB, a technique originally developed to partition unstructured grids. Although researchers have shown other unstructured-grid decomposition approaches to be generally superior to RCB when applied to unstructured grids [40,57,63], RCB exploits spatial locality, the precise trait desired to reduce unstructured communications. To apply this technique to overset grids, the individual structured grids are treated together as a single grid by ordering the non-hole cells of all grids based on coordinate location. The ordered list is then bisected and each of the two sub-lists is then re-ordered. The process continues until the desired number of sub-lists is obtained. The cells of each sub-list are then assigned to a processor thereby allowing cells from any number of grids to reside on a single processor. Since the cells on a processor are physically close, the unstructured communication requirements can be dramatically reduced. This reduction can come at a price, however, since the individual structured grids are most likely not divided along constant-coordinate lines. In such instances, there is a corresponding increase in the number of cut structured edges in the communication graph. In order to examine this possible adverse impact, a standard structured-grid block-type decomposition was also implemented and compared to the RCB results.

5.3.2 Computer Code Development

As mentioned previously, the RCB method does not necessarily produce domains that are partitioned along constant-coordinate lines. In fact, in the most general sense, the cells comprising a single domain are not guaranteed to be contiguous. This fact, coupled with the unstructured communication requirements discussed in Section 5.1, mandated that the capabilities of the SIGMA code be significantly upgraded. Consequently, a substantial effort was devoted into developing an algorithm specifically designed for solving overset-grid-based problems in parallel environments. The resulting code is known as CHARGE[1]. CHARGE uses the same FVSRK2 numerical scheme found in SIGMA; however, it is much more flexible and allows for completely arbitrary grid configurations to reside on a single processor. Sub-domains can be blocked, irregular, or even non-contiguous. Furthermore, portions of multiple grids can reside on one processor with the maximum number of grids limited only by available memory. Efficient implementation of such a capability required a substantial departure from the array-type data structures used in SIGMA. The actual domain decomposition algorithms are not part of CHARGE, but instead form a separate program which functions to partition the overset grid as a preprocessing step prior to execution of CHARGE. A complete discussion of the workings of the domain decomposition and CHARGE codes is not fundamental to the

1. *CH*imera grid, *AR*bitrary *GE*ometry

focus of this chapter and is therefore deferred to Chapter 8.

In addition to the flexibility in handling arbitrarily shaped domains, CHARGE is also capable of completely unstructured communications and makes no assumptions on the relative logical or physical locations of the domains. Overall, these capabilities allow for an extremely flexible implementation which is consistent with the main objectives of this research.

5.3.3 Test Case

The problem chosen to test the RCB algorithm was the computation of the scattered electromagnetic fields about two spheres in close proximity. The two spheres were each gridded with a body-conformal grid and then embedded within a uniform Cartesian background grid. The background and spherical grids were comprised of approximately 96,000 and 35,000 cells respectively, which resulted in a computational domain consisting of roughly 166,000 cells. The dual sphere overset grid geometry is depicted in Figure 5.3.

Timing data was collected by allowing the computer code to run for a sufficient time to mitigate any transient machine-loading effects. Multiple runs were made for each decomposition on each machine, and the fastest observed time was used when reporting speedup and efficiency. All timing results were found to be repeatable to well within 1 percent.

5.3.4 Target Architectures

The machines used for the study presented in this chapter were the same as those used in Chapter 4 with the exception of the SP2. This work was completed approximately one year after the completion of the work of Chapter 4. In the interval, the WPAFB Major Shared Resource Center was brought on line and the facility acquired an SP2 possessing faster (but far fewer) processors and a higher-bandwidth interconnection network than that available on the MHPCC SP2. This, coupled with the extremely heavy loading on the Maui machine, prompted the change in platform usage. Furthermore, as proof of the increasing popularity of parallel machines, it became extremely difficult to obtain all 128 nodes of the Eglin Air Force Base T3D. For these reasons, the maximum number of processors used for the present study on the Paragon, T3D, and SP2 was 128, 64, and 32, respectively. In terms of code optimizations on the machines, attempts were made to invoke as much compiler optimization as possible; the compiler flag settings used were *-O3 -Knoieee -Minlines* for the Paragon, *-O3* for the T3D, and *-q arch=pwr2 -O3 -Q+inlines* for the SP2. The *inlines* variable denotes those functions that were explicitly inlined.

5.4 Results

Results presented in this chapter focus on the parallel-performance aspects of the decomposition algorithm rather than the resolution of the electromagnetic fields surrounding the spheres. Those issues are discussed in detail in Chapter 7.

5.4.1 Decomposition Performance

The decomposition performance of the two algorithms was assessed by computing the number of cut structured and unstructured edges assigned to each processor. This was accomplished using the formula

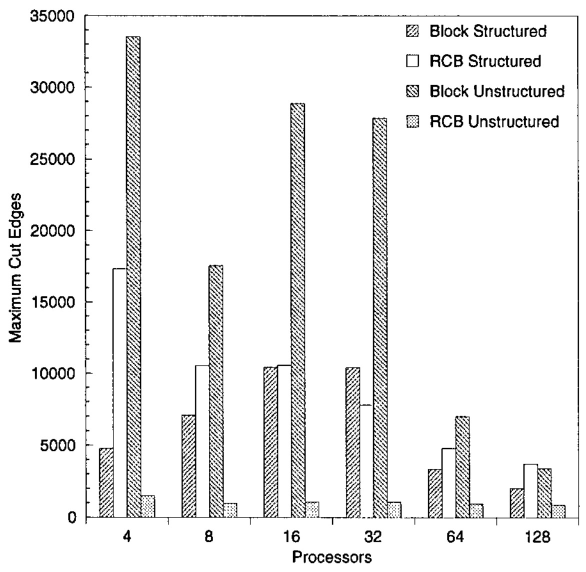
$$\mathbf{E}_{cut,\,p_i}\Big|_{(un)str} = \sum_{c_i}\left(\sum_{e(c_i,\,c_j)} \delta e_{(un)str}(c_i, c_j)\right) \tag{5.17}$$

where $\mathbf{E}_{cut,\,p_i}\Big|_{(un)str}$ is the number of cut (un)structured edges assigned to processor p_i, c_i is a grid cell residing on p_i, and

$$\delta e_{(un)str}(c_i, c_j) = \begin{cases} 0 \text{ if } c_j \text{ resides on } p_i \\ 1 \text{ if } c_j \text{ does not reside on } p_i \end{cases} \tag{5.18}$$

By taking the maximum or minimum of equation (5.17) over all processors, the maximum and minimum number of cut edges for a given decomposition was determined.

Because all processors must simultaneously perform the data exchange associated with each type of communication, the processor with the most data to exchange can become the limiting factor in terms of execution speed. For this reason, it is useful to examine the maximum number of cut edges residing on a processor for a given decomposition. This data is contained in Figure 5.5*a*. Although it is clear from the figure that the number of each type of cut edges is decomposition specific, it is also evident that the block-type decomposition tended to produce fewer cut structured edges than did RCB. This was expected for the reasons described in Section 5.3.1. In terms of cut unstructured edges, the RCB algorithm proved to be dramatically superior to the block-decomposition algorithm for all decompositions and in fact produced fewer total



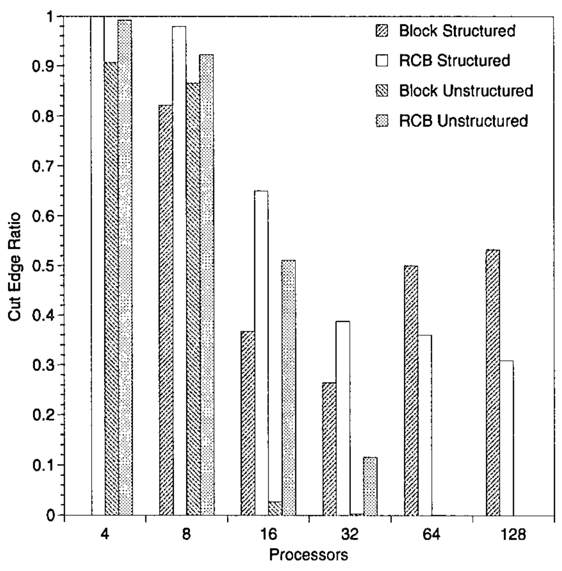


Figure 5.5: Decomposition Results a) Maximum Cut Edges, b) Cut-Edge Ratios

cut edges than did block decomposition. Furthermore, the RCB algorithm produced much more uniform trends than did block decomposition. This is due to the fact that the block decomposition algorithm can suffer from load-balancing problems in cases where the number of grid cells along a coordinate direction is not evenly divided by the number of processors assigned along that direction. In contrast, RCB always assigns a nearly equal number of cells to all processors. In the most simple sense, load balance can be measured by taking the ratio of the minimum to the maximum number of grid cells assigned to any two processors. For this test case, RCB produced ratios always in excess of 0.994 while block decomposition produced ratios ranging between 0.73 and 0.89. The load-balance-ratio comparisons are contained in Figure 5.6.

While the maximum number of cut edges provides a measure of the length of time required for the limiting processor to exchange its data, the issue of communication load balance is more readily assessed by examining the ratio of the minimum to the maximum number of cut edges assigned to any two processors. This information is contained in Figure 5.5*b*. In this figure, a ratio of one indicates that all processors contained the same number of cut edges of a given type. Thus, in contrast to Figure 5.5*a*, longer bars represent a more favorable decomposition. Again, RCB generally outperformed the block decomposition for structured and unstructured cut edges. Note that as the number of processors was increased, the communication imbalance associated with the unstructured edges becomes much more pronounced, and in fact, the ratio drops to zero for the 64- and 128-node decompositions. This is due to the fact that the interpolation regions are phys-

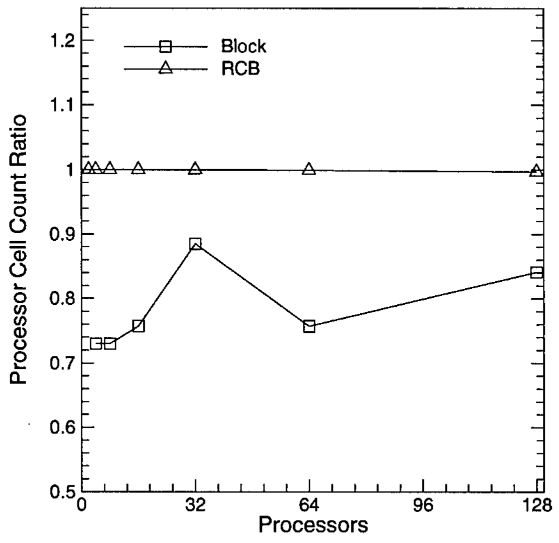


Figure 5.6: Load-Balance-Ratio Comparisons

ically localized and as processor count increases and the domains become physically smaller, the likelihood increases that at least one processor possess no cells that are updated via interpolation. Also note that two processors possessed no cut structured edges for the four-processor block decomposition since, in this instance, each of the spherical grids was assigned to its own processor and thus no structured edges were cut on those grids.

5.4.2 Parallel Performance

The parallel speedup and efficiency comparisons for each of the two decomposition approaches on the three architectures are presented in Figures 5.7 and 5.8, respectively. No two-processor block-decomposition data is presented due to the fact that the block decomposition did not allow cells from two separate grids to reside on the same processor. From a performance standpoint, because CHARGE was specifically designed for distributed architectures, both decomposition approaches achieved very good parallel scalability. Still, the parallel performance of the RCB method was uniformly superior. In terms of architecture comparisons, the efficiencies observed on the Paragon and T3D were very similar and consistently above 90 percent even for a large number of processors on this relatively small problem size. The SP2, because of its lower interprocessor- network-connection bandwidth, suffered somewhat diminished efficiency. This is not

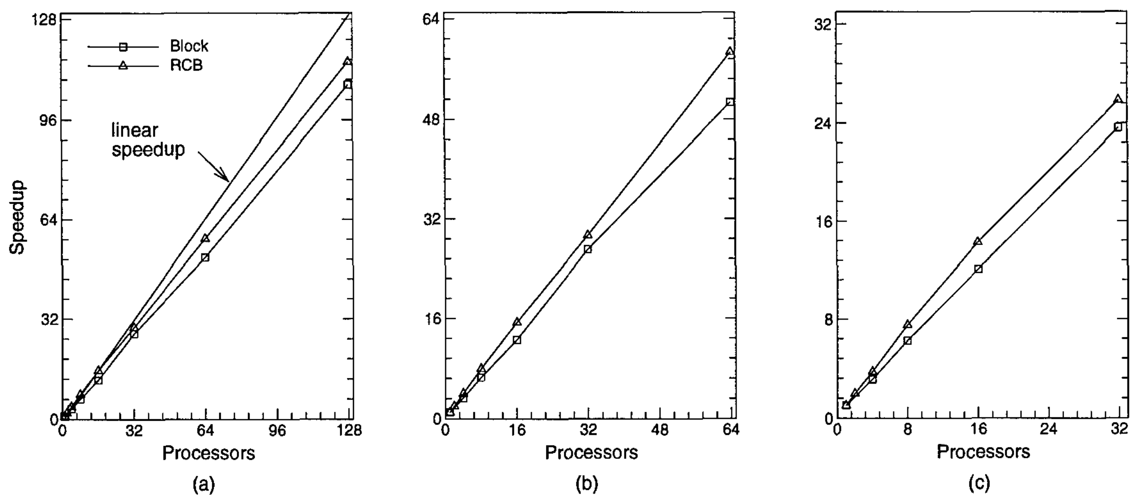


Figure 5.7: Speedup Results: a) Paragon, b) T3D, c) SP2

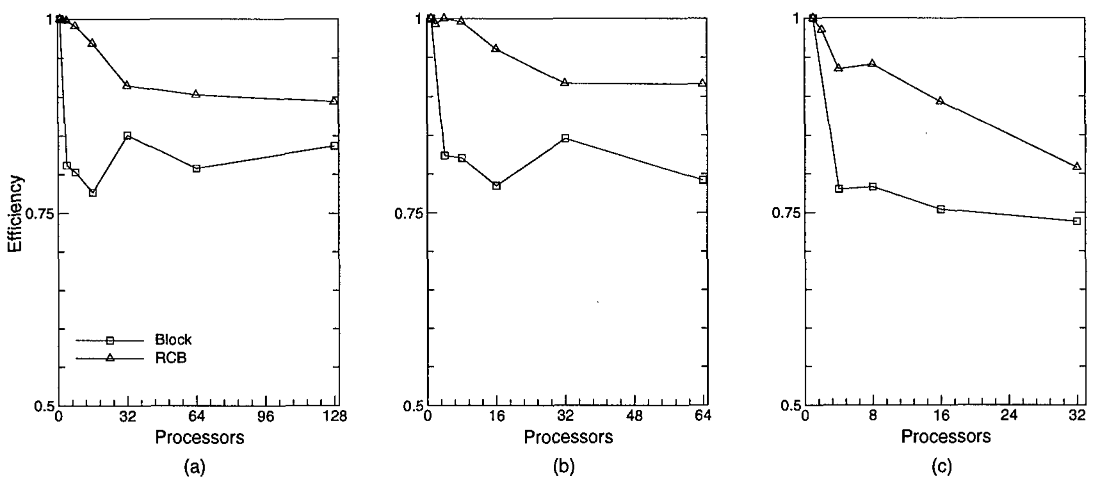


Figure 5.8: Efficiency Results: a) Paragon, b) T3D, c) SP2

to say, however, the Paragon and T3D outperform the SP2 in terms of raw execution speed. In fact, the opposite was observed to be true. The SP2 was consistently three-to-four times faster than the T3D and nearly eight times faster than the Paragon for identical processor counts and decompositions.

One may wonder why the efficiencies and speedups observed for the overset grid case presented here are superior to the single-grid results presented in Chapter 4, especially when considering the arguments presented in the first section of this chapter. This is due to the fundamentally different organization and structure of CHARGE as compared to SIGMA. The former was targeted from the initial design phase as a code optimized to run on RISC-based parallel architectures while the latter was developed using vector-machine-type data structures and programming practices. Again, those issues are discussed in more detail in Chapter 8.

5.5 Overset-Grid Decomposition-Study Conclusions

It has been shown using both an analytical approach and through the use of a model problem that the communications associated with the interpolation phase of an overset-grid calculation can play a significant role in the parallel performance of the algorithm. The present work focused on using RCB to exploit the spatial locality of the interpolation phase and thereby reduce the amount of information which must be communicated between processors. This reduction in communication, in conjunction with improved load balancing, resulted in marked improvements in parallel speedup and efficiency over the block-type decomposition approach. Care must be exercised when attempting to generalize the results presented here since they can be very problem specific. Nevertheless, it is apparent that unstructured techniques such as RCB have the potential of reducing communication overhead and improving parallel performance in overset-grid-based applications.

5.6 Chapter Summary

The discussion presented in this chapter focused on the analysis and implementation of domain-decomposition methods for overset grids. Both the analysis and the implementation presented here differs substantially from previous approaches and represents the first time an unstructured-grid-type decomposition approach has been applied to overset grids. This chapter, like Chapter 4, concentrated on parallel algorithm development rather than on the numerical issues of the single- or overset-grid FVTD process. Because the numerical issues associated with single-grid problems form a logical precursor to assessing the numerical performance of overset-grid cases, the next chapter presents results for the FVSRK2 procedure as applied to single-grid problems.

VI. Single-Grid FVTD Algorithm-Characterization Study

In this chapter, the focus shifts substantially from the parallel emphasis of the preceding two chapters. Here, the performance of the MUSCL-based FVSRK2 algorithm which forms the core solver contained in CHARGE is addressed. Performance issues of this chapter deal only with the capability of the method to produce accurate results on computational problems using single structured grids. Accuracy is assessed by comparing results obtained from CHARGE for two different geometries–the cylindrical rod and the flat plate–to results obtained from other validated numerical methods. Key issues examined in this study include the effects of grid-cell density, mesh stretching, excitation frequency, and Fourier sampling period.

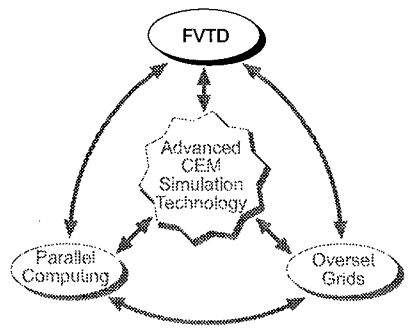


Figure 6.1: Chapter 6 Emphasis

6.1 Algorithm Performance Characterization Issues

6.1.1 Issues Examined

Before the performance of the MUSCL-based FVSRK2 scheme employed at the heart of CHARGE can be assessed in an overset grid environment, it is important that the performance of the method be carefully documented for single-grid problems. Weber [132] has quantified performance aspects of the algorithm using a two-dimensional square cylinder and an oscillating dipole while Shang, et al. [89,92,93,94,96] have characterized certain algorithm features for waveguides, spheres, dipoles, and an ogive/frustrum missile configuration. Specifically, these authors have examined certain accuracy-related issues pertaining to outer boundary placement and mesh refinement. Nevertheless, during the CHARGE validation process, discrepancies arose between observed and published results in terms of required mesh densities, and consequently, a comprehensive study was undertaken in order to quantify the pertinent performance parameters of the algorithm. Those parameters include

- required mesh densities in directions both tangential and normal to the scattering surface,
- mesh stretching effects,

- excitation frequency effects, and
- Fourier sampling effects.

Although several geometries were used to validate CHARGE–including oscillating electric dipoles, waveguides, spheres, cubes, flat plates, and cylindrical rods–only results from the latter two geometries are presented here since they had not been previously examined using this algorithm. Results from the other geometries have been adequately discussed in the above-cited references.

6.1.2 Performance Assessment Metrics

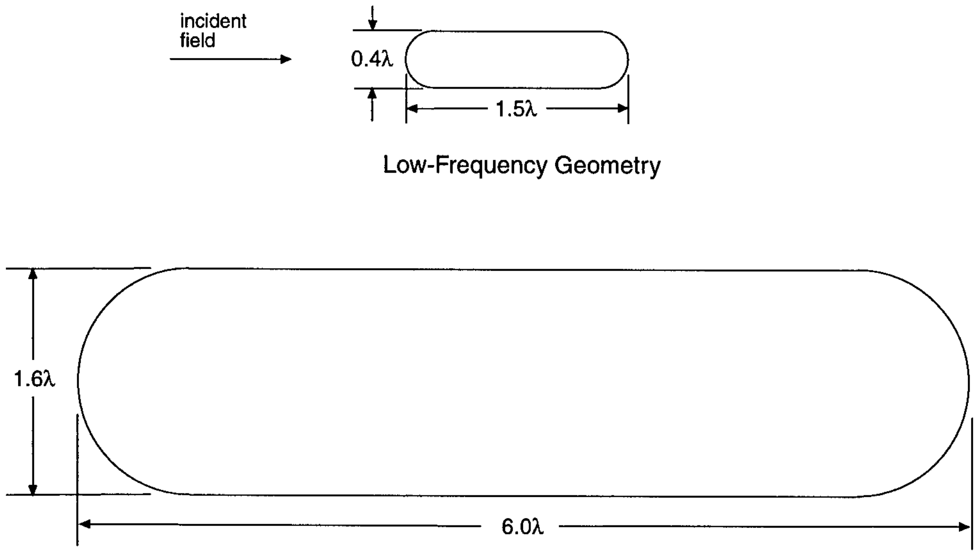
In this chapter, "performance" refers to accuracy rather than speed of execution, and thus it was necessary to determine an adequate metric with which to assess computational accuracy. For this work, the chosen metric was the computed RCS of the scattering body. The choice of this metric requires the consideration of several important issues. To begin, the RCS of a given body does not provide a measure of the global accuracy of the solution. This is due to the fact that the RCS is obtained by integrating the charge and current distributions over a surface which encloses the scattering body. No consideration is made to field values in other regions of the computational domain. Furthermore, the Fourier transformation of the electromagnetic field values necessary for the computation of the RCS introduces an additional source of error distinct from the errors in the computed field values. Despite these concerns, the RCS is generally the quantity sought from a numerical simulation. Furthermore, it is the quantity for which experimental and other numerical data is most readily available. For these reasons, choosing the RCS as the performance metric fits the objectives of this chapter. Because the FVTD methodology requires a new simulation for each incident field angle, unless otherwise noted, results were generated in the form of *bistatic RCS profiles*[1] in order to maintain computer run times at acceptable levels.

6.2 Cylindrical Rod Studies

6.2.1 Geometry and Problem Description

The cylindrical rod is a geometry constructed by placing a hemispherical endcap on both ends of a circular cylinder. This geometry was chosen for the bulk of the grid-refinement and algorithm-characteriza-

1. A bistatic RCS profile is obtained by varying the location of the receiving antenna for a given transmitting antenna location. This is contrasted to a monostatic profile in which the transmitting and receiving antennas are in coincident locations.



High-Frequency Geometry

Figure 6.2: Cylindrical Rod Geometries

tion analysis because comparative MoM data was readily available [6] and because the cylindrical rod forms the body of the finned missile used in the case study appearing in Chapter 8. For the test cases presented in this section, the rod is illuminated with a single-frequency, sinusoidally varying incident field which impinges on the geometry as shown in Figure 6.2. The figure also shows the relative electrical sizes of the two cases that were examined in order to study frequency issues. Although the results do not depend on the exact frequency (it is the electrical size of the object that is important), the low- and high-frequency cases correspond to 500-Mhz and 2-Ghz incident signals, respectively.

6.2.2 Low-Frequency Cylindrical Rod Study

6.2.2.1 Test Case Description and Grid Characteristics

The matrix of computer runs for the low-frequency cylindrical rod (LFCR) study is given in Table 6.1. A total of 30 cases were examined for grid-refinement-study purposes. Each row of the table corresponds to six test cases, one for each of the wall spacings listed in the right-most column of the table. Ascending numerical test-case identifiers within a table row indicate tighter wall spacings in terms of cells per wavelength (cpλ). Grid-cell counts are reported in the table for the radial direction, r, along the axis of the body, θ, and circumferentially around the body, ϕ. In addition, the cpλ are reported for the θ and ϕ directions. The

Case Number	Grid Cells			Wall Spacing (cpλ)
	r	θ (cpλ)	ϕ (cpλ)	
LFCR1 - 6	21	15 (8)	11 (8)	20, 100, 250, 500, 1000, 2000
LFCR7 - 12	21	21 (12)	15 (12)	20, 100, 250, 500, 1000, 2000
LFCR13 - 18	21	34 (20)	25 (20)	20, 100, 250, 500, 1000, 2000
LFCR19 - 24	21	43 (25)	31 (25)	20, 100, 250, 500, 1000, 2000
LFCR25 - 30	21	87 (50)	63 (50)	20, 100, 250, 500, 1000, 2000

Table 6.1: LFCR Test Matrix

range of studied mesh densities was designed to test the gamut from below the nominal 10-cpλ requirement typically reported [4,96] to a substantially greater value in order to ensure grid convergence. As is evident from the table, an extremely wide range of cell spacings in the normal direction was examined since it was found that this parameter plays a critical role in the accuracy of the computed results.

Because of the relative simplicity of the cylindrical rod geometry, the grids were generated analytically. The grid was body conformal and the grid lines in the radial direction were everywhere normal to the body at the surface. No mesh stretching or packing was performed in the θ and ϕ directions; however, an exponential stretching function was used in the *r* direction. Wall spacing was controlled by specifying the desired distance as an input variable into the grid generation program developed for this geometry. The outer boundary of the computational domain was fixed at three wavelengths from the body, and 21 cells were used in the normal direction for all cases reported here[1]. A cutaway view of a sample grid appears in Figure 6.3.

6.2.2.2 Solution Behavior as a Function of Grid Refinement

The HH- and VV- polarization plane[2] RCS results for several selected LFCR cases are depicted in Figures 6.4 and 6.5. The figures are meant to convey the wide range of solutions obtained with the different grids and to emphasize the importance of a proper grid-resolution study prior to any production code runs. Since grid refinements in the normal and tangential directions were observed to produce vastly different

1. Radial direction cell counts in excess of 21 cells were examined when quantifying the mesh stretching effects presented in Section 6.2.3.
2. HH and VV polarization planes are discussed in Section 7 of Appendix A.

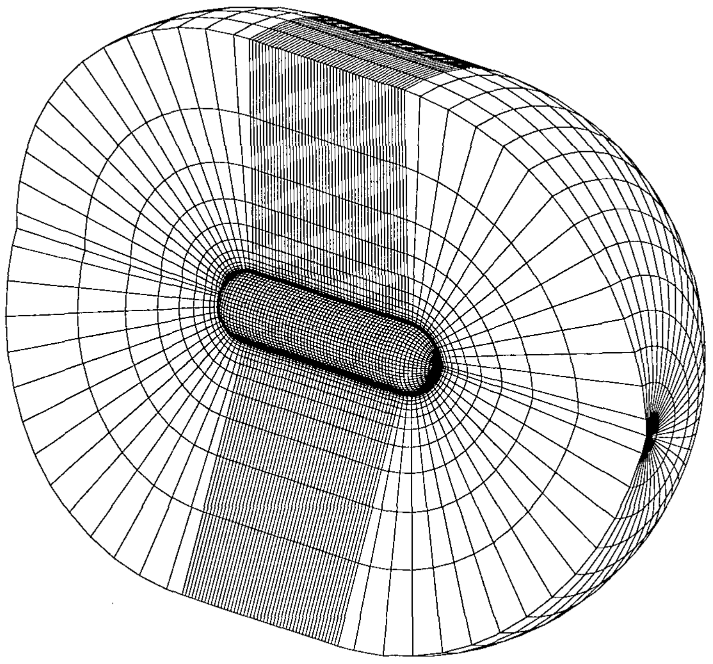


Figure 6.3: Cylindrical Rod Sample Grid

results, each of these topics is treated individually in the following sections.

6.2.2.3 Surface-Mesh-Refinement Effects (Fixed Wall Spacing)

The effects of surface-grid refinement for fixed wall spacings of 20, 100, 250, and 500 cpλ are presented in parts *a*, *b*, *c*, and *d*, respectively, of Figure 6.6. Note that even the coarsest wall spacing is twice the nominal 10 cpλ often cited (e.g., [4]) as being sufficient for resolving wave motion. Part *a* of the figure clearly shows the algorithm is incapable of resolving the physics of the problem using a wall spacing of 20 cpλ regardless of the level of refinement of the surface mesh. In fact, for this level of wall refinement, the computed solution actually worsens as the surface mesh is refined. Although the solutions for the 25- and 50- cpλ surface-mesh spacings are in good agreement, they still deviate substantially from the results obtained using CICERO[1], a MoM code developed by McDonnell Douglas specifically for bodies of revolution [78]. This same agreement in the 25- and 50-cpλ values carries throughout all the wall-spacing results presented in Figure 6.6. Note that no 1000- or 2000-cpλ wall-spacing results are presented in the figure since

1. The results of CICERO were found to be nearly identical to those of Andreason [6], and therefore the latter results are omitted in the figures in an effort to improve clarity.

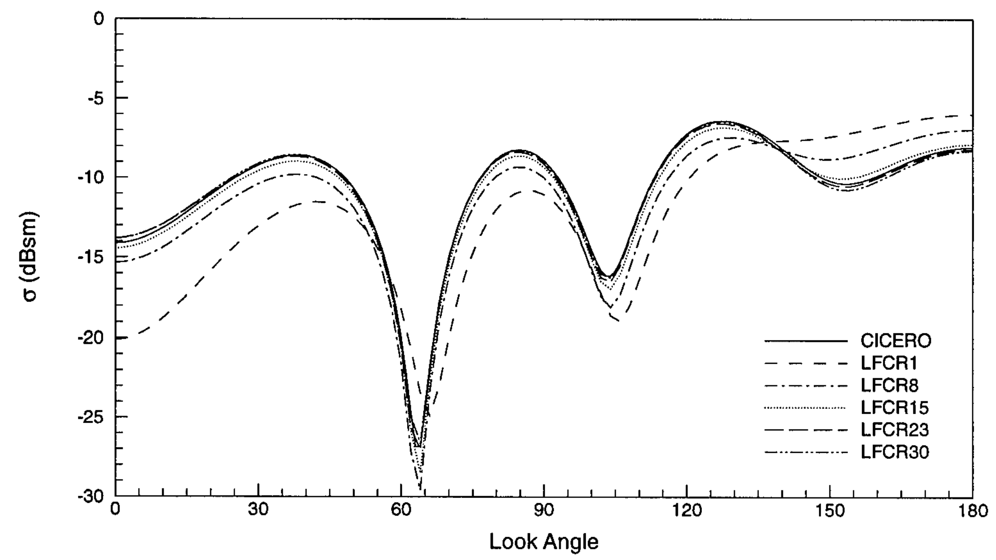


Figure 6.4: Sample LFCR RCS Results (HH polarization)

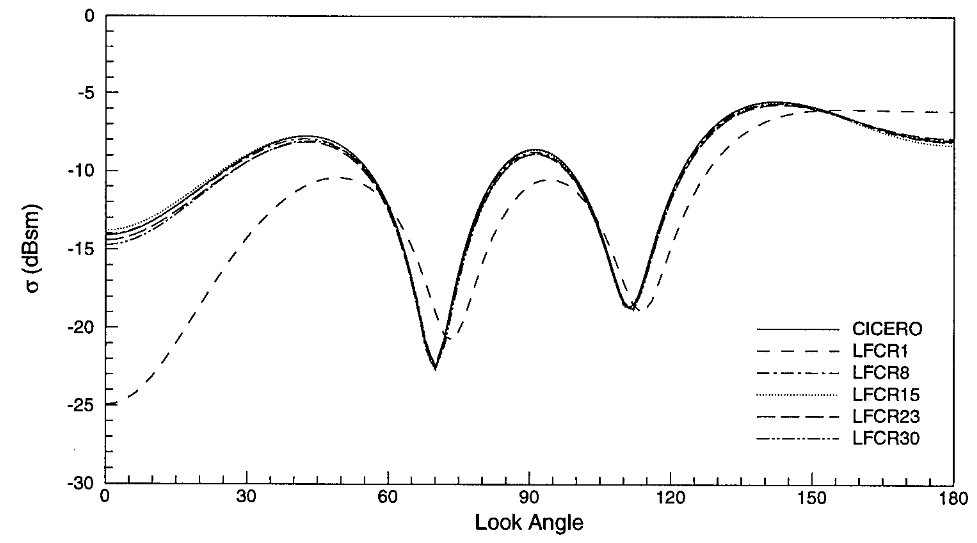
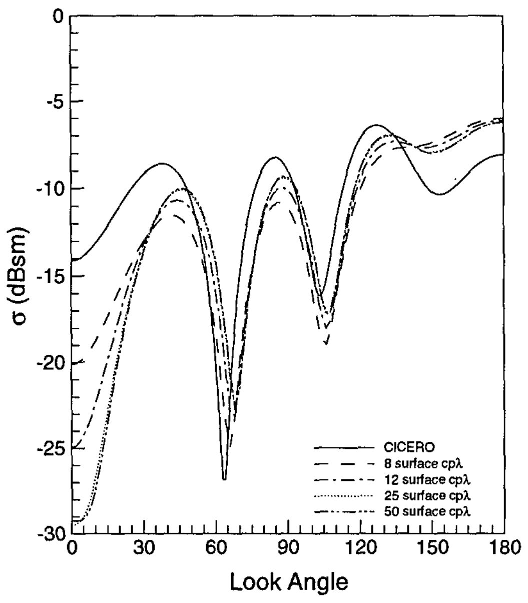
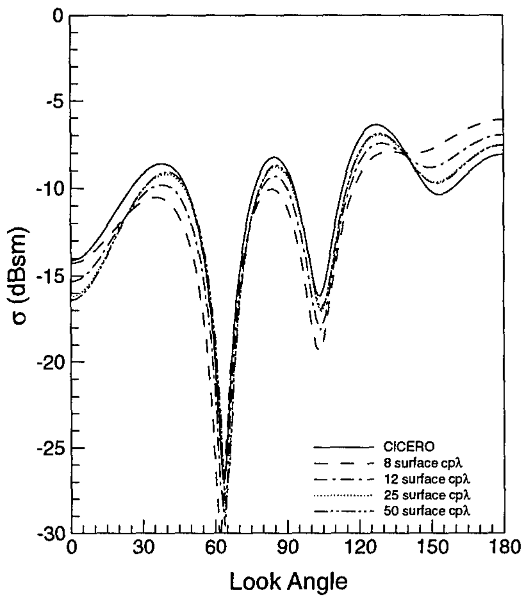


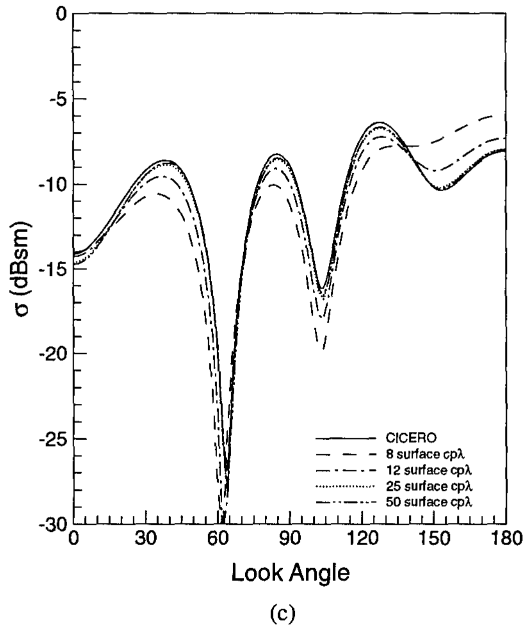
Figure 6.5: Sample LFCR RCS Results (VV polarization)



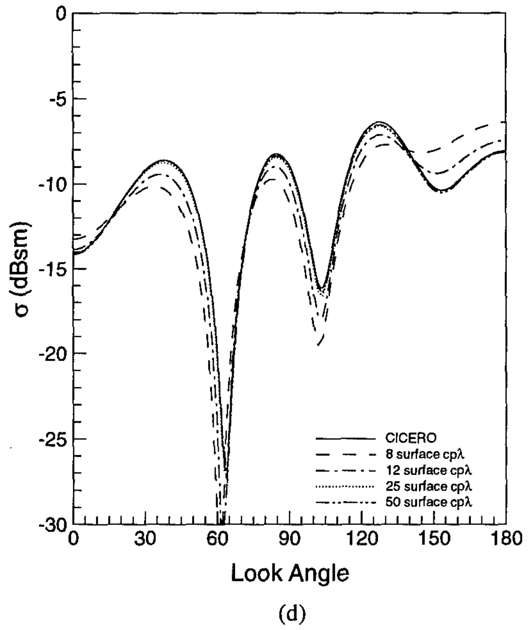
(a)



(b)



(c)



(d)

Figure 6.6: LFCR Surface-Mesh-Refinement Effects for Fixed Wall Spacing of a) 20 cpλ, b) 100 cpλ, c) 250 cpλ, and d) 500 cpλ

Wall Spacing	Backscatter Difference (dBsm)	Forward Scatter Difference (dBsm)	CICERO Backscatter difference (dBsm)	CICERO Forward Scatter Difference (dBsm)
20	NA	NA	15.14	1.84
100	12.84	1.32	2.30	0.52
250	1.69	0.43	0.61	0.09
500	0.56	0.17	0.05	0.08
1000	0.27	0.10	0.22	0.17

Table 6.2: Solution Variation Comparisons for 25 cpλ Surface-Mesh Density

they were observed to be nearly indistinguishable from the 500- cpλ results shown in part *d*.

6.2.2.4 Wall-Spacing-Refinement Effects (Fixed Surface-Mesh Density)

The effect of varying wall spacing while holding the surface spacing fixed is shown in Figure 6.7. Parts *a*, *b*, *c*, and *d* of the figure correspond to surface mesh densities of 8, 12, 25 and 50 cpλ, respectively. Note that the solution is largely unchanging after the wall spacing reaches 500 cpλ. Variations are summarized for the 25-cpλsurface-mesh cases (part *c* of Figure 6.7) in Table 6.2. The Backscatter Difference and Forward Scatter Difference columns in the table contain the magnitude of the difference in RCS values between two subsequent levels of wall-spacing refinement for the indicated look angle. From the table, it is apparent that the backscatter solution changes by only 0.27 dB per square meter (dBsm) and 0.10 dBsm in the backscatter and forward-scattering directions, respectively, as the wall spacing is refined from 500 to 1000 cpλ. The last two columns of the table indicate the magnitude of the difference between the CHARGE and CICERO results for a given wall spacing. Note that the CHARGE and CICERO results agree within 1 dBsm for wall spacings of 250 cpλ or less.

From the results of the wall- and surface-refinement analyses, it is apparent that the low-frequency cylindrical rod requires approximately 20 to 25 cpλ of surface-mesh resolution and a wall spacing of roughly 250 to 500 cpλ in order to achieve a result which deviates from the MoM result by less than 1 dBsm. The exceptionally tight wall spacing requirement is a consequence of the first-order-accurate extrapolation boundary condition used with the current scheme.

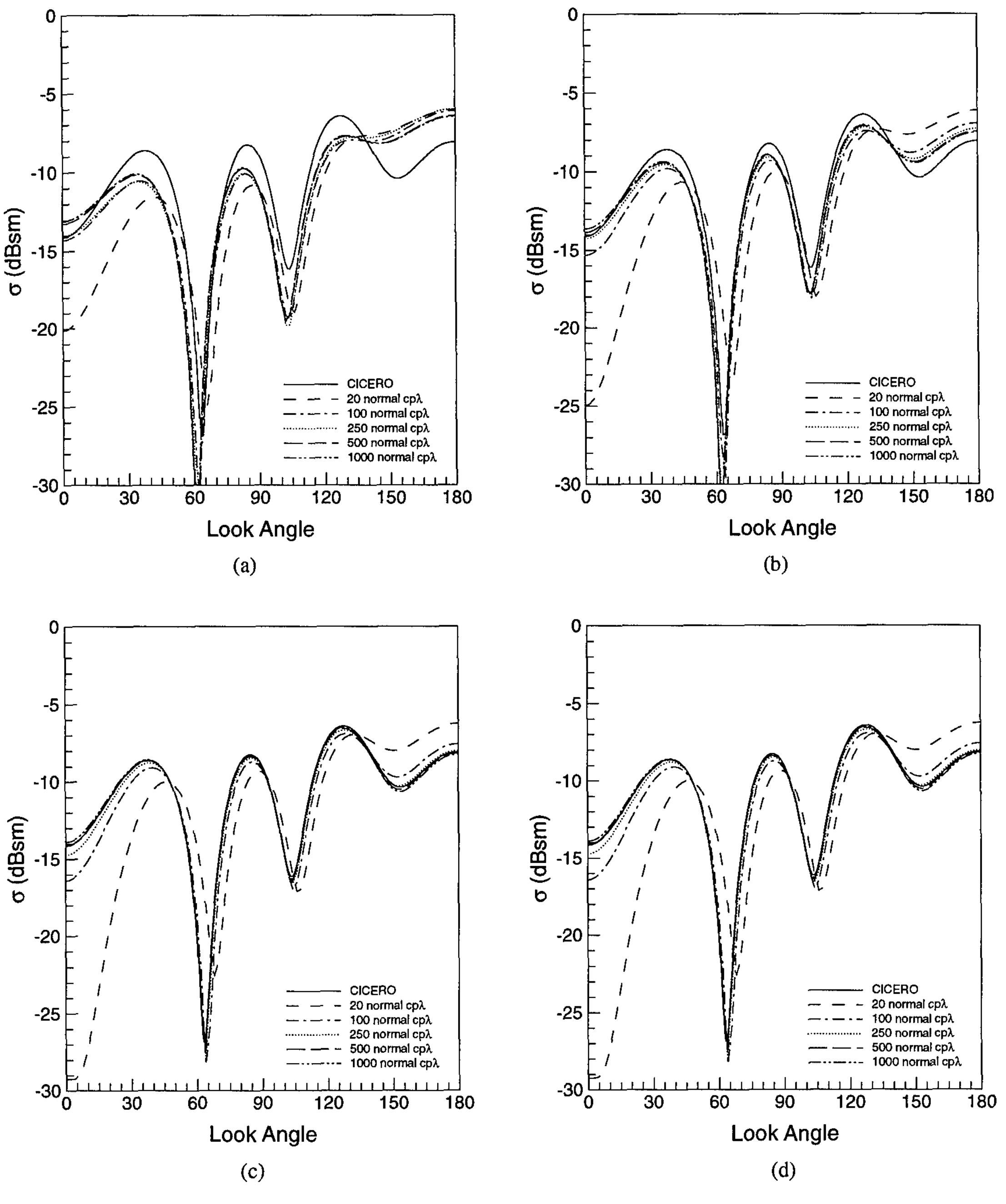


Figure 6.7: LFCR Wall-Spacing-Refinement Effects for Constant Surface-Mesh Density of a) 8 cpλ, b) 12 cpλ, c) 25 cpλ, and d) 50 cpλ

6.2.2.5 RCS Convergence Histories

In addition to affecting the accuracy of the final solution, the level of grid refinement also plays a role in the convergence behavior of the RCS profiles. Because RCS is a frequency-domain measurement, it is necessary to convert the computed electromagnetic field values from the time domain to the frequency domain via a Fourier transform. Before Fourier sampling begins, start-up transients in the surface currents should be allowed to diminish [118]. Experience gained in this work showed that waiting for two periods before beginning the Fourier sampling is generally sufficient (this issue is addressed in Section 6.2.4.4). Once the sampling begins, the computed time-domain fields are transformed for one period of wave motion and the results are used to compute the RCS following the procedure outlined in Appendix A. Sampling is then continued for another period (for a total of two sampling periods) and the RCS is again computed. This continues for up to 25 sampling periods or until the maximum difference in magnitudes of the computed RCS for any look angle between two subsequent Fourier sampling periods falls below a predefined level which is based on the minimum obtainable residual for a given geometry and incident field frequency. For the cylindrical rod cases, a value of 0.00005 was used.

The RCS convergence histories (shown for both the HH and VV polarizations) of cases LFCR1 and LFCR22 are depicted in parts *a* and *b*, respectively, of Figure 6.8. Computed RCS profiles are provided for each Fourier sampling period (a sampling period equates to within one timestep of one period of wave motion). In the case of the coarse grid, the results did not converge within the maximum 25 sampling periods; however, the more-refined grid required only 9 sampling periods for convergence. For the LFCR22 case, the results from each of the nine sampling periods are nearly indistinguishable indicating the computations could have been stopped after a single sampling period. Note that the LFCR22 case corresponds to 25-cpλ surface-mesh refinement and 500-cpλ wall spacing which was deemed to be the grid required to demonstrate a grid-independent solution in the previous sections.

In terms of computational effort required for the two cases, the computed time step for the LFCR1 and LFCR22 cases was 0.0237 and 0.00107, respectively. Because of the smaller time step, the LFCR22 case required approximately 22.15 times the number of time steps as the LFCR1 case in order to reach the same physical time. Furthermore, because the more-refined grid contained 8.08 times the number of grid cells as the coarse grid, the LFCR22 grids required roughly 179 times the amount of computational work as the coarse grid for the same physical simulation time. This increase in workload was offset in part, however,

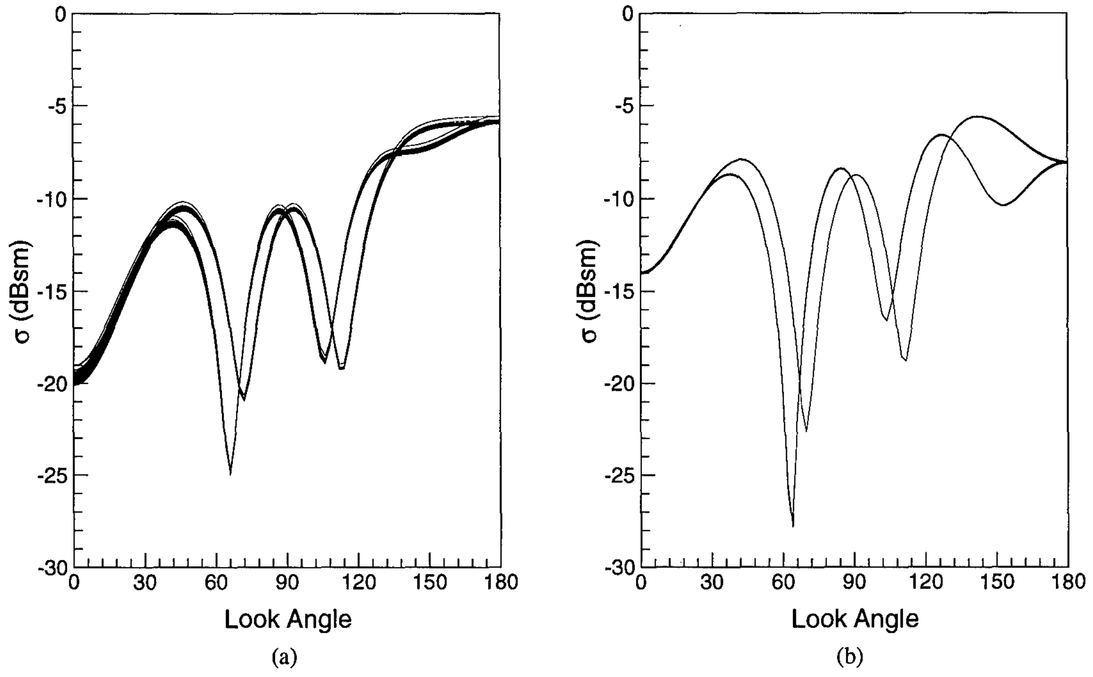


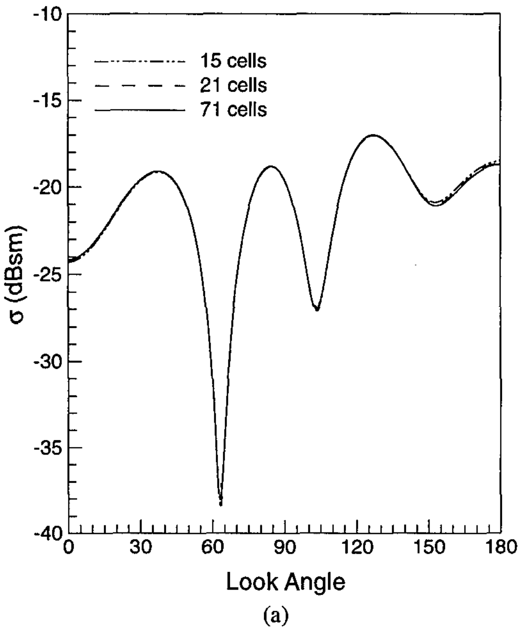
Figure 6.8: LFCR RCS Convergence Histories: a) LFCR1, b) LFCR22

by the reduced number of sampling periods necessary to obtain a converged solution.

6.2.3 Mesh-Stretching Effects

The flux-reconstruction formula used in the FVTD algorithm is derived for a uniformly spaced mesh [52]. Should the mesh be highly stretched, then the accuracy of the method can be degraded. However, because the algorithm requires such a tight wall spacing, and because the outer boundary is typically placed at roughly three wavelengths from the scattering surface, it is not possible to maintain a uniform normal spacing without excessive grid-cell counts. For example, a uniform radial-mesh spacing of 500 cpλ over 3 wavelengths results in 1500 cells in the radial direction. If the surface of the cylindrical rod is gridded at 25 cpλ, then nearly two-million grid cells are required for this simple geometry. Such a requirement would render the algorithm impractical for any realistic engineering analysis.

To overcome this difficulty, the mesh cells can be clustered near the surface. For the cylindrical rod, this was accomplished via an exponential stretching function. This stretching approach was shown to be highly effective in substantially reducing the number of grid cells required to obtain accurate results for this



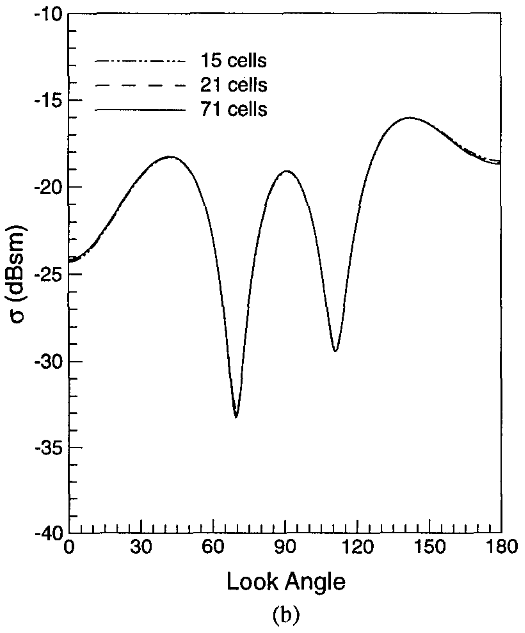


Figure 6.9: Effect of Mesh Stretching Normal to Body: a) HH Polarization, b) VV Polarization

type of convex scattering body. Figure 6.9 shows the effect of mesh stretching on the cylindrical rod. The figure depicts results for the same surface mesh density and wall spacing as used by test case LFCR9, but the number of cells is varied (15-, 21-, and 71-cell results are shown) in the radial direction while keeping the wall spacing constant. Clearly, the results are nearly identical even though the radial cell count varies by almost a factor of five. These results are quite fortuitous and of considerable importance due to the tight wall spacing mandated by the first-order extrapolation surface boundary condition.

A similar beneficial effect was observed when points were simply packed near the boundary. An illustration of cell packing, constant spacing, and stretching appears in Figure 6.10. Packing was examined early in the study and found to yield results of much greater accuracy than non-packed meshes but was discontinued after stretching was found to produce meshes requiring fewer cells. Because all cylindrical rod cases were examined after the benefits of stretching were determined, no test cases were conducted with packing. However, several single-sphere cases were examined using both packing and stretching schemes and a sample comparison of the results produced by the various wall-spacing schemes is provided in Figure 6.11. For the cases presented in the figure, the sphere was originally gridded using 71 cells, constantly spaced, in the radial direction. The grid was then modified by packing 10 additional cells within the first radial cell spacing

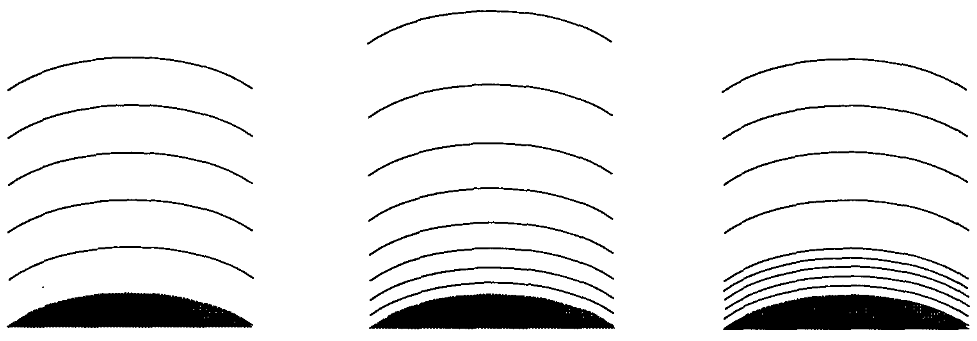
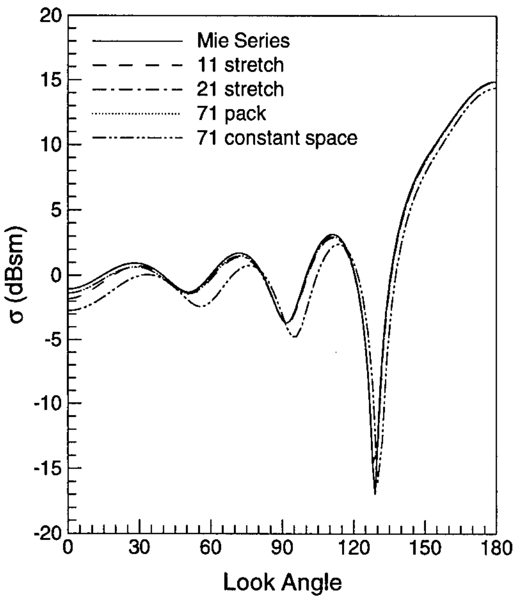


Figure 6.10: Wall-Mesh-Spacing Approaches

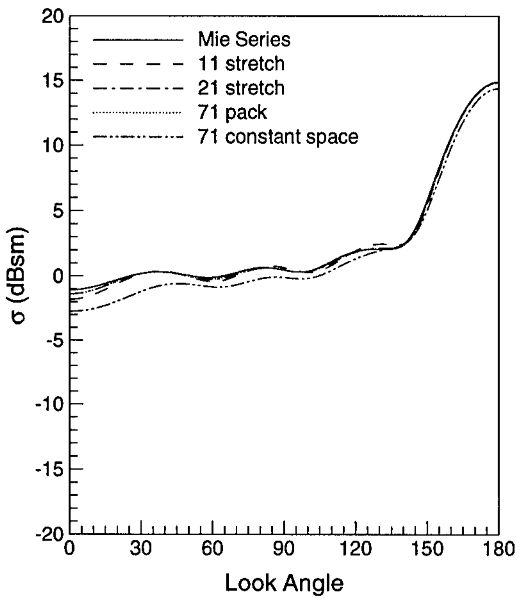
of the original grid. Results obtained from this grid were compared against stretching using 11 and 21 radial cells, maintaining a consistent wall spacing between the three cases. For reference purposes, the results of the original constantly spaced grid as well as the analytical Mie Series solution are also contained in the figure. The packed results compare extremely well to the stretched mesh using 21 cells in the normal direction. Thus, despite the jump in the metric values between the tenth and eleventh cells in the radial direction, the solution remains well behaved. Note that all cases which employ a level of wall-spacing refinement produce results far superior to the constantly spaced mesh. In the extreme case of only 11 radial cells, the wave is resolved at less than 1 cpλ at the outer boundary. Clearly, this is below the Nyquist rate and thus the computed solution in this region is invalid. Nevertheless, these results show the solution at the body (where the RCS integration is performed) is of utmost importance and that away from the body large inaccuracies can be tolerated. This is provided, of course, that the wave which propagates through these extremely coarse mesh regions does not impinge upon another scattering body.

6.2.4 High-Frequency Cylindrical Rod Study

In order to ascertain the effect of excitation frequency on the phenomena observed in the preceding sections, the tests were repeated using the same cylindrical rod while increasing the incident field frequency by a factor of four. These tests are denoted as the high-frequency cylindrical rod (HFCR) tests, and the associated test matrix appears in Table 6.3. Because the HFCR exhibited better grid-convergence properties than did the LFCR, fewer test cases were required. Furthermore, because many of the results were similar to



(a)



(b)

Figure 6.11: Effect of Mesh Stretching and Cell Packing on Spherical Geometry: a) HH Polarization, b) VV Polarization

those reported for the LFCR, fewer graphical results are presented in the following section. The grid characteristics for the high-frequency cases were identical to the low-frequency characteristics described in Section 6.2.2.1

6.2.4.1 Surface-Mesh-Refinement Effects (Fixed Wall Spacing)

Figure 6.12 is the high-frequency counterpart of the low-frequency results presented in Figure 6.6. Parts *a* and *b* of the figure contain the RCS results for 20- and 250-cpλ wall spacing, respectively. From the figure, it is apparent that the solution is nearly identical for the 18- and 25- cpλ surface-mesh refinements. This is in contrast to the low-frequency cases which required roughly 25- cpλ surface-mesh refinement in order to achieve grid convergence.

6.2.4.2 Wall-Spacing-Refinement Effects (Fixed Surface-Mesh Density)

The effects of wall spacing for the 12- and 18-cpλ surface-spacing HFCR cases is demonstrated in Figure 6.13. For both surface-mesh densities, the solution is quite similar for wall spacings in excess of 100 cpλ. In fact, the difference in the backscatter RCS between the 100- and 500-cpλ wall-spacing cases is a

Case Number	Grid Cells			Wall Spacing (cpλ)
	r	θ (cpλ)	ϕ (cpλ)	
HFCR1 - 4	21	55 (8)	41 (8)	20, 100, 250, 500
HFCR5 - 8	21	83 (12)	61 (12)	20, 100, 250, 500
HFCR9 - 12	21	125 (20)	91 (20)	20, 100, 250, 500
HFCR13 - 16	21	172 (25)	125 (25)	20, 100, 250, 500

Table 6.3: High-Frequency Cylindrical Rod Test Matrix

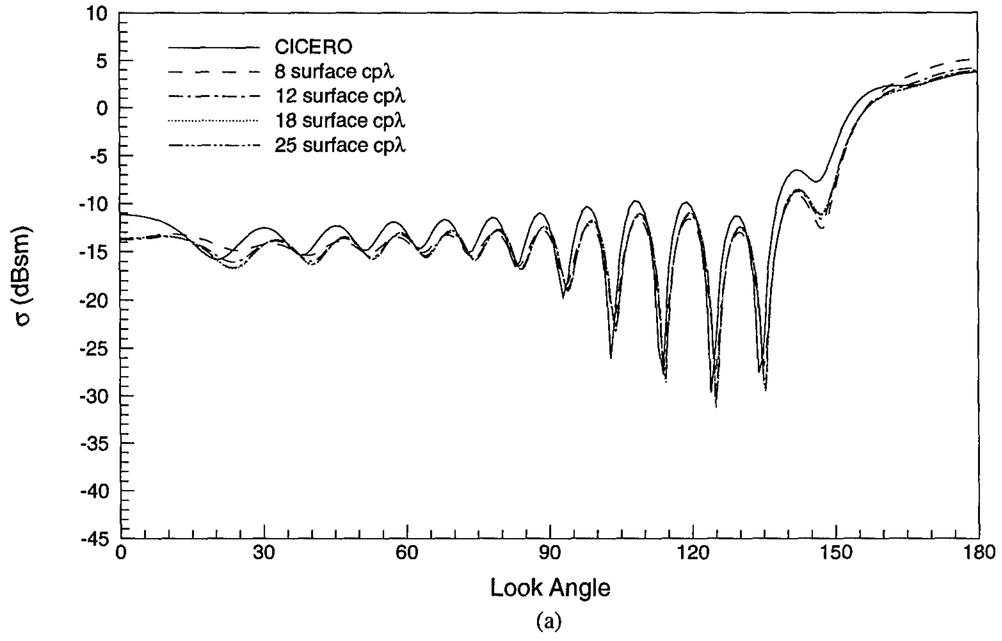
mere 0.24 dBsm. Differences in the forward-scattering RCS calculations is even more impressive–only 0.05 dBsm separates any two calculations using greater than 100-cpλ wall spacing. Taken in conjunction with the required surface-mesh spacings presented in the previous section, these results clearly show that the higher-frequency illuminations require fewer cells per wavelength in order to capture the problem physics. This is due to the fact that surface phenomena–including travelling and creeping waves–are much more pronounced at lower frequencies. Because the extrapolation boundary condition brings information to a given cell on the surface only from the normal direction, these surface-tangential effects are largely lost. Appendix A provides an in-depth discussion on the implementation of the surface boundary condition in CHARGE.

6.2.4.3 RCS Convergence Histories

A final comparison of the low-vs.-high-frequency behavior of the algorithm is given by Figure 6.14 wherein the convergence histories in both the HH- and VV-polarization RCS results for cases HFCR2 and HFCR12 are presented. Comparing these results to those contained in Figure 6.8, it is apparent that the high-frequency solutions exhibit a much greater variation in the RCS profiles–even for the more-refined grid case–than do the low-frequency solutions. It is believed that this effect is largely attributable to the increased surface-current transients caused by the higher energies (which are manifest as greater magnitudes in scattered-field amplitudes) associated with the higher-frequency incident field.

6.2.4.4 Fourier Sampling Start Period

If Fourier sampling begins while there are still large surface-current transients, it is possible that significant errors can be introduced into the RCS computation. It was noted in Section 6.2.2.5 that waiting two periods of wave motion before beginning Fourier sampling was generally sufficient; however, given the



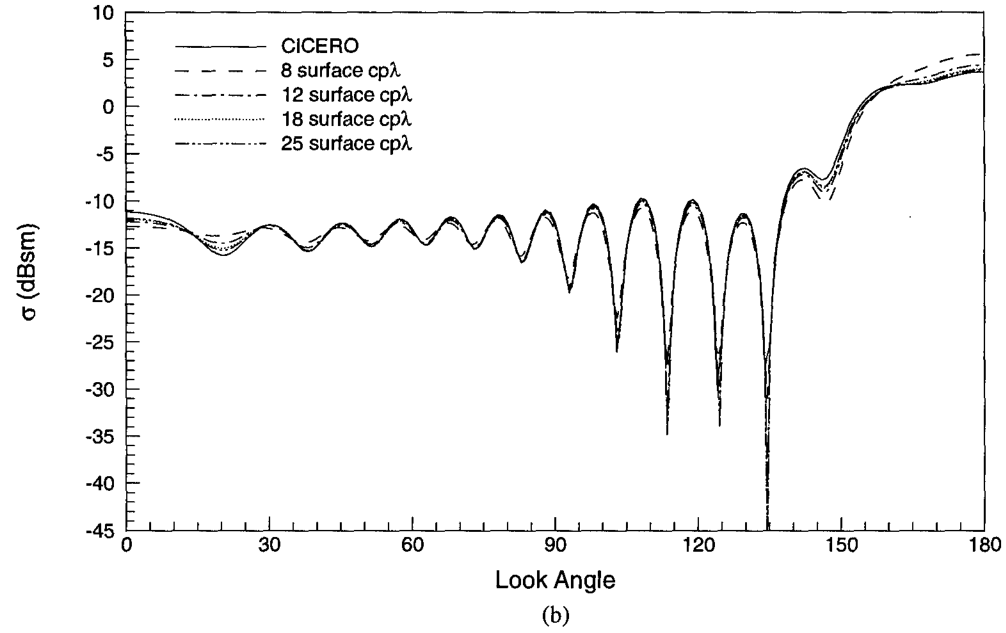
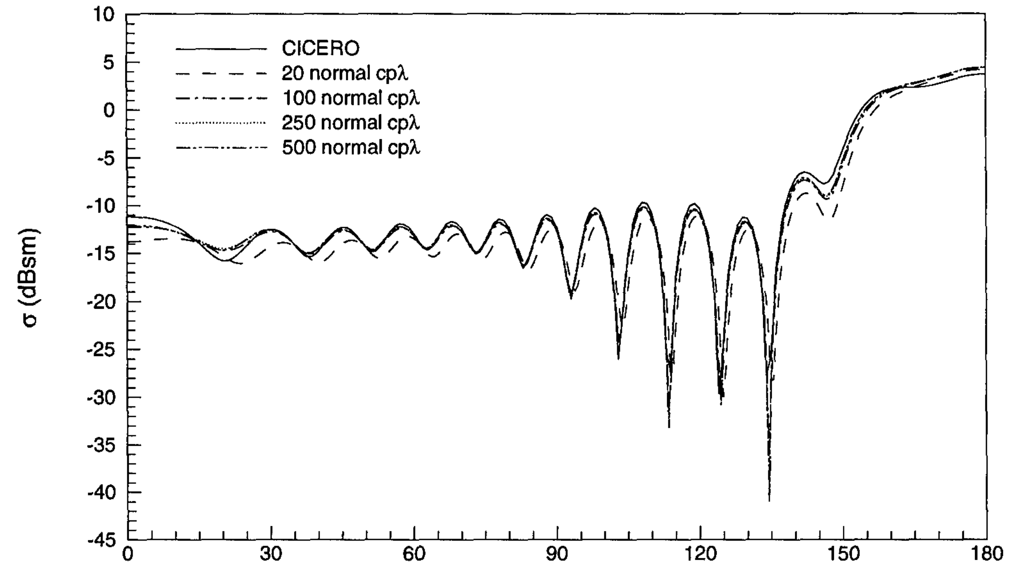
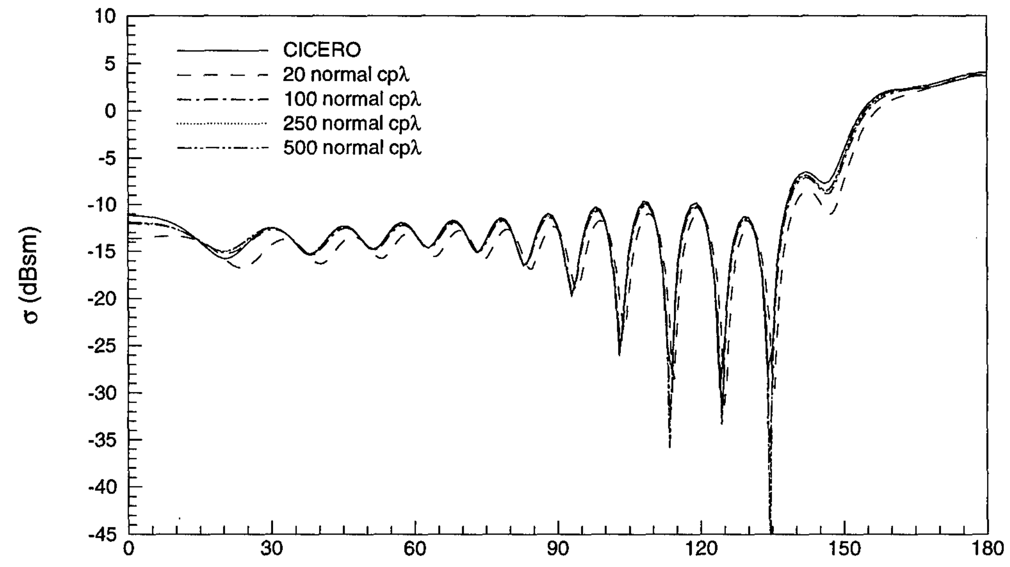


Figure 6.12: HFCR Surface-Mesh-Refinement Effects for Fixed Wall Spacing of a) 20 cpλ, and b) 250 cpλ

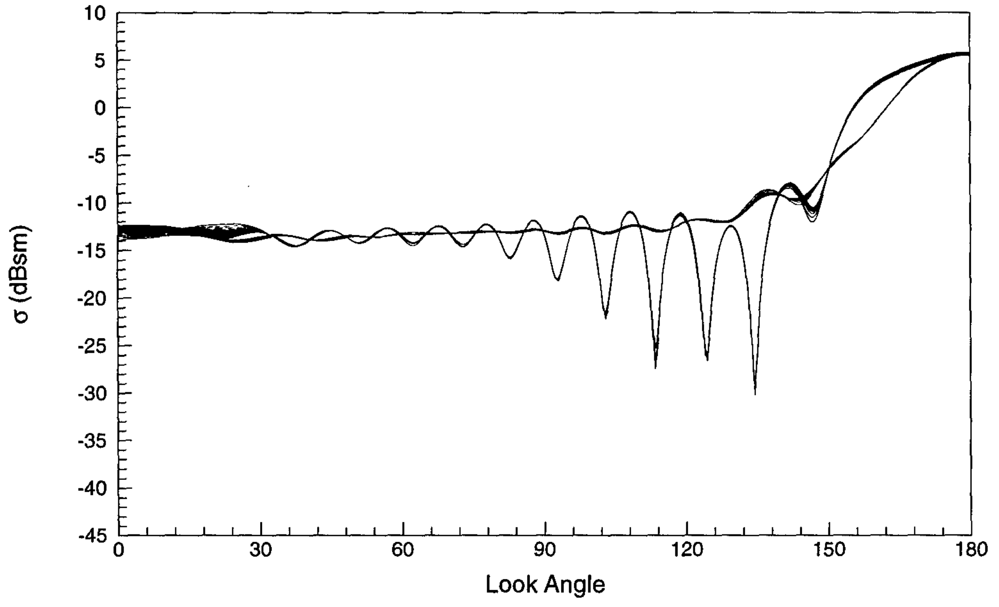


(a)

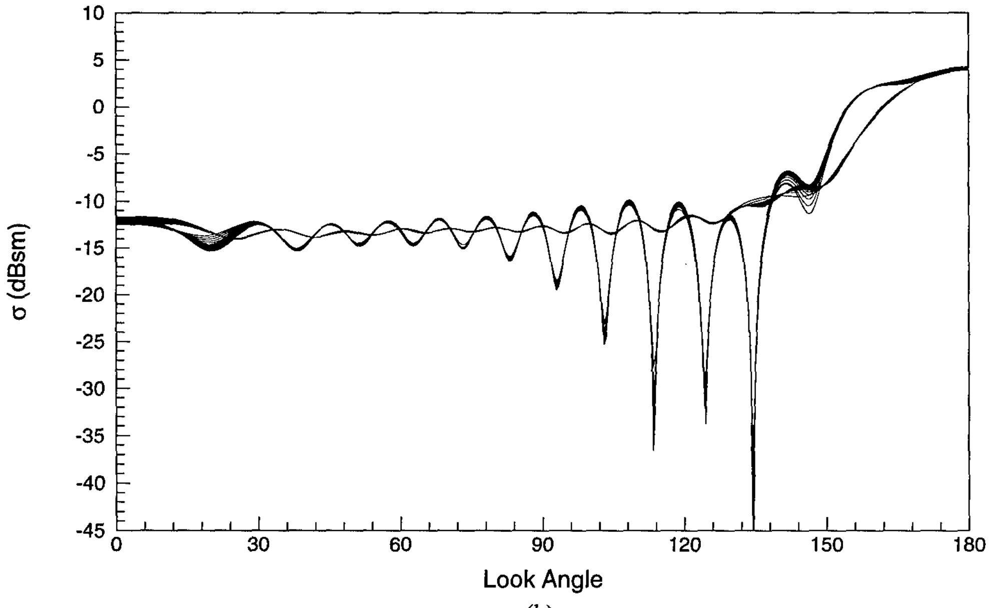


(b)

Figure 6.13: HFCR Wall-Spacing-Refinement Effects for Constant Surface-Mesh Density of a) 12 cpλ, and b) 18 cpλ



(a)



(b)

Figure 6.14: HFCR RCS Convergence Histories: a) HFCR2, b) HFCR12

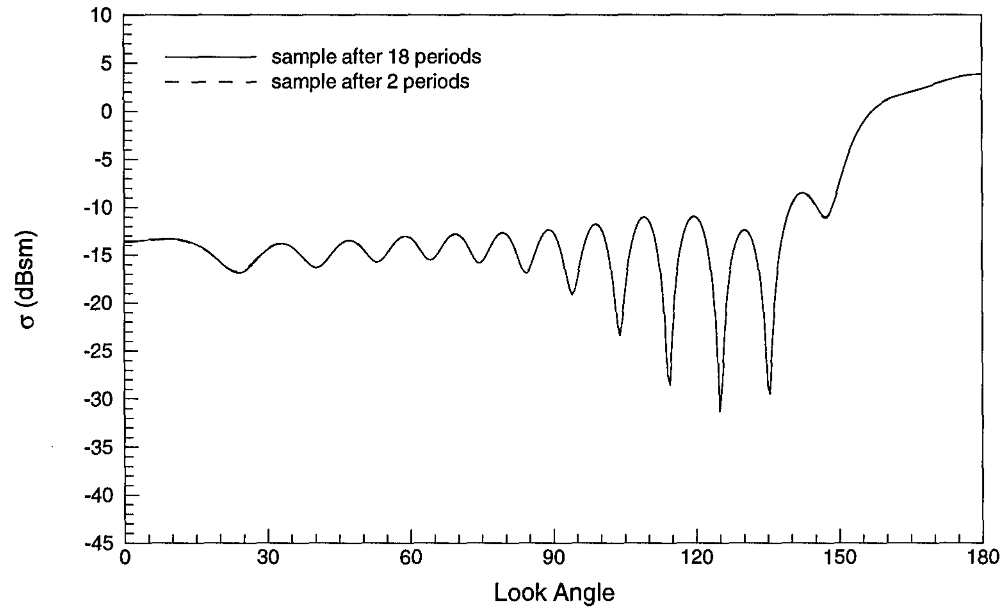


Figure 6.15: HFCR Fourier Sampling Starting Period Comparisons

more dramatic variations in RCS convergence histories for the high-frequency test cases, an analysis was conducted to determine the effects, if any, of the choice of initial sampling period on the RCS calculations. To examine these effects, the results of case HFCR13 (in which Fourier sampling was begun after two periods) were compared against an identical-grid case in which Fourier sampling was postponed until after 18 wave periods (a number of periods equal to 3 times the electrical length of the body). The computed RCS for both cases after 27 total wave periods is presented in Figure 6.15. Clearly, the plots are nearly indistinguishable. This indicates the start-up transients effects on the RCS computations are not severe, and if present when Fourier sampling begins, they do not permanently affect the quality of the RCS computation.

6.3 Flat Plate Studies

The finite-thickness flat plate represents a dramatically different geometry (in terms of electrical features) than the cylindrical rod. While the surface of cylindrical rod is first-derivative (C1) continuous, the edge discontinuities of the flat plate produce large diffraction phenomena. It is for this reason that Shankar [103] comments on the difficulty of accurately computing the scattered fields from the flat plate.

Because the edge-diffraction phenomena becomes increasingly important as the electrical size of the plate decreases, several plate sizes were examined. Results are presented here for a large ($5\lambda \times 5\lambda \times 0.031711\lambda$) and small ($0.5\lambda \times 0.5\lambda \times 0.02\lambda$) plate. The large plate was chosen in order to compare with a physical-optics-based result presented by Balanis [8] while the small plate was chosen to approximate the dimensions of the missile fins used in Chapter 8.

6.3.1 Large Plate Study

6.3.1.1 Grid Generation and Problem Description

A description of the geometry and incident-field orientations used by Balanis are depicted in Figure 6.16. From the figure, it can be seen that the incident wave strikes the plate at an angle of 30 degrees from the z axis. For this study, both TMx and TEx polarizations[1] were examined with the understanding that the method used by Balanis is exact only for an infinite plate and thus those results become increasingly inaccurate as look angles move away from the specular-reflection direction.

Like the cylindrical rod, a grid-refinement study was conducted in which the mesh-cell counts were varied between 75 and 150 cells across the height/width of the plate and 4 and 8 cells across the thickness of the plate. The mesh was clustered near the edges of the plate to approximately 150 cpλ in order to ensure proper capturing of the edge-diffraction phenomena. Additionally, wall spacings were varied between 175 and 350 cpλ. The outer boundary of the grid was placed at two wavelengths from the plate in all directions, and 20 cells were used to extend from the surface of the plate to the outer boundary. An xy-plane projection of a typical grid is shown in Figure 6.17. Because of the electrically large size of the plate, virtually no differences were noted in solutions obtained from the various mesh resolutions. This finding is consistent with observations for the high-frequency cylindrical rod since the electrical size of the longest dimension of the two bodies is very similar. Furthermore, because the incident wave strikes the plate nearly broadside, there are virtually no travelling wave phenomena. Finally, it is worth noting that the geometry itself precludes any creeping wave effects and thus the surface boundary condition does not adversely impact the solution to the degree that was seen in cylindrical rod cases. Because the results of the various mesh resolutions were virtually identical, only coarse-mesh results are presented here.

1. TMx: Magnetic field vector is transverse to x-coordinate direction.
   TEx: Electric field vector is transverse to x-coordinate direction.

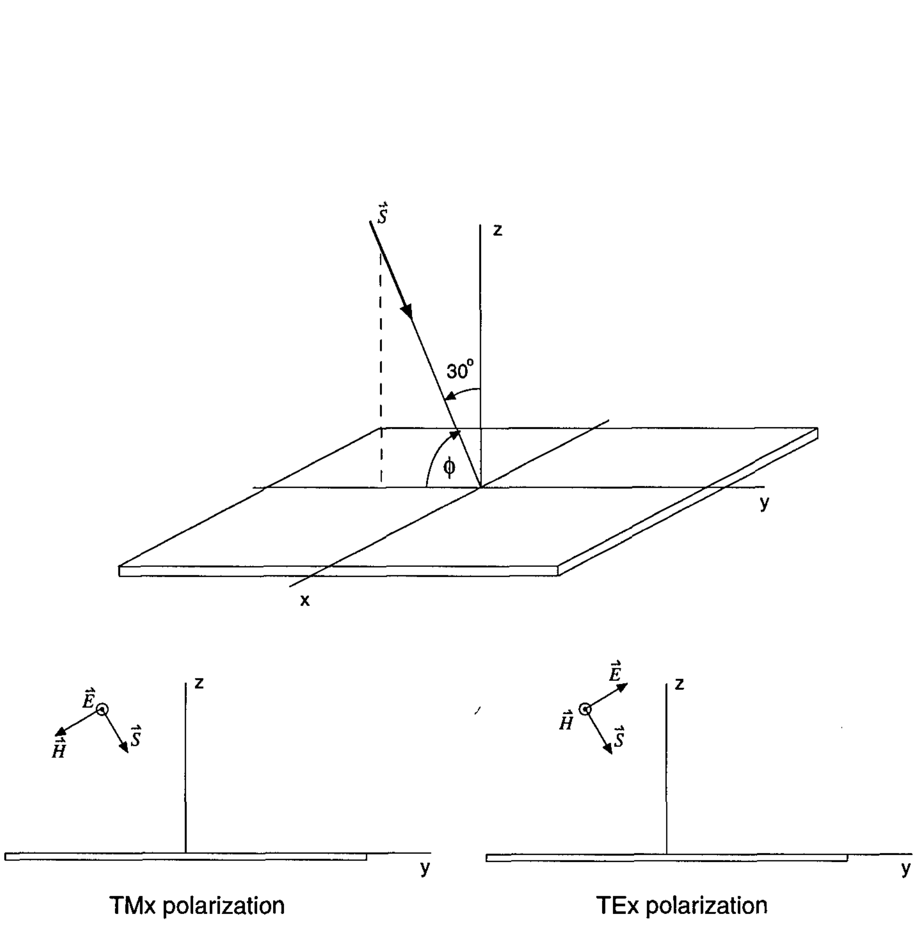


Figure 6.16: Flat Plate Problem Illustration

6.3.1.2 Results

In order to compare directly to the results presented by Balanis, two changes in RCS reporting convention were made: first, results are presented in dB per square wavelength (dBsw) rather than dBsm (this corresponds to a simple rescaling of the RCS data), and second, the look angle is defined with respect to the angle ϕ appearing in Figure 6.16. Note that this definition of look angle causes the backscatter direction to be shifted from 0 to 60 degrees, and thus the specular-reflection direction is given by $\phi = 120$ degrees.

The RCS results for the TMx and TEx polarizations are presented in parts *a* and *b*, respectively, of Figure 6.18. For both polarization cases, the CHARGE results agree extremely well with the results of Balanis for look angles near the specular-reflection direction; the results differ by only 0.11 and 0.04 dBsw for the TMx and TEx cases, respectively. As expected, there in an increasing disparity in the results as the look angle moves away from the specular direction. Because the results of Balanis are approximate, it is not entirely meaningful to conduct a detailed comparison of the two solutions for edge-on look angles. Nevertheless, it can be noted that the trends of the two solution schemes are similar for all look angles, although

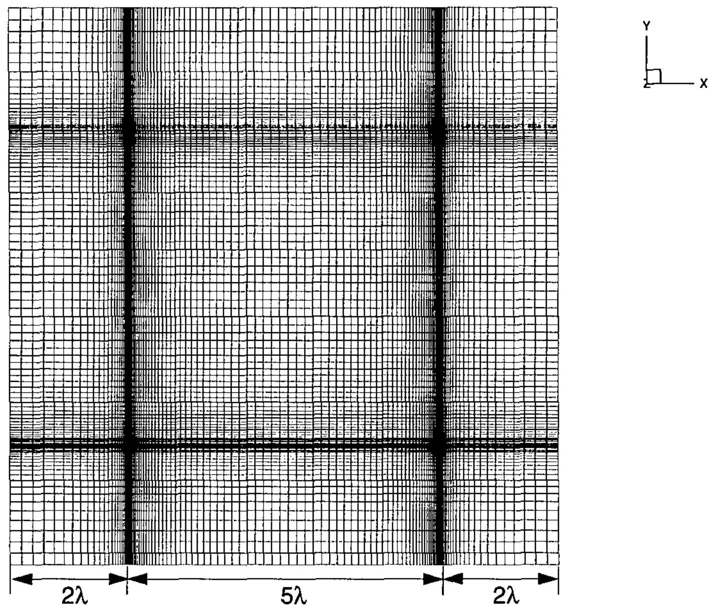
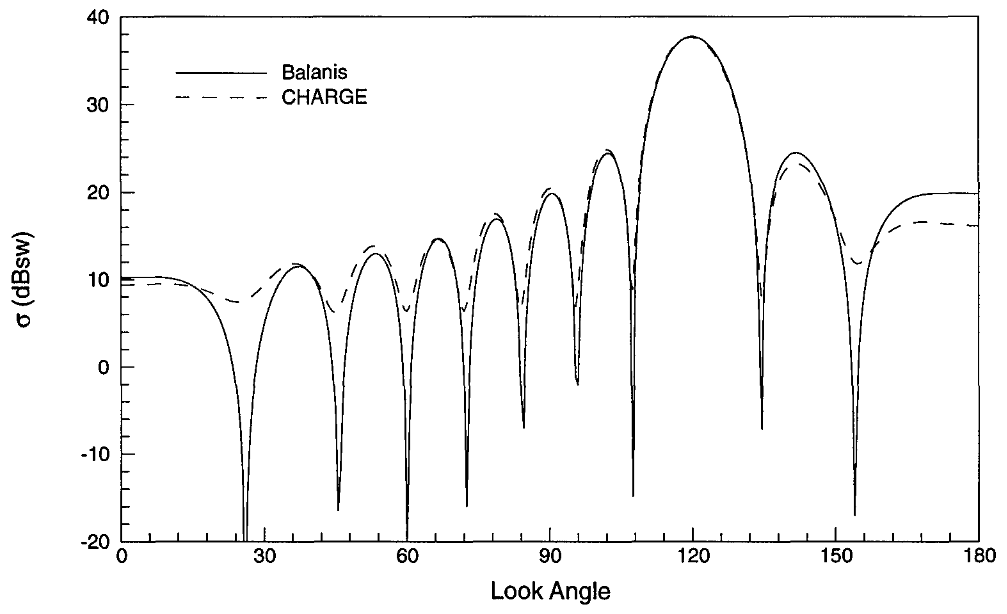


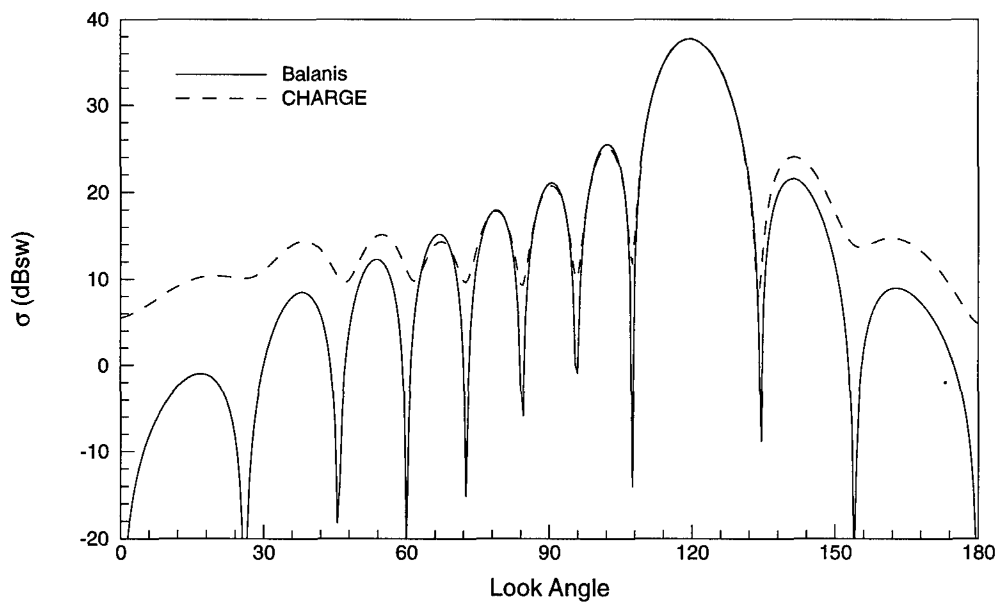
Figure 6.17: Large Plate Grid

the nulls obtained from CHARGE are substantially less pronounced than those from Balanis. It is interesting to note that plate thickness affected the results only slightly. Although the results are not presented here, reducing the plate thickness by a factor of 10 had virtually no effect on the RCS profile near the specular direction and only a 1-to-2-dBsw effect for look angles corresponding to edge-on views.

Further insight into the scattering characteristics of the flat plate geometry can be gained by examining the contours of the electric fields in the near zone surrounding the plate. These contours for the TMx and TEx polarizations are provided in parts *a* and *b*, respectively, of Figure 6.19. Contours are shown on the surface of the plate as well as on a *yz*-cutting plane passing through the plate approximately 1.25 wavelengths from a edge of constant *x*. In both parts of the figure, the edge of the plate is indicated by the dark line. The high degree of scattering in both the specular and forward-scattering directions is apparent for both polarizations. Furthermore, the edge diffraction from the front of the plate is quite strong for the TMx case and noticeably absent for the TEx case. Both of these observations are consistent with theoretical results [8].

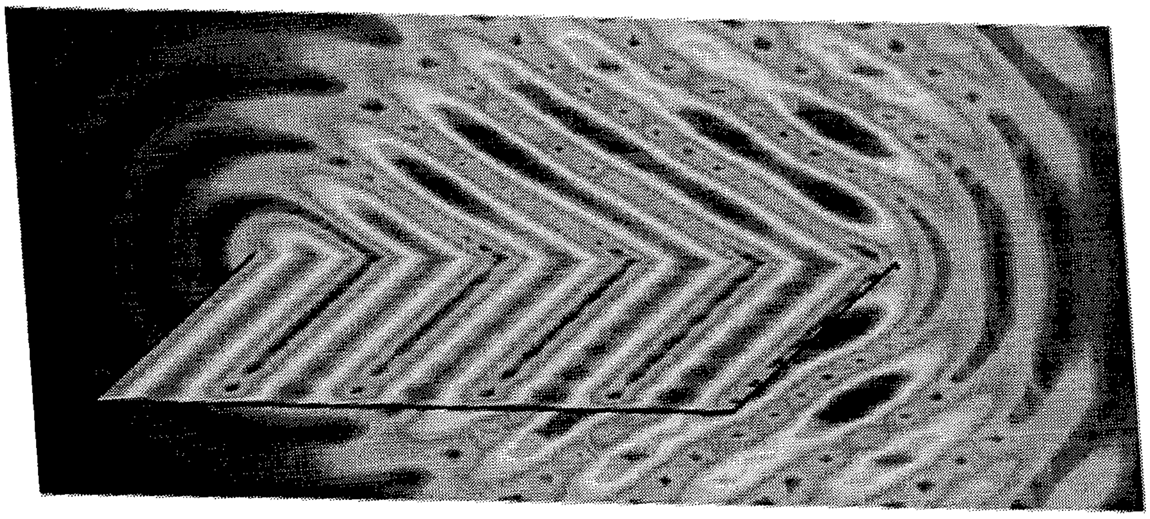


(a)

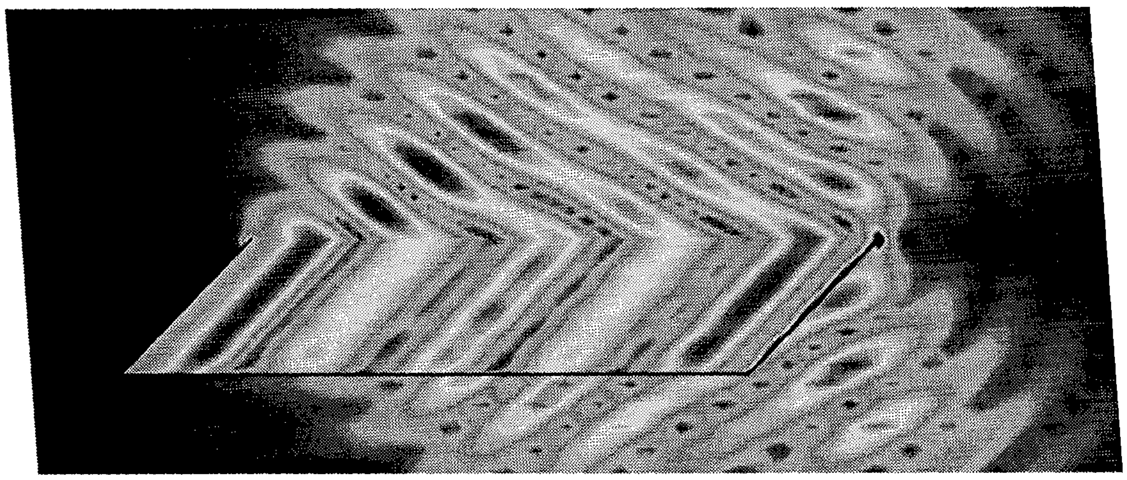


(b)

Figure 6.18: Large Flat Plate RCS Results: a) TMx Polarization, b) TEx Polarization



(a)



(b)

Figure 6.19: Near Zone Scattered-Electric-Field Contours: a) TMx Polarization, b) TEx Polarization

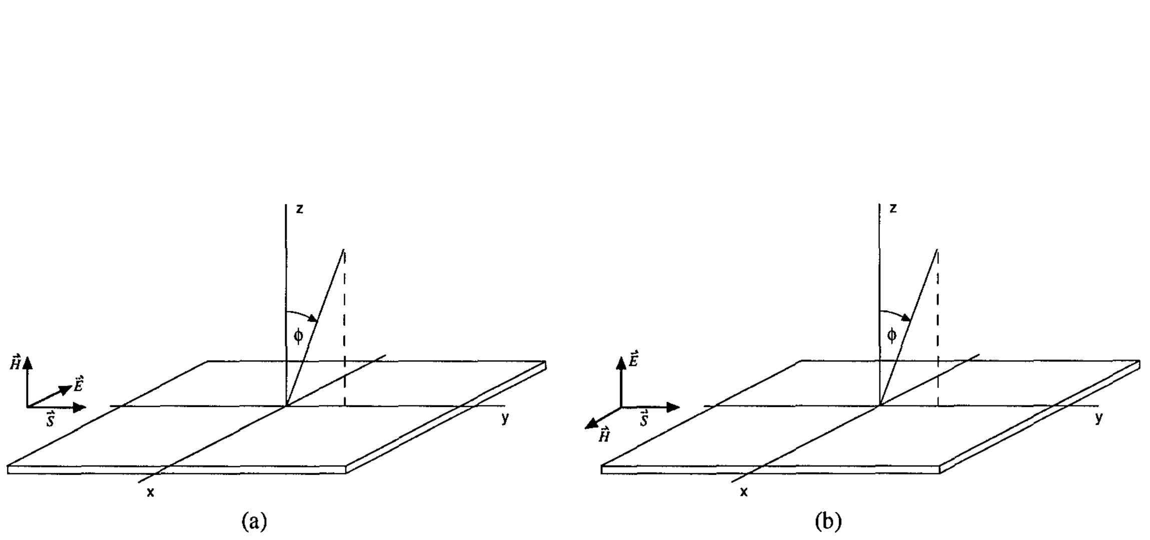


Figure 6.20: Small Plate Edge-On Incidence Descriptions: a) TMx Polarization, b) TEx Polarization

6.3.2 Small Plate Study

6.3.2.1 Grid Generation and Problem Description

With the exception of the size of the plate and the subsequent difference in the number of required grid cells, the geometry-generation process for the small plate was identical to that previously described for the large plate. For the small plate, 20 cells across the height and width of the plate and 4 cells across the thickness were found to produce a grid-converged solution provided the wall spacing was maintained at roughly 250 cpλ. For this geometry, two separate field-incidence angles were examined: broadside and edge on. For the broadside incidence, the Poynting vector was oriented along the z axis, and the RCS HH- and VV-polarization cuts were defined precisely as was done for the cylindrical rod studies. For the edge-on incidence, both TMx and TEx polarizations were examined. These polarizations, along with the convention used for the look angle, ϕ, are shown in Figure 6.20. The change in the look angle definition was required in order to facilitate comparison with results obtained using CARLOS[1] [77]. Note that because of this definition for RCS look angle, no backscatter results are reported for the edge-on incidence cases.

6.3.2.2 Broadside Incidence Results

The RCS results for broadside incidence are provided in Figure 6.21. For the VV-polarization cutting plane, which contains the relatively strong scattering from the y_{min} and y_{max} edges of the plate, the CARLOS and CHARGE results differ by only 0.72 to 0.93 dBsw across the entire range of look angles. For the HH-

1. CARLOS is a MoM code developed at McDonnell Douglas and is capable of computing the RCS of arbitrarily shaped bodies. All CARLOS and CICERO results appearing in this document were supplied by Wright Labs/XPN.

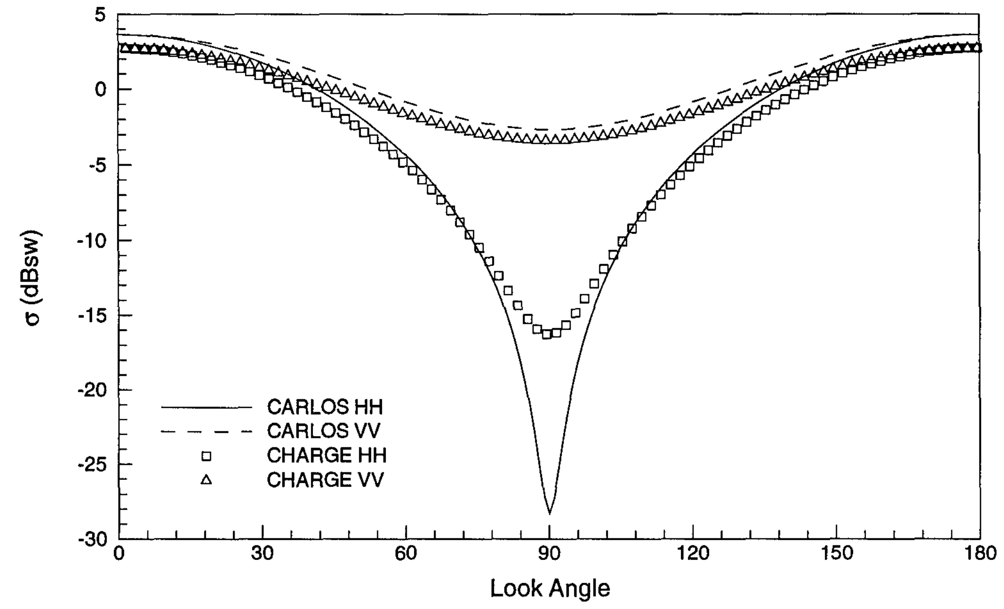


Figure 6.21: Small Plate Broadside-Incidence RCS Comparisons

polarization plane, however, the difference is more substantial, especially as ϕ approaches 90 degrees which corresponds to an edge-on view of the plate. At $\phi = 90$ degrees, the results differ by approximately 11.7 dBsw. This disparity in null regions between results obtained using the flux-vector-splitting algorithm and the MoM has been observed by other researchers [48,133], and is most likely related to the extrapolation surface-boundary-condition implementation.

6.3.2.3 Edge-On Incidence Results

The general trends in agreement between the CARLOS and CHARGE results observed for broadside incidence were found to remain generally unchanged for the edge-on incidence cases. This is evident from results contained in Figure 6.22. For the TMx polarization case (in which edge diffraction is more dominant) the results differ by a maximum of 7.9 dBsw in the null regions. For $\phi = 90$ degrees, which corresponds to the forward-scattering direction, the results agree to within 1.2 dBsw. This can be compared to a difference of 3.5 dBsw in the forward-scattering direction for the TEx polarization case. Although these differences may seem large, in actuality the RCS computations for the TMx and TEx polarization cases agree to two and five decimal places, respectively. It is apparent from these results that the FVSRK2 algorithm is capable of computing the general scattering behavior from flat plates of various electrical size; however, there are clear

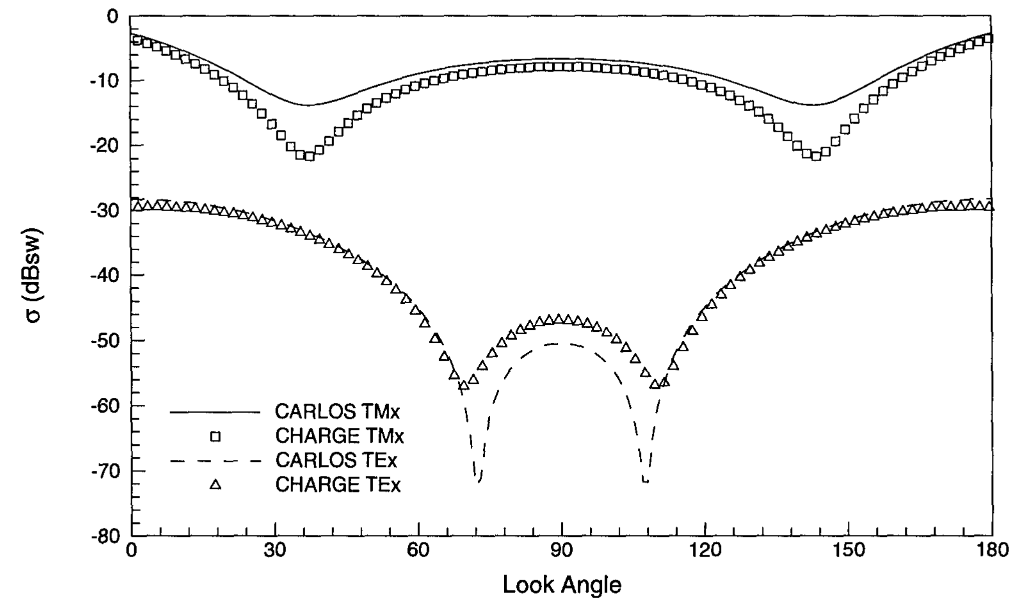


Figure 6.22: Small Plate Edge-On-Incidence RCS Comparisons

differences, especially in null regions, between results obtained from CHARGE and CARLOS. These discrepancies, although not overly disconcerting, warrant further investigation and must be considered when assessing the finned-missile results presented in Chapter 8.

6.4 Chapter Summary and Conclusions

This chapter focused on computational-accuracy issues surrounding the FVSRK2 algorithm on single-grid problems. Results were presented primarily for the cylindrical rod and the flat plate since these geometries serve to incrementally develop the work in the following chapters and because they have not been previously investigated using the FVSRK2 algorithm. The results show the algorithm is capable of resolving the physics associated with electromagnetic scattering using surface-mesh densities of 20 to 25 cpλ for lower-frequency simulations and 15 to 20 cpλ for the higher-frequency cases. This work documents the previously unexplored areas of mesh stretching and surface packing and identifies wall spacing as the single most important factor in obtaining accurate results given the present surface-boundary-condition implementation. Wall spacings as high as 1000 cpλ were found to be required for simulations in which surface-wave phenomena are important. Using these results as a comparative baseline, the analysis is expanded in the following chapter to include issues particular to overset grids.

VII. Investigation of Overset Grid Applicability to FVTD

This chapter extends the single-grid analysis appearing in the previous chapter to include overset grids. Specifically, a number of wave-propagation and scattering problems are presented. Wave-propagation simulations center on the rectangular waveguide and are designed to compare errors generated by the interpolation procedure, the extrapolation boundary condition, and the FVSRK2 algorithm. Once the errors are quantified, results from single- and dual-sphere scattering simulations are presented. The single-sphere simulations are used to compare RCS results obtained using single and overset grids and to ascertain whether the interpolation errors significantly affect RCS computations. The dual-sphere simulations extend this analysis to a more complex configuration and demonstrate the utility of the overset-grid process.

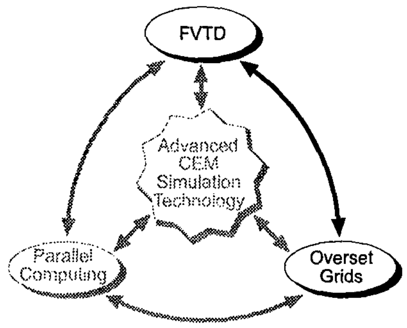


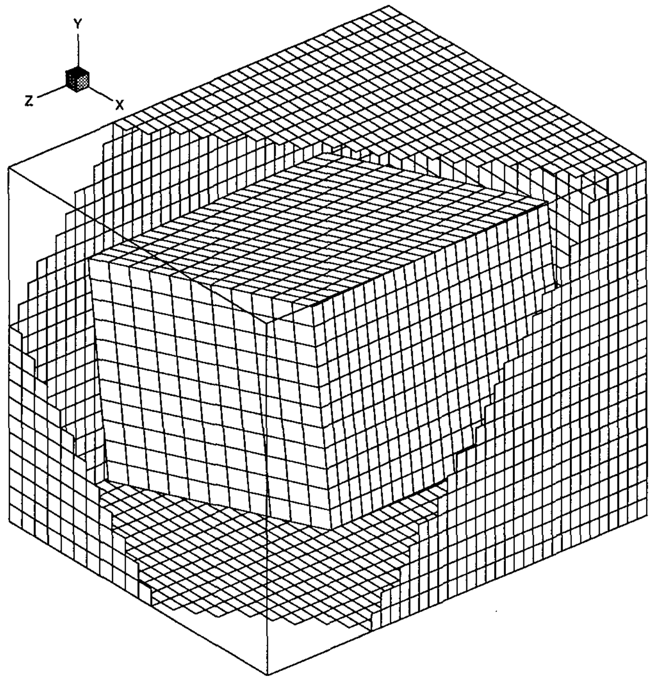
Figure 7.1: Chapter 7 Emphasis

7.1 Wave Propagation Simulations (The Rectangular Waveguide)

The rectangular waveguide was previously discussed during the domain-decomposition study of Chapter 4 (see Figure 4.4 and the accompanying discussion in Section 4.3.1.1). Here in Chapter 7, the geometry is used for a different purpose: namely, to conduct an analysis designed to compare the errors generated by the various cell-updating mechanisms (e.g., boundary condition, FVSRK2, and interpolation) contained in CHARGE. Selection of the waveguide for this analysis follows naturally due to the simplicity of the geometry and the existence of an analytical solution [111] which facilitates a ready assessment of any numerical error. For the waveguide simulations, error analysis is conducted in the time domain by examining the exact and computed electromagnetic field values. This is in contrast to the error analysis for scattering simulations appearing in Chapter 6 which was conducted in the frequency domain by studying RCS results. By examining the electromagnetic fields directly, additional insight can be gained which might otherwise be masked by the Fourier-transformation process.

7.1.1 Geometry Generation and Problem Description

The primary difference between the configuration studied here and that presented in Chapter 4 lies in



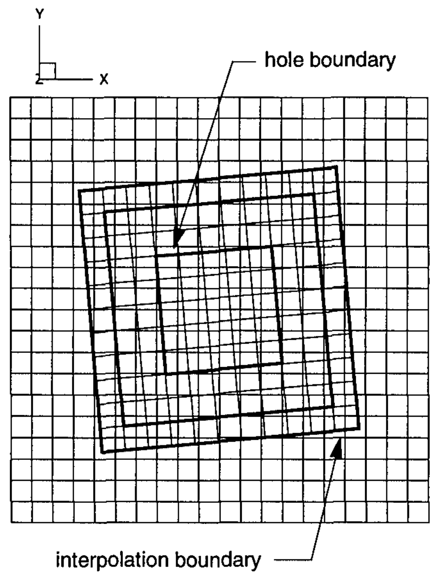


Figure 7.2: Waveguide Embedded-Grid Configuration: a) Three-Dimensional Cutaway View, b) Two-Dimensional Projection Showing Interpolation and Hole-Cutting Boundaries

the gridding scheme used. While the work of Chapter 4 utilized a single structured grid, for this study the computational domain of the waveguide was comprised of two uniform Cartesian grids, one of which was embedded within the other. The embedded grid could be scaled and rotated so that its local coordinate system was arbitrarily oriented with respect to the outer grid. The outer grid was dimensioned to $\pi \times \pi \times 1$ units, and each side of the embedded grid was dimensioned to 0.65 times the length of the corresponding side of the outer grid. In order to ensure a variation in spacing between interpolation-data-receiving and donor cells, the inner grid was rotated by five degrees about its y and z axes. A three-dimensional cutaway view of this configuration appears in Figure 7.2. In the figure, the z axis has been magnified to show the grid detail with greater clarity. Also contained in the figure is a two-dimensional xy-plane projection of a cross section of the waveguide. This illustration depicts the hole-cutting boundary which was defined with respect to the embedded grid. Any cells in the outer grid which fell inside this boundary were classified as hole cells and removed from the computational domain. Furthermore, those cells which bordered the hole boundary were updated via tri-linearly interpolated data from the embedded grid. Finally, those cells on the periphery

Case Number	Resolution (cpλ)	Surface Boundary Condition	Grid Configuration
WG1	40	exact	single
WG2	40	extrapolated	single
WG3	40	exact	embedded
WG4	40	extrapolated	embedded
WG5	80	exact	single
WG6	80	extrapolated	single
WG7	80	exact	embedded
WG8	80	extrapolated	embedded

Table 7.1: Rectangular Waveguide Case Studies

of the embedded grid were updated via interpolated data from the outer grid. Surface boundary conditions were enforced on the periphery of the outer grid which was coincident with the waveguide walls, and the analytical solution was used for the two entrance and exit planes located at $z = 0$ and $z = 1$. The waveguide was excited in the TMZ_{11} mode, and the excitation frequency was chosen so that the overall electrical dimensions of the waveguide were $0.5\lambda \times 0.5\lambda \times 0.97\lambda$. In order to ensure accurate resolution of the fields inside the guide and to examine the error reduction with respect to mesh refinement, grid-cell densities of 40 and 80 cpλ were used. Because of the electrically small size of the geometry, this level of mesh refinement did not result in excessive computational-resource requirements.

Recalling the primary objective of the waveguide study was to characterize the various sources of error, several cases were examined. An investigation of the effects of interpolation was conducted by contrasting results obtained from the embedded-grid configuration described above with those obtained from a single-grid configuration. Furthermore, effects of the surface boundary condition were assessed by comparing results obtained using the extrapolation boundary condition with those obtained using the analytical (exact) boundary condition [111]. A summary of the cases appears in Table 7.1.

7.1.2 Error Norm Definition

The primary sources of error in CHARGE can be categorized into three groups: errors associated with the MUSCL-based FVSRK2 algorithm, errors associated with the extrapolation boundary condition, and errors associated with interpolation. In order to better characterize each of these error sources, the cells in the computational domain were categorized according to their method of update to a new time level: interpolation, surface boundary condition, or application of the FVSRK2 scheme. A norm for each cell type was then defined, viz.

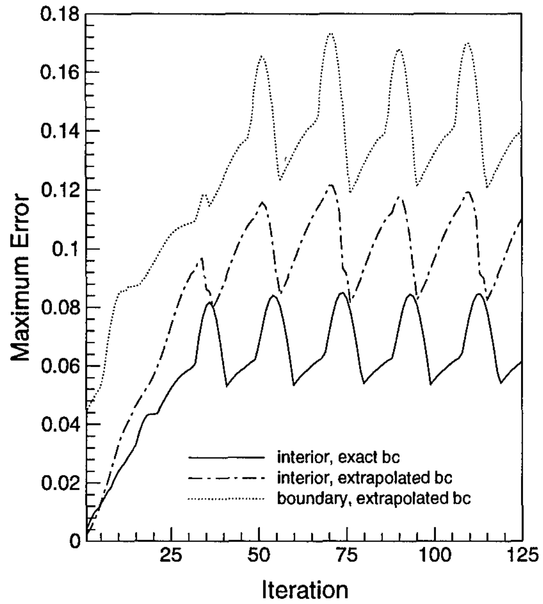
$$L_k = max_k|B_{ex} - B_{comp}| \quad (7.1)$$

where B_{ex} and B_{comp} represent the magnitude of the total exact and computed magnetic fields, respectively, and the *max* (maximum) operation was taken over all cells type k (k being either interpolation, boundary, or interior cells). Note that each cell in the computational domain was thus uniquely defined. With the geometry generated and the norm suitably defined, the waveguide was initialized to the exact solution at time $t = 0$, and the simulation was allowed to progress for a period long enough to establish a periodic error behavior.

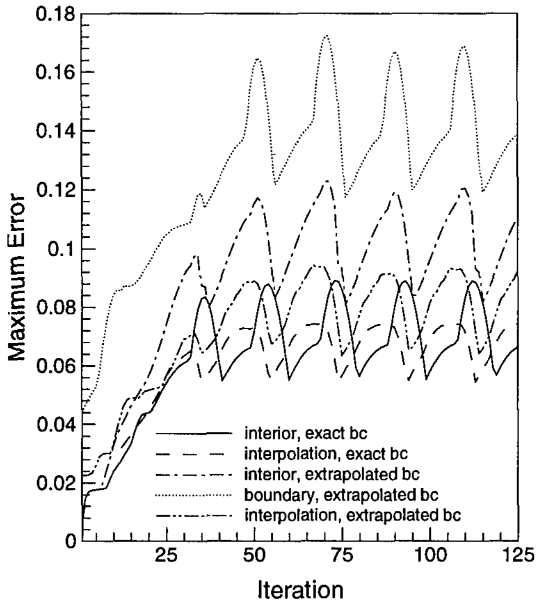
7.1.3 Results

The error results for the 40-cpλ waveguide simulations for the single- and embedded-grid configurations appear in parts *a* and *b*, respectively, of Figure 7.3. From Figure 7.3*a*, it is apparent that the maximum error for the single-grid, exact-boundary-condition case is approximately 0.084. This case (denoted by WG1 in Table 7.1) serves as a baseline since it represents the instance devoid of any interpolation or surface-boundary-condition error. When the extrapolation boundary condition is introduced (case WG2), the maximum error for the single-grid configuration jumps dramatically to 0.172, an increase of approximately 105 percent. Similarly large increases are evident for the embedded-grid configuration where a comparison of the maximum errors associated with the exact- and extrapolated-boundary-condition cases (cases WG3 and WG4) shows the extrapolation condition causes the error to increase by approximately 95 percent, from 0.088 to 0.172.

The fairly dramatic increase in error caused by the extrapolation surface boundary condition is contrasted by the small error increases affected by the interpolation process. This is evident by examining case WG1 in conjunction with case WG3. As stated previously, the maximum error for the each of two cases was 0.084 and 0.088, respectively. Thus, the addition of interpolation error to the baseline case causes the maximum error to increase by only 4.8 percent. This slight increase due to interpolation error disappears com-



(a)



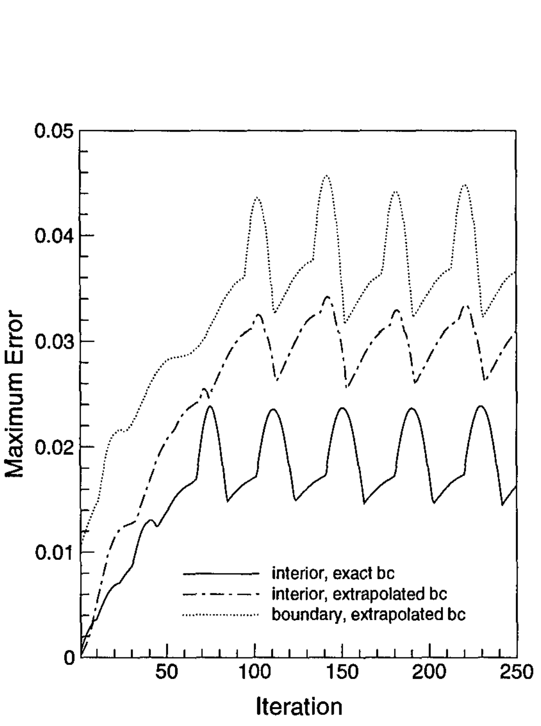
(b)

Figure 7.3: Rectangular Waveguide Error Summary for 40 cpλ Mesh Density: a) Embedded Grid, b) Single Grid

pletely for the extrapolation-boundary-condition cases (WG2 and WG4) where the maximum error remains constant at 0.172.

The same effects were apparent for the results of the 80-cpλ cases which are presented Figure 7.4. For the baseline 80-cpλ case (case WG5), the maximum error was approximately 0.024. Case WG7 demonstrates that the addition of the embedded grid caused the maximum error to increase by only 4.2 percent, to 0.025. As in the case of the coarse mesh, in the presence of errors associated with the extrapolation boundary condition, the additional interpolation error had no effect on the overall accuracy of the simulation. From these results, there is little doubt that boundary condition errors are of far greater impact than interpolation errors for this type of wave-propagation simulation.

Although the effect of the interpolation error was shown to be slight, it is nevertheless beneficial to determine if the interpolation process degrades the practical order of accuracy of the overall solution scheme. To this end, a measure of the error behavior as a function of mesh refinement is presented in Tables 7.2 and 7.3. The tables list the approximate maximum-error values for each cell type for the various mesh



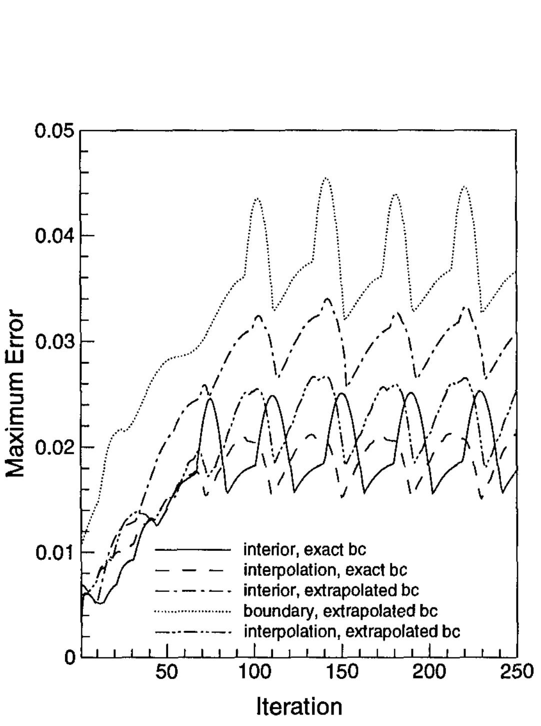


Figure 7.4: Rectangular Waveguide Error Summary for 80 cpλ Mesh Density: a) Embedded Grid, b) Single Grid

configurations. The error-reduction factor is simply the quotient of the maximum errors of the coarse and fine grids for a given cell type. In both embedded- and single-grid configurations, the errors decreased by slightly less than a factor of four as the mesh density was doubled. At first, this may seem contradictory to the third-order accuracy of the numerical scheme. In fact, however, the data reported in the table is a verification of the temporally second-order-accurate Runge-Kutta scheme since the time step was halved as the mesh density doubled. Of key importance is a comparison of the error reduction factors between the two tables. The embedded-grid cases performed as well as the single-grid cases as the mesh density increased. This indicates the interpolation errors do not reduce the practical accuracy of the global system for the cases studied.

7.2 Scattering Simulations

The wave-propagation simulations of the previous section were designed to assess the relative magnitudes of the errors associated with the FVSRK2, boundary condition, and interpolation procedures. With

Cell Type	40 cpλ Maximum Error	80 cpλ Maximum Error	Error-Reduction Factor
Interior (extrapolated bc)	0.119	0.033	3.606
Interior (exact bc)	0.082	0.024	3.417
Boundary (extrapolated bc)	0.172	0.045	3.822

Table 7.2: Error Reduction with Mesh Refinement for Single-Grid Waveguide

Cell Type	40 cpλ Maximum Error	80 cpλ Maximum Error	Error-Reduction Factor
Interior (extrapolated bc)	0.120	0.033	3.636
Interior (exact bc)	0.088	0.025	3.520
Interpolation (extrapolated bc)	0.093	0.027	3.444
Interpolation (exact bc)	0.072	0.021	3.429
Boundary (extrapolated bc)	0.172	0.045	3.822

Table 7.3: Error Reduction with Mesh Refinement for Embedded-Grid Waveguide

those errors quantified, several scattering simulations were devised in order to analyze the impact of the interpolation error on RCS results. In order to conduct this analysis, both single- and dual-sphere geometries were examined. The single sphere was selected for examination since it possesses an analytical solution [114] and because both single and overset grids are easily generated for the geometry. This facilitates a ready comparison of results obtained using the two gridding schemes. Like the single sphere, the dual sphere possesses an analytical solution [68]; however, its increased geometrical complexity serves to demonstrate the utility of the overset-grid process.

7.2.1 The Single Sphere

7.2.1.1 Geometry Generation and Problem Description

The primary objective of the single-sphere simulations was to compare results obtained using both single and overset grids in an effort to quantify interpolation error effects on RCS calculations. The single-

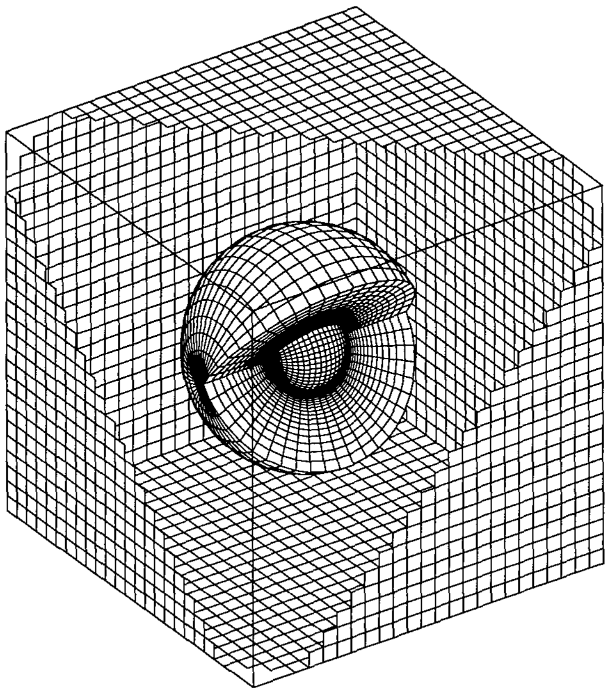


Figure 7.5: Single Sphere Chimera-Grid Configuration

grid configurations served as baselines for the comparisons and were generated using a body-conformal spherical grid. From this baseline geometry, the overset-grid configurations were obtained by simply embedding the spherical grid in a uniform Cartesian background grid. A typical overset-grid configuration is presented in Figure 7.5. Spherical grid mesh densities of 10 and 20 cpλ and wall spacings of 100 and 500 cpλ were used to examine grid-refinement issues. Furthermore, in order to study outer-boundary-placement effects, the distance between the sphere surface and the outer boundary was varied over the range of 0.5 to 3.0 and 0.5 to 2.0 wavelengths for the single- and overset-grid configurations, respectively. In all overset-grid cases, the outer boundary of the Cartesian grid was fixed at 3.0 wavelengths at its closest point to the sphere surface. Finally, in order to examine background grid mesh-resolution effects, grid-cell counts were varied between 31 and 71 cells along each coordinate direction. A summary of the cases examined appears in Table 7.4. Cells on the outer boundary of the spherical grid were updated via the far-field boundary condition described in Appendix A for the single-grid cases and via interpolation for the embedded-grid cases. Results in the following section are presented for the case in which the sphere was illuminated by a linearly polarized plane wave (ka = 5.235 where k is the wavenumber and a is the sphere radius).

Case Number	Spherical Grid			Background Grid	
	Resolution	Outer Boundary Placement	Wall Spacing (cpλ)	Resolution	Outer Boundary Placement
SS1	21 x 27 x 53	3 λ	100	N/A	N/A
SS2	21 x 27 x 53	2 λ	100	31 x 31 x 31	3 λ
SS3	21 x 57 x 105	.5 λ	500	N/A	N/A
SS4	21 x 57 x 105	1 λ	500	N/A	N/A
SS5	21 x 57 x 105	2 λ	500	N/A	N/A
SS6	21 x 57 x 105	3 λ	500	N/A	N/A
SS7	21 x 57 x 105	.5 λ	500	71 x 71 x 71	3 λ
SS8	21 x 57 x 105	1 λ	500	41 x 41 x 41	3 λ
SS9	21 x 57 x 105	2 λ	500	31 x 31 x 31	3 λ
SS10	21 x 57 x 105	2 λ	500	51 x 51 x 51	3 λ
SS11	21 x 57 x 105	2 λ	500	71 x 71 x 71	3 λ

Table 7.4: Single Sphere Case Studies

7.2.1.2 Results

Cases SS1, SS2, SS6, and SS9 were designed to determine if an overset-grid configuration was capable of producing results of similar accuracy (in terms of RCS calculations) to a single-grid configuration. Note that configurations SS1 and SS2 utilize identical spherical grid mesh densities and wall spacings. In case SS2, the spherical grid outer boundary is simply moved toward the body in order to facilitate embedding the grid within the Cartesian background grid. A similar discussion holds for cases SS6 and SS9. A comparison of the HH-polarization RCS profiles of these configurations is presented in Figure 7.6. From the figure, it is apparent that the results of the SS1 and SS2 cases are nearly identical. The largest difference in the results of the two cases occurred in the null at 129 degrees and was limited to a mere 0.07 dBsm. Similarly, the SS6 and SS9 results differ by no more than 0.04 dBsm across the entire range of look angles. Clearly, in these instances, the overset-grid configurations produce results nearly identical to their single-grid counterparts. Although the purpose of this study was not to compare directly to the analytical solution,

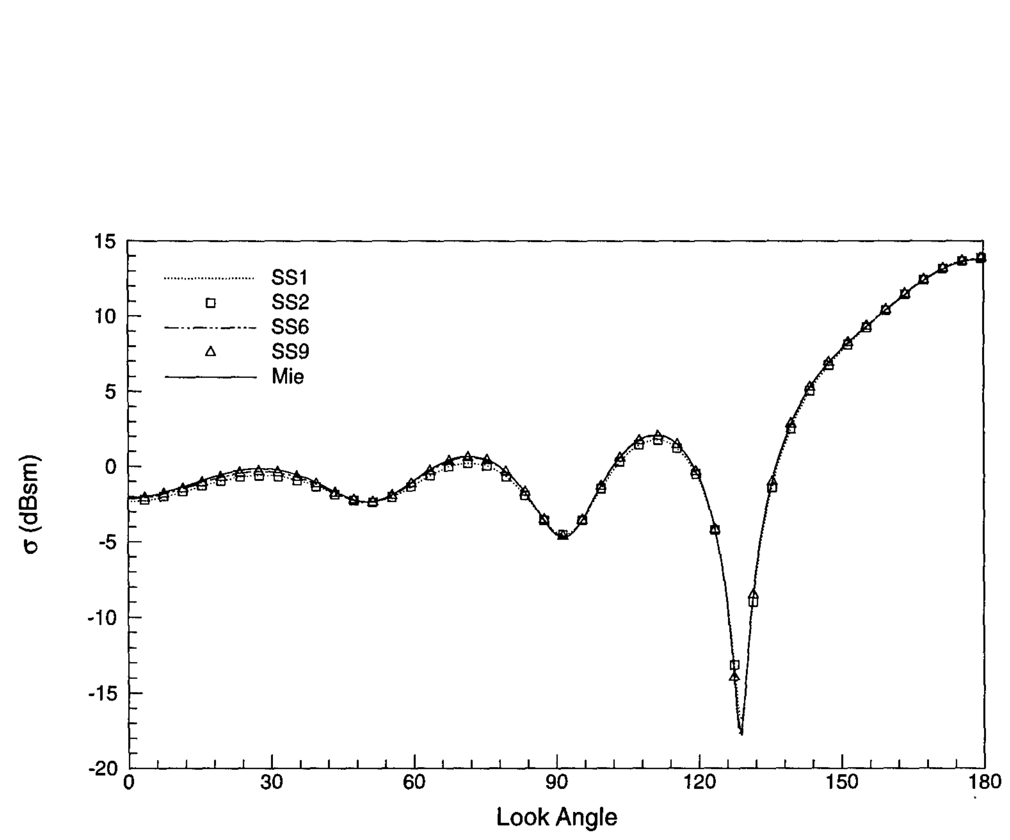


Figure 7.6: Single Sphere Overset Grid/Single Grid RCS Comparisons (HH Polarization)

it can be noted that the SS1/SS2 and the SS6/SS9 results agree with the Mie series calculation to within 0.8 and 0.19 dBsm, respectively, with the largest differences for both sets of results occurring in the null region.

In the results presented above, the outer boundary of the spherical grid was at least 2 wavelengths from the surface of the sphere. For single-grid simulations, the outer boundary of the computational domain must be placed far enough from the scattering object so that numerical reflections from the boundary are minimized [118]. Unfortunately, for overset-grid simulations, it is often not possible to place the outer boundary of an embedded grid more than a fraction of a wavelength from the scattering surface. Cases SS7, SS8, and SS9 were designed to examine any possible adverse effects occurring as a result of placing the interpolation boundary close to the body surface. Cases SS3, SS4, and SS5 were used as single-grid references against which to compare the overset-grid results. By examining results from the two sets of cases together, the relative effects of numerical reflection from the far-field and interpolation boundaries could be assessed.

To examine these effects, the spherical grid mesh density and wall spacing were fixed at 20 and 500 cpλ, respectively, and the surface-to-outer-boundary distance of the spherical grid was varied between 0.5 and 3.0 wavelengths for single-grid configurations and between 0.5 and 2.0 wavelengths for overset-grid

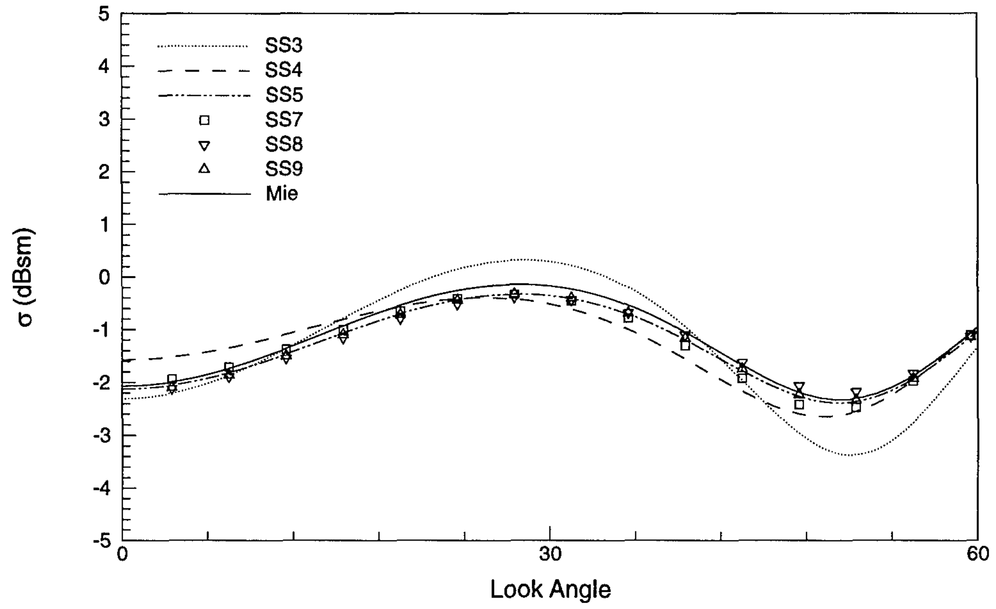


Figure 7.7: Single Sphere Outer-Boundary-Placement Effects for Single and Embedded Grids

configurations. The results for these cases for look angles between 0 and 60 degrees are presented in Figure 7.7. For a given look angle, the single-grid cases exhibit a variation of as much as 1.01 dBsm. In contrast, the overset-grid results differ by no more than 0.29 dBsm. Clearly, there is a slight-but-noticeable degradation of the solution. Thus, although numerical reflection from the interpolation boundary is not as severe as that from the far-field boundary, it cannot be neglected.

It should be noted that as the outer boundary of the spherical grid was brought closer to the sphere surface, the resolution of the background grid was increased. The increase in background-grid resolution was necessitated by the PEGSUS[1] code which requires sufficient spacing (in terms of number of mesh cells) between interpolation and hole-cutting boundaries. This increase in background-grid resolution was believed not to affect the results presented in Figure 7.7, since for single-body problems such as the sphere, the resolution of the background grid does not significantly affect the RCS results. This is evident from the data presented in Figure 7.8 which shows the effect of varying the background grid resolution while keeping all other parameters fixed. The results for the three different background grid resolutions differ by no more than

1. PEGSUS is discussed in Chapter 8.

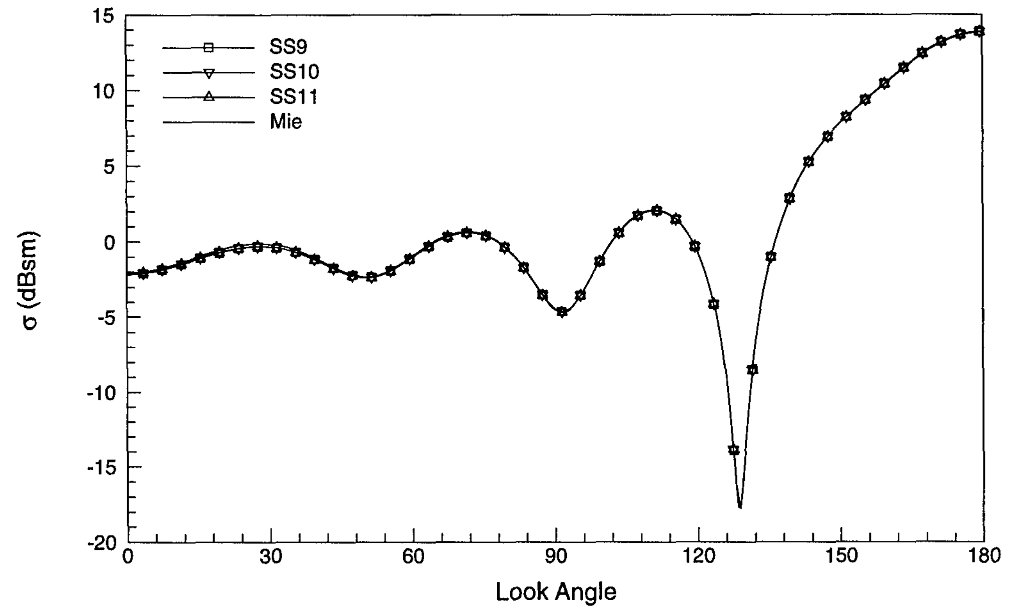


Figure 7.8: Single Sphere Background-Mesh-Resolution Effects

0.05 dBsm.

7.2.2 The Dual-Sphere Problem

Although overset grids can be used with the single-sphere problem, clearly the geometry does not require such a gridding scheme. The benefits of the Chimera methodology are much more apparent when the scattering geometry is more complex as is the case for the dual-sphere problem. The dual-sphere scattering problem has been investigated analytically by Liang and Lo [66] and both analytically and experimentally by Brunig and Lo [26,27]. In addition, Ludwig [68] has developed a generalized multipole technique (GMT) for computing the RCS of the dual-sphere geometry, and finally, Jurgens and Taflove [59] and Shankar [103] have investigated the problem using single-grid, time-domain algorithms. Because this problem has been investigated so thoroughly, the present study does not propose to provide new insight into the electromagnetic scattering phenomena associated with the geometry but rather to demonstrate a new approach for solving this and other similarly complex configurations.

7.2.2.1 Geometry and Problem Specification

The dual-sphere geometry was first introduced in Chapter 5 where it was used to investigate overset-

Case Number	Spherical Grid Surface Mesh cpλ	Spherical Grid Radial Cells	Spherical Grid Wall Spacing (cpλ)	Normal Spacing	Background Grid cpλ
DS1	15	20	20	Uniform	7.5
DS2	20	20	20	Uniform	10
DS3	20	20	500	Stretched	5.6
DS4	20	20	1000	Stretched	5.6
DS5	20	40	1000	Stretched	10
DS6	20	40	2000	Stretched	10
DS7	20	30	2000	Stretched	10

Table 7.5: Dual Sphere Grid Configurations

grid domain-decomposition techniques. Here, the focus is placed on accuracy issues. Referring to Figure 5.3, the two PEC spheres, each of diameter 1 wavelength, were centered on the z axis with a center-to-center separation distance of 2 wavelengths. The spheres were illuminated by a linearly polarized plane wave ($ka = \pi$) travelling in the z direction. Each sphere was gridded with a body-conformal grid and then embedded within a uniform Cartesian background grid. The outer boundary of each spherical grid was placed at one wavelength from the sphere surface and the Cartesian grid was sized so that its outer boundary was no less than three wavelengths from the nearest points on the surfaces of the spheres. Cells on the outer boundaries of the spherical grids were updated via interpolated data from the Cartesian grid. Furthermore, hole-cutting boundaries were defined with respect to each spherical grid, and cells in the Cartesian grid which fell inside those boundaries were removed from the computational domain. Cartesian grid cells on the periphery of any hole regions were updated via interpolated data from the appropriate spherical grid. Clearly, generating a single structured grid for this problem can become quite complex. The Chimera grid-generation process, on the other hand, is relatively simple.

A sample of the dual-sphere cases examined appears in Table 7.5. Several different mesh densities and spacing schemes were used in the radial direction on the spherical grids in an attempt to ensure the electromagnetic fields were properly resolved in the region between the spheres. In all cases, the background grid was kept relatively coarse since previous work showed that the background grid mesh density has only a

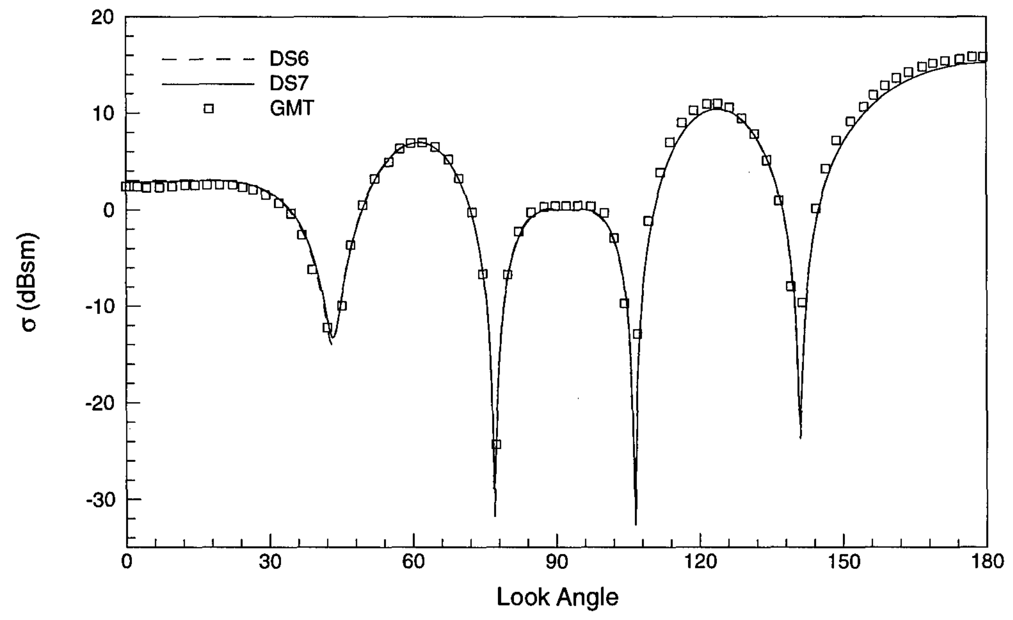


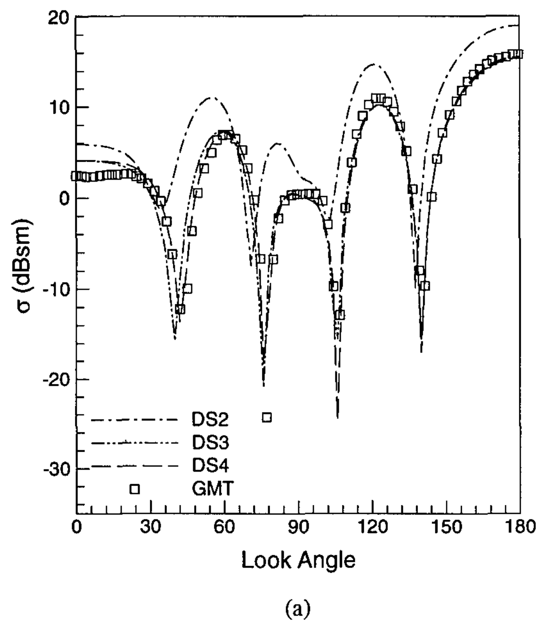
Figure 7.9: Dual Sphere Chimera Grid RCS Results

minor effect on solution accuracy.

7.2.2.2 Results

The HH-polarization RCS results for cases DS6 and DS7 are presented in Figure 7.9. The excellent agreement between the two sets of results indicate that 30 cells in the radial direction was sufficient to achieve grid convergence for this geometry. This translates to a mesh resolution of 30 cpλ in the region between the two spheres. The CHARGE solutions also exhibit excellent agreement to the GMT solution[1] and differ by only 0.46 and 0.66 dBsm in the backscatter and forward-scattering directions, respectively. It should be noted, however, that these results were obtained with wall spacings of 2000 cpλ. Other results obtained with less-refined wall spacings were not as satisfactory and are presented in Figure 7.10. An examination of Figure 7.10*a* reveals the effect of decreasing the wall spacing of the spherical grids. Beginning with the least-refined grid, case DS2 shows that a wall spacing of 20 cpλ produces results of extremely poor quality. As the wall spacing was refined to 500 and 1000 cpλ (cases DS3 and DS4), the results improved dramatically yet still differed from the GMT solution by approximately 2 and 4 dBsm, respectively, in the back-

1. The GMT results were digitized from the work of Jurgens and Taflove [59].



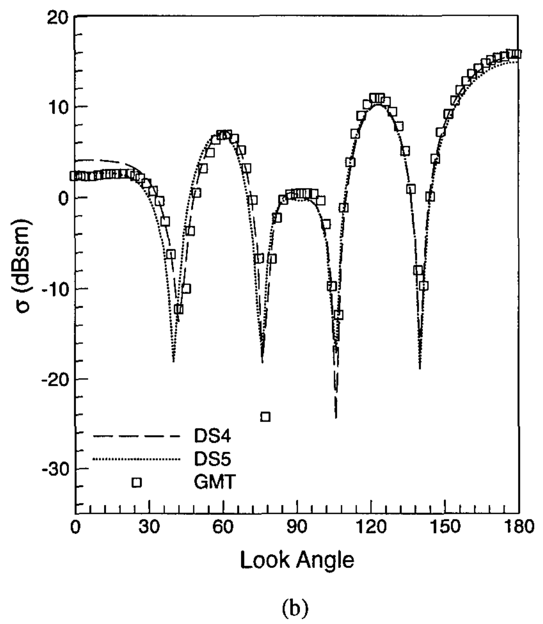


Figure 7.10: Effects of Wall Spacing and Inter-Sphere Mesh Refinement on Dual Sphere RCS Results: a) Wall Spacing Effects, b) Inter-Sphere Mesh Refinement Effects

scatter direction.

The effect of increasing the mesh density in the region between the spheres while holding the wall spacing fixed is shown in Figure 7.10*b*. Interestingly, although the results of case DS5 are in better agreement with the GMT solution in the backscatter direction, the results of the DS4 solution more closely match the trends of the GMT solution for look angles greater than approximately 30 degrees. These results, taken in conjunction with those presented in Figure 7.9 and 7.10a, underscore the fact that a sufficiently refined wall spacing is equally important in Chimera and single-grid simulations. Clearly, any errors introduced by the interpolation process affect the RCS computations to a much lesser extent than do inaccuracies introduced as a result of inadequate wall spacing.

7.3 Chapter Conclusions and Summary

Results were presented for overset-grid simulations conducting using CHARGE, the first-of-its-kind program capable of utilizing overset grids in conjunction with a FVTD Maxwell equation solver. Both wave-propagation and scattering simulations were conducted. The wave-propagation problems were designed to quantify the relative magnitudes of the errors associated with the extrapolation boundary condition, FVSRK2, and interpolation algorithms contained in CHARGE. It was shown that the interpolation errors are very small, especially when compared to the errors introduced as a result of the extrapolation boundary condition. For scattering simulations, results obtained using overset grids were found to be of comparable accuracy to those obtained via a single grid. Furthermore, the utility of the Chimera approach was demonstrated using the dual-sphere problem. In this case, no single-grid comparisons were presented due to the difficulty of generating such a grid for this geometry–a testament to the utility of the Chimera gridding process.

This chapter concludes the error analysis and study of the Chimera-grid implementation contained in CHARGE, and taken in conjunction with the results presented in Chapters 4 through 6, marks the end of the focus on the three separate objective areas undertaken as part of this work. Throughout these chapters, no emphasis has been placed on the CEM simulation environment created by the combination of these three separate but highly interrelated areas of study. That environment is discussed in the next chapter.

VIII. The CEM Simulation Environment

Before beginning the discussion on this chapter's topic, it is beneficial to summarize the results thus far presented. Each of the preceding four chapters emphasized one or more of this study's primary areas of focus: FVTD, parallel computing, and overset grids. These areas were identified in Chapter 1 as the principle components to be investigated in order to construct an environment for conducting complex numerical simulations of electromagnetic phenomena. While aspects of each of these disciplines were developed and associated results were presented, the exact method by which that CEM simulation capability was realized was not discussed. This chapter serves to fill that void. In the course of the discussion, some implementation details are provided; however, the focus is on the *solution process.* The discussion begins by identifying the important design criteria and then discusses the simulation process which was developed to satisfy those criteria. A case study is then provided which follows the process from generating the initial geometry to obtaining a computed solution.

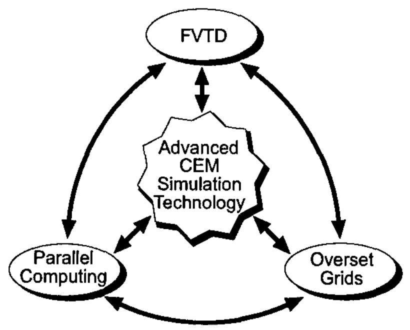


Figure 8.1: Chapter 8 Emphasis

8.1 Simulation Environment Design Objectives

The design objectives for the CEM simulation process are consistent with the overriding objective appearing in Chapter 1, namely, to create a versatile and flexible process for conducting complex numerical simulations of electromagnetic phenomena. In order to achieve this objective, several features were deemed necessary:

- to compartmentalize the functionality of each step of the solution process so that future changes directed in one area do not force modification of the entire process,
- to make the process general enough so that advances in solver technologies could be easily incorporated,
- to allow for a variety of approaches to discretization of the computational domain, and
- to exploit RISC-based parallel architectures.

Under these design objectives, the *P*arallel, *H*igh-performance, *A*rbitrary grid *S*imulation environment for *E*lectromagnetics (PHASE) was created. The following section summarizes the key components of the process encapsulated within PHASE.

8.2 The Solution Process Within PHASE

8.2.1 A Categorization of the Solution Steps

Consistent with the first design objective stated above, the steps of the process for conducting an FVTD simulation were categorized into four groups: geometry generation, interpolation-stencil computation, computational-domain-to-architecture mapping, and solution extraction. By compartmentalizing all functionality into one of the four groups, the goal of versatility is achieved. For example, by isolating all geometry-specific details to the geometry-generation step, the solver need not be modified each time a new geometry is investigated. In this manner, PHASE can dramatically reduce engineering problem-solving time[1]. The process within PHASE seeks to minimize problem-solving time rather than execution time for the solver. This is a subtle yet extremely important point, for under this precept, the solver is generalized to accept any type of geometry and grid configuration. This lack of specificity may result in slower execution times; however, the contention is that this reduction in execution speed is more than offset by the increase in the number and types of problems that can be effectively analyzed. The method by which this generality is achieved is discussed in the following section.

8.2.2 The Core Philosophy from an Object Perspective

The focus of this section is not to discuss the data structures used in any of the components of PHASE, but rather to illustrate the key difference between this approach and known earlier FVTD implementations. The approach begins with a basic examination of the nature of the computational domain. For the present study, the computational domain is defined in the manner described in Chapter 5, namely, as a collection or set of *cells*. Associated with each cell are data, namely, the six scalar components of the electric and magnetic fields. Also associated with each cell is a set of operations or *methods* that are performed on the cell in order to update the cell's data from one time level to another. Under this paradigm, a cell is an *object* and is classified according to its functionality within the domain. The current cell types that are sup-

1. Problem solving time, for the purposes of this work, is defined as the time that expires from the inception of geometry generation to time an accurate result is obtained.

Cell Classification	Description
FIELD	Cell updated via the core solver
SURFACE	Cell updated via surface boundary condition
FAR FIELD	Cell updated via far-field boundary condition
INSIDE CORNER	Cell which falls on an inside corner which cannot be updated via the standard surface boundary condition. Updated via averaged extrapolation
ORPHAN*	Cell which falls on an interpolation boundary, but for which PEGSUS cannot find a suitable interpolation stencil
INTERPOLATION*	Cell updated via interpolation

*Classification performed by PEGSUS

Table 8.1: Cell Classifications Used in PHASE

ported by CHARGE are given in Table 8.1. Classification of cells according to functionality rather than physical or logical location is a feature which permeates the PHASE process and is, in large part, the reason for its superior flexibility. While the cell types identified in the table were sufficient for all geometries examined in the present study, PHASE and CHARGE are designed so as to facilitate the addition of new cell types on an as-needed basis. To do so, the user simply defines a new cell type during geometry generation, and provides the appropriate set of methods to CHARGE with which to update the data for that cell. No concerns are paid in CHARGE to the logical or physical location of the cell. This is especially important in a parallel environment where the computational domains can be highly irregular which can lead to problems with more conventional cell-indexing strategies.

8.2.3 The Components of PHASE

Consistent with the categorization presented in Section 8.2.1, PHASE is comprised of four distinct components: geometry generation, interpolation stencil determination, domain decomposition, and FVTD solver. An overall view of the flow of data through PHASE is given in Figure 8.2, and a discussion of the functionality of the components of PHASE appears in the following paragraphs.

8.2.3.1 Step 1: Geometry Generation

The geometry-generation portion of PHASE requires more than just grid generation since a computa-

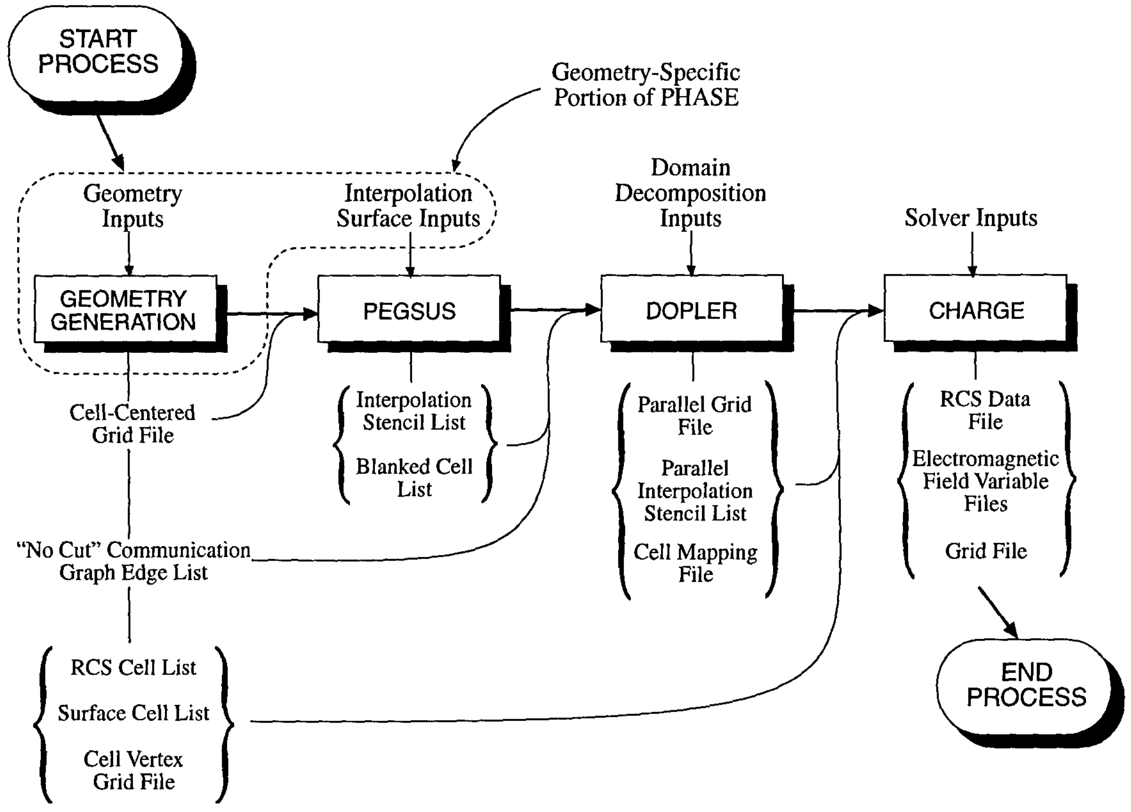


Figure 8.2: PHASE Process-Flow Diagram

tional grid is simply a collection of (x,y,z) locations of the centers or vertices of the cells which comprise the grid along with connectivity information as to the relative physical locations of the cells.[1] This amount of information is insufficient to specify a geometry since it omits an explicit definition of the surfaces (including surface normals) comprising the geometry. The geometry-generation step of PHASE requires that all this information be specified for later use. This is accomplished, in part, by appropriately categorizing each cell in the domain according to one of the types contained in Table 8.1. Furthermore, for those cells which comprise a surface, an appropriate surface normal must be specified. This information is contained in the "Surface Cell List" file depicted in Figure 8.2.

In most cases, the surface over which the RCS is computed corresponds to the surface of the scattering body. However, because situations can arise in overset grid environments for the need to separately

1. For a structured grid, the connectivity information is implicit; however, for an unstructured grid it must be explicitly stated.

define these surfaces, the cells in the computational domain which comprise the RCS surface must be separately identified via the "RCS Cell List" depicted in Figure 8.2. Note that there is no cell type "RCS" appearing in Table 8.1. This is due to the fact that cells are classified based on the method used to update the cell to the new time level rather than on any post-processing requirements of the scheme.

In addition to the issues discussed in the previous two paragraphs, additional information which must be produced during the geometry-generation phase is contained in the "Cell-Centered Grid File" and the "'No Cut' Communication Graph Edge List" file shown in Figure 8.2. These items are discussed in Sections 8.2.3.2 and 8.2.3.3, respectively.

8.2.3.2 Step 2: PEGSUS

In order to shorten development time, it was decided early in the design process to utilize existing computer code capabilities wherever possible. This motivated the choice of PEGSUS as the interpolation-stencil-computation step of PHASE. Because of this decision, the geometry-generation step must accommodate the requirements of PEGSUS. Thus, in order for PEGSUS to properly compute the interpolation distances which are referenced to cell centers, a cell-centered grid must also be generated in a format appropriate for PEGSUS. PEGSUS then uses the cell-centered grid to produce a set of interpolation stencils for those cells which must be updated via interpolation. To do so, the regions of the grids which form the surfaces must be identified to PEGSUS via the "Interpolation Surface Inputs" file depicted in Figure 8.2. Generation of this file (in a format acceptable to PEGSUS) marks the end of the geometry-specific input to PHASE. The functionality of PEGSUS and data-input requirements are fully documented by Suhs and Tramel [116] and are therefore not repeated here.

*8.2.3.3 Step 3: D*omain Decomposition

The first two steps of PHASE are focused primary on geometry specification rather than implementation. This step of PHASE provides the requisite mapping of the computational domain created in the first two steps to the computational architecture. This mapping is accomplished through the use of DOPLER[1], a computer code developed as part of the present study which embodies algorithms developed for partitioning structured, unstructured, and overset grids. DOPLER currently operates as a serial preprocesssor to CHARGE. Using the data provided by the geometry-generation and PEGSUS steps, DOPLER performs the

1. *DO*main decomposer for *P*aralle*L* *E*nvi*R*onments

following tasks:

- It removes from the computational domain any hole cells identified by PEGSUS.
- It reclassifies any cells identified as type ORPHAN or INTERPOLATION by PEGSUS.
- It builds an unstructured-grid representation of any input structured grids (required by CHARGE).
- It partitions the computational domain according to the method selected by the user while ensuring the integrity of any edges of the communication graph that have been identified as "no-cut" edges.
- It identifies all data which must be communicated by interprocessor exchange and builds the appropriate data structures to manage that exchange.
- It maps all structured- or overset-grid cell-indexing references (including those supplied by PEGSUS for the interpolation stencils) to a partition-based system which allows for partitions of arbitrary structure and size.
- It converts the grid files produced during geometry generation for use in a parallel environment.
- It produces a map table delineating the processor and local partition-based index for all cells in the computational domain (the "Cell Mapping File" of Figure 8.2).
- It produces the output files containing the grid- and interpolation-stencil-related data structures ("Parallel Grid File" and "Parallel Interpolation Stencil List" of Figure 8.2).

Consistent with the third design objective stated in Section 8.1, the actual domain-decomposition method is localized to a single function which is accessed via a function pointer within DOPLER. Thus, in order to expand the decomposition repertoire beyond that currently contained in the code, it is only necessary to add a new function containing the desired decomposition algorithm and set the function pointer (which is accomplished in the input file). The decomposition algorithm must take as its input a list of the cells in the computational domain. The list is then modified by placing those cells which are to reside on the same processor contiguously within the list. Finally, the decomposition algorithm returns to the main body of DOPLER a table which delineates the indices in the list where a new processor's cells begin. No other restructuring or modifications to DOPLER are necessary. Furthermore, because the mapping process is completely compartmentalized within DOPLER, changes to the decomposition approach do not force any modification to CHARGE.

Due to certain requirements of either the geometry or the solver, it may be necessary to specify limitations on how the compositional domain is decomposed. DOPLER provides the option of specifying any edges in the communication graph which must not be cut by the decomposition[1]. These edges are specified as part of the geometry-generation step in the "'No Cut' Communication Graph Edge List" file shown in Figure 8.2.

8.2.3.4 Step 4: CHARGE

With the computational domain suitably generated and mapped to the parallel architecture, the simulation is performed by using the methods described in Section 8.2.2 to advance each cell's solution in time until a suitable termination criteria is reached. As discussed in previous chapters, the termination criteria used in scattering simulations was the RCS convergence check. For wave-propagation simulations, the maximum number of time steps was specified. The numerics contained in CHARGE which are involved with the cell solution-updating process are discussed in depth in Appendix A and are therefore not repeated here. However, some of the important capabilities of CHARGE include

- the ability to operate on structured single- or multi-zone structured or overset grids,
- the ability to operate on an arbitrarily shaped or non-contiguous computational domain,
- the use of a fully unstructured data-structure paradigm that facilitates an easy path to a fully unstructured solver,
- an abstraction of the communication routines from the core solver which effectively isolates changes in the decomposition approach from changes in the solver,
- the flexibility for implementation on a variety of parallel or serial architectures using an arbitrary number of processors,
- the flexibility for the easy addition of cell-type definitions should the need arise, and
- the flexibility to allow for the replacement of the core solver with a new explicit algorithm if desired.

The last item in the list is especially important when considered in context with the second design objective of Section 8.1. This feature allows the process to be used for problems from disciplines other than CEM. In fact, the FVSRK2 algorithm is isolated to three functions within CHARGE. Furthermore, the CEM-specific portions of the FVSRK2 algorithm are isolated to a single function which computes the flux matrices associated with the Maxwell equations. By replacing this function with one designed to compute

1. Refer to Section 5.2 for the definition of a cut edge.

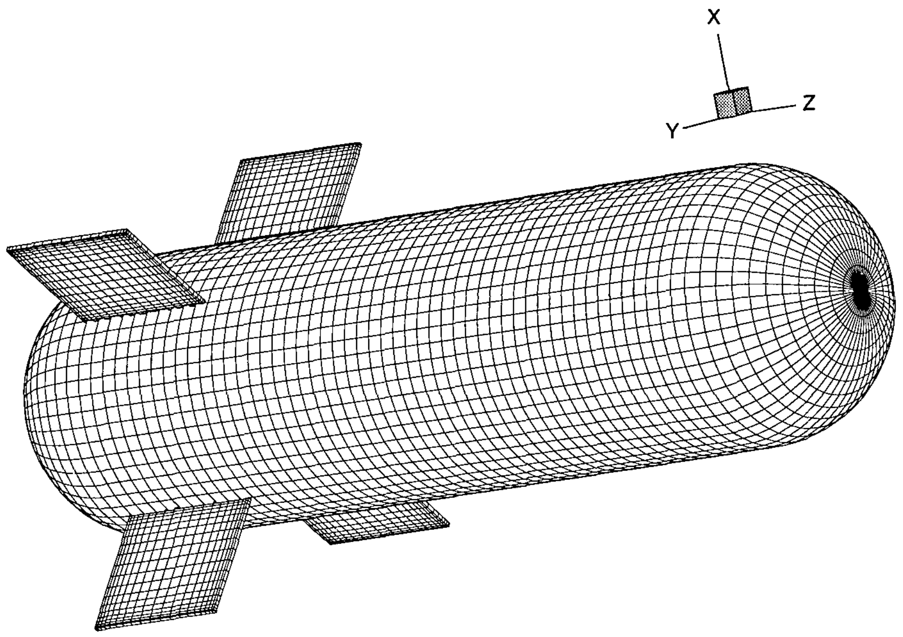


Figure 8.3: Finned Missile Geometry

the flux matrices of the Euler equations, for example, a completely different problem set can be solved.

8.2.4 The Capabilities of PHASE

It is believed that PHASE represents the most advanced and versatile parallel overset-grid implementation currently available for any solver type. It is the only known implementation for overset grids in conjunction with an FVTD solver. As a testament to the effectiveness of the process encapsulated within PHASE, all geometry studies presented in Chapters 6 and 7 were conducted using PHASE. Despite the obvious differences in both physical and gridding configurations, *no code modifications* were required to run any of the scattering simulations on any number of processors on any of the parallel machines used.

8.3 A Case Study: The Missile Problem

As a final demonstration of the capability of PHASE, a missile-like geometry (as depicted in Figure 8.3) was examined with an emphasis towards illustrating the solution *process.* This particular geometry was chosen for investigation both because of its relative complexity which lends itself to solution via overset grids, and because both fin and finless configurations have been previously investigated by Shaeffer [61,82] using a MoM code. This provided a comparative basis for results obtained from CHARGE and allowed for an incremental development path since the effects of the fins could be examined once the body-only solution was obtained. This is an inherent advantage to the Chimera-gridding process since it allows for geometries to be assembled on a component-wise basis with few changes required to the base gridding scheme as the geometry becomes more complex.

8.3.1 Step 1: Geometry Description

Like the previously examined cases, computation of the scattered electromagnetic fields surrounding the missile began with the definition of the missile geometry. The missile-body geometry was simply a cylindrical rod of length 2.6λ and radius 0.344λ. A cell-vertex-based grid was first generated analytically, and in the case of the finned missile, a cell-centered grid was derived from that grid by using the vertices for each cell to compute the coordinates of the cell centers. A cell-centered grid was not necessary for the finless configuration because a single grid was used, and therefore the PEGSUS portion of PHASE was unnecessary. Cells of the grid which defined the missile body were classified as type SURFACE while cells on the outer boundary of the body grid were classified as type FARFIELD (see Table 8.1). The remainder of the cells in the grid were classified as type FIELD. The exact mesh-density requirements were determined via a grid-resolution study which is discussed in Section 8.3.4.1.

For the finned-missile case, the fins were modeled as flat plates of finite thickness. The fins were 0.330λ in height (perpendicular to the axis of the missile), 0.390λ in length (along the axis of the missile), and 0.02λ in thickness. The aft ends of the fins were positioned at the juncture between the cylinder and the trailing hemispherical endcap as shown in Figure 8.3. The fin grids were constructed using mesh stretching so that cells were clustered near the fin edges in order to appropriately capture the expected diffraction patterns. A single fin grid was constructed which was then duplicated, rotated, and translated to the appropriate location on the missile body. An example of the fin grids showing two-dimensional front and side projections appears in Figure 8.4. The lower edge of the fin grid was conformal to the missile body. Cells along that boundary were defined as type SURFACE so as to be updated via application of the surface boundary condition within CHARGE. Cells comprising the fins themselves were also defined as type SURFACE noting the distinction between the surface normals of each side of the fin. Furthermore, cells located at the juncture of the fin/body were classified as INSIDECORNERCELLS. The remaining cells of the fin grid were defined as type FIELD with the realization that cells which would eventually be updated via interpolation

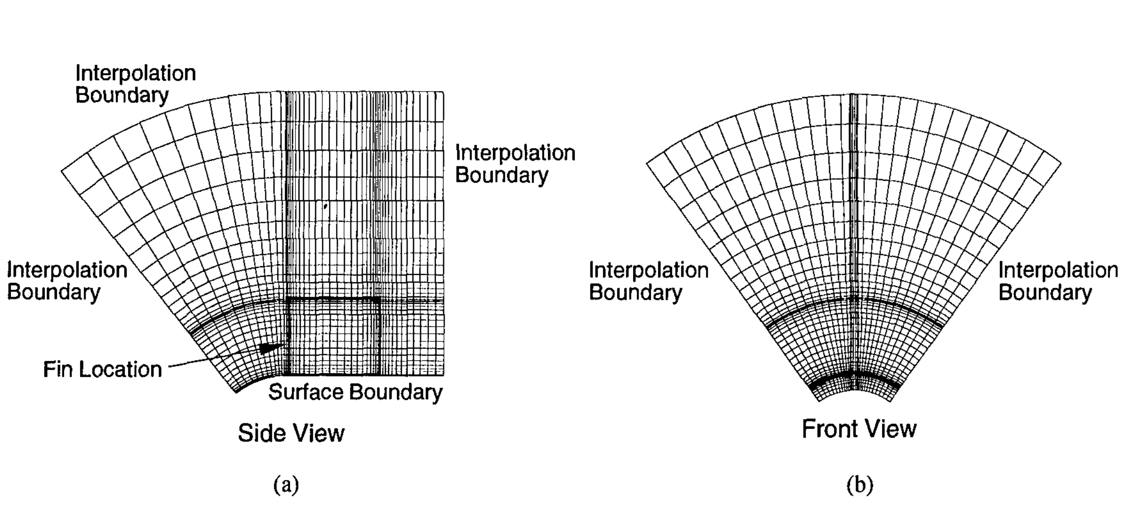


Figure 8.4: Missile Fin-Grid Detail: a) Side View (along missile axis), b) Front View

would be identified by PEGSUS and then appropriately reclassified by DOPLER.

In addition to grid generation, the first step of PHASE required the specification of the surface over which the current and charge distributions were integrated as part of the RCS calculation. Although the physical surface of the missile was used for this computation, the RCS cell specification differed from the list of surface cells in the finned-missile case due to the body-conformal boundary of the fin. Despite the fact that cells along this boundary were classified as type SURFACE, they were not included in the RCS calculation. This is because those cells were collocated with SURFACE cells on the missile body grid which were included in the RCS cell list.

The final step in the geometry-generation process was the identification of any groups of cells which were required to reside together on a given processor. Cells which must remain together are identified to DOPLER in the "'No Cut' Communication Graph Edge List" file depicted in Figure 8.2 using the notation

numcells gridnum1 cellnum1 gridnum2 cellnum2 ...

where *numcells* is the number of cells within the group to remain together, and *gridnumX* and *cellnumX* ($1 \le X \le numcells$) are the grid and cell identifiers for each of the cells in the group.

A situation where this grouping became imperative was for those cells of type SURFACE. As mentioned previously, the current surface-boundary-condition implementation performs an extrapolation of the electromagnetic field variables to a cell on the surface of the body from one cell off the body. This requires the cell from which the extrapolated information is derived to be at the new time level before the extrapolation can be performed. In a parallel environment, this requirement can effectively double the amount of interprocessor communication unless special precautions are taken. By grouping each surface cell with its corresponding neighbor one cell off the body, these additional communication requirements were avoided.

In the finned-missile case, additional cell "no cut" groupings were required for those cells identified as type INSIDECORNER. These cells at the fin/body juncture were updated by performing one-sided extrapolations of the surface values along the missile body and normal to the body along the fin and averaging the result. Thus, INSIDECORNER cells were forced to be updated *after* application of the surface boundary condition to the surrounding cells. Since two neighboring cells were required along each extrapolation direction, this translated to a total of nine cells which were required to remain together in each inside corner cell group.

With the "no cut" edges in the communication identified, the geometry-generation process for the missile was complete. A sample surface-grid configuration for the finned missile is presented in Figure 8.3 while a more complete cutaway view of the embedded-grid configuration is provided in Figure 8.5.

8.3.2 Step 2: PEGSUS

In the case of the finned missile, the 5-cell computational stencil required that the outer two grid planes on the periphery of the fin grid (with the exception of the planes on the body-conformal edge of the grid) be updated via interpolation. By specifying these grid planes as boundary planes, PEGSUS then appropriately computed intepolation stencils for the cells contained in the planes. By this method, information was transferred from the missile body grid to the fin grids. Conversely, by defining the periphery of the fins as hole-cutting boundaries, a means was created for transferring information from the fin grids to the missile-body grid. Cells on the missile-body grid which fell inside the hole-cutting boundaries were identified as hole cells by PEGSUS and interpolation stencils were computed for the cells surrounding the hole regions. A sample of the hole created in the missile body grid as a result of a fin hole-cutting boundary is shown in Figure 8.6.

8.3.3 Step 3: Domain Decomposition

With the interpolation stencils and hole cells identified, the computational domain was then decomposed via DOPLER. Although the algorithms and data structures of DOPLER are the most complex portion

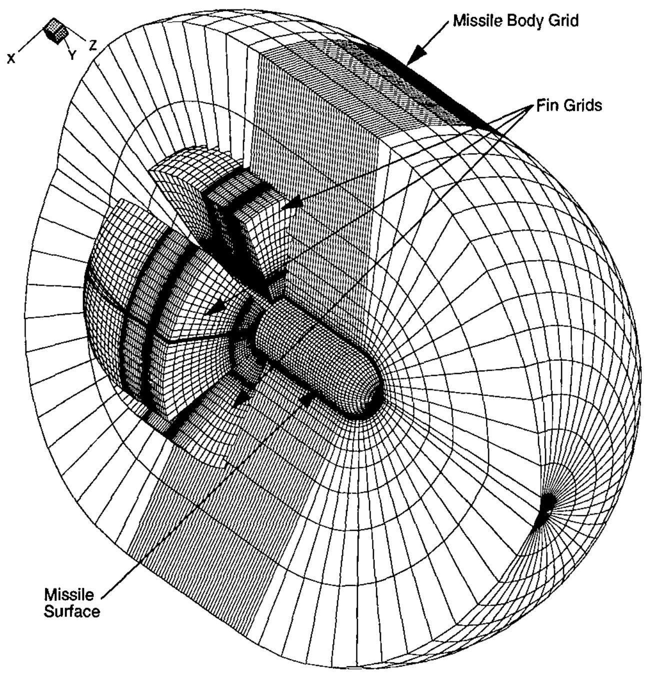


Figure 8.5: Finned Missile Embedded-Grid Configuration

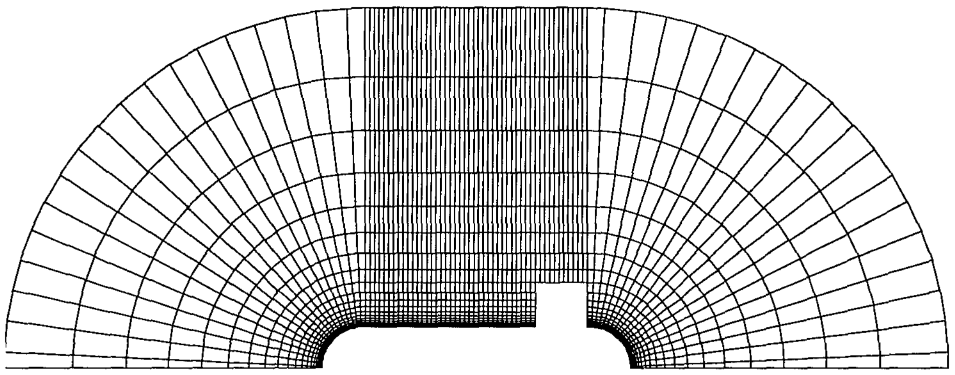


Figure 8.6: Cells Removed from Missile Body Grid by Fin Hole-Cutting Boundaries

of PHASE, from a user's perspective the decomposition process is extremely straightforward. The only user-controlled inputs to the program are the number of processors in the parallel environment and the decomposition method desired. For the missile geometry, cases were examined using all machines listed in Appendix B with processor counts ranging from 8 to 128. Because of the decomposition findings of Chapter 5, RCB

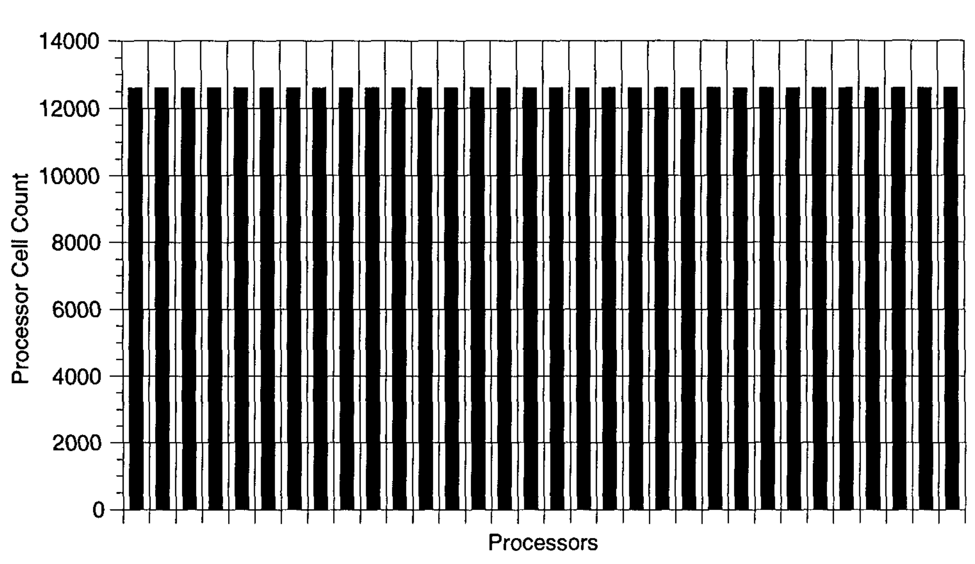


Figure 8.7: Sample Load-Balance Result for Finned Missile on 32 Processors

was used exclusively for all decompositions. The RCB approach allowed for portions of one or more fin grids to reside on the same processor as the missile body grid, thus reducing the communication requirements associated with the interpolation routines contained in CHARGE. Furthermore, the load imbalance that arose because of the "no cut" edges discussed in Section 8.3.1 was extremely small as shown by the sample decomposition given in Figure 8.7 which depicts the number of grid cells assigned to each processor using the RCB algorithm. Disregarding any differences in computational workload due to variations in the updating mechanisms for the different cell types, the load balance is nearly perfect.

8.3.4 Step 4: CHARGE Calculations

8.3.4.1 Electromagnetic Problem Description

Unlike the results presented in Chapters 6 and 7, both monostatic and bistatic calculations were performed for the missile in order to compare with the results of Shaeffer. Generation of the monostatic results required a large number of computer runs since each data point on a monostatic RCS curve required its own FVTD simulation. Before these monostatic runs were completed, a grid-convergence study was conducted on the finless missile configuration, which, although not presented here, indicated that roughly 25 cpλ surface-mesh density and approximately 1000 cpλ wall spacing was required to achieve a grid-converged solu-

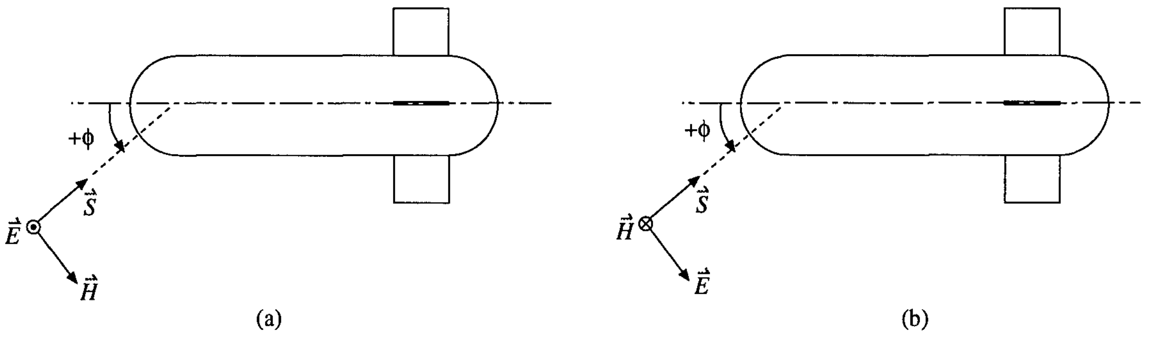


Figure 8.8: Incident Field Descriptions for Missile Geometry: a) VV Polarization, b) HH Polarization

tion. These values were ultimately used to generate the grid in the geometry-generation step described in Section 8.3.1.

For this study, the incident field was a linearly polarized, sinusoidally varying, uniform plane wave of frequency 2.6 Ghz ($ka = 2.16$, where k and a are the wavenumber and missile body radius, respectively). The method used to define the direction of incidence of the field is depicted in Figure 8.8. Using this definition, monostatic results were obtained for the finless missile for incidence angles over the range $0 \leq \phi \leq 90$ at 5-degree intervals. Comparisons were then performed between the results of CHARGE, Shaeffer, and CICERO. In the case of the finned missile configuration, bistatic comparisons were conducted with results obtained from CARLOS [77]. No finned-missile results from Shaeffer are presented for reasons discussed in Section 8.3.4.3.

8.3.4.2 The Finless Missile Monostatic RCS Results

Monostatic RCS HH- and VV-polarization RCS results obtained from CHARGE, Shaeffer, and CICERO are presented in Figure 8.9 and 8.10, respectively. From the figures, it is clear that the CICERO and CHARGE results agree extremely well. In fact, the maximum difference between the two sets of results was only 1 dBsm and occurred for a look angle of 25 degrees for the HH-polarization case. In contrast, the results of Shaeffer differ substantially from those of CICERO and CHARGE, especially for incidence angles less than 40 degrees. For example, the location of the first null in the VV-polarization case occurs at $\phi = 30$ and 35 degrees, respectively, for the CICERO/CHARGE and Shaeffer data. Furthermore, for $\phi = 0$ degrees, the RCS magnitudes differ by approximately 3.5 dBsm. The excellent agreement between the CHARGE and CICERO results generated a high degree of confidence in their quality and cast some doubt on the accuracy

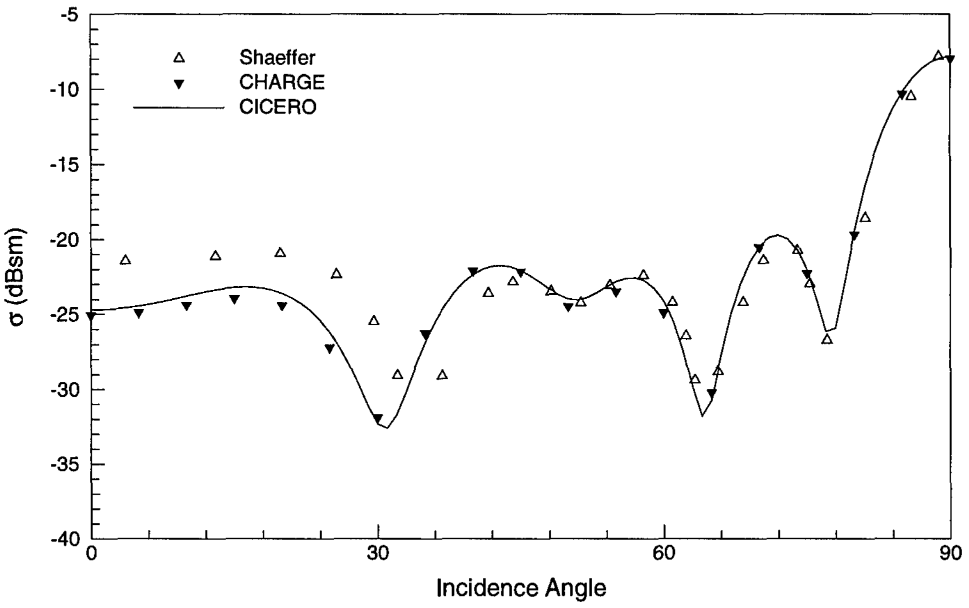


Figure 8.9: Finless Missile Monostatic RCS Results (HH Polarization)

of the data presented by Shaeffer. Furthermore, given the poor agreement between the CHARGE/CICERO and Shaeffer results for the finless missile, the results of the finned-missile configuration were not expected to compare favorably.

8.3.4.3 The Finned Missile Bistatic RCS Results

Because of the poor agreement between the results of Shaeffer and those obtained by CHARGE and CICERO for the finless missile, attempts were made to corroborate the finned-missile data of Shaeffer using CARLOS. Since Shaeffer used wire-loop fins rather than flat plates, some discrepancy was expected; however, the monostatic RCS results of the two MoM codes were found to exhibit little correlation and differed by as much as 30 dBsm. Because of these discrepancies, the finned-missile results of Shaeffer were not used to assess the accuracy of the CHARGE results. Instead, bistatic comparisons were performed for nose-on incident fields ($\phi = 0$) using both CHARGE and CARLOS. These results are found in Figures 8.11 and 8.12. Two sets of CHARGE results are presented to show the effects of mesh refinement. In the case of the coarse grid, the missile body grid used a surface-mesh resolution and wall spacing of 30 and 100 cpλ, respectively. Furthermore, packing was used rather than stretching in order to maintain adequate body-grid resolution in regions near the tips of the fins. The fin grids were resolved at 40 cpλ or greater in all directions

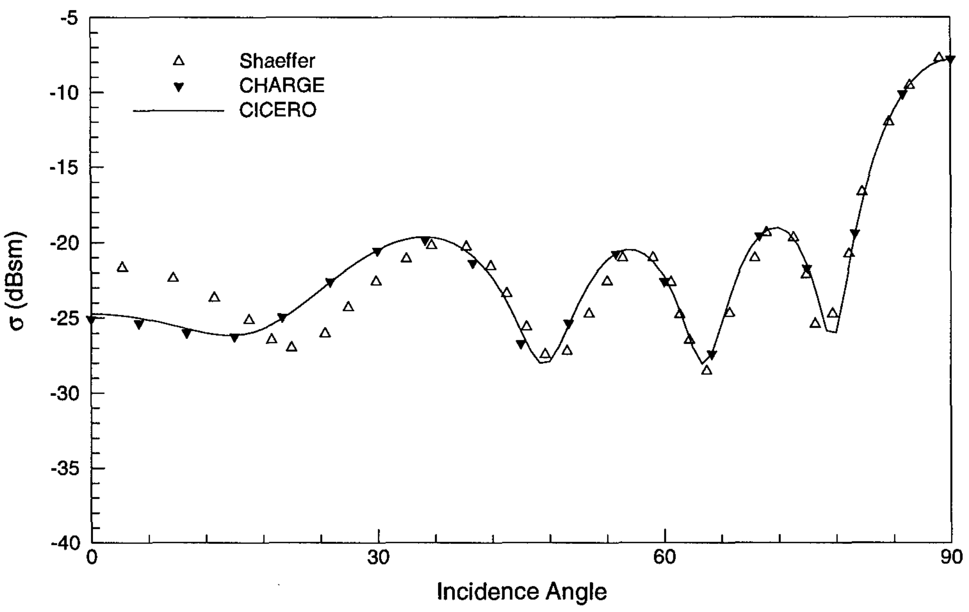


Figure 8.10: Finless Missile Monostatic RCS Results (VV Polarization)

with a wall spacing of 100 cpλ. For the fine-grid case, the missile body grid surface-mesh density was increased to 45 cpλ along the body of the missile and to 60 cpλ circumferentially in an attempt to better capture any interactions between the fins. Furthermore, the wall spacing was decreased to 300 cpλ. Although the fin grid density was kept constant, the wall spacing was reduced to 500 cpλ. This resulted in a grid of approximately 1.2 million grid points. Although the results from the two meshes differ by as much as 3.5 dBsw in the nulls, the overall trends are in close agreement.

The agreement in the general trends extends to the CARLOS results as well. With the exception of the first null in the HH-polarization results, the locations of the peaks and nulls as predicted by the two codes differ by no more than 3.5 degrees. Although there is a considerable difference in the predicted depth of certain nulls, it should be noted that computer problems prevented completion of a grid-convergence study for the CARLOS data [51], and thus it is difficult to precisely quantify the differences in the results.

Additional insight to the effectiveness of the overset-grid process can be gained by examining the magnitude of the electric field on the surface of the finned missile. This data is depicted in Figure 8.13 for the case where the incident field impinges on the missile body from an angle of ϕ = 30 degrees. The dark-

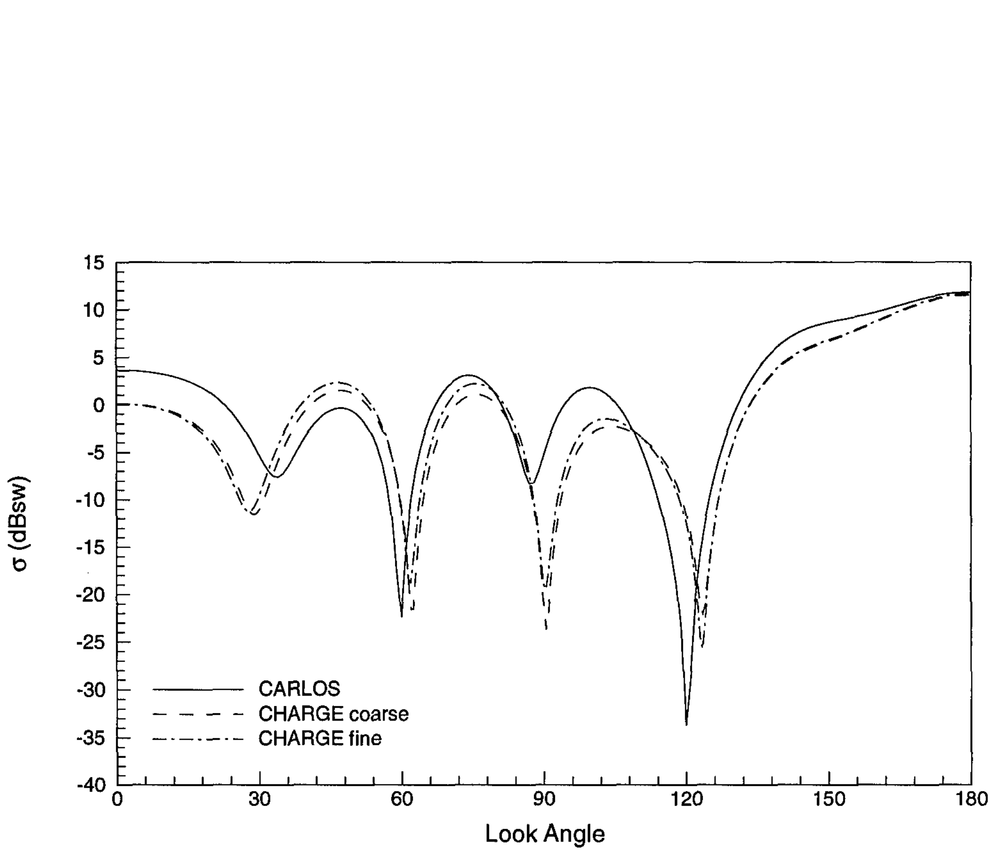


Figure 8.11: Finned Missile Bistatic RCS Results (HH Polarization, Nose-on Incident Field)

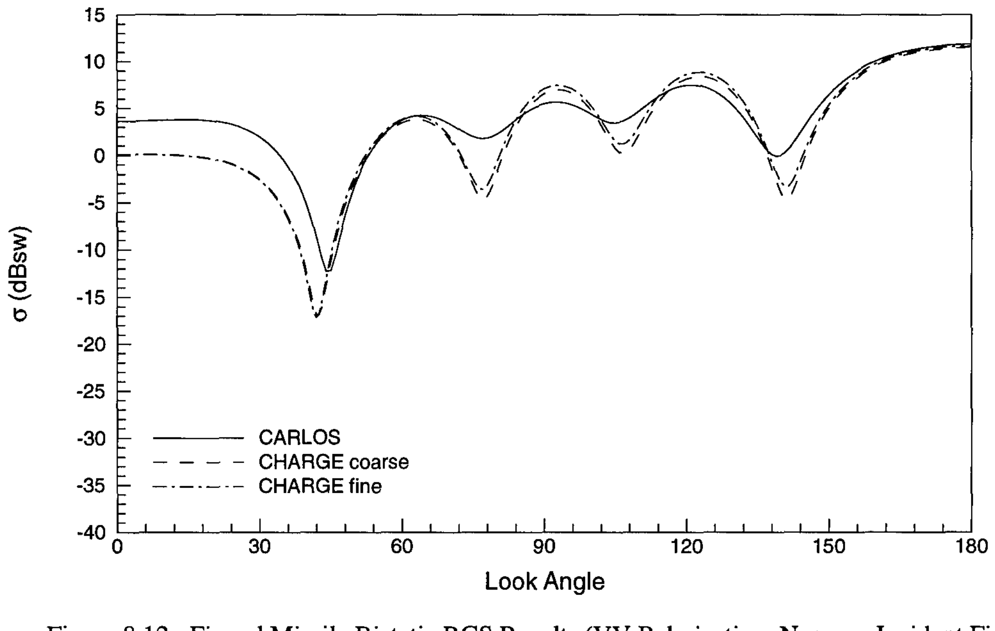


Figure 8.12: Finned Missile Bistatic RCS Results (VV Polarization, Nose-on Incident Field)

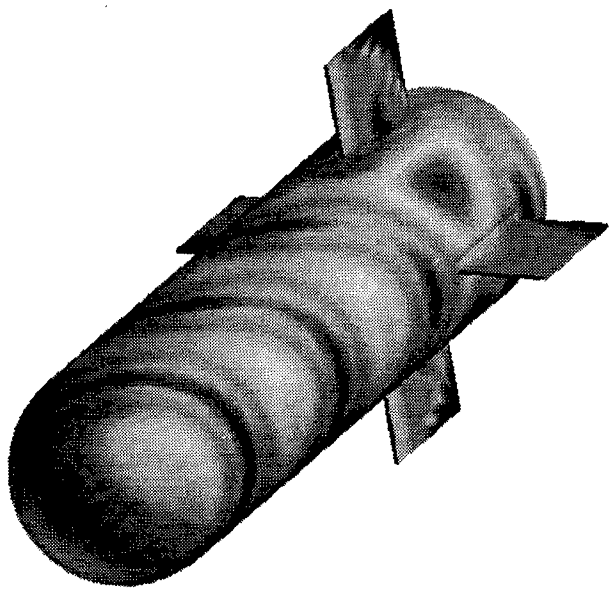


Figure 8.13: Electric Field Contours on Finned-Missile Surface

ened regions at the tips of the fins indicate the locations of high scattering caused by edge diffraction. Furthermore, the interaction between the fins and the missile body is clearly evident in the figure. The smoothness of the solution between the fins and the body demonstrates that the first-order-accurate tri-linear interpolation process is capable of accurately transferring information between the various grids.

8.4 Chapter Conclusions and Summary

This chapter focused on the methodology developed to conduct complex numerical simulations of electromagnetic scattering and wave-propagation phenomena. This methodology, encapsulated within PHASE, represents the culminating effort of the present study. PHASE is the only process currently capable of performing FVTD simulations using overset grids on massively parallel computing platforms. The versatility and practicality of the method have been successfully demonstrated via the finned-missile simulations of this chapter as well as the simulations contained in Chapter 7. This process provides a viable tool for analyzing complex engineering problems of interest.

IX. Conclusions and Recommendations for Further Study

This chapter provides a synopsis of the motivation and objectives for the research and summarizes the primary areas of study emphasis. Conclusions relating to each study area are then formulated, and several areas for further investigation are recommended.

9.1 Objective Review

As a basis for summarizing the findings of this study, it is helpful to restate the overarching objective of this work as it appears in Chapter 1, namely,

> *To develop a versatile and practical capability for conducting time-domain numerical simulations of electromagnetic scattering and wave-propagation phenomena.*

In order to achieve this objective, a process was developed which is versatile in that it is platform independent and capable of solving a large number of problems with no modification to the overall process and few, if any, modifications to any computer code. It is also practical as has been shown through its ability to compute the electromagnetic fields scattered from a relatively complex, multi-finned missile. Furthermore, these desirable characteristics were not achieved at the expense of usability, for a process that is highly capable but unduly complex is little better than one which is simple but cannot be used to solve practical problems.

This advanced simulation capability leverages technologies from several disciplines including computational fluid dynamics, computational electromagnetics, and parallel computing. Where sufficient capability was perceived to previously exist, that capability was simply applied to the new problem area (such was the case with the PEGSUS code and the FVSRK2 algorithm). On the other hand, where gaps were apparent, effort was expended to fill those gaps. Under this precept, work was accomplished to quantify the effects of decomposition dimensionality on the parallel performance of hyperbolic PDE solvers, to develop a new method of partitioning overset grids for parallel environments, to develop the first-of-its-kind overset-grid CEM solver, and to provide the first detailed assessment of the applicability of overset grids to CEM. By accomplishing this work, the objectives set forth in Chapter 1 were met.

Because the nature of the work and the variety of disciplines used, several specialized investigations were conducted. These investigations were conceived and executed using an incremental design philosophy which yielded a number of useful conclusions that are summarized in the following section. Consistent with the flow of this document, the significant findings are grouped by technology area: parallel algorithm development, FVTD algorithm analysis, and overset grid/FVTD applicability assessment.

9.2 Statement of Conclusions

9.2.1 Parallel Algorithm Development

For this study, parallel algorithm development focused on designing domain-decomposition techniques for improved parallel performance. For computational domains comprised of single structured grids, partitioning the computational domain in a block-type fashion so that the dimensionalities of the decomposition and the problem space were equal produced an optimum theoretical isoefficiency function; however, because the additive-cost-function communication model used in the development of the theoretical isoefficiency function neglects such issues as load imbalance and message contention, measured parallel performance often fell below the theoretical predictions [17,18]. In practice it was found that although the theoretical model predicted general parallel-performance trends, efficiencies associated with a given decomposition approach were closely related to the interprocessor-connection topologies of the parallel machines. Thus, higher-dimensional decompositions were favored by machines such as the Cray T3D which has a greater interprocessor connectivity than the Intel Paragon or IBM SP2. For improved efficiency across a range of parallel platforms, it is therefore important to allow the dimensionality of the decomposition to vary according to the architecture on which the computer code is executed.

Although block-type decompositions were adequate for single grids, they were found lacking for overset grids. This was demonstrated theoretically through a new overset grid communication-graph-based theoretical parallel performance model in which intra- and inter-grid communications were classified as structured and unstructured, respectively. The theoretical model showed unstructured communications (which are a result of the interpolation requirement of an overset grid) can comprise a significant portion of the total communication requirements. By utilizing RCB instead of a block-type decomposition for an overset grid computational domain surrounding two spheres in close proximity, unstructured communications were significantly reduced while load balance was markedly improved. This led to improvements in parallel efficiency by as much as 25 percent [20].

9.2.2 FVTD Algorithm Analysis

With parallel algorithm development completed, the capabilities of the FVSRK2 algorithm were assessed using several single-grid simulations. These simulations showed the algorithm to be capable of producing accurate results for scattering simulations using surface-mesh densities of 15 to 25 cpλ depending on the electrical size of the body. Generally, a large body-length-to-wavelength ratio (higher frequency) required less mesh resolution due to the less-pronounced effect of surface-wave phenomena at these frequencies. At the same time, however, higher-frequency simulations were found to exhibit much more dramatic variations in RCS convergence behavior. Although the RCS convergence history was found to be affected by frequency, it was relatively insensitive to the choice of initial Fourier sampling period.

While surface-mesh densities of 15 to 25 cpλ were sufficient, wall spacings of 200 to 2000 cpλ were required in order to obtain accurate RCS profiles. As in the case of the surface-mesh density, lower refinements corresponded to higher excitation frequencies. Mesh stretching and packing were found to be capable of producing the requisite wall spacings while substantially lowering the overall number of cells. Of the two methods, stretching was generally preferred since it produced identical results to constantly spaced or packed meshes while requiring fewer cells. Where packing was employed, no adverse effects in the RCS calculations were observed due to the jump in the metric values at the edge of the packed region.

9.2.3 Overset Grid/FVTD Applicability Assessment

The surface-mesh-density and wall-spacing requirements of the single-grid cases were found to be equally applicable to overset-grid simulations. These simulations were conducted using CHARGE, the first FVTD computer code capable of utilizing overset grids. In waveguide simulations conducted with CHARGE, the largest error increase (comparing an embedded-grid to a single-grid configuration) due to the interpolation process was limited to approximately 5 percent and occurred for the case in which analytical boundary conditions were used. In cases where extrapolation boundary conditions were used, there was no discernible difference in the maximum-error values between the embedded- and single-grid cases. This degree of accuracy extended to scattering simulations where single- and overset-grid configurations produced RCS profiles that differed by no more than 0.07 dBsm for comparable wall spacings and surface-mesh densities. Thus, the overset-grid process was found to be capable of producing results of comparable accuracy to single-grid simulations while greatly simplifying the grid-generation procedure for complex geometries [19,21].

9.3 Suggestions for Improvements and Further Research

Although the capability of the FVTD technique has been dramatically extended as a consequence of the present study, there remain several areas for further improvements. Several of the more important issues are detailed in the following paragraphs.

9.3.1 Parallel Algorithm Research Areas

The domain-decomposition studies demonstrated that significant performance improvements are possible using appropriate decomposition schemes. This was especially true when RCB was applied to overset grids. It should not be concluded, however, that RCB represents an optimal overset-grid-partitioning scheme. Further research is needed into decomposition methods tailored specifically to overset grids. For example, the feature contained in DOPLER which allows the specification of "no cut" edges could be exploited by requiring that a cell which is updated via interpolation be located on the same processor as those cells which supply interpolated data. In contrast to RCB which merely reduces unstructured communications, such an approach completely eliminates them.

Features like the "no cut" edge specification make DOPLER highly flexible; however, several improvements to DOPLER are necessary. Specifically, the code currently functions only as a serial preprocessor to CHARGE. There is opportunity for research into parallelizing the sorting and decomposition algorithms contained within DOPLER. A parallel version of DOPLER could then decompose the computational domain at run time, thus eliminating the costly DOPLER/CHARGE data-migration portion of the PHASE process. Other more minor enhancements to DOPLER include better error checking and more memory-efficient data structures. Currently, the data structure used to determine cell neighbors is based on a completely unstructured-grid approach. This design allows for the utmost flexibility; however, it results in large storage requirements. It is possible to develop new data structures which are more tailored to a structured-grid approach yet still allow a high degree of flexibility for future modifications.

9.3.2 FVTD Solver Improvements

While the FVSRK2 algorithm contained in CHARGE was shown to be capable of providing accurate results, the extrapolation surface boundary condition remains the single greatest hindrance to the FVTD methodology realizing its full potential. The extremely tight wall spacing required to achieve accurate results necessitates time steps so small that thousands of iterations are often required in order to advance the solution a single wave period. Research in the area of explicit surface boundary conditions for collocated grids is absolutely imperative.

In addition to improvements to the surface boundary condition, it is necessary to develop new algorithms capable of reducing the number of unknowns which must be solved in a given simulation. This can be accomplished either by reducing the 15-to-25 cpλ mesh-density requirement or by reducing the size of the computational domain. The former solution scheme dictates that methods be developed which can more accurately resolve the wave motion using fewer cells than the FVSRK2 algorithm while the latter requires research in the area of far-field boundary condition formulations which suppress numerical reflection while allowing the outer boundary of the computational domain to be brought close to the body.

9.3.3 Overset Grid Interpolation Process Improvements

Although the interpolation errors were shown to be largely masked by those associated with the extrapolation surface boundary condition, they could become a leading source of error if more accurate surface boundary conditions are developed. For this reason, it is beneficial to consider alternative methods of exchanging information between overset and embedded grids. Work has been accomplished in developing conservative methods for treating grid interfaces [14,128,129], but a detailed study in conjunction with a FVTD technique remains to be completed.

Appendix A: An Overview of the FVTD Methodology

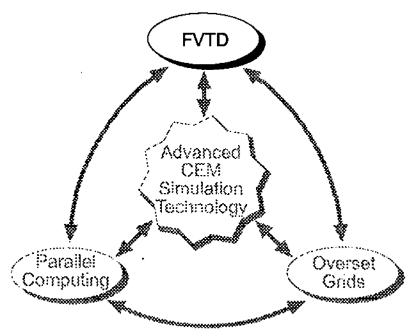


Figure A.1: Appendix A Emphasis

This appendix describes, on a fairly introductory level, the theory and numerics behind the FVTD methodology. It is intended as a concise yet relatively complete reference so that one might construct a basic FVTD computer code with minimal additional reading. The appendix begins with a statement of the governing equations of electromagnetics–the Maxwell equations–as found in nearly all classic electromagnetics texts. The equations are then cast into conservative form for a general curvilinear coordinate system in order to provide a foundation on which to build a general FVTD solver. A derivation of the scattered-field form of the equations is presented, and the motivation for using this form of the equations in the FVTD solver is discussed.

The equation development serves as one of the two fundamental components of the FVTD methodology, the second being the finite-volume discretization procedure. To begin the discussion on this second component of the process, a generalized finite-volume scheme is presented. Then, moving from the general to the specific, the particular finite-volume scheme used in the body of this work is presented and applied to the proper form of the governing equations. That scheme consists of two parts: determination of cell fluxes via a flux-vector-splitting approach, and numerical time integration via a Runge-Kutta procedure.

With the equations placed the proper form and the finite-volume discretization procedure applied, the basic components of the FVTD methodology are in place and the governing equations can be solved numerically. Typically, however, the desired objective of a FVTD simulation is not to simply integrate the equations but to examine how an electromagnetic wave interacts with a body placed in an electromagnetic field. This is usually accomplished by extracting RCS data from the FVTD simulation. Bearing this in mind, the appendix concludes by examining the process for determining the RCS of a scattering body using the numerically computed time-varying electromagnetic fields.

A.1 The Maxwell Equations

The behavior of electromagnetic phenomena is modeled by the set of four equations collectively known as Maxwell's equations. In differential vector form for a Cartesian reference frame, these equations are written as [120]

Faraday's Law:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{a.1}$$

Ampere's Law:

$$\nabla \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J} \tag{a.2}$$

Gauss's Law:

$$\nabla \cdot \vec{D} = \rho \tag{a.3}$$

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

where:

$\vec{B}$	= magnetic flux density (webers/square meter)
$\vec{D}$	= electric flux density (coulombs/square meter)
$\vec{H}$	= magnetic field intensity (amperes/meter)
$\vec{E}$	= electric field intensity (volts/meter)
$\vec{J}$	= conduction electric current density (amperes/square meter)
ρ	= electric charge density (coulombs/cubic meter)

By taking the divergence of (a.1) and (a.2) and applying the continuity equation,

$$\nabla \cdot \vec{J} = -\frac{\partial \rho}{\partial t} \tag{a.5}$$

one can obtain equations (a.3) and (a.4). Thus, the two divergence equations are not independent of the two curl equations and are therefore unnecessary for describing the wave motion. For this reason, the system to be solved is comprised of equations (a.1) and (a.2) and constitutes a coupled set of six scalar equations with twelve unknowns. To provide closure, the flux densities can be related to the field intensities through the constitutive relations

$$\vec{D} = \underset{\sim}{\varepsilon}\vec{E} \text{ and } \vec{B} = \underset{\sim}{\mu}\vec{H} \tag{a.6}$$

where

$\underset{\sim}{\varepsilon}$	= electric permittivity (farad/meter)
$\underset{\sim}{\mu}$	= magnetic permeability (henry/m)

In the most general sense, the electric permittivity and magnetic permeability are tensors; however, for the cases studied in this work where the region over which the electromagnetic wave propagates is considered to be linear, homogeneous, nondispersive and isotropic, the quantities can be considered scalar constants [8]. Consequently, the subscript '~' tensor notation is dropped for the remainder of this appendix.

By carrying through the curl operations and incorporating equation (a.6) into equations (a.1) and (a.2), the following result is obtained:

$$\vec{Q}_t + \vec{U}_x + \vec{V}_y + \vec{W}_z = \vec{J} \tag{a.7}$$

where

$$\vec{Q} = \begin{bmatrix} Bx \\ By \\ Bz \\ Dx \\ Dy \\ Dz \end{bmatrix} \quad \vec{U} = \begin{bmatrix} 0 \\ -Dz/\varepsilon \\ Dy/\varepsilon \\ 0 \\ Bz/\mu \\ -By/\mu \end{bmatrix} \quad \vec{V} = \begin{bmatrix} Dz/\varepsilon \\ 0 \\ -Dx/\varepsilon \\ -Bz/\mu \\ 0 \\ Bx/\mu \end{bmatrix} \quad \vec{W} = \begin{bmatrix} -Dy/\varepsilon \\ Dx/\varepsilon \\ 0 \\ -By/\mu \\ -Bx/\mu \\ 0 \end{bmatrix} \quad \vec{J} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -Jx \\ -Jy \\ -Jz \end{bmatrix} \tag{a.8}$$

The notations $\vec{Z}_x$ and Zx should not be confused. The former is shorthand notation for $\partial\vec{Z}/\partial x$ while the latter represents the x component of the vector $\vec{Z}$. In the previous equation, the vector $\vec{Q}$ is known as the *dependent-variable vector* while the vectors $\vec{U}$, $\vec{V}$, and $\vec{W}$ are termed the *flux vectors.*

Because a Cartesian coordinate system is inadequate for describing most practical shapes, it is desirable to transform equation (a.7) to a general (ξ, η, ζ) coordinate system. The transformation should ensure that the equation remains in *conservative form* so that any field discontinuities can be properly captured. The process of transforming the equation is simply an exercise in algebraic manipulation after application of the chain rule of differentiation. Because Hoffman [54] provides an excellent discussion of coordinate transformations and conservative formulations, the process is not illustrated here; however application of the process to equation (a.7) leads to the conservative form of the two Maxwell curl equations for a general curvilinear coordinate system, namely

$$\tilde{Q}_t + \tilde{U}_\xi + \tilde{V}_\eta + \tilde{W}_\zeta = \tilde{J} \qquad \text{(a.9)}$$

where

$$\begin{aligned}
\tilde{Q} &= \vec{Q}/J \\
\tilde{U} &= (\xi_x \vec{U} + \xi_y \vec{V} + \xi_z \vec{W})/J \\
\tilde{V} &= (\eta_x \vec{U} + \eta_y \vec{V} + \eta_z \vec{W})/J \\
\tilde{W} &= (\zeta_x \vec{U} + \zeta_y \vec{V} + \zeta_z \vec{W})/J \\
\tilde{J} &= \vec{J}/J
\end{aligned} \qquad \text{(a.10)}$$

Terms of the form ξ_x are known as the *metrics* of the coordinate transformation while J represents the *Jacobian* of the coordinate transformation and is given by

$$J = \begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} \qquad \text{(a.11)}$$

Although equation (a.9) is the desired form of the Maxwell equations, it was derived from equations (a.1) and (a.2) which govern the behavior of the *total* electric and magnetic fields. Unfortunately, numerically solving the Maxwell equations in *total-field form* can lead to large inaccuracies. For this reason, the following section discusses an alternative form of the Maxwell equations that is more suitable to solution via numerical means.

A.2 Scattered-Field Formulation of the Maxwell Equations

As mentioned in the previous section, equation (a.9) governs the behavior of the total electric and magnetic fields. The total fields are simply the sum of all magnetic and electric fields in the region of interest. For a typical electromagnetic scattering problem, an incident wave of some form (perhaps a pulse or sinusoid) impinges on a body of interest and is scattered in some fashion. In this situation, the linearity of the Maxwell equations allows one to treat the electromagnetic field as consisting of an incident portion and a scattered portion, i.e.

$$\vec{E}_t = \vec{E}_s + \vec{E}_i \tag{a.12}$$

where the subscripts t, s, and i refer to the total, scattered, and incident fields, respectively. A similar expression holds for the magnetic field. By definition, the incident field is that portion of the field that would exist in the absence of the scattering body [8].

Translating the above-stated scattering scenario to a computer simulation, one would expect to numerically propagate the incident wave from the outer region of the computational domain to the scattering surface. Unfortunately, the diffusion and dispersion errors inherent in most numerical schemes introduce phase and amplitude deviations into the signal that can adversely impact the solution quality [94]. Since the incident field is typically a known analytic function, it is therefore more effective to specify the incident field as a boundary condition on the surface of the scattering object and to subsequently solve for the scattered rather than the total fields. It is therefore necessary to develop a form of the Maxwell equations that governs the behavior of the scattered fields. To obtain the scattered field form of equation (a.9), the two curl equations are first written for a source-free region in the absence of any scattering bodies. In this case, the total field is the incident field, and the equations are given by

$$\nabla \times \vec{E}_i = -\frac{\partial(\mu_0 \vec{H}_i)}{\partial t} \tag{a.13}$$

$$\nabla \times \vec{H}_i = \frac{\partial(\varepsilon_0 \vec{E}_i)}{\partial t} \tag{a.14}$$

where ε_0 and μ_0 are the free-space permittivity and permeability, respectively.

Now, introducing a scattering body (a body of different characteristic impedance than the medium surrounding it) into the region, the curl equations are applied to the total fields to yield

$$\nabla \times \vec{E}_t = -\frac{\partial(\mu \vec{H}_t)}{\partial t} \tag{a.15}$$

$$\nabla \times \vec{H}_t = \frac{\partial(\varepsilon \vec{E}_t)}{\partial t} + \vec{J} \tag{a.16}$$

where

$$\epsilon, \mu = \begin{cases} \epsilon_0, \mu_0 & \text{outside the scatterer} \\ \epsilon_1, \mu_1 & \text{inside the scatterer} \end{cases} \tag{a.17}$$

By subtracting (a.13) and (a.14) from (a.15) and (a.16), performing some algebraic manipulation, and substituting (a.12) appropriately, two curl equations are obtained, namely

$$\nabla \times \vec{E}_s = -\frac{\partial(\mu_0 \vec{H}_s)}{\partial t} - \frac{\partial}{\partial t}(\mu - \mu_0)\vec{H}_t \tag{a.18}$$

$$\nabla \times \vec{H}_s = \frac{\partial(\epsilon_0 \vec{E}_s)}{\partial t} + \vec{J} + \frac{\partial}{\partial t}(\epsilon - \epsilon_0)\vec{E}_t \tag{a.19}$$

In cases where the scattering body is constructed of a perfectly-electrically-conducting (PEC) material, the second time derivative term in the above two equations vanishes since no time-varying fields can exist within such a body [74]. This allows the equations to be simplified, viz.

$$\nabla \times \vec{E}_s = -\frac{\partial(\mu_0 \vec{H}_s)}{\partial t} \tag{a.20}$$

$$\nabla \times \vec{H}_s = \frac{\partial(\epsilon_0 \vec{E}_s)}{\partial t} + \vec{J} \tag{a.21}$$

Equations (a.20) and (a.21) are identical in form to equations (a.1) and (a.2). Thus, in cases where the scattering bodies are PEC, the conservative formulation presented in equation (a.9) can be used unmodified for the scattered-field formulation of the Maxwell equations.

A.3 A General Finite-Volume Formulation

With the governing equations placed into the desired form, a numerical technique is now developed for their solution. To do so, equation (a.9) can be integrated over an arbitrary volume V to obtain

$$\iiint_V \tilde{Q}_t dV + \iiint_V (\tilde{U}_\xi + \tilde{V}_\eta + \tilde{W}_\zeta) dV = \iiint_V \tilde{J} dV \tag{a.22}$$

which can be written as

$$\iiint\limits_V \tilde{Q}_t dV + \iiint\limits_V (\nabla \cdot \tilde{F}) dV = \iiint\limits_V \tilde{J} dV \quad \text{(a.23)}$$

where $\tilde{F} = \tilde{U}\hat{\xi} + \tilde{V}\hat{\eta} + \tilde{W}\hat{\zeta}$. Application of the divergence theorem to the second integral gives

$$\iiint\limits_V \tilde{Q}_t dV + \iint\limits_S \tilde{F} \cdot \hat{n} dS = \iiint\limits_V \tilde{J} dV \quad \text{(a.24)}$$

where $\hat{n}$ is the unit surface normal to the volume bounding surface, S.

Using equation (a.24), a generalized finite-volume scheme can be constructed by assuming the arbitrary volume to be a time-invariant hexahedral cell in which the flux is constant over each cell face, and the current-density and dependent-variable vectors are constant over the cell volume. In this case, the equation becomes

$$\tilde{Q}_t V + \sum_{k=1}^{6} \tilde{F}_k \cdot d\vec{A}_k - \tilde{J} V = 0 \quad \text{(a.25)}$$

where $d\vec{A}_k$ is a vector normal to face k with a magnitude equal to the area of the face.

Figure A.2 depicts a general hexahedral cell with vertices and cell faces numbered according to the convention used in the present study. Faces 1 and 2 are faces of constant ξ, faces 3 and 4 are faces of constant η, and faces 5 and 6 are faces of constant ζ. From the figure, the term $d\vec{A}_2$ can be computed via the cross product of the face diagonals $\vec{D}_{58}$ and $\vec{D}_{67}$ [124], viz.

$$d\vec{A}_2 = 0.5(\vec{D}_{67} \times \vec{D}_{58}) \quad \text{(a.26)}$$

where

$$\vec{D}_{ij} = (x_j - x_i)\hat{i} + (y_j - y_i)\hat{j} + (z_j - z_i)\hat{k} \quad \text{(a.27)}$$

Because $d\vec{A}_2$ is in the $\hat{\xi}$ direction,

$$\tilde{F}_2 \cdot d\vec{A}_2 = \tilde{U}_2 \cdot d\vec{A}_2 = \left\| \tilde{U}_2 \right\| \left\| d\vec{A}_2 \right\| = 0.5 \left\| \tilde{U}_2 \right\| \left\| \vec{D}_{67} \times \vec{D}_{58} \right\| \equiv U_2 A_2 \quad \text{(a.28)}$$

Similar expressions hold for each of the remaining five cell faces. Care must be exercised, however, when computing the cell face vectors for faces 1, 3, and 5. If the cross product of the face diagonals is taken as was

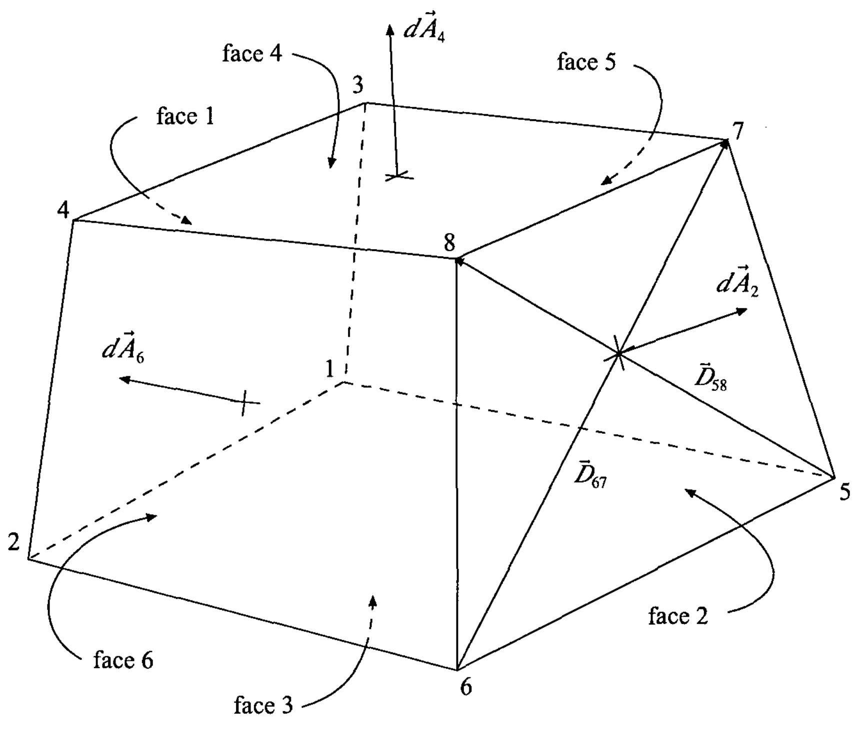


Figure A.2: Finite Volume Cell

shown for face 2, then the resulting face vector points inward rather than outward and the resulting computed flux is an influx rather than an outflux. In practice, it is more efficient to compute these fluxes as influxes and simply reverse the sign to yield

$$\tilde{Q}_t + \frac{1}{V}[U_2A_2 - U_1A_1 + V_4A_4 - V_3A_3 + W_6A_6 - W_5A_5] - \tilde{J} = 0 \qquad \text{(a.29)}$$

The value for the cell volume, V, appearing in (a.29) can be computed via several different methods which do not necessarily produce identical results [124]. In the present study, the volume is computed via

$$V = \frac{1}{3}(d\vec{A}_1 + d\vec{A}_3 + d\vec{A}_5) \cdot \vec{D}_{18} \qquad \text{(a.30)}$$

Equation (a.29) represents a general finite-volume formulation under the stated assumptions and is simply a result of applying equation (a.9) to a finite control volume. The differentiating factor in the myriad of finite-volume techniques lies in the treatment of the time integration term, $\tilde{Q}_t$, and in the determination of the flux vector, $\tilde{F}$, in order to achieve the desired numerical properties [127]. Treatment of the time integration term for an explicit scheme is usually relatively straightforward; however, computation of the fluxes can be somewhat more complex. For this reason, the flux-calculation procedure is discussed in detail in the following sections.

A.4 The Nature of the Fluxes

In order to gain insight into the physical nature of the flux vector term appearing in equation (a.23), it is helpful to write (a.1) and (a.2) in the form

$$\frac{\partial}{\partial t}\begin{bmatrix}\vec{B}\\ \vec{D}\end{bmatrix}+\begin{bmatrix}\nabla\times\vec{B}/\mu\\ -\nabla\times\vec{D}/\varepsilon\end{bmatrix}=\vec{J} \qquad \text{(a.31)}$$

Integrating (a.31) over an arbitrary volume and applying the relation

$$\iiint_V(\nabla\times\vec{A})\,dV=\iint_S(\hat{n}\times A)\,dS \qquad \text{(a.32)}$$

where $\vec{A}$ is an arbitrary vector gives

$$\iiint_V\frac{\partial}{\partial t}\begin{bmatrix}\vec{B}\\ \vec{D}\end{bmatrix}dV+\iint_S\begin{bmatrix}\hat{n}\times\vec{B}/\mu\\ -\hat{n}\times\vec{D}/\varepsilon\end{bmatrix}dS=\iiint_V\vec{J}\,dV \qquad \text{(a.33)}$$

Comparing the surface integrals appearing in equations (a.24) and (a.33) reveals the fluxes through the faces of a volumetric element are comprised solely of the tangential components of the electromagnetic field.

A.5 The Flux-Calculation and Time-Integration Procedures

A.5.1 Flux-Calculation Procedure

From the general finite-volume formulation presented in the previous sections, a specific procedure for computing the flux vector is now presented. The flux vectors $\tilde{U}$, $\tilde{V}$, and $\tilde{W}$ possess the property

$$\begin{aligned}\tilde{U}(\lambda\tilde{Q}) &= \lambda\tilde{U}(\tilde{Q})\\ \tilde{V}(\lambda\tilde{Q}) &= \lambda\tilde{V}(\tilde{Q})\\ \tilde{W}(\lambda\tilde{Q}) &= \lambda\tilde{W}(\tilde{Q})\end{aligned}\qquad\text{(a.34)}$$

where λ is an arbitrary scalar quantity. Because of this property, the flux vectors are termed *homogeneous of degree one in* $\tilde{Q}$ and can be written as [5,32,130]

$$\begin{aligned}\tilde{U}(\tilde{Q}) &= \frac{\partial\tilde{U}(\tilde{Q})}{\partial\tilde{Q}}\tilde{Q} \equiv A\tilde{Q}\\ \tilde{V}(\tilde{Q}) &= \frac{\partial\tilde{V}(\tilde{Q})}{\partial\tilde{Q}}\tilde{Q} \equiv B\tilde{Q}\\ \tilde{W}(\tilde{Q}) &= \frac{\partial\tilde{W}(\tilde{Q})}{\partial\tilde{Q}}\tilde{Q} \equiv C\tilde{Q}\end{aligned}\qquad\text{(a.35)}$$

The matrices A, B, and C are known as the *flux Jacobian matrices* and for a general curvilinear coordinate system are given by [90]

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & -\frac{\xi_z}{\varepsilon} & \frac{\xi_y}{\varepsilon}\\ 0 & 0 & 0 & \frac{\xi_z}{\varepsilon} & 0 & -\frac{\xi_x}{\varepsilon}\\ 0 & 0 & 0 & -\frac{\xi_y}{\varepsilon} & \frac{\xi_x}{\varepsilon} & 0\\ 0 & \frac{\xi_z}{\mu} & -\frac{\xi_y}{\mu} & 0 & 0 & 0\\ -\frac{\xi_z}{\mu} & 0 & \frac{\xi_x}{\mu} & 0 & 0 & 0\\ \frac{\xi_y}{\mu} & -\frac{\xi_x}{\mu} & 0 & 0 & 0 & 0\end{bmatrix}\quad B = \begin{bmatrix} 0 & 0 & 0 & 0 & -\frac{\eta_z}{\varepsilon} & \frac{\eta_y}{\varepsilon}\\ 0 & 0 & 0 & \frac{\eta_z}{\varepsilon} & 0 & -\frac{\eta_x}{\varepsilon}\\ 0 & 0 & 0 & -\frac{\eta_y}{\varepsilon} & \frac{\eta_x}{\varepsilon} & 0\\ 0 & \frac{\eta_z}{\mu} & -\frac{\eta_y}{\mu} & 0 & 0 & 0\\ -\frac{\eta_z}{\mu} & 0 & \frac{\eta_x}{\mu} & 0 & 0 & 0\\ \frac{\eta_y}{\mu} & -\frac{\eta_x}{\mu} & 0 & 0 & 0 & 0\end{bmatrix}\quad C = \begin{bmatrix} 0 & 0 & 0 & 0 & -\frac{\zeta_z}{\varepsilon} & \frac{\zeta_y}{\varepsilon}\\ 0 & 0 & 0 & \frac{\zeta_z}{\varepsilon} & 0 & -\frac{\zeta_x}{\varepsilon}\\ 0 & 0 & 0 & -\frac{\zeta_y}{\varepsilon} & \frac{\zeta_x}{\varepsilon} & 0\\ 0 & \frac{\zeta_z}{\mu} & -\frac{\zeta_y}{\mu} & 0 & 0 & 0\\ -\frac{\zeta_z}{\mu} & 0 & \frac{\zeta_x}{\mu} & 0 & 0 & 0\\ \frac{\zeta_y}{\mu} & -\frac{\zeta_x}{\mu} & 0 & 0 & 0 & 0\end{bmatrix}\qquad\text{(a.36)}$$

The flux Jacobian matrices become important in developing a flux-calculation routine since the eigenvalues of the matrices embody information concerning the direction and speed of the wave propagation. Those eigenvalues are given by

$$\begin{aligned}\Lambda_A &= \text{diagonal}\left\{\frac{\alpha}{\sqrt{\mu\varepsilon}}, \frac{\alpha}{\sqrt{\mu\varepsilon}}, -\frac{\alpha}{\sqrt{\mu\varepsilon}}, -\frac{\alpha}{\sqrt{\mu\varepsilon}}, 0, 0\right\}\\ \Lambda_B &= \text{diagonal}\left\{\frac{\beta}{\sqrt{\mu\varepsilon}}, \frac{\beta}{\sqrt{\mu\varepsilon}}, -\frac{\beta}{\sqrt{\mu\varepsilon}}, -\frac{\beta}{\sqrt{\mu\varepsilon}}, 0, 0\right\}\\ \Lambda_C &= \text{diagonal}\left\{\frac{\gamma}{\sqrt{\mu\varepsilon}}, \frac{\gamma}{\sqrt{\mu\varepsilon}}, -\frac{\gamma}{\sqrt{\mu\varepsilon}}, -\frac{\gamma}{\sqrt{\mu\varepsilon}}, 0, 0\right\}\end{aligned}\qquad\text{(a.37)}$$

where Λ_A, Λ_B, and Λ_C are the eigenvalues of the A, B, and C matrices, respectively, and

$$\alpha = \sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2} \tag{a.38}$$

$$\beta = \sqrt{\eta_x^2 + \eta_y^2 + \eta_z^2} \tag{a.39}$$

$$\gamma = \sqrt{\zeta_x^2 + \zeta_y^2 + \zeta_z^2} \tag{a.40}$$

In order to take advantage of the directional propagation properties which are manifested by the signs of the eigenvalues, Λ_A, Λ_B, and Λ_C are split into components which contain only the positive and negative eigenvalues, respectively. For the Λ_A matrix, the splitting yields

$$\begin{aligned} \Lambda_A^+ &= \text{diagonal}\left\{\frac{\alpha}{\sqrt{\mu\varepsilon}}, \frac{\alpha}{\sqrt{\mu\varepsilon}}, 0, 0, 0, 0\right\} \\ \Lambda_A^- &= \text{diagonal}\left\{0, 0, -\frac{\alpha}{\sqrt{\mu\varepsilon}}, -\frac{\alpha}{\sqrt{\mu\varepsilon}}, 0, 0\right\} \end{aligned} \tag{a.41}$$

Similar expressions hold for the other two eigenvalue matrices.

With the eigenvalue matrices split, a similarity transformation is performed on the flux Jacobian matrices which, for the A matrix, is of the form

$$A = P_A^{-1}\Lambda_A P_A = P_A^{-1}(\Lambda_A^+ + \Lambda_A^-)P_A = A^+ + A^- \tag{a.42}$$

where

$$\begin{aligned} A^+ &= P_A^{-1}\Lambda_A^+ P_A \\ A^- &= P_A^{-1}\Lambda_A^- P_A \end{aligned} \tag{a.43}$$

The columns of the matrices P_A, P_B, and P_C are formed from the *non-unique* eigenvectors of A, B, and C, respectively. Using the symbolic mathematics package *Maple V* [28], the matrices and their respective inverses are found to be

$$P_A = \begin{bmatrix} -\frac{\alpha\sqrt{\mu}}{\xi_z\sqrt{\varepsilon}} & \frac{\xi_x}{\xi_z} & \frac{\alpha\sqrt{\mu}}{\xi_z\sqrt{\varepsilon}} & \frac{\xi_x}{\xi_z} & 1 & 0 \\ \frac{\alpha\xi_x\sqrt{\mu}}{\xi_z\xi_y\sqrt{\varepsilon}} & -\frac{\xi_x^2+\xi_z^2}{\xi_z\xi_y} & -\frac{\alpha\xi_x\sqrt{\mu}}{\xi_z\xi_y\sqrt{\varepsilon}} & -\frac{\xi_x^2+\xi_z^2}{\xi_z\xi_y} & \frac{\xi_y}{\xi_x} & 0 \\ 0 & 1 & 0 & 1 & \frac{\xi_z}{\xi_x} & 0 \\ \frac{\xi_x}{\xi_y} & -\frac{\alpha\sqrt{\varepsilon}}{\xi_y\sqrt{\mu}} & \frac{\xi_x}{\xi_y} & \frac{\alpha\sqrt{\varepsilon}}{\xi_y\sqrt{\mu}} & 0 & \frac{\xi_x}{\xi_y} \\ 1 & 0 & 1 & 0 & 0 & 1 \\ -\frac{\xi_x^2+\xi_y^2}{\xi_z\xi_y} & \frac{\alpha\xi_x\sqrt{\varepsilon}}{\xi_z\xi_y\sqrt{\mu}} & -\frac{\xi_x^2+\xi_y^2}{\xi_z\xi_y} & -\frac{\alpha\xi_x\sqrt{\varepsilon}}{\xi_z\xi_y\sqrt{\mu}} & 0 & \frac{\xi_z}{\xi_x} \end{bmatrix} \qquad (a.44)$$

$$P_B = \begin{bmatrix} 1 & 0 & 1 & 0 & \frac{\eta_x}{\eta_y} & 0 \\ \frac{\eta_y}{\eta_x} & -\frac{\beta\sqrt{\mu}}{\eta_x\sqrt{\varepsilon}} & \frac{\eta_y}{\eta_x} & \frac{\beta\sqrt{\mu}}{\eta_x\sqrt{\varepsilon}} & 1 & 0 \\ -\frac{\eta_x^2+\eta_y^2}{\eta_x\eta_z} & \frac{\beta\eta_y\sqrt{\mu}}{\eta_x\eta_z\sqrt{\varepsilon}} & -\frac{\eta_x^2+\eta_y^2}{\eta_x\eta_z} & -\frac{\beta\eta_y\sqrt{\mu}}{\eta_x\eta_z\sqrt{\varepsilon}} & \frac{\eta_z}{\eta_y} & 0 \\ \frac{\beta\eta_y\sqrt{\varepsilon}}{\eta_x\eta_z\sqrt{\mu}} & -\frac{\eta_y^2+\eta_z^2}{\eta_x\eta_z} & -\frac{\beta\eta_y\sqrt{\varepsilon}}{\eta_x\eta_z\sqrt{\mu}} & -\frac{\eta_y^2+\eta_z^2}{\eta_x\eta_z} & 0 & \frac{\eta_x}{\eta_z} \\ -\frac{\beta\sqrt{\varepsilon}}{\eta_z\sqrt{\mu}} & \frac{\eta_y}{\eta_z} & \frac{\beta\sqrt{\varepsilon}}{\eta_z\sqrt{\mu}} & \frac{\eta_y}{\eta_z} & 0 & \frac{\eta_y}{\eta_z} \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad (a.45)$$

$$P_C = \begin{bmatrix} -\frac{\zeta_y^2+\zeta_z^2}{\zeta_x\zeta_y} & \frac{\gamma\zeta_z\sqrt{\mu}}{\zeta_x\zeta_y\sqrt{\varepsilon}} & -\frac{\zeta_y^2+\zeta_z^2}{\zeta_x\zeta_y} & -\frac{\gamma\zeta_z\sqrt{\mu}}{\zeta_x\zeta_y\sqrt{\varepsilon}} & 1 & 0 \\ 1 & 0 & 1 & 0 & \frac{\zeta_y}{\zeta_x} & 0 \\ \frac{\zeta_z}{\zeta_y} & -\frac{\gamma\sqrt{\mu}}{\zeta_y\sqrt{\varepsilon}} & \frac{\zeta_z}{\zeta_y} & \frac{\gamma\sqrt{\mu}}{\zeta_y\sqrt{\varepsilon}} & \frac{\zeta_z}{\zeta_x} & 0 \\ 0 & 1 & 0 & 1 & 0 & \frac{\zeta_x}{\zeta_y} \\ \frac{\zeta_z\sqrt{\varepsilon}}{\zeta_x\zeta_y\sqrt{\mu}} & -\frac{\zeta_x^2+\zeta_z^2}{\zeta_x\zeta_y} & -\frac{\zeta_z\sqrt{\varepsilon}}{\zeta_x\zeta_y\sqrt{\mu}} & -\frac{\zeta_x^2+\zeta_z^2}{\zeta_x\zeta_y} & 0 & 1 \\ -\frac{\gamma\sqrt{\varepsilon}}{\zeta_x\sqrt{\mu}} & \frac{\zeta_z}{\zeta_x} & \frac{\gamma\sqrt{\varepsilon}}{\zeta_x\sqrt{\mu}} & \frac{\zeta_z}{\zeta_x} & 0 & \frac{\zeta_z}{\zeta_y} \end{bmatrix} \qquad (a.46)$$

$$P_A^{-1} = \begin{bmatrix} -\frac{\xi_z\sqrt{\varepsilon}}{2\alpha\sqrt{\mu}} & 0 & \frac{\xi_x\sqrt{\varepsilon}}{2\alpha\sqrt{\mu}} & -\frac{\xi_x\xi_y}{2\alpha^2} & \frac{\xi_x^2+\xi_z^2}{2\alpha^2} & -\frac{\xi_y\xi_z}{2\alpha^2} \\ -\frac{\xi_x\xi_z}{2\alpha^2} & -\frac{\xi_y\xi_z}{2\alpha^2} & \frac{\xi_x^2+\xi_y^2}{2\alpha^2} & -\frac{\xi_y\sqrt{\mu}}{2\alpha\sqrt{\varepsilon}} & \frac{\xi_x\sqrt{\mu}}{2\alpha\sqrt{\varepsilon}} & 0 \\ \frac{\xi_z\sqrt{\varepsilon}}{2\alpha\sqrt{\mu}} & 0 & -\frac{\xi_x\sqrt{\varepsilon}}{2\alpha\sqrt{\mu}} & -\frac{\xi_x\xi_y}{2\alpha^2} & \frac{\xi_x^2+\xi_z^2}{2\alpha^2} & -\frac{\xi_y\xi_z}{2\alpha^2} \\ -\frac{\xi_x\xi_z}{2\alpha^2} & -\frac{\xi_y\xi_z}{2\alpha^2} & \frac{\xi_x^2+\xi_y^2}{2\alpha^2} & \frac{\xi_y\sqrt{\mu}}{2\alpha\sqrt{\varepsilon}} & -\frac{\xi_x\sqrt{\mu}}{2\alpha\sqrt{\varepsilon}} & 0 \\ \frac{\xi_x^2}{\alpha^2} & \frac{\xi_x\xi_y}{\alpha^2} & \frac{\xi_x\xi_z}{\alpha^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\xi_x\xi_y}{\alpha^2} & \frac{\xi_y^2}{\alpha^2} & \frac{\xi_y\xi_z}{\alpha^2} \end{bmatrix} \qquad \text{(a.47)}$$

$$P_B^{-1} = \begin{bmatrix} \frac{\eta_y^2+\eta_z^2}{2\beta^2} & -\frac{\eta_x\eta_y}{2\beta^2} & -\frac{\eta_x\eta_z}{2\beta^2} & 0 & -\frac{\eta_z\sqrt{\mu}}{2\beta\sqrt{\varepsilon}} & \frac{\eta_y\sqrt{\mu}}{2\beta\sqrt{\varepsilon}} \\ \frac{\eta_y\sqrt{\varepsilon}}{2\beta\sqrt{\mu}} & -\frac{\eta_x\sqrt{\varepsilon}}{2\beta\sqrt{\mu}} & 0 & -\frac{\eta_x\eta_z}{2\beta^2} & -\frac{\eta_y\eta_z}{2\beta^2} & \frac{\eta_x^2+\eta_y^2}{2\beta^2} \\ \frac{\eta_y^2+\eta_z^2}{2\beta^2} & -\frac{\eta_x\eta_y}{2\beta^2} & -\frac{\eta_x\eta_z}{2\beta^2} & 0 & \frac{\eta_z\sqrt{\mu}}{2\beta\sqrt{\varepsilon}} & -\frac{\eta_y\sqrt{\mu}}{2\beta\sqrt{\varepsilon}} \\ -\frac{\eta_y\sqrt{\varepsilon}}{2\beta\sqrt{\mu}} & \frac{\eta_x\sqrt{\varepsilon}}{2\beta\sqrt{\mu}} & 0 & -\frac{\eta_x\eta_z}{2\beta^2} & -\frac{\eta_y\eta_z}{2\beta^2} & \frac{\eta_x^2+\eta_y^2}{2\beta^2} \\ \frac{\eta_x\eta_y}{\beta^2} & \frac{\eta_y^2}{\beta^2} & \frac{\eta_y\eta_z}{\beta^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\eta_x\eta_z}{\beta^2} & \frac{\eta_y\eta_z}{\beta^2} & \frac{\eta_z^2}{\beta^2} \end{bmatrix} \qquad \text{(a.48)}$$

$$P_C^{-1} = \begin{bmatrix} -\frac{\zeta_x\zeta_y}{2\gamma^2} & \frac{\zeta_x^2+\zeta_z^2}{2\gamma^2} & -\frac{\zeta_y\zeta_z}{2\gamma^2} & \frac{\zeta_z\sqrt{\mu}}{2\gamma\sqrt{\varepsilon}} & 0 & -\frac{\zeta_x\sqrt{\mu}}{2\gamma\sqrt{\varepsilon}} \\ 0 & \frac{\zeta_z\sqrt{\varepsilon}}{2\gamma\sqrt{\mu}} & -\frac{\zeta_y\sqrt{\varepsilon}}{2\gamma\sqrt{\mu}} & \frac{\zeta_y^2+\zeta_z^2}{2\gamma^2} & -\frac{\zeta_x\zeta_y}{2\gamma^2} & -\frac{\zeta_x\zeta_z}{2\gamma^2} \\ -\frac{\zeta_x\zeta_y}{2\gamma^2} & \frac{\zeta_x^2+\zeta_z^2}{2\gamma^2} & -\frac{\zeta_y\zeta_z}{2\gamma^2} & -\frac{\zeta_z\sqrt{\mu}}{2\gamma\sqrt{\varepsilon}} & 0 & \frac{\zeta_x\sqrt{\mu}}{2\gamma\sqrt{\varepsilon}} \\ 0 & -\frac{\zeta_z\sqrt{\varepsilon}}{2\gamma\sqrt{\mu}} & \frac{\zeta_y\sqrt{\varepsilon}}{2\gamma\sqrt{\mu}} & \frac{\zeta_y^2+\zeta_z^2}{2\gamma^2} & -\frac{\zeta_x\zeta_y}{2\gamma^2} & -\frac{\zeta_x\zeta_z}{2\gamma^2} \\ \frac{\zeta_x^2}{\gamma^2} & \frac{\zeta_x\zeta_y}{\gamma^2} & \frac{\zeta_x\zeta_z}{\gamma^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\zeta_x\zeta_y}{\gamma^2} & \frac{\zeta_y^2}{\gamma^2} & \frac{\zeta_y\zeta_z}{\gamma^2} \end{bmatrix} \qquad \text{(a.49)}$$

Using the homogeneity property defined in equation (a.35), the flux vectors can now be written as

$$\begin{aligned} \tilde{U}(\tilde{Q}) &= (A^+ + A^-)\tilde{Q} \equiv \tilde{U}^+(\tilde{Q}) + \tilde{U}^-(\tilde{Q}) \\ \tilde{V}(\tilde{Q}) &= (B^+ + B^-)\tilde{Q} \equiv \tilde{V}^+(\tilde{Q}) + \tilde{V}^-(\tilde{Q}) \\ \tilde{W}(\tilde{Q}) &= (C^+ + C^-)\tilde{Q} \equiv \tilde{W}^+(\tilde{Q}) + \tilde{W}^-(\tilde{Q}) \end{aligned} \qquad \text{(a.50)}$$

where $\tilde{U}^+(\tilde{Q}), \tilde{V}^+(\tilde{Q}), \tilde{W}^+(\tilde{Q})$ and $\tilde{U}^-(\tilde{Q}), \tilde{V}^-(\tilde{Q}), \tilde{W}^-(\tilde{Q})$ are the positive and negative components, respectively, of the flux vectors. Using equations (a.41), (a.44), and (a.47), the positive and negative components of the $\tilde{U}(\tilde{Q})$ flux vector are thus found to be

$$\tilde{U}^+(\tilde{Q}) = \frac{1}{J}\begin{bmatrix} \frac{\xi_y^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & 0 & -\frac{\xi_z}{2\varepsilon} & \frac{\xi_y}{2\varepsilon} \\ -\frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_z}{2\varepsilon} & 0 & -\frac{\xi_x}{2\varepsilon} \\ -\frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x^2+\xi_y^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y}{2\varepsilon} & \frac{\xi_x}{2\varepsilon} & 0 \\ 0 & \frac{\xi_z}{2\mu} & -\frac{\xi_y}{2\mu} & \frac{\xi_y^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} \\ -\frac{\xi_z}{2\mu} & 0 & \frac{\xi_x}{2\mu} & -\frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} \\ \frac{\xi_y}{2\mu} & -\frac{\xi_x}{2\mu} & 0 & -\frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x^2+\xi_y^2}{2\alpha\sqrt{\mu\varepsilon}} \end{bmatrix}\begin{bmatrix} Bx \\ By \\ Bz \\ Dx \\ Dy \\ Dz \end{bmatrix} \qquad \text{(a.51)}$$

$$\tilde{U}^{-}(\tilde{Q}) = \frac{1}{J}\begin{bmatrix} -\frac{\xi_y^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & 0 & -\frac{\xi_z}{2\varepsilon} & \frac{\xi_y}{2\varepsilon} \\ \frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_z}{2\varepsilon} & 0 & -\frac{\xi_x}{2\varepsilon} \\ \frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x^2+\xi_y^2}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_y}{2\varepsilon} & \frac{\xi_x}{2\varepsilon} & 0 \\ 0 & \frac{\xi_z}{2\mu} & -\frac{\xi_y}{2\mu} & -\frac{\xi_y^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} \\ -\frac{\xi_z}{2\mu} & 0 & \frac{\xi_x}{2\mu} & \frac{\xi_x\xi_y}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x^2+\xi_z^2}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} \\ \frac{\xi_y}{2\mu} & -\frac{\xi_x}{2\mu} & 0 & \frac{\xi_x\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & \frac{\xi_y\xi_z}{2\alpha\sqrt{\mu\varepsilon}} & -\frac{\xi_x^2+\xi_y^2}{2\alpha\sqrt{\mu\varepsilon}} \end{bmatrix}\begin{bmatrix} Bx \\ By \\ Bz \\ Dx \\ Dy \\ Dz \end{bmatrix} \qquad \text{(a.52)}$$

The remaining positive and negative flux-vector components are identical to equations (a.51) and (a.52) with the exception that the ξ metric terms are replaced by η terms in the case of the $\tilde{V}(\tilde{Q})$ flux-vector components and by ζ terms in the case of the $\tilde{W}(\tilde{Q})$ flux vector components. For a finite volume scheme, the metrics can be considered as either the direction cosines between the Cartesian and general curvilinear coordinate systems, or as the areas of the cell faces projected into the *x*, *y*, and *z* coordinate directions. In the latter case, the term ξ_x for cell face 2, for example, is found from

$$\xi_x = d\vec{A}_2 \cdot \hat{x} \qquad \text{(a.53)}$$

where $\hat{x}$ is a unit vector in the *x*-coordinate direction. Should the metrics appearing in the expressions for the flux matrices be computed via this procedure, then the flux calculation of equation (a.29) becomes

$$\tilde{Q}_t + \frac{1}{V}[\tilde{U}_2 - \tilde{U}_1 + \tilde{V}_4 - \tilde{V}_3 + \tilde{W}_6 - \tilde{W}_5] - \tilde{J} = 0 \qquad \text{(a.54)}$$

At this point, the flux calculation procedure is nearly complete; the only issue remaining is the determination of the dependent-variable values at the cell faces. For the finite-volume procedure used in this study, the dependent variables are defined with respect to cell centers rather than at cell faces. Thus, a means must exist to transfer data from cell centers to cell faces. This can be accomplished in one of two ways: the fluxes can be evaluated at cell centers and then extrapolated to the cell faces, or the dependent-variable data can be extrapolated to the cell faces and the fluxes subsequently evaluated. The latter method is known as *Monotone Upstream-centered Schemes for Conservation Laws* (MUSCL) and is the method used in this

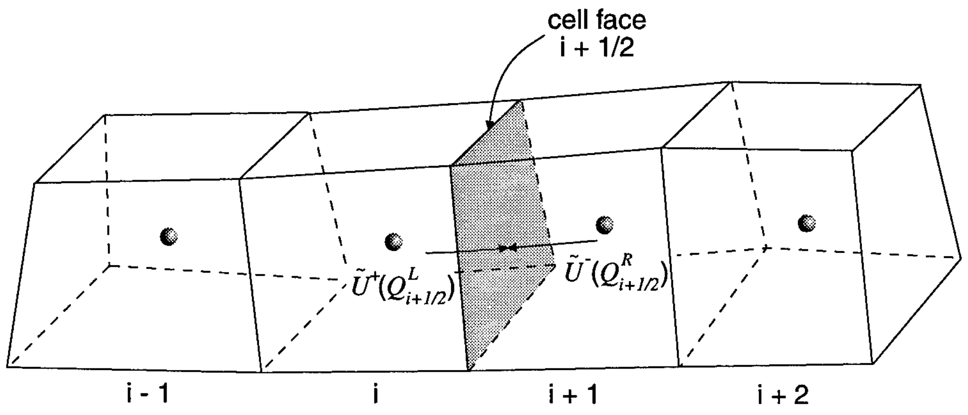

cell face
i + 1/2
$\tilde{U}^{+}(Q^{L}_{i+1/2})$
$\tilde{U}^{-}(Q^{R}_{i+1/2})$
i - 1
i
i + 1
i + 2

Figure A.3: Flux Extrapolation Example

work since it is more in keeping with the spirit of finite volume [124]. Referring to Figure A.3 which portrays a row of finite volume cells, the dependent-variable values at a cell face can be considered to consist of a left state, $\tilde{Q}^{L}$, and a right state, $\tilde{Q}^{R}$. To determine these states, the variable-extrapolation technique of van Leer is used [122]. For cell face $i + 1/2$, the van Leer scheme yields

$$\begin{aligned} \tilde{Q}^{L}_{i+1/2} &= \tilde{Q}_i + \frac{\phi}{4}[(1-\kappa)(\tilde{Q}_i - \tilde{Q}_{i-1}) + (1+\kappa)(\tilde{Q}_{i+1} - \tilde{Q}_i)] \\ \tilde{Q}^{R}_{i+1/2} &= \tilde{Q}_i - \frac{\phi}{4}[(1+\kappa)(\tilde{Q}_{i+1} - \tilde{Q}_i) + (1-\kappa)(\tilde{Q}_{i+2} - \tilde{Q}_{i+1})] \end{aligned} \qquad \text{(a.55)}$$

The desired spatial order of accuracy can be varied by choosing appropriate values for κ and ϕ. Setting $\kappa = 1/3$ and $\phi = 1$ produces a spatially third-order-accurate scheme. Note, however, that the formal order of accuracy is derived assuming a Cartesian mesh of uniform spacing. If the mesh is highly skewed or stretched, the accuracy of the variable-extrapolation procedure is degraded.

A.5.2 Time-Integration Procedure

A.5.2.1 The Runge-Kutta Scheme

With the cell fluxes determined, emphasis is now placed on the time-integration term appearing in equation (a.25). For the present effort, that term is computed via a two-stage, second-order-accurate Runge-Kutta scheme which advances the solution from time level n to time level $n + 1$ according to

$$\begin{aligned}\tilde{Q}^0 &= \tilde{Q}^n \\ \tilde{Q}^1 &= \tilde{Q}^0 - \frac{\Delta t}{V}(R(\tilde{Q}^0)) \\ \tilde{Q}^2 &= \tilde{Q}^0 - \frac{\Delta t}{2V}(R(\tilde{Q}^0) + R(\tilde{Q}^1)) \\ \tilde{Q}^{n+1} &= \tilde{Q}^2\end{aligned} \tag{a.56}$$

where, using the cell-face-indexing notation from equation (a.29),

$$R(\tilde{Q}^k) = (\tilde{U}_2(\tilde{Q}^k) - \tilde{U}_1(\tilde{Q}^k) + \tilde{V}_4(\tilde{Q}^k) - \tilde{V}_3(\tilde{Q}^k) + \tilde{W}_6(\tilde{Q}^k) - \tilde{W}_5(\tilde{Q}^k)) \tag{a.57}$$

For the present study, all simulations were performed in source-free regions (i.e., $\vec{J} = 0$). Thus, the two time-integration stages are given by

Stage 1:

$$\tilde{Q}^1 = \tilde{Q}^0 - \frac{\Delta t}{V}(\tilde{U}_2(\tilde{Q}^0) - \tilde{U}_1(\tilde{Q}^0) + \tilde{V}_4(\tilde{Q}^0) - \tilde{V}_3(\tilde{Q}^0) + \tilde{W}_6(\tilde{Q}^0) - \tilde{W}_5(\tilde{Q}^0)) \tag{a.58}$$

Stage 2:

$$\tilde{Q}^2 = \frac{1}{2}\left(\tilde{Q}^0 + \tilde{Q}^1 - \frac{\Delta t}{V}(\tilde{U}_2(\tilde{Q}^1) - \tilde{U}_1(\tilde{Q}^1) + \tilde{V}_4(\tilde{Q}^1) - \tilde{V}_3(\tilde{Q}^1) + \tilde{W}_6(\tilde{Q}^1) - \tilde{W}_5(\tilde{Q}^1))\right) \tag{a.59}$$

A.5.2.2 Time Step Determination

The time step, Δt, appearing in equations (a.58) and (a.59) is computed so that the *stability condition* is met. Enlightening discussions on stability can be found in works by Anderson, et al. [4] and Strikwerda [115]. Furthermore, Weber [132] investigates the stability condition for the numerical procedure detailed in this appendix. Therefore, the stability condition is simply stated here as

$$\frac{c\Delta t}{\Delta d} \leq 0.87 \tag{a.60}$$

where c is the speed of the wave propagation (equal to the magnitude of the eigenvalues of the Jacobian matrices) and Δd is an appropriate cell-based distance. For this work, Δd was calculated using the formula

$$\Delta d = min\left(\frac{V}{\alpha}, \frac{V}{\beta}, \frac{V}{\gamma}\right) \tag{a.61}$$

where the *min* (minimum) operation is taken over all cells in the computational domain.

With the time step calculated, equations (a.58) and (a.59) are applied to each finite volume cell in the interior of the computational domain. This, however, is insufficient for determining a unique solution until proper boundary conditions are specified on the surface of the scattering object and at the outer periphery of the domain. The boundary conditions are treated in the next section.

A.6 Boundary Conditions

It is through the application of the boundary conditions that a unique solution is obtained to any partial differential equation [76]. This section briefly mentions the appropriate analytical boundary conditions for the Maxwell equations and then discusses the numerical boundary conditions used in this work.

A.6.1 Analytical Boundary Conditions

A.6.1.1 Surface Boundary Conditions

Maxwell's equations require that the tangential electric- and normal magnetic-field components be continuous across an interface which separates two media of differing characteristic impedances. Furthermore, the magnitude of the jumps in the tangential magnetic and normal electric fields must be equal to the magnitudes of the conduction current and electric-charge densities, respectively. Mathematically, these conditions are stated as [65]

$$\hat{n} \times \vec{E}_t = 0 \qquad \text{(a.62)}$$

$$\hat{n} \cdot \vec{B}_t = 0 \qquad \text{(a.63)}$$

$$\hat{n} \times \vec{B}_t / \mu = \vec{J} \qquad \text{(a.64)}$$

$$\hat{n} \cdot \vec{E}_t / \varepsilon = \rho \qquad \text{(a.65)}$$

Although equations (a.62) and (a.63) specify two components of the surface electric and one component of the surface magnetic field, respectively, the surface charge and current distributions are usually unknown *a priori* and therefore equations (a.64) and (a.65) do not provide any information useful for a numerical boundary condition implementation. Unfortunately, this leaves three of the six electromagnetic field components unresolved.

A.6.1.2 Far-Field Boundary Condition

The far-field boundary condition is simply a condition which must be applied as a result of the truncated computational space. Physically, the condition simply states that the scattered field must be outwardly propagating (i.e., propagating away from the body) at the outer boundary.

A.6.2 Numerical Boundary Conditions

A.6.2.1 Surface boundary condition

Unlike that for the analytical boundary conditions, the numerical boundary conditions must specify a surface condition on all six dependent variables. This poses a problem on the surface of a PEC body since only three conditions can be gleaned from the analytical conditions embodied in equations (a.62) and (a.63). Those are

$$E_{t1} = E_{t2} = 0 \tag{a.66}$$

and

$$B_n = 0 \tag{a.67}$$

where E_{t1} and E_{t2} are the two components of the total-electric-field vector which are tangential to the scatterer surface and B_n is the component of the total magnetic field normal to the surface. In order to determine the remaining three field components, Shang and Gaitonde [89] have developed the extrapolation conditions given by

$$\hat{n} \cdot \nabla(\hat{n} \times \vec{B}_t/\mu) = 0 \tag{a.68}$$

$$\hat{n} \cdot \nabla(\hat{n} \cdot \vec{E}_t/\varepsilon) = 0 \tag{a.69}$$

The statements contained in equations (a.68) and (a.69) imply that the surface changes and currents exist in infinitesimally thin sheets at the surface of the body. For illustrative purposes, a description of the process for determining the unknown electric-field component is obtained using equation (a.69) in conjunction with equation (a.62). To begin, the total electric field in Cartesian space is written as

$$\vec{E}_t = Ex\hat{i} + Ey\hat{j} + Ez\hat{k} \tag{a.70}$$

where $(\hat{i}, \hat{j}, \hat{k})$ are an orthonormal vector set which spans the Cartesian space. Note that this orthonormal vector set can be expressed in terms of another orthonormal vector set ($\hat{\xi}$, $\hat{\eta}$, $\hat{\zeta}$) in which one of the vectors is oriented normal to the scattering body as follows:

$$\begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix} = \begin{bmatrix} \xi i & \eta i & \zeta i \\ \xi j & \eta j & \zeta j \\ \xi k & \eta k & \zeta k \end{bmatrix} \begin{bmatrix} \hat{\xi} \\ \hat{\eta} \\ \hat{\zeta} \end{bmatrix} \tag{a.71}$$

where the terms of the form $\xi i \equiv \hat{\xi} \cdot \hat{i}$ are simply the direction cosines between the two coordinate systems. Substituting equation (a.71) into equation (a.70) and rearranging gives

$$\vec{E}_t = E1\hat{\xi} + E2\hat{\eta} + E3\hat{\zeta} \tag{a.72}$$

where

$$\begin{aligned} E1 &= Ex(\xi i) + Ey(\xi j) + Ez(\xi k) \\ E2 &= Ex(\eta i) + Ey(\eta j) + Ez(\eta k) \\ E3 &= Ex(\zeta i) + Ey(\zeta j) + Ez(\zeta k) \end{aligned} \tag{a.73}$$

Although the statement was made that one of the coordinate directions in the general system is directed normal to the body, it is convenient to assume for a moment that the body unit normal, $\hat{n}$, can be written as

$$\hat{n} = n1\hat{\xi} + n2\hat{\eta} + n3\hat{\zeta} \tag{a.74}$$

Substituting equations (a.72) and (a.74) into equation (a.62) yields the surface conditions

$$\begin{aligned} n2E3 - n3E2 &= 0 \\ n1E3 - n3E1 &= 0 \\ n1E2 - n2E1 &= 0 \end{aligned} \tag{a.75}$$

and performing a similar substitution into equation (a.69) gives

$$n1\frac{\partial}{\partial\hat{\xi}}(En) + n2\frac{\partial}{\partial\hat{\eta}}(En) + n3\frac{\partial}{\partial\hat{\zeta}}En = 0 \tag{a.76}$$

where

$$En = n1E1 + n2E2 + n3E3 \tag{a.77}$$

If a first-order-accurate approximation to the derivatives appearing in equation (a.76) is now employed, then

$$\frac{n1}{\Delta\xi}[n1(\delta E1)_\xi + n2(\delta E2)_\xi + n3(\delta E3)_\xi] + \frac{n2}{\Delta\eta}[n1(\delta E1)_\eta + n2(\delta E1)_\eta + n3(\delta E1)_\eta] + \frac{n3}{\Delta\zeta}[n1(\delta E1)_\zeta + n2(\delta E1)_\zeta + n3(\delta E1)_\zeta] = 0 \tag{a.78}$$

where, for example,

$$\begin{aligned}(\delta E1)_\xi &= E1_{i+1} - E1_i \\ (\delta E1)_\eta &= E1_{j+1} - E1_j \\ (\delta E1)_\zeta &= E1_{k+1} - E1_k\end{aligned} \tag{a.79}$$

Equations (a.75) and (a.78) hold for an arbitrarily oriented surface. If it is now assumed that the surface normal is oriented in the ξ direction, then $n2 = n3 = 0$ and from (a.75),

$$E2 = E3 = 0 \tag{a.80}$$

Furthermore (a.78) becomes

$$(\delta E1)_\xi = 0 \tag{a.81}$$

Denoting the surface cell as cell 1, then equations (a.80) and (a.81) can be written as the system

$$\begin{aligned}(Ex(\xi i) + Ey(\xi j) + Ez(\xi k))_1 &= (Ex(\xi i) + Ey(\xi j) + Ez(\xi k))_2 \\ (Ex(\eta i) + Ey(\eta j) + Ez(\eta k))_1 &= 0 \\ (Ex(\zeta i) + Ey(\zeta j) + Ez(\zeta k))_1 &= 0\end{aligned} \tag{a.82}$$

which can be solved for the three unknown components of the electric field on the surface.

A.6.2.2 Second-Cell Calculations

Figure A.4*a* shows the geometry of the surface and second cells. From the figure, it is apparent that the third-order-accurate flux-calculation scheme given in equation (a.55) would require dependent-variable information from inside the body in order to compute the positive flux component at the cell 1/cell 2 interface. To circumvent this problem, the dependent-variable values at the cell face can be obtained via several methods including one-sided extrapolation or by averaging the values of the dependent variables in the first and second cells. Both method were investigated during this work, and the results were found to be comparable.

A.6.2.3 Far-Field Boundary Condition

Implementing the condition that the scattered field must be outgoing at the edge of the computational

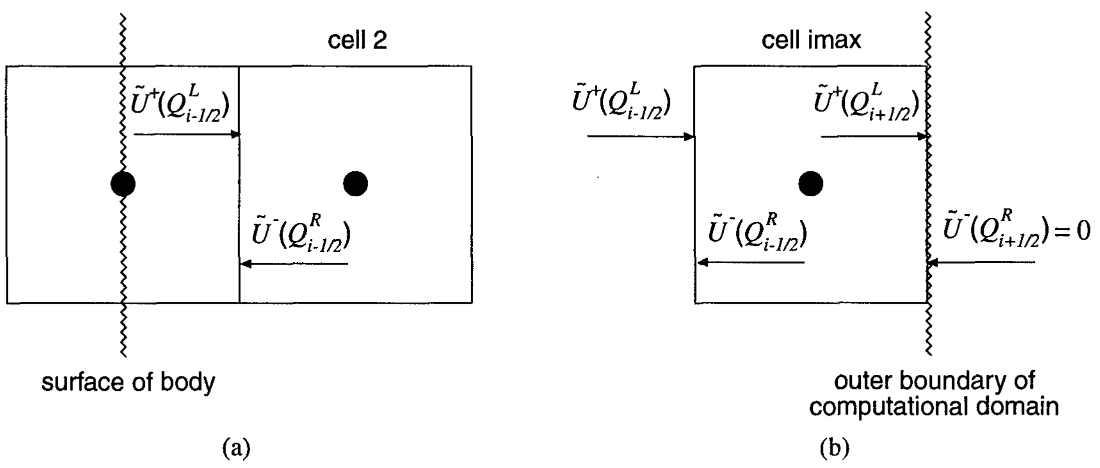

cell 2
$\tilde{U}^{+}(Q^L_{i-1/2})$
$\tilde{U}^{-}(Q^R_{i-1/2})$
surface of body
(a)
cell imax
$\tilde{U}^{+}(Q^L_{i-1/2})$
$\tilde{U}^{+}(Q^L_{i+1/2})$
$\tilde{U}^{-}(Q^R_{i-1/2})$
$\tilde{U}^{-}(Q^R_{i+1/2}) = 0$
outer boundary of computational domain
(b)

Figure A.4: Numerical Boundary Condition Details: a) Surface, b) Far Field

domain is accomplished by setting the incoming flux component to zero as shown in Figure A.4b. This condition is only exact if the direction of the wave propagation is aligned with the normal to the outer boundary of the computational domain[1]. Of the four flux components depicted in Figure A.4b, only the positive component on the $i - 1/2$ face can be computed via a third-order-accurate extrapolation; there is insufficient data for that degree of extrapolation for the remaining terms. For the present study, the positive flux component on the $i + 1/2$ face was computed via a one-sided extrapolation, and the negative component at the $i - 1/2$ face was computed via averaging.

Specification of the boundary conditions completes the description of the numerical procedure, and provides the necessary description in order to conduct a time-domain simulation of an electromagnetic scattering or wave-propagation problem. It is usually the objective of such a simulation, however, to extract the radar cross section of the target. The following section details the procedure used in the current study.

A.7 Radar Cross Section (RCS) Computations

A.7.1 RCS Definition

In most scattering simulations, the ultimate objective is to determine the radar cross section of the

1. Weber [132] provides a thorough discussion of the outer boundary condition implementation for a two-dimensional problem.

scattering object. For a three-dimensional target, the RCS, σ, is defined as [35]

$$\sigma = \lim_{R \to \infty} 4\pi R^2 \left| \frac{\vec{E}_s^2}{\vec{E}_i^2} \right| \qquad \text{(a.83)}$$

where R is the magnitude of a vector $\vec{R}$ from the origin of the coordinate system (presumed on, in, or near the scattering body) to the point of observation. Note that the magnetic fields can be used instead of the electric fields in the above definition with equal validity. From equation (a.83), it is clear the RCS can only be computed once the scattered fields far from the body are known. The scattered fields are given by the Stratton-Chu integral equations [114], namely,

$$\vec{E}_s(\vec{R}) = \frac{1}{4\pi} \iint_S [j\omega\mu(\hat{n} \times \vec{H}_t)G + (\hat{n} \times \vec{E}_t) \times \nabla G + (\hat{n} \cdot \vec{E}_t)\nabla G] dS \qquad \text{(a.84)}$$

$$\vec{H}_s(\vec{R}) = \frac{1}{4\pi} \iint_S [(\hat{n} \times \vec{H}_t) \times \nabla G + (\hat{n} \cdot \vec{H}_t)\nabla G - j\omega\varepsilon(\hat{n} \times \vec{E}_t)G] dS \qquad \text{(a.85)}$$

where $j = \sqrt{-1}$, S is the surface over which the integration is performed, $\hat{n}$ is the surface unit normal, ω is the angular frequency of the wave, and G is the free-space Green's function given by

$$G = \frac{e^{j\omega\sqrt{\mu\varepsilon}|\vec{R} - \vec{R}'|}}{|\vec{R} - \vec{R}'|} \qquad \text{(a.86)}$$

The vectors $\vec{R}$, $\hat{r}$, and $\vec{R}'$ are shown in Figure A.5.

By allowing R to grow arbitrarily large, the Stratton-Chu integral equations become [35],

$$\vec{E}_s\Big|_{R \to \infty} = \frac{jke^{jkR}}{4\pi R} \iint_S [\sqrt{\mu/\varepsilon}(\hat{n} \times \vec{H}_t) - (\hat{n} \times \vec{E}_t) \times \hat{r} - (\hat{n} \cdot \vec{E}_t)\hat{r}] e^{-jk(\hat{r} \cdot \vec{R}')} dS \qquad \text{(a.87)}$$

$$\vec{H}_s\Big|_{R \to \infty} = \frac{-jke^{jkR}}{4\pi R} \iint_S [\sqrt{\varepsilon/\mu}(\hat{n} \times \vec{E}_t) + (\hat{n} \times \vec{H}_t) \times \hat{r} - (\hat{n} \cdot \vec{H}_t)\hat{r}] e^{-jk(\hat{r} \cdot \vec{R}')} dS \qquad \text{(a.88)}$$

where

$$k = \omega\sqrt{\mu\varepsilon} \qquad \text{(a.89)}$$

In this study, only the scattered electric field was used to compute the reported RCS results. Equation

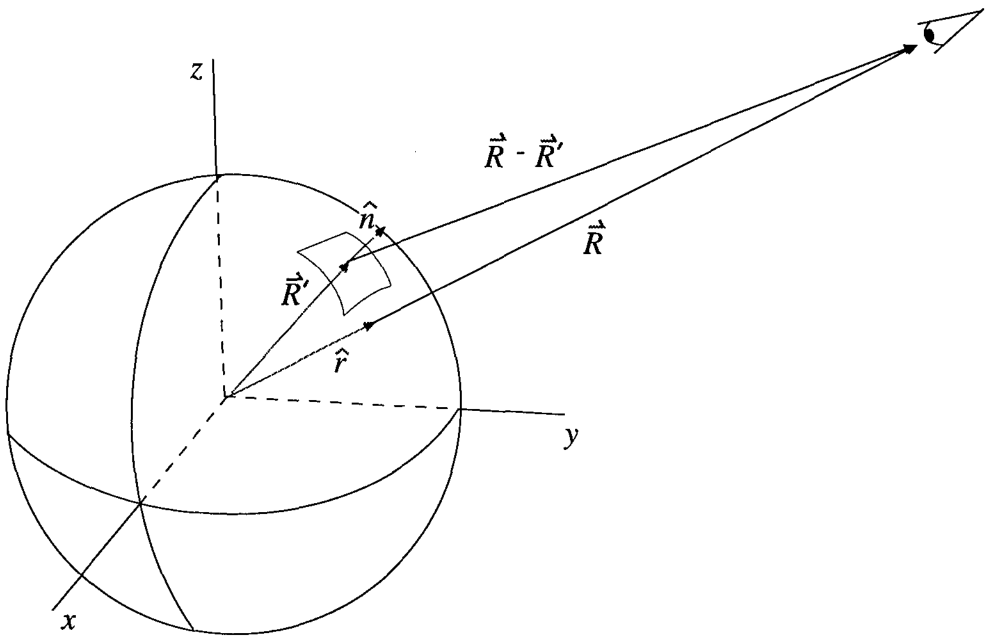


Figure A.5: RCS Geometry Detail

(a.88) is merely presented for completeness. Note that the scattered fields are a function of the position vector $\vec{R}$, and therefore the integration contained in equation (a.88) must be performed for *each* desired observation point. A discussion of the commonly used values for $\vec{R}$ appears in Section A.7.3.

A.7.2 The Fourier Transform

The radar cross section of a target is a frequency-domain response, and consequently, all terms appearing in the above equations must be computed in the frequency domain. Since the FVTD methodology is, by definition, conducted in the time domain, the fields for the cells comprising the surface S described in Section A.7.1 must be converted to the frequency domain using a discrete Fourier transform defined by [131]

$$Q(\omega) = \frac{1}{N}\sum_{n=0}^{N} (Q(t))e^{-j\omega n\Delta t} \tag{a.90}$$

where N is the number of time steps over which the fourier sampling takes place. Note that N must equate to an integral number of excitation periods. Because the total fields appear in equations (a.87) and (a.88) and

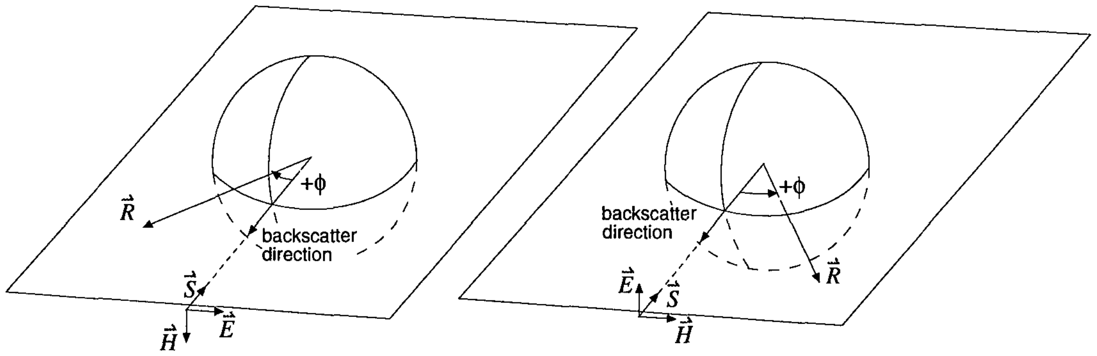


Figure A.6: HH- and VV-Plane Detail

the dependent variables in the FVTD scheme are scattered-field quantities, the incident field must be added to the scattered field during the sampling. Furthermore, it must be remembered the magnitude of the field in the frequency domain does not necessarily correspond to the magnitude in the time domain. For example, a sinusoidally varying incident field having an amplitude of one in the time domain has a magnitude of one-half in the frequency domain.[1]

A.7.3 RCS Profiles

Although the RCS can be reported for any observation point relative to the scattering object, it is usually reported in the two cutting planes depicted in Figure A.6. The HH-cutting plane (sometimes referred to as simply HH or HH polarization) is formed from the observation and electric field vectors while the VV plane is formed from the observation and magnetic field vectors. In all cases, the Poynting vector, $\vec{S}$, lies in the plane of observation. The figures show the convention for increasing look angle, ϕ, used thoughout this work.

1. In actuality, a frequency-domain plot of a sinusoidal wave is two delta functions located at $\pm\omega$, each of magnitude 1/2 [131].

Appendix B: Computer Resource Description

This appendix provides a description of the parallel machines and message-passing libraries used during the course of this investigation. The material is meant to supplement the parallel domain-decomposition studies presented in Chapters 4 and 5.

B.1 Target Architectures

The machines discussed in this Appendix represent the state of the art in massively parallel computing platforms at the outset of this study. Although newer and faster machines are being introduced as of this writing, they generally represent upgrades in central processing unit (CPU) speeds and do not differ radically from the machines used in this study in terms of interprocessor-connection topology and general machine characteristics. The variety of paradigms and architectures present in today's parallel computing machines necessitates a detailed understanding of the unique features of each machine if maximal performance is to be extracted. The following paragraphs discuss the key aspects of the primary parallel machines used in this work: the Intel Paragon, the IBM SP2, and the Cray T3D. A summary of the characteristics of each machine is contained in Table B.1.

B.1.1 Intel Paragon XP/S

The Paragon at WPAFB is comprised of 368 computational, 7 service, and 24 I/O nodes. Each node contains two Intel i/860XP RISC processors, one of which functions as a message-routing chip. Each superscalar processor operates at 50 megahertz (MHz) and is capable 75-million floating point operations (MFlops) per second. All computational nodes have 32 megabytes of memory with the exception of sixteen MP nodes which have 64. The nodes are connected in a two-dimensional mesh topology via a high-speed bus capable of 200 MBytes/sec throughput [44]. Forty-four of the nodes are grouped into one partition which can run jobs interactively. The remainder of the nodes must be utilized through a batch queueing system with the largest number of nodes currently available to a single job limited to 128. Message passing on the Paragon can be accomplished through a variety of libraries including Parallel Virtual Machine (PVM) [10], Message Passing Interface (MPI) [110], and the proprietary Intel NX library [73].

B.1.2 Maui IBM SP2

The 400-node IBM SP2 located in Maui, Hawaii is a distributed-memory platform based on the IBM RS6000 RISC microprocessor running at 66.5 Mhz and capable of 266 MFlops/second peak performance. The machine is assembled as a series of frames with up to 8 *wide* or 16 *thin* nodes per frame. Wide nodes typically possess more memory and greater expansion capability than thin nodes. Memory configuration varies across the nodes with thin nodes possessing between 64 and 128 megabytes and wide nodes between 256 and 1024 megabytes. Communication between the nodes is accomplished either via a high-speed switch [69] which is capable of transmission rates up to 40 megabytes per second or via ethernet which is significantly slower. Like the Paragon, the SP2 is divided into various partitions (termed *pools*). Currently, the maximum number of processors available to a single job is limited to 128. Message-passing libraries available on the machine include PVM, PVM enhanced (PVMe), MPI, and IBM's proprietary Message Passing Library (MPL).

B.1.3 WPAFB MSRC SP2

The SP2 at the WPAFB MSRC is an upgraded yet smaller version of the Maui SP2. The machine is based on the RS6000 processor operating at 77 Mhz with a peak performance rating of 308 MFlops/second. Of the 80 nodes currently installed in the machine, 57 are dedicated compute nodes, 4 are interactive nodes, and the remaining are server and gateway nodes. The backplane is significantly upgraded from that found in the Maui SP2. While still an omega network, the bandwidth has been increased to approximately 100 Mbytes/sec [69,126]. Each node has 1024 megabytes of RAM installed.

B.1.4 Cray T3D

The Cray T3D is based on the DECchip 21064 RISC microprocessor which is rated at 150 MFLOPS peak performance. Up to 2048 such processors (termed *processing elements* by Cray Research) can be connected together via a 300 Mbyte/sec bus configured as an interleaved three-dimensional torus. Each computational node of the T3D is comprised of two processing elements with their associated local memory, a block transfer engine which is used to efficiently move data to and from remote processing elements, and a network interface. The memory structure of the T3D is significantly different than that of either the Paragon or the SP2 in that it is formally classified as a shared-distributed memory system. Although the memory is physically distributed, a global address space is used and any processing element can directly access the memory contents of any other such element. Memory hierarchy consists of on-board CPU registers, an 8 KByte direct-mapped cache, up to 64 MBytes of local memory, and finally, remote memory. Access to cache is one order of magnitude faster than access to local memory which in turn is one order of magnitude faster than a remote memory access [34]. The Cray T3D is a flexible machine that allows for a variety of programming methods including data sharing, work sharing, message passing, and explicit shared memory [33]. For the purposes of this work, the message-passing paradigm provides the highest degree of commonality with the other architectures on which the computer code is designed to run. On the T3D, explicit message passing is accomplished through the use of PVM [75] or MPI.

B.2 Message-Passing Libraries

As discussed in the previous section, a variety of message-passing libraries are available on each of the machines. During the early phases of this work (such as for the study presented in Chapter 4), the libraries used represented what was considered to be an optimal trade-off between performance, portability and support. PVM for the T3D was actively supported by Cray Research and was therefore selected for use despite its relatively poor performance (see Chapter 4). A similar high level of support existed for the NX library on the Paragon. In contrast to PVM on the T3D, however, NX represented the fastest library available on the Paragon. The SP2 from the outset supported a wide range of libraries. Ordered from worst to best performance, they are given by PVM, Linda, PVMe, MPI, and MPL. Because the difference in performance between MPI and MPL was only roughly ten percent, and because MPI is much more portable, MPI was used on the SP2 from the outset.

As of this writing, MPI has become a message-passing-library standard due to its fine mix of good performance and high portability. For this reason, that library was used on all machines for the latter phases of this study.

	Cray T3D (Eglin Air Force Base, FL)	IBM SP2 (WPAFB MSRC)	IBM SP2 (Maui High Performance Computing Center)	Intel Paragon XP/S (Wright Patterson Air Force Base, OH)
central processing unit (CPU)	DEC Alpha 21064 @ 150 megahertz	IBM RS/6000 @ 77 megahertz	IBM RS/6000 @ 66.7 megahertz	Intel i/860XP @ 50 megahertz
single processor megaflop rating (double precision)	150	308	266	75
number of compute processors	128	57	400	368
memory per CPU (megabytes)	64	1024	64-1024	32-64
interprocessor connection topology	3D torus configured 8 x 4 x 4	Omega network	Omega network	Two-dimensional mesh
communication bandwidth (megabytes/sec)	300	100	40	200
communication library used	PVM, MPI	MPI	MPI	NX, MPI
maximum processors available to a single job (without special request)	128	57	128	128

Table B.1: Target Machine Characteristics

Bibliography

1. Adve, V. S. and Vernon, M. K., "Performance Analysis of Mesh Interconnection Networks with Deterministic Routing," *Transactions on Parallel and Distributed Systems*, Vol. 5, No. 3, March 1994.

2. Aftosmis, M. J., "Two Compact Cell-Vertex Methods for Computational Electromagnetics," AIAA Paper 91-1504, 1991.

3. Ahuja, V. and Long, L. N., "A Message Passing Finite Volume Algorithm for Maxwell's Equations on Parallel Machines," AIAA Paper 95-1967, 1995.

4. Anderson, D. A., Tannehill, J. C. and Pletcher, R. H., *Computational Fluid Mechanics and Heat Transfer*, Hemisphere Publishing Corporation, 1984.

5. Anderson, W. K., Thomas, J. L. and van Leer, B., "A Comparison of Finite Volume Flux Vector Splittings for the Euler Equations," AIAA Paper 85-0122, 1985.

6. Andreasen, M. G., "Scattering from Bodies of Revolution," *IEEE Transactions on Antennas and Propagation*, Vol. AP-13, No. 2, pp. 303-10, March 1965.

7. Bailey, D. H., Barszcz, E., Dagum L. and Simon, H. D., "NAS Parallel Benchmark Results 3-94," RNR Technical Report RNR-94-006, March 21, 1994.

8. Balanis, C. A., *Advanced Engineering Electromagnetics*, John Wiley & Sons, 1989.

9. Barszcz, E., Weeratunga, S. K. and Meakin, R. L., "Dynamic Overset Grid Communication on Distributed Memory Parallel Processors," AIAA Paper 93-3311, 1993.

10. Beguelin, A., Dongarra J., Geist, A., Manchek, R., Sunderam, V., "A User's Guilde to Parallel Virtual Machine," ORNL/TM-11826, March 1992.

11. Benek, J. A., Steger, J. L. and Dougherty, F. C., "A Flexible Grid Embedding Technique with Application to the Euler Equations." AIAA Paper 83-1944, 1983.

12. Benek, J. A., Buning, P. G., and Steger, J. L., "A 3-D Chimera Grid Embedding Technique," AIAA Paper 85-1523, 1985.

13. Benek, J. A., Steger, J. L., Dougherty, F. C. and Buning, P. G., "Chimera: A Grid-Embedding Technique," AEDC-TR-85-64, April 1986.

14. Berger, M. J., "On Conservation at Grid Interfaces," *SIAM Journal of Numerical Analysis*, Vol. 24, No. 5, October 1987.

15. Bishop, D. G. and Anderson, D. A., "A Comparison of Finite-Volume Time-Domain Schemes for the Maxwell Equations," AIAA Paper 92-0456, 1992.

16. Bishop, D. G., "Analysis of Material-Based Limiters for the Time-Domain Maxwell Equations," AIAA Paper 93-0369, 1993.

17. Blake, D. C. and Buter, T. A., "Domain Decomposition Strategies for Solving Hyperbolic Systems on Distributed Parallel Architectures," AIAA Paper 96-0834, 1996.

18. Blake, D. C. and Buter, T. A., "Domain Decomposition Strategies for Solving the Maxwell Equations on Distributed Parallel Architectures," to appear in the Fall 1997 issue of the *Applied Computational Electromagnetics Society Journal.*

19. Blake, D. C. and Buter, T. A., "Overset Grid Methods Applied to a Finite-Volume Time-Domain Max-

well Equation Solver," AIAA Paper 96-2338, 1996.

20. Blake, D. C., "Application of Unstructured Grid Domain Decomposition Techniques to Overset Grids," Eighth SIAM Conference on Parallel Processing for Scientific Computing, Minneapolis, MN, 14-17 March, 1997.

21. Blake, D. C. and Buter, T. A., "Electromagnetic Scattering Simulations Using Overset Grids on Massively Parallel Computing Platforms," invited paper to the 1997 IEEE AP-S International Symposium, Montreal, Canada, 13-18 July, 1997.

22. Blake, R. J. and Hu, Y. F., "Numerical experiences with partitioning of unstructured meshes," *Parallel Computing*, Vol. 20, pp. 815-29, 1994.

23. Blosch E. L. and Shyy, W., "Parallel Efficiency of Sequential Pressure-Based Navier-Stokes Algorithms on the CM-2 and MP-1 SIMD Computers," AIAA Paper 94-0409, 1994.

24. Bokhari, S. H., "Multiphase Complete Exchange on Paragon, SP2, and CS-2," *IEEE Parallel & Distributed Technology*, Fall 1996.

25. Bouche, D. P., Molinet, R. A. and Mittra, R., "Asymptotic and Hybrid Techniques for Electromagnetic Scattering," *Proceedings of the IEEE*, Vol. 81, No. 12, Dec 1993.

26. Brunig, J. H. and Lo, Y. T., "Multiple Scattering of EM Waves by Spheres Part I–Multipole Expansion and Ray-Optical Solutions," *IEEE Transactions on Antennas and Propagation*, Vol. AP-19, No. 3, May 1971.

27. Brunig, J. H. and Lo, Y. T., "Multiple Scattering of EM Waves by Spheres Part II–Numerical and Experimental Results," *IEEE Transactions on Antennas and Propagation*, Vol. AP-19, No. 3, May 1971

28. Char, B. W., Geddes, K. O., Gonnet, G. H., Leong, B. L., Monagan, M. B. and Watt, S. M., *Maple V Library Reference Manual*, Springer-Verlag, 1991.

29. Chawla, K., "Aerodynamic Optimization Studies on Advanced Architecture Computers," NASA-CR-198045, Feb. 28, 1995.

30. Chessire, G. and Naik, V. K., "An environment for parallel and distributed computation with application to overlapping grids," *IBM J. Res. Develop.*, Vol. 38, No. 3, May, 1994.

31. Cooke-Yarborough, E. H., "Countermeasures receiver techniques," *Radar Development to 1945*, Burns, R., ed., XXXXXX, XXXX.

32. Courant, R., *Differential and Integral Calculus*, 1936.

33. *Cray Research MPP Software Guide*, Cray Research, Inc., 1994.

34. *Cray MPP Fortran Reference Manual*, Cray Research, Inc., 1994.

35. Crispin, J. W. and Siegel, K. M., *Methods of Radar Cross Section Analysis*, Academic Press, 1968.

36. Dongarra, J. L., Meuer, H. W. and Strohmaier, E.., "TOP 500 Supercomputer Sites," Nov. 11, 1993.

37. Drikakis, D., Schreck, E. and Durst, F., "A Comparative Study of Numerical Methods for Incompressible and Compressible Flows on Different Parallel Machines," AIAA Paper 94-0412, 1994.

38. Fletcher, C. A. J., *Computational Techniques for Fluid Dynamics, Vol. II*, Springer-Verlag, 1991.

39. Foster, I., *Designing and Building Parallel Programs*, Addison-Wesley Publishing Company, 1995.

40. Fox, G. C., Williams, R. D. and Messina, P. C., *Parallel Computing Works!*, Morgan Kaufmann Publishers, Inc., 1994.

41. Fuhs, A. E., *Radar Cross Section Lectures*, American Institute of Aeronautics and Astronautics, 1982.

42. Fusco, M., "FDTD Algorithm in Curvilinear Coordinates," *IEEE Transactions on Antennas and Propagation*, Vol. 38, No. 1, Jan., 1990.

43. Gaitonde, D. and Shang, J. S., "High-Order Finite-Volume Schemes in Wave Propagation Phenomena," AIAA Paper 96-2335, 1996.

44. Goosby, D. and Lamont, G. B., "AFIT/ENG Intel Paragon Quick Reference Manual, Version 1.0," Air Force Institute of Technology, Wright-Patterson AFB, April 6, 1994.

45. Grama, A. Y., Gupta, A., and Kumar, V., "Isoefficiency: Measuring the Scalability of Parallel Algorithms and Architectures," *IEEE Parallel & Distributed Technology*, August 1993.

46. Halliday, D. and Resnick, R. *Fundamentals of Physics*, 2nd ed., Extended Version, John Wiley & Sons, 1981.

47. Hammandi, L., Lee, R., and Özgüner, F., "Review of Domain-Decomposition Methods for the Implementation of FEM on Massively Parallel Computers," *IEEE Antennas and Propagation Magazine*, Vol. 37, No. 1, February 1995.

48. Harmon, F. G., "Application of a Finite-Volume Time-Domain Maxwell Equation Solver to Three-Dimensional Objects," Master's Thesis AFIT/GE/ENG/96D-06, Air Force Institute of Technology, 1996.

49. Harrington, R. F., *Time-Harmonic Electromagnetic Fields*, McGraw-Hill Book Co., 1961.

50. Harrington, R. F., *Field Computations by Moment Method in Electromagnetics*, Krieger, 1982.

51. Hill, K. C., personal correspondence, 25 Feb. 1997.

52. Hirsch, C., *Numerical Computation of Internal and External Flows – Computational Methods for Inviscid and Viscous Flows*, John Wiley & Sons, 1990.

53. Hockney, R. W., "The Communication Challenge for MPP: Intel Paragon and Meiko CS-2," *Parallel Computing*, Vol. 20, No. 3, pp. 389-98, March 1994.

54. Hoffman, K. A., *Computational Fluid Dynamics for Engineers*, Engineering Education System, Wichita, KS, 1989.

55. Holland, R. and Simpson, L., "Finite-Difference Analysis of EMP Coupling to Thin Struts and Wires," *IEEE Transactions on Electromagnetic Compatability*, Vol. 23, May 1981.

56. Holland, R., "Finite-Difference Solution of Maxwell's Equations in Generalized Nonorthogonal Coordinates," *IEEE Transactions on Nuclear Science*, Vol. NS-30, No. 6, pp. 4589-4591, December 1983.

57. Hu, Y. F. and Blake, R. J., "Numerical experiences with partitioning of unstructured meshes," *Parallel Computing* 20, pp. 815-829, 1994.

58. Huh, K. S., Shu, M. and Agarwal, R. K., "A Compact High-Order Finite-Volume Time Domain/Frequency-Domain Method for Electromagnetic Scattering," AIAA Paper 92-0453, 1992.

59. Jurgens, T. G. and Taflove, A. "Three-Dimensional Contour FDTD Modeling of Scattering from Single and Multiple Bodies," *IEEE Transactions on Antennas and Propagation*, Vol. 41, No. 12, December 1993.

60. Kaddoura, M., Ou, C. W. and Ranka, S., "Partitioning Unstructured Computational Graphs for Nonuniform and Adaptive Environments," *IEEE Parallel & Distributed Technology*, Fall 1995.

61. Knott, E. F., Shaeffer, J. F., and Tuley, M. T., *Radar Cross Section*, 2nd ed., Artech House, 1993.

62. Kong, J. A., *Electromagnetic Wave Theory*, John Wiley & Sons, 1986.

63. Kumar, V., Grama, A., Gupta, A. and Karypis, G., *Introduction to Parallel Computing*, Benjamin/Cummings Publishing Company, 1994.

64. Kunz, K. S. and Simpson, L. "A Technique for Increasing the Resolution of Finite-Difference Solutions of the Maxwell Equations," *IEEE Transactions on Electromagnetic Compatability*, Vol. 23, No. 4, November 1981.

65. Kraus, J. D., *Electromagnetics*, 4th ed., McGraw-Hill, Inc., 1992.

66. Liang, C. and Lo, Y. T., "Scattering by Two Spheres," *Radio Science*, Vol. 2, No. 12, December 1967.

67. Liu Y., "Fourier Analysis of Numerical Algorithms for the Maxwell Equations," AIAA Paper 93-0368, 1993.

68. Ludwig, A. C., "Scattering by Two and Three Spheres Computed by the Generalized Multipole Technique," *IEEE Transactions on Antennas and Propagation*, Vol. 39, pp. 703-705, May 1991.

69. Miguel, J., Arruabarrena, A., Beivide, R. and Gregorio, J. A., "Assessing the Performance of the New IBM SP2 Communication Subsystem," *IEEE Parallel & Distributed Technology*, Winter 1996.

70. Mohammadian, A. H., Shankar, V. and Hall, W. F., "Application of Time-Domain Finite-Volume Method to Some Radiation Problems in Two and Three Dimensions", *IEEE Transactions on Magnetics*, Vol. 27, No. 5, September 1991.

71. Naik, V. K., personal correspondence, June, 1995.

72. Noack, R. W., and Anderson, D. A., "Time Domain Solutions of Maxwell's Equations Using a Finite-Volume Formulation," AIAA Paper 92-0451.

73. *Paragon C System Calls Reference Manual*, Intel Corporation, Order Number 312487-003, June 1994.

74. Paul, C. R. and Nasar, S. A., *Introduction to Electromagnetic Fields*, McGraw-Hill, 1987.

75. *PVM and HeNCE Programmer's Manual*, SR-2501 5.0, Cray Research, Inc., 1994.

76. Powers, D. L., *Boundary Value Problems*, 3rd ed., Harcourt Brace Jovanovich, 1987.

77. Putnam, J. N. and Gedera, M. B., "CARLOS-3DTM: A General-Purpose Three-Dimensional Method-of-Moments Scattering Code," in the EM Programmer's Notebook, Volakis, J. L., ed., *IEEE Antennas and Propagation Magazine*, Vol. 35, No. 2, April 1993.

78. Putnam, J. N. and Medgyesi-Mitschang, L. N., "Combined Field Integral Equation Formulation for Axially Inhomogeneous Bodies of Revolution (Combined Field Formulation of CICERO)," prepared by McDonnell Douglas Research Laboratories for Sandia National Laboratory under Contract No. 33-4257, MDCQA003, 1 December 1987.

79. Ruck, G. T., Barrick, D. E., Stuart, W. D. and Kirchbaum, C. K., *Radar Cross Section Handbook, Volume I*, Plenum Press, 1970.

80. Ryan, J. S. and Weeratunga, S. K., "Parallel Computation of 3-D Navier-Stokes Flowfields for Supersonic Vehicles," AIAA Paper 93-0064, 1993.

81. Scherr, S. J., "Implementation of an Explicit Navier-Stokes Algorithm on a Distributed Memory Parallel Computer," AIAA Paper 93-0063, 1993.

82. Shaeffer, J. F., "EM Scattering from Bodies of Revolution with Attached Wires," *IEEE Transactions on Antennas and Propagation*, Vol. AP-30, No. 3, May 1982.

83. Shang, J. S. and Scherr, S. J., "Navier-Stokes Solution of the Flow Field Around a Complete Aircraft," AIAA Paper 85-1509, 1985.

84. Shang, J. S., "Characteristic-Based Methods for the Time-Domain Maxwell Equations," AIAA Paper 91-0606, 1991.

85. Shang, J. S., "A Characteristic-Based Algorithm for Solving 3-D, Time-Domain Maxwell Equations," AIAA Paper 92-0452, 1992.

86. Shang, J. S. and Gaitonde, D., "Characteristic-Based, Time-Dependent Maxwell Equations Solvers on a General Curvilinear Frame," AIAA Paper 93-3178, July 1993.

87. Shang, J. S., Hill, K. C., and Calahan, D., "Performance of a Characteristic-Based, 3-D, Time-Domain Maxwell Equations Solver on a Massively Parallel Computer," AIAA Paper 93-3179, 1993.

88. Shang, J. S., Calahan, D. A. and Vikstrom, B., "Performance of a Finite Volume CEM Code on Multicomputers," AIAA Paper 94-0236, 1994.

89. Shang, J. S. and Gaitonde, D., "Scattered Electromagnetic Field of a Reentry Vehicle," AIAA Paper 94-0231, 1994.

90. Shang, J. S. and Fithen, R. M., "A Comparative Study of Numerical Algorithms for Computational Electromagnetics," AIAA Paper 94-2410, 1994.

91. Shang, J. S., "A Fractional-Step Method for Solving 3-D Time-Domain Maxwell Equations," *Journal of Computational Physics* 118, pp. 109-19, 1995.

92. Shang, J. S., "Characteristic-Based Algorithms for Solving Maxwell's Equations in the Time-Domain," *IEEE Antennas and propagation Magazine*, Vol. 37, No. 3, pp. 15-25, June 1995.

93. Shang, J. S., and Shang, C. C., "Concurrent Computation of Electromagnetic Phenomena on the Paragon," AIAA Paper 95-0592, 1995.

94. Shang, J. S. and Scherr, S. J., "Time-Domain Electromagnetic Scattering Simulations on Multicomputers," AIAA Paper 95-1966, 1995.

95. Shang, J. S. and Gaitonde, D., "On High-Resolution Schemes for Time-Dependent Maxwell Equations," AIAA Paper 96-0832, 1996.

96. Shang, J. S., Gaitonde, D., and Wurtzler, K., "Scattering Simulations of Computational Electromagnetics,", AIAA Paper 96-2337, 1996.

97. Shankar, V. and Chakravarthy, S., "Development and Application of Unified Algorithms for Problems in Computational Science," NASA CP 2454, 1987.

98. Shankar, V., Hall, W. and Mohammadian, A. H., "A CFD-Based Finite-Volume Procedure for Computational Electromagnetics – Interdisciplinary Applications of CFD Methods," AIAA Paper 89-1987, 1989.

99. Shankar, V., Mohammadian, A. H. and Hall, W. F., "A Time-Domain, Finite-Volume Treatment for the Maxwell Equations," *Electromagnetics* 10: pp. 127-45, 1990.

100. Shankar, V., Mohammadian, A. H., Hall, W. F. and Erickson, R., "CFD-Spinoff – Computational Electromagnetics for Radar Cross Section (RCS) Studies," AIAA Paper 90-3055, 1990.

101. Shankar, V., "Research to Application – Supercomputing Trends for the 90's – Opportunities for Interdisciplinary Computation," AIAA Paper 91-0002, 1991.

102. Shankar V., "A Gigaflop Performance Algorithm for Solving Maxwell's Equations of Electromagnetics," AIAA Paper 91-1578, 1991.

103. Shankar, V., Hall, W. F., Mohammadian, A. and Rowell, C., "Computational Electromagnetics: Development of a Finite-Volume, Time-Domain Solver for Maxwell's Equations," Final Contract Report of Contract N62269-90-C-0257, May 1993.

104. Shankar, V., Hall, W. F., Mohhamadian, A., Rowell, C., Chakravarthy, S., and Palaniswamy, S., "An Unstructured Grid-Based CEM Solver," Report prepared for Contract DI-SC-92-0010/MDA-908-88-C-9416, Rockwell International Science Center, November 1993.

105. Shankar, V., Hall, W. F., Mohammadian, A. and Rowell C., "Algorithmic Aspects and Supercomputing Trends in Computational Electromagnetics," AIAA Paper 93-0367, 1993.

106. Shankar, V., Hall, W. F., Mohammadian, A., Rowell, C. and Palaniswamy, S., "Advances in Time-Domain CEM Using Structured/Unstructured Formulations and Massively Parallel Architectures," AIAA Paper 95-1963, 1995

107. Shankar, V., Hall, W. F., Rowell, C., Mohammadian, A. and Palaniswamy, S., "Development and Implementation of Computational Electromagnetic Techniques on Massively Parallel Computing Architectures," WL-TR-96-6003, January 1996.

108. Skolnik, M., *Radar Handbook*, 2nd ed., McGraw-Hill, Inc., 1990.

109. Smith, M. H. and Pallis, J. M., "MEDUSA–An Overset Grid Flow Solver for Network-Based Parallel Computer Systems," AIAA Paper 93-3312, 1993.

110. Snir, M., Otto, S., Huss-Lederman, S., Walker, D. W. and Dongarra, J., *MPI: The Complete Reference*, MIT Press, 1996.

111. Southworth, G. C., *Principles and Applications of Waveguide Transmission*, D. Van Nostrand Co., Princeton, NJ, 1956.

112. Stagg, A. K., Cline, D. D. and Carey, G. F., "Parallel, Scalable Parabolized Navier-Stokes Solver for Large-Scale Simulations," *AIAA Journal*, Vol. 33, No. 1, January 1995.

113. Steger, J. L. and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamics Equations with Application to Finite Difference Methods," *Journal of Computational Physics*, Vol. 40, pp. 263-93, April 1981.

114. Stratton, J. A., *Electromagnetic Theory*, McGraw-Hill Book Company, 1941.

115. Strikwerda, J. C., *Finite Difference Schemes and Partial Differential Equations*, Wadsworth & Brooks/Cole, 1989.

116. Suhs, N. E. and Tramel, R. W., "PEGSUS 4.0 User's Manual," AEDC-TR-91-8, November 1991.

117. Taflove, A. "Re-Inventing Electromagnetics: Supercomputing Solution of Maxwell's Equations via Direct Time Integration on Space Grids," AIAA Paper 92-0333, 1992.

118. Taflove, A., *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, Artech House, 1995.

119. Torenbeek, E., *Synthesis of Subsonic Airplane Design*, Delft University Press, 1986.

120. Umashankar, K. and Taflove, A., *Computational Electromagnetics*, Artech House, Inc., 1993.

121. Van Driessche, R. and Roose, D., "An improved spectral bisection algorithm and its application to dynamic load balancing," *Parallel Computing*, Vol. 21, pp. 26-48, 1995.

122. van Leer, B., "Flux-Vector Splitting for the Euler Equations," Technical Report 82-30, ICASE, September 1983.

123. Vinh, H., van Dam, C. P. and Dwyer, H. A., "Shape Optimization for Aerodynamic Efficiency and Low Observability," AIAA Paper 93-3115, 1993.

124. Vinokur, M., "An Analysis of Finite-Difference and Finite-Volume Formulations of Conservation Laws," NASA CR 177416, Palo Alto, CA, June 1986.

125. Volakis, J. L. and Kempel, L. C., "Electromagnetics: Computational Methods and Considerations," *IEEE Computational Science and Engineering*, Vol. 2, No. 1, Spring 1995.

126. Wagner, M., IBM Analyst, personal correspondence, December 1996.

127. Wang, J. C. T. and Widhopf, G. F., "A High-Resolution TVD Finite Volume Scheme for the Euler Equations in Conservative Form," *Journal of Computational Physics* 84, pp. 145-173, 1989.

128. Wang, Z. J. and Yang, H. Q., "A Unified Conservative Zonal Interface Treatment for Arbitrarily Patched and Overlapped Grids," AIAA Paper 94-0320, 1994.

129. Wang, Z. J., Yang, H. Q., and Przekwas, A. J., "Implicit Conservative Interfacing for 3D Overlapped Chimera Grids," AIAA Paper 95-1683, 1995.

130. Warming, R. F. and Beam, R. M., "On the Construction and Application of Implicit Factored Schemes for Conservation Laws," SIAM-AMS Proceedings, Vol. II, 1978.

131. Weaver, H. J., *Applications of Discrete and Continuous Fourier Analysis*, John Wiley & Sons, 1983.

132. Weber, Y. S., "Investigations on the Properties of a Finite-Volume, Time-Domain Method for Computational Electromagnetics," AIAA Paper 95-1963, 1995.

133. Weber, Y. S., Hill, K. C. and Young, J. L., "The Application of Finite-Volume, Time-Domain Techniques to EM Scattering from Cavities and Inlets," AIAA Paper 96-2336, 1996.

134. Weeratunga, S. and Chawla, K. "Overset Grid Applications on Distributed Memory MIMD Computers," AIAA Paper 95-0573, 1995.

135. Weeratunga, S., Barszcz, E. and Chawla, K., "Moving Body Overset Grid Applications on Distributed Memory MIMD Computers," AIAA Paper 95-1751, 1995.

136. Woodward, P. R., "Perspectives on Supercomputing: Three Decades of Change," *Computer*, October 1996.

137. Wong, C. C., Blottner, F. G., Payne, J. L. and Soetrisno, M., "A Domain Decomposition Study of Massively Parallel Computing in Compressible Gas Dynamics," AIAA Paper 95-0572, 1995.

138. Xu, Z. and Hwang, K., "Modeling Communication Overhead: MPI and MPL Performance on the IBM SP2," *IEEE Parallel & Distributed Technology*, Spring 1996.

139. Yadlin Y. and Caughey, D. A., "Parallel Computing Strategies for Block Multigrid Implicit Solution of the Euler Equations," *AIAA Journal*, Vol. 30, No. 8, August 1992.

140. Yee, K. S. "Numerical Solution of Initial Boundary Value Problems Involving Maxwell's Equations in Isotropic Media," *IEEE Transactions on Antennas and Propagation*, Vol. AP-14, pp. 302-307, May, 1966.

141. Yee, K. S., Chen, J. S. and Chang, A. H., "Conformal Finite-Difference Time-Domain (FDTD) with Overlapping Grids," *IEEE Transactions on Antennas and Propagation*, Vol. 40, No. 9, September 1992.

142. Zivanovic, S. S., Yee, K. S. and Mei, K. K., "A Subgridding Method for the Time-Domain Finite-Difference Method to Solve Maxwell's Equations," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 39, No. 3, March 1991.

Vita

Captain Doug Blake [redacted] Ralph E. and Patricia A. Blake [redacted] He joined the Air Force in 1982 to become a Korean Cryptologic Linguist and was initially assigned to the 6903 Electronic Security Group at Osan Air Base, Korea. There he worked as a Korean clear speech voice-intercept operator and ground mission analyst. He was recognized as Group Airman of the Year in 1984. In 1986, he was accepted into the Airman Education and Commissioning Program and attended the University of Missouri, Rolla, completing a Bachelor of Science Degree in Aerospace Engineering and graduating Summa Cum Laude in 1989. After completing Officer Training School as his class's top honor graduate, he was assigned to the Ogden Air Logistics Center Operating Location at Vandenberg Air Force Base, California. There he worked for three years as a project engineer on the Follow On Test and Evaluation Programs for the Minuteman III and Peacekeeper intercontinental ballistic missiles.

In 1992, Captain Blake entered the Air Force Institute of Technology (AFIT) where he worked towards a Master's Degree in Aeronautical Engineering, specializing in the areas of computational fluid dynamics and parallel computing. He graduated in 1993 as the top engineering student in his class. Turning his efforts to the area of computational electromagnetics, he then continued his AFIT studies at the doctoral level. Upon graduation, Captain Blake will be assigned to the Flight Dynamics Directorate of Wright Labs where he is planning on continuing his work in distributed-memory parallel algorithm development for integrated computational electromagnetics/computational fluid dynamics design environments.

Captain Blake is happily married to the former Terri Wissman of Cape Girardeau, Missouri. Together they have two beautiful daughters: Danielle, [redacted] and Taylor [redacted]

[redacted]

[redacted]