

AD-A182 414

A THREE-DIMENSIONAL VECTOR BOUNDARY ELEMENT FORMULATION
FOR MULTIPOLE SCA. (U) UTAH UNIV SALT LAKE CITY DEPT OF
ELECTRICAL ENGINEERING M L TRACY FEB 87 UTEC-MD-86-067

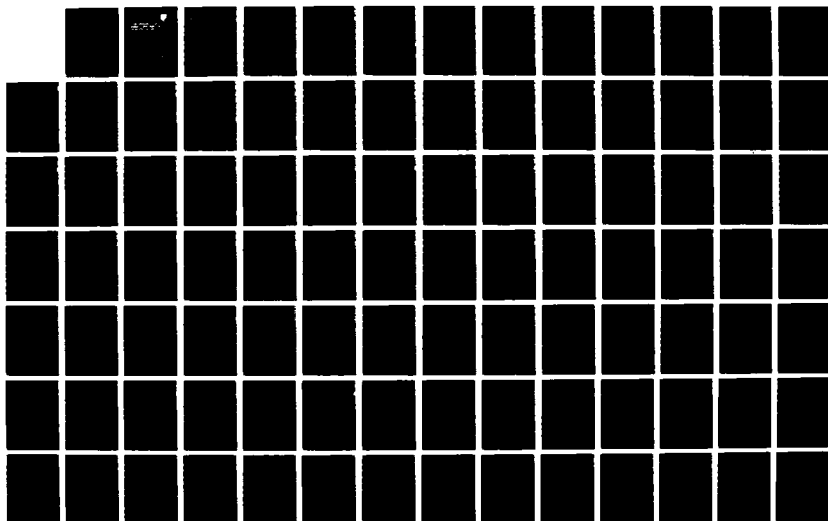
1/2

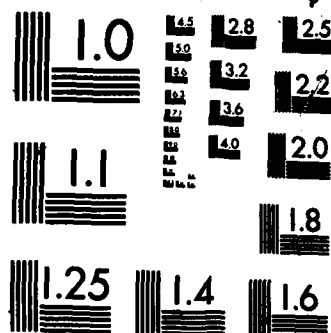
UNCLASSIFIED

RADC-TR-87-24 F30602-82-C-0161

F/G 20/3

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A182 414

RADC-TR-87-24
Final Technical Report
February 1987

DTIC FILE COPY



12

A THREE-DIMENSIONAL, VECTOR, BOUNDARY ELEMENT FORMULATION FOR MULTIPOINT SCATTERING

University of Utah

Michael L. Tracy

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

DTIC
ELECTE
JUL 16 1987
S **D**
E

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

ADA182414

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS N/A		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UTEC MD-86-067			5. MONITORING ORGANIZATION REPORT NUMBER(S) RADC-TR-87-24		
6a. NAME OF PERFORMING ORGANIZATION University of Utah Dept of Electrical Engineering		6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (OCTP)		
6c. ADDRESS (City, State, and ZIP Code) Microwave Device & Physical Electronics Lab Salt Lake City UT 84112			7b. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR		8b. OFFICE SYMBOL (If applicable) NE	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F30602-82-C-0161		
8c. ADDRESS (City, State, and ZIP Code) Bolling AFB Wash DC 20332			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO. 61102F	PROJECT NO. 2305	TASK NO. J9
					WORK UNIT ACCESSION NO. 16
11. TITLE (Include Security Classification) A THREE-DIMENSIONAL, VECTOR, BOUNDARY ELEMENT FORMULATION FOR MULTIPOINT SCATTERING					
12. PERSONAL AUTHOR(S) Michael L. Tracy					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM Sep 82 TO Oct 86		14. DATE OF REPORT (Year, Month, Day) February 1987	
				15. PAGE COUNT 108	
16. SUPPLEMENTARY NOTATION Research accomplished in conjunction with Air Force Thermionics Engineering Research Program (AFTER) AFTER-20. Michael L. Tracy was an AFTER student from Hughes (Cont'd)					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	Waveguide Theory Computer Code		
09	03		Three-Dimensional, Multiport Problem Boundary Element Problem		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Recent work by S. Kagami and I. Fukai and also work by E. Tonye and H. Baudrand has demonstrated the application of the boundary element method for solving planar (two-dimensional) electromagnetic multiport scattering problems. The boundary element method is shown by these workers to be computationally more efficient than using finite element methods for two-dimensional cases. This report extends application of the boundary element method to the three-dimensional case, requiring a field vector integral equation formulation as opposed to the scalar field equation formulation used by the workers above. Computational details are discussed and numerical results are presented for a few simple test cases. The computational storage costs are shown to be proportional to the discontinuity volume raised to the four-thirds power.					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Andrew E. Chrostowski, 1Lt, USAF			22b. TELEPHONE (Include Area Code) (315) 330-4381		22c. OFFICE SYMBOL RADC (OCTP)

DD Form 1473, JUN 86

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

UNCLASSIFIED

Block 16 (Cont'd)

Aircraft Company. This report was submitted in partial fulfillment of the requirements of the degree of Electrical Engineer.

UNCLASSIFIED

ACKNOWLEDGMENTS

This work was supported by the AFTER program at the University of Utah and by the Power Tube Department of the Electron Dynamics Division of Hughes Aircraft Company.

I would like to thank Dr. J. Mark Baird,* my project adviser, from whom I learned normal mode theory. His careful if somewhat delinquent review of this report was appreciated and is responsible for improving the professionalism of the final draft. I would also like to acknowledge Dr. Steven A. Johnson* for showing me the ART algorithm and for never clipping my wings; thank you.

Finally, I would like to dedicate this paper to all of my friends from the AFTER program, for many of you, I hope to remain a continued source of amusement.

Accession	
NTIS	☒
DTIC	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
and/or	
Special	
A-1	



* University of Utah, Department of Electrical Engineering.

TABLE OF CONTENTS

	<u>Page</u>
LIST OF ILLUSTRATIONS AND TABLES	iv
I. INTRODUCTION	1
II. DERIVATION OF AN APPROPRIATE SURFACE INTEGRAL EQUATION . . .	3
III. APPLICATION OF THE METHOD OF MOMENTS	12
IV. RESULTS	26
1. Straight Shot Data	27
2. Straight Shot with Perfect Short	31
3. Straight Shot with Asymmetric Iris	34
V. CONCLUSIONS AND RECOMMENDATIONS	43
APPENDIX A	46
APPENDIX B	54
APPENDIX C	58
REFERENCES	94

LIST OF ILLUSTRATIONS AND TABLES

<u>Figure</u>		<u>Page</u>
1	Geometry for specializing $\bar{x}'$ to a smooth surface S	5
2	Schematic diagram of a N port junction indicating placement of field boundaries defining the discontinuity volume V	13
3	Waveguide cross section for straight shot data	28
4	Computed transmission and reflection amplitudes for the straight shot case without the conservation of power constraint enforced	29
5	Computed versus theoretically exact phase for the transmission data in Fig. 4	29
6	Computed transmission and reflection amplitudes for the straight shot case <u>with</u> the conservation of power constraint enforced	30
7	Computed versus theoretically exact phase for the transmission data in Fig. 6	30
8	Reflection amplitude for the straight shot case with a perfect shorting plane on one side and also without enforcing the conservation of power constraint	33
9	Computed versus theoretically exact phase for the reflection data in Fig. 8	33
10	Computed versus theoretically exact phase for the straight shot with shorting plane case <u>with</u> the conservation of power constraint enforced	34
11	Geometry for asymmetric iris	35
12	Reflection amplitude, case $\delta/a = 0.75$	37
13	Reflection phase, case $\delta/a = 0.75$	37
14	Transmission amplitude, case $\delta/a = 0.75$	38
15	Transmission phase, case $\delta/a = 0.75$	38
16	Reflection amplitude, case $\delta/a = 0.625$	39
17	Reflection phase, case $\delta/a = 0.625$	39

<u>Figure</u>		<u>Page</u>
18	Transmission amplitude, case $\delta/a = 0.625$	40
19	Transmission phase, case $\delta/a = 0.625$	40
20	Average error per equation, case $\delta/a = 0.75$	42
21	Average error per equation, case $\delta/a = 0.625$	42

I. INTRODUCTION

This report presents a three-dimensional, vector, boundary element, or surface integral moment-method formulation for scattering inside waveguides and represents a fairly logical extension of two-dimensional work done by S. Kagami and I. Fukai,¹ and also work by E. Tonye and H. Baudrand.² The method uses mode expansions in the regular waveguide of each port, and allowance for multiple ports incurs no additional theoretical cost, as this is a multiport formulation. Therefore, at least in principle, multimode multiport scattering matrix information can be obtained. The primary restrictions are that the discontinuity boundary be perfectly conducting and that the material filling the discontinuity volume be homogeneous and isotropic. These restrictions may, to some extent, be lifted.¹

Other numerical schemes that produce multimode scattering data are mode matching^{4,5} at abrupt transitions from one waveguide to another, and other methods⁶ based upon Solyman's⁷ equations for tapering transitions between waveguides. These are excellent tools that have recently been developed to the point of having practical application. Both methods apply only to two-port transitions, and the central axis of the two connected guides must be parallel. A boundary element formulation overcomes these restrictions, albeit at an even more intensive computational cost.

Another approach that could be potentially more flexible than the one described in this report would be based on finite element or finite difference schemes. Initially, we thought a boundary element scheme

would require less computer storage to implement than a corresponding difference approach to the same problem. This is probably true for two-dimensional problems, but in three dimensions it is not. It turns out that for a boundary element scheme, one needs to compute and store coupling coefficients from every surface patch to every other surface patch. This means the storage required is proportional to the surface area squared or the enclosed volume to the four-thirds power, whereas a corresponding difference approach requires storage proportional to the volume. Without knowing the coefficients of proportionality for the two different cases, nothing more can be said by way of comparison. It is important to realize however that a boundary element formulation in three dimensions does not a priori require less storage than a finite element formulation.

An interesting and possibly less computationally intensive approach, based on variational techniques for irregular waveguide transitions, is presented by Bernstein, et al.⁸

This work was motivated by the need for a tool capable of analyzing the rapid transition from a regular waveguide to a slow wave circuit. Although the theory presented is not sufficiently well developed to do this job, the goal was simply to develop a framework upon which such a problem could properly be considered. This is primarily an exposition on the theoretical and computational aspects of the boundary element method as it applies to waveguide discontinuities. Numerical results are given for a few simple but important test cases to verify that the work given here does have substance and to demonstrate, rather specifically, what types of computational expense to expect.

II. DERIVATION OF AN APPROPRIATE SURFACE INTEGRAL EQUATION

In this section, Green's theorem is applied to a scalar wave function, thereby obtaining the wave function interior to a volume in terms of its boundary values. The point of evaluation is then "pushed" to the surface of the volume by a limiting process, yielding a surface integral equation for the wave function. It is then noted that the electric and magnetic field intensities satisfy vector wave equations, and that the result derived for a scalar wave function applies to the rectangular components of these equations. Vector integral equations are then obtained by recombining the component equations. To wrap up, the integrand is simplified for the case when part of the boundary is perfectly conducting.

Green's theorem,

$$\oint_S (\phi \nabla \psi - \psi \nabla \phi) \cdot \hat{n} d\alpha = \int_V (\phi \nabla^2 \psi - \psi \nabla^2 \phi) d^3x \quad (1)$$

applies on a volume V with volume element d^3x , S is a closed surface bounding V , with area element $d\alpha$ and unit outward normal $\hat{n}$ at $d\alpha$. Take ψ to be a solution to the inhomogeneous wave equation,

$$(\nabla^2 + k^2)\psi(\bar{x}) = -F(\bar{x}) \quad (2)$$

for all points $\bar{x}$ in the volume V . Solving for $\nabla^2 \psi$ in this equation and substituting the result into Eq. 1 gives

$$\oint_S \{ \phi(\bar{x}) \nabla \psi(\bar{x}) - \psi(\bar{x}) \nabla \phi(\bar{x}) \} \cdot \hat{n} d\alpha = - \int_V \{ F(\bar{x}) \phi(\bar{x}) + \psi(\bar{x}) [\nabla^2 + k^2] \phi(\bar{x}) \} d^3x \quad (3)$$

Choice of the function $\phi(\bar{x})$ is quite arbitrary, but Eq. 3 indicates that some reduction in complexity is possible by choosing ϕ such that

$$(\nabla^2 + k^2) \phi(\bar{x}) = - \delta(\bar{x} - \bar{x}') \quad (4)$$

where $\phi(\bar{x} - \bar{x}')$ is the Dirac delta function. In effect, take $\phi(\bar{x}) = \phi(\bar{x}, \bar{x}')$ to be a "Green's function." If the geometry under consideration is excessively simple, then the geometrically correct Green's function, satisfying either Dirichlet or Neumann boundary conditions, is known explicitly. For this case, Eq. 3 reduces to the well known result,

$$\phi(\bar{x}') = \int_V F(\bar{x}) \phi(\bar{x}, \bar{x}') d^3x \quad (5)$$

Usually, the physically appropriate Green's function is beyond knowing explicitly, and so it makes sense to choose $\phi(\bar{x}, \bar{x}')$ such that it has no geometrical bias. Clearly, the "best" choice in this case is the free space Green's function,

$$\phi(\bar{x}, \bar{x}') = \frac{e^{\pm jk|\bar{x}' - \bar{x}|}}{4\pi|\bar{x}' - \bar{x}|} \quad (6)$$

In making this choice, it has been assumed that k is constant throughout V . As you might have guessed, the $\pm$ sign in Eq. 6 is arbitrary. But

for $e^{j\omega t}$ time dependence, the $-$ sign is the standard choice and the one used here. For future reference,

$$\nabla\phi(\bar{x}, \bar{x}') = \frac{e^{-jkR}}{4\pi R^2} \left[jk + \frac{1}{R} \right] (\bar{x}' - \bar{x}) \quad (7)$$

where $R = |\bar{x}' - \bar{x}|$. When Eq. 6 is used in Eq. 3 and $\bar{x}'$ is taken inside V , the result is

$$\psi(\bar{x}') = \int_V F(\bar{x})\phi(\bar{x}, \bar{x}') d^3x + \oint_S \{ \phi(\bar{x}, \bar{x}')\nabla\psi(\bar{x}) - \psi(\bar{x})\nabla\phi(\bar{x}, \bar{x}') \} \cdot \hat{n}(\bar{x}) d\bar{a} \quad (8)$$

For $\bar{x}'$ outside V , the left-hand side of this equation is zero. Comparing Eq. 8 to the result when the geometrically correct Green's function is known (Eq. 5), we see that the price paid for not having the correct Green's function is in the addition of a surface integral to the integral equation.

Next, the evaluation point $\bar{x}'$ in Eq. 8 is specialized to a point on S by a limiting process that is outlined by Brebbia.³ Figure 1 is a

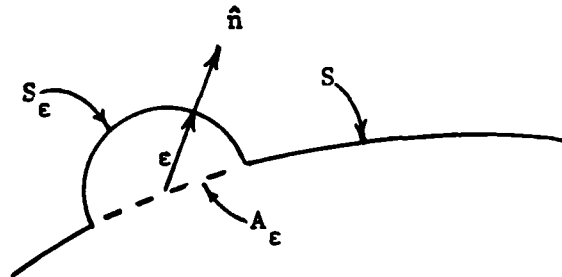


Fig. 1. Geometry for specializing $\bar{x}'$ to a smooth surface S .

blown-up view depicting a small section of the smooth surface S distorted outward by a small hemisphere having radius ϵ and surface S_ϵ . The point $\bar{x}'$ is taken to be at the center of the hemisphere and on the original surface S . That part of the original surface now displaced by S_ϵ will be referred to as A_ϵ . In this geometry, $\bar{x}'$ is inside the distorted volume, so that Eq. 8 remains valid. Now let us see what change the terms in this equation go through in the limit as the distorting radius ϵ goes to zero. Consider the surface integration in the far right term of Eq. 8,

$$\begin{aligned} & \lim_{\epsilon \rightarrow 0} \int_{S-A_\epsilon} \psi(\bar{x}) \nabla \phi(\bar{x}, \bar{x}') \cdot \hat{n} d\alpha + \lim_{\epsilon \rightarrow 0} \int_{S_\epsilon} \psi(\bar{x}) \nabla \phi(\bar{x}, \bar{x}') \cdot \hat{n} d\alpha \\ &= \int_S \psi(\bar{x}) \nabla \phi(\bar{x}, \bar{x}') \cdot \hat{n} d\alpha + \lim_{\epsilon \rightarrow 0} \int_0^{2\pi} \int_0^{\pi/2} \psi(\bar{x}' + \epsilon \hat{r}) \frac{e^{-jk\epsilon}}{4\pi\epsilon^2} \left[jk + \frac{1}{\epsilon} \right] (-\epsilon) \\ & \quad \cdot \epsilon^2 \sin \theta \, d\theta \, d\phi \\ &= \int_S \psi(\bar{x}) \nabla \phi(\bar{x}, \bar{x}') \cdot \hat{n} d\alpha - \frac{1}{2} \psi(\bar{x}') \end{aligned}$$

The $1/2 \psi$ term, resulting from the limiting process above, is the classic result obtained when $\bar{x}'$ is specialized to any smooth point on S . When specializing to corners or edges on S , a more general term of the form $\eta\psi$ is obtained, where η is between 0 and 1. The explicit value of η can be obtained for each case by the method used here for smooth surfaces. For edges, we have $\eta = \theta/2\pi$, where θ is the edge angle as measured from outside the enclosing volume. For right angle corners, $\eta = 1/8$ and $7/8$ for inside and outside corners, respectively. Turning

our attention now to the two remaining integrals in Eq. 8, we see that the limiting process produces no more new terms because the order of the singularity in the integrands is too low in both cases. With these results, Eq. 8 can be specialized to

$$\frac{1}{2} \psi(\bar{x}') = \int_V F(\bar{x}) \phi(\bar{x}, \bar{x}') d^3x + \int_S \{ \phi(\bar{x}, \bar{x}') \nabla \psi(\bar{x}) - \psi(\bar{x}) \nabla \phi(\bar{x}, \bar{x}') \} \cdot \hat{n} d\alpha \quad (9)$$

where $\bar{x}'$ must now belong only to the smooth subset of points making up S .

It was mentioned earlier that the electric and magnetic field intensities satisfy vector wave equations. This is simple to show and, in doing so, an important assumption is made obvious. The harmonic, $e^{j\omega t}$, form for Maxwell's equations is

$$\nabla \times \bar{E} = -j\omega\mu\bar{H} \quad (10)$$

$$\nabla \times \bar{H} = \bar{J} + j\omega\epsilon\bar{E} \quad (11)$$

For μ and ϵ constant

$$\nabla \cdot (\epsilon\bar{E}) = \rho \quad \nabla \cdot \bar{E} = \rho/\epsilon \quad (12)$$

$$\nabla \cdot (\mu\bar{H}) = 0 \quad \nabla \cdot \bar{H} = 0 \quad (13)$$

Take the curl of Eq. 11 and assume ϵ constant,

$$\nabla \times \nabla \times \bar{H} = \nabla \times \bar{J} + j\omega\epsilon \nabla \times \bar{E}$$

Substitute for $\nabla \times \bar{E}$ from Eq. 10,

$$\nabla \times \nabla \times \bar{H} - \omega^2 \mu \epsilon \bar{H} = \nabla \times \bar{J}$$

Use the vector identity,

$$\nabla \times \nabla \times \bar{H} = \nabla(\nabla \cdot \bar{H}) - \nabla^2 \bar{H}$$

to obtain

$$\nabla(\nabla \cdot \bar{H}) - \nabla^2 \bar{H} - k^2 \bar{H} = \nabla \times \bar{J}$$

where $k^2 \equiv \omega^2 \mu \epsilon$. Now assume constant μ and use Eq. 13 to get

$$\nabla^2 \bar{H} + k^2 \bar{H} = -\nabla \times \bar{J} \quad (14)$$

The parallel result for the electric field is

$$\nabla^2 \bar{E} + k^2 \bar{E} = j\omega\mu \bar{J} + \frac{1}{\epsilon} \nabla \rho \quad (15)$$

Equations 14 and 15 are the inhomogeneous, vector, wave equations for the electric and magnetic field intensities, and they are valid at all points where μ and ϵ are constant, i.e., homogeneous and isotropic regions.

All that remains to show now is how the result for a scalar wave function, Eq. 9, can be applied to the vector wave Eqs. 14 and 15. To do so depends only on the fact that in a rectangular coordinate system, the Laplacian operator may be written

$$\nabla^2 \bar{H} = \hat{e}_1 \nabla^2 H_1 + \hat{e}_2 \nabla^2 H_2 + \hat{e}_3 \nabla^2 H_3$$

where $\hat{e}_1$, $\hat{e}_2$, and $\hat{e}_3$ form a right-handed set of orthogonal unit vectors in a rectangular system. In component form, we may then write Eq. 14 as

$$\nabla^2 H_i + k^2 H_i = - [\nabla \times \bar{J}]_i$$

where $i = 1, 2$, and 3 . This is a scalar wave equation to which we can immediately apply Eq. 9 to yield

$$\begin{aligned} \frac{1}{2} H_i(\bar{x}') &= \int_V [\nabla \times \bar{J}(\bar{x})]_i \phi(\bar{x}, \bar{x}') d^3x \\ &+ \oint_S \left\{ \phi(\bar{x}, \bar{x}') \frac{\partial}{\partial n} H_i(\bar{x}) - H_i(\bar{x}) \frac{\partial}{\partial n} \phi(\bar{x}, \bar{x}') \right\} da \end{aligned}$$

Recombining the component parts of this equation, it is not difficult to show that, in general,

$$\begin{aligned} \frac{1}{2} \bar{H}(\bar{x}') &= \int_V [\nabla \times \bar{J}(\bar{x})] \phi(\bar{x}, \bar{x}') d^3x \\ &+ \oint_S \{ \phi(\bar{x}, \bar{x}') [\hat{n} \cdot \nabla] \bar{H}(\bar{x}) - \bar{H}(\bar{x}) [\hat{n} \cdot \nabla] \phi(\bar{x}, \bar{x}') \} da \quad (16) \end{aligned}$$

Equation 16 is a coordinate free result despite recourse to a rectangular system for its derivation, it can be applied in any coordinate system. The parallel result to Eq. 16 for the electric field can be written down by inspection.

Up until this point, the source terms, $\bar{J}$ and ρ , in Maxwell's equations have been retained throughout. There are two reasons for this, one is simply that the added generality required very little additional work and, secondly, because the equations may be useful for examining cavity or circuit interaction with an electron beam. With regard to this second reason, great care must be exercised in the application of an equation such as Eq. 16 because, in high space charge regions, the effective permittivity may vary with position and direction; i.e., plasma permittivities are usually tensor quantities. So, except in low space charge devices, the equations of this section may not be applicable. From here on, it will be assumed that there are no source terms in V.

The only thing remaining to this section is to show how the integrand of Eq. 16 simplifies when all or part of the bounding surface, S, is perfectly conducting. At the surface of a perfect conductor, we know that the tangential components of the electric field and the normal component of the magnetic field are zero; $\bar{E}_t = 0$, $H_n = 0$. So, using Eq. 11 with $\bar{J} = 0$,

$$\nabla \times \bar{H}_t = j\omega\epsilon \hat{n}$$

From this, it follows that the normal derivative of the tangential components of $\bar{H}$ are 0; $\frac{\partial}{\partial n} \bar{H}_t = 0$. Let that part of the surface that is perfectly conducting be represented by S_c and represent the remaining part of the enclosing surface by S_f , so that symbolically $S = S_c + S_f$. With this understanding, Eq. 16 may be written as

$$\frac{1}{2} \bar{H} = \int_{S_c} \left\{ \phi \left(\frac{\partial}{\partial n} H_n \right) \bar{n} - \bar{H}_t \left(\frac{\partial \phi}{\partial n} \right) \right\} da + \int_{S_f} \left\{ \phi \left(\frac{\partial}{\partial n} \bar{H} \right) - \bar{H} \left(\frac{\partial \phi}{\partial n} \right) \right\} da \quad (17)$$

where now the volume integration has been dropped and the explicit evaluation points are implied by reference back to Eq. 16. Note that the integral over S_c contains only three unknown functions, the two transverse components of H and the normal derivative of the normal component of H , as opposed to what appears to be six unknown functions on the general boundary S_f .

In the next section, some approximations will be made regarding the nature of the unknown functions on both boundary types, and the number of algebraic equations required to find these functions will be made clear.

III. APPLICATION OF THE METHOD OF MOMENTS

The intent of this section is to demonstrate a method for extracting useful information from the vector surface integral equation of Section II, Eq. 17. The approach outlined here is just one very simple application of the general technique called the method of moments, or MoM.⁹

First, the field boundaries are considered to be cross sections of regular uniform waveguides for which the normal mode functions are known, and the surface connecting the uniform waveguides, the junction surface, is assumed perfectly conducting in keeping with Section II. Pulse basis sets are used to expand the unknown H field on the conducting boundaries, and normal mode expansions are used on the field boundaries in the regular waveguides. The system is excited, or driven, by a pure mode incident on the junction from any one of the ports. An over-determined linear system of equations is obtained by delta testing, point matching, on both the conducting and field boundaries. The unknown set contains the scattered mode amplitudes at all the ports along with the field amplitudes for the pulse basis expansions on the conducting boundaries.

Figure 2 illustrates a general N port junction and indicates how the discontinuity volume is defined by placement of planar field boundaries perpendicular to the axis of each regular waveguide entering the junction. These field boundaries can be moved far enough away from the junction and into the waveguide so that only propagating modes need to be considered. Alternatively, they can be moved as close to the

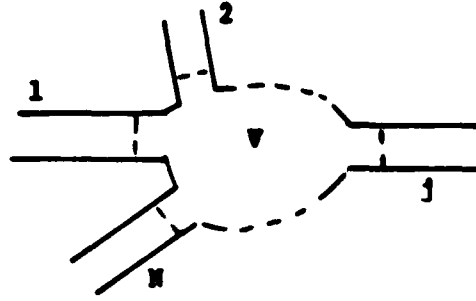


Fig. 2. Schematic diagram of an N port junction indicating placement of field boundaries defining the discontinuity volume V.

junction as possible, while still remaining inside the waveguide, by including an appropriately large set of evanescent modes along with the propagating modes on the field plane. This minimizes the discontinuity volume.

Explicitly, formal mode expansions can be used to express the field at boundary j,

$$\bar{H}_j = \delta_{ij} \hat{h}_{[j,\xi]} + \sum_{[j,n]} a_{[j,n]} \hat{h}_{[j,n]} \quad (18)$$

where δ_{ij} is the Kronecker delta function and is used to indicate that the incident mode is in the i th waveguide. Other definitions are:

- i, j Indices over field boundaries.
- n, ξ Integer indices used for ordering and counting modes.
Each index corresponds to a unique mode at each field boundary. Mode ξ at field plane 1 is the incident or driving mode.

- $\hat{h}_{[1,n]}$ Normalized mode function at field boundary 1 and for mode n . The back arrow is used to indicate that this mode is propagating in the $-\hat{n}$ direction. This function has the units of magnetic field intensity.
- $\hat{h}_{[1,n]}$ Same as above, except the forward arrow indicates propagation in the $+\hat{n}$ direction. This function is used for all reflected and transmitted modes.
- $a_{[1,n]}$ Amplitude of mode n at field plane 1. Carries no units, but the power carried by the associated mode is proportional to aa^* .

The distinction between mode functions propagating along $+$ or $-\hat{n}$ directions in the regular waveguides is necessary due to a subtle difference between them as a result of assumed propagation direction. This distinction is important; it means the difference between success and failure. Appendix A clarifies this distinction and also gives details on the normalization of the mode functions both above and below cutoff.

Equation 17 also requires the normal derivative of h on the field boundaries. To this end,

$$\frac{\partial}{\partial n} \bar{H}_j = -j\delta_{1j} \beta_{[j,\xi]} \hat{h}_{[j,\xi]} - j \sum_{[j,n]} a_{[j,n]} \beta_{[j,n]} \hat{h}_{[j,n]} \quad (19)$$

where $\beta_{[j,n]}$ is the propagation constant for mode n at field plane j and will be purely imaginary below cutoff. The formula for computing the β 's are given in Appendix A.

Now suppose that the conducting surface can be broken down into a set of simple geometric regions. That is, subdivide the conducting surface into rectangles, triangles, sections of cylinders, etc., such that all of the regions combined describe or approximate the conducting surface of interest. The point of this is to use shapes that can be defined on a computer with a minimum of effort. On each region, define a local set of orthogonal curvilinear surface coordinates (p,s) . The local coordinates correspond uniquely to a point in the absolute coordinate system (x,y,z) by the surface mapping M , which depends on the surface shape, orientation, and location. Symbolically, $M(p,s) \rightarrow (x,y,z)$ for all p,s on the defined region; i.e., M takes the surface coordinates (p,s) into the absolute system (x,y,z) . The surface tangent unit vectors should be such that $\hat{p} \times \hat{s} = \hat{n}$, where $\hat{n}$ is the outward unit normal to the enclosed volume V . The unit vectors $\hat{p}$ and $\hat{s}$ are later referred to as the unit primary and unit secondary directions.

The transverse components of H on the conducting surface can now be expanded, approximated, by a pulse basis expansion on each region.

$$\bar{H}_t(a;p,s) = \sum_{[a,kl]} \bar{H}_{t[a,kl]} P_{[a,kl]}(p,s) \quad (20)$$

where the following definitions are used:

- a Integer index used for ordering and counting the simple geometric regions used to build the conducting surface.

k,l Patch position indices used for ordering and counting the patch positions on region a .

p,s Local coordinates on region a .

$\sum_{[a,kl]}$ Sum over patch positions k,l on region a .

$\bar{H}_t[a,kl]$ Constant, transverse magnetic field amplitude at position k,l on region a . Carries units of magnetic field intensity.

$\bar{H}_t(a;p,s)$ Approximation to transverse H field at local coordinate positions p,s on region a .

$P_{[a,kl]}(p,s)$ 1 if p,s is on patch k,l in region a . 0 otherwise.

This last equation defines the pulse function. Note that nothing here requires the patch shape to be rectangular or flat; indeed, the sub-domain patches ordered by indices k,l on region a need not be flat or rectangular. The pulse function is simply a function having value 1 when the local coordinates p,s are inside patch k,l , and 0 otherwise. The normal derivative of the normal component of H can also be expanded on a pulse basis. Let $H'_n \equiv \frac{\partial}{\partial n} H_n$. Then,

$$H'_n(a;p,s) = \sum_{[a,kl]} H'_{n[a,kl]} P_{[a,kl]}(p,s) \quad (21)$$

where all the same notation applies.

While the pulse basis set may be the most flexible and convenient, it is certainly not the most accurate basis expansion available. The

pulse expansions in Eqs. 20 and 21 approximate smooth functions by functions with abrupt jumps at patch boundaries. Clearly, some physics has been lost by this approximation. Reasonably simple subdomain basis sets that are capable of C^0 approximations,¹⁰ smooth approximations, are linear interpolation on a triangular net and bilinear interpolation on a rectangular net. On a triangular patch, the linear interpolating polynomial g has the general form, $g = a + bp + cs$, where p and s are the surface primary and secondary coordinates as before. The three coefficients, a , b , and c , are uniquely related to the values of g at the three vertices of the triangle by the index equation $g_i = a + bp_i + cs_i$; $i = 1, 2, 3$. This is trivial to invert for the polynomial coefficients in terms of the vertex values g_i . The underlying basis function for this interpolant is sometimes called the rooftop function and can be found by setting two of the three values for g_i to 0 and the third value to 1, then solving for a , b , and c . For rectangular patches, a bilinear interpolating polynomial, $g = a + bp + cs + dps$, is used and again the four coefficients are uniquely determined by the four vertex values. The underlying basis for this interpolant is called the pagoda function and can be found as before by setting three of the four vertex values to 0, while setting the fourth equal to 1. These basis functions are almost as flexible as pulse functions. The additional organizational complexity required to implement a scheme using more appropriate basis sets may prove worthwhile by producing more accurate solutions.

At this point, we have approximated all the surface fields using expansion functions. This limits the domain of the function space. Now we have only a finite set of unknown coefficients to look for, not

entire functions. On the field boundaries, the unknowns are the mode amplitudes, and on the conducting boundaries, the unknowns are the pulse function amplitudes. The next job is to relate these unknowns through the physics of the situation.

Clearly, the surface integral equation from Section II, Eq. 17, provides the relationship we need between the surface fields. Using the field expansions, Eqs. 18-19 and 20-21, in the right-hand side of Eq. 17 yields a formidable looking result

$$\begin{aligned}
 \frac{1}{2} \bar{H} = & \sum_a \sum_{[a,kl]} \left\{ H_n'_{[a,kl]} \int_{[a,kl]} \hat{n} \, d\alpha \right. \\
 & - H_p_{[a,kl]} \int_{[a,kl]} \hat{p} (\nabla \phi \cdot \hat{n}) \, d\alpha \\
 & \left. - H_s_{[a,kl]} \int_{[a,kl]} \hat{s} (\nabla \phi \cdot \hat{n}) \, d\alpha \right\} \\
 & + \int_{S_1} \left\{ j\beta_{[1,\xi]} \phi - \nabla \phi \cdot \hat{n} \right\} \hat{h}_{[1,\xi]} \, d\alpha \\
 & - \sum_j \sum_{[j,n]} a_{[j,n]} \int_{S_j} \left\{ j\beta_{[j,n]} \phi + \nabla \phi \cdot \hat{n} \right\} \hat{h}_{[j,n]} \, d\alpha \quad (22)
 \end{aligned}$$

Where the transverse H field has been decomposed into its primary and secondary surface components,

$$\bar{H}_t_{[a,kl]} = \hat{p} H_p_{[a,kl]} + \hat{s} H_s_{[a,kl]}$$

Note that the integrands contain only explicitly known functions; the free space Green's function is given by Eq. 6 and its gradient by Eq. 7.

We can now generate a systems of linear equations by systematically enforcing Eq. 22 for a discrete number of points in the range of $\bar{x}'$ on S. This is called delta testing or point matching. So, test Eq. 22 at the center of every patch on each region, points [c,mn], and use expansion Eq. 20 on the left-hand side to obtain three scalar equations, one for each of the three vector components. They are:

The primary equation,

$$\begin{aligned} \hat{p}_{[c,mn]} \cdot \bar{B}_{[i,\xi]}^{[c,mn]} = & \sum_j \sum_{[j,n]} a_{[j,n]} \hat{p}_{[c,mn]} \cdot \bar{F}_{[j,n]}^{[c,mn]} \\ & + \sum_a \sum_{[a,kl]} \left\{ - H'_{n[a,kl]} \hat{p}_{[c,mn]} \cdot \bar{G}_{[a,kl]}^{[c,mn]} \right. \\ & + H_{p[a,kl]} \left[\hat{p}_{[c,mn]} \cdot \bar{g}_{p[a,kl]}^{[c,mn]} + \frac{1}{2} \delta_{[z,kl]}^{[c,mn]} \right] \\ & \left. + H_{s[a,kl]} \hat{p}_{[c,mn]} \cdot \bar{g}_{s[a,kl]}^{[c,mn]} \right\} \quad (23) \end{aligned}$$

Secondary equation,

$$\begin{aligned} \hat{s}_{[c,mn]} \cdot \bar{B}_{[i,\xi]}^{[c,mn]} = & \sum_j \sum_{[j,n]} a_{[j,n]} \hat{s}_{[c,mn]} \cdot \bar{F}_{[j,n]}^{[c,mn]} \\ & + \sum_a \sum_{[a,kl]} \left\{ - H'_{n[a,kl]} \hat{s}_{[c,mn]} \cdot \bar{G}_{[a,kl]}^{[c,mn]} \right. \\ & + H_{p[a,kl]} \hat{s}_{[c,mn]} \cdot \bar{g}_{p[a,kl]}^{[c,mn]} \\ & \left. + H_{s[a,kl]} \left[\hat{s}_{[c,mn]} \cdot \bar{g}_{s[a,kl]}^{[c,mn]} + \frac{1}{2} \delta_{[a,kl]}^{[c,mn]} \right] \right\} \quad (24) \end{aligned}$$

Normal equation,

$$\begin{aligned}
 \hat{n}_{[c,mn]} \cdot \bar{B}_{[i,\xi]}^{[c,mn]} = & \sum_j \sum_{[j,n]} a_{[j,n]} \hat{n}_{[c,mn]} \cdot \bar{F}_{[j,n]}^{[c,mn]} \\
 & + \sum_a \sum_{[a,kl]} \left\{ - H'_{n[a,kl]} \hat{n}_{[c,mn]} \cdot \bar{G}_{[a,kl]}^{[c,mn]} \right. \\
 & + H_{p[a,kl]} \hat{n}_{[c,mn]} \cdot \bar{g}_{p[a,kl]}^{[c,mn]} \\
 & \left. + H_{s[a,kl]} \hat{n}_{[c,mn]} \cdot \bar{g}_{s[a,kl]}^{[c,mn]} \right\} \quad (25)
 \end{aligned}$$

In Eqs. 23, 24, and 25, the following definitions have been used:

- $[c,mn]$ Used to indicate that the test point is at the center of patch mn on region c.
- $[a,kl]$ Evaluation region, indicates that the integration variable $\bar{x}$ runs over patch kl on region a.
- $\delta_{[a,kl]}^{[c,mn]}$ Kronecker delta, has value 1 when $a = c$ and $m = k, n = 1$; 0 otherwise.
- S_j Indicates that the surface integration runs over field boundary j.

$$\bar{B}_{[i,\xi]}^{[c,mn]} = \int_{S_i} \left\{ j\beta_{[i,\xi]} \phi - \nabla \phi \cdot \hat{n} \right\} \hat{h}_{[i,\xi]}^+ da$$

$$\bar{F}_{[j,\eta]}^{[c,mn]} = \int_{S_j} \left\{ j\beta_{[j,\eta]} \phi + \nabla \phi \cdot \hat{n} \right\} \hat{h}_{[j,\eta]}^+ da$$

$$\bar{G}_{[a,kl]}^{[c,mn]} = \int_{[a,kl]} \phi \hat{n} da$$

$$\bar{g}_p^{[c,mn]}_{[a,kl]} = \int_{[a,kl]} \hat{p}(\nabla \phi \cdot \hat{n}) da$$

$$\bar{g}_s^{[c,mn]}_{[a,kl]} = \int_{[a,kl]} \hat{s}(\nabla \phi \cdot \hat{n}) da$$

We now have as many equations as there are surface patch unknowns, and require at least as many more equations as there are modes on each port. Again, this can be done using point matching; only this time, the matching is done on the field boundaries. Use Eq. 18 on the left-hand side of Eq. 22 and test at point mn on the surface of field boundary c, point [c,mn], where c is used here as an index over ports.

$$\begin{aligned} \bar{B}_{[i,\xi]}^{[c,mn]} - \frac{1}{2} \delta_{ic} \hat{h}_{[c,\xi]}^{+[c,mn]} &= \sum_a \sum_{[a,kl]} \left\{ -H'_{n[a,kl]} \bar{G}_{[a,kl]}^{[c,mn]} \right. \\ &\quad + H_{p[a,kl]} \bar{g}_p^{[c,mn]}_{[a,kl]} \\ &\quad \left. + H_{s[a,kl]} \bar{g}_s^{[c,mn]}_{[a,kl]} \right\} \\ &\quad + \sum_j \sum_{[j,\eta]} a_{[j,\eta]} \left\{ \bar{F}_{[j,\eta]}^{[c,mn]} + \frac{1}{2} \delta_{cj} \hat{h}_{[c,\eta]}^{+[c,mn]} \right\} \quad (26) \end{aligned}$$

The testing points $[c, mn]$ in Eq. 26 should be fairly uniformly distributed on each field boundary to optimize the condition number of the resulting system of equations. If you have K modes on each port, the number of test points L on each port required to determine a solution is given by $L > K/3$; L must be an integer. The factor of 3 results from the fact that Eq. 26 is a vector equation, so three scalar equations are generated for each test point.

The equations that result from testing on the conducting boundaries, Eqs. 23, 24, and 25, and those due to testing on the field boundaries, Eq. 26, form an overdetermined linear system of equations and will later be referred to, respectively, as type A and type B equations. If no approximations had been made obtaining these equations, they would be perfectly consistent and a unique solution satisfying all the equations would be possible. But the pulse basis expansions are not smooth and, hence, cannot perfectly mimic the physically exact solution. As a result, our system of equations is inconsistent. In a strict mathematical sense, the system of equations developed here has no solution; that is, there exists no solution vector that can perfectly satisfy every equation in the system. However, from a practical point of view, the system has minimum residual error solutions where the residual error is measured by some suitable vector norm.

Classically, a least-squares residual error solution can be obtained by premultiplying the overdetermined system, $Ax = b$, by the conjugate transpose of A , A^+ , and solving the resulting square system of equations using a standard method. This results however in squaring the condition number of the original system of equations and should be

avoided, since moment-method matrices are notoriously ill-conditioned in the first place. Methods for finding the pseudo-inverse of A that are based on finding its singular value decomposition, which is nondependent on premultiplication by A^+ , and are available as library subroutines which should work. But they run at substantial overhead cost and require a significant amount of additional storage.

For the results in Section 3, a variant on the iterative algorithm called ART was employed. The synonym ART stands for "algebraic reconstruction technique" and was first used extensively on early X-ray CT scanners; it was the algebraic tool used to reconstruct cross-section profiles from X-ray attenuation data, and hence its name. This algorithm and some variants thereon are explained in detail in Appendix B. Briefly, it is a row by row steepest descent algorithm that can be relaxed to give minimum norm error solutions. Being a "row action" method, ART can be implemented with just one row of matrix elements in the computer core at a time, keeping all other rows on disc space. Because disc access time is slow, it is best to read in as many coefficient rows as possible, so that the number of disc access requests is minimized. Although ART exhibits only a linear rate of convergence, guarantee of convergence does not depend on degree of diagonal dominance as with, for example, the SOR algorithm;¹¹ indeed, ART converges almost unconditionally.

If the system of equations developed in this section were perfectly consistent and very well conditioned, there would be no need to look for any more help in finding the solution because all the physics would be represented and mimicked by the equation system. But, as was

noted earlier, some physics has been lost. To some extent, we can reintroduce the physics by making our approximate solution have at least some property in common with the exact physical solution. This takes the form of enforcing a conservation of power constraint on the solution. Mathematically we require that all the propagating power, real power flow, in the reflected and transmitted modes must be the same as the power carried by the incident mode. Since the normal mode functions in Eq. 18 have all been normalized to carry the same amount of power, we can write,

$$1 = \sum_j \sum_{[\beta]} a_{[j,\beta]}^{(k)} a_{[j,\beta]}^{(k)*} \quad (27)$$

where β is the summing index over only the propagating modes on port j . This constraint was enforced at the end of each ART update of the entire system by a constant phase adjustment. The update takes the following form: Let $a_{[i,\eta]}^{(k)}$ be the predicted mode amplitudes at the end of the k th ART pass, then adjust the mode amplitudes to $a_{[i,\eta]}^{(k+)}$ before starting the next ART pass by using

$$a_{[i,\eta]}^{(k+)} = \frac{a_{[i,\eta]}^{(k)}}{\left\{ \sum_j \sum_{[\beta]} a_{[j,\beta]}^{(k)} a_{[j,\beta]}^{(k)*} \right\}^{1/2}} \quad (28)$$

Doing this, the values of $a_{[i,\eta]}^{(k+)}$ always satisfy Eq. 27 without altering the phase of the initial values. Enforcing this constraint also tends to accelerate the overall convergence rate for the algorithm by reducing the solution space to only that class of coefficients that satisfy Eq.

27. Other constraints¹² may also be used; if, for example, you have symmetric ports, then constraints on the solution based on that symmetry should be used.

As an aside, Spielman¹³ used a formulation similar to the one described here (but in two dimensions) to find the cutoff resonances of arbitrary cross-section waveguides. When there are no ports, Eqs. 23, 24, and 25 reduce to a square matrix system of the form $Ax = 0$; this is an eigenvalue problem for resonances in a three-dimensional cavity. Spielman found the eigen or resonance values by searching for frequencies where the magnitude of the determinant of A has minimums. Computing a matrix determinant is an N cubed operation, where N is the edge dimension of the matrix. Embedding an N cubed operation into a search algorithm means slow business. This is also a nonlinear eigenvalue problem. Since all the matrix elements vary in a complex way with frequency, it cost N squared numerical integrations over surface patches to update A to a new frequency. Spielman gets away with all this because his matrix size is small, only 21 by 21 elements. It seems unlikely that a practical algorithm for finding eigen frequencies in three-dimensional cavities, based on this approach, would be possible.

IV. RESULTS

This section gives numerical results for a few simple test cases based on Eqs. 23, 24, 25, and 26 in Section III. The job was broken down into three sections. A Fortran code was written to implement each. The first code allows one to build a data file specifying arbitrary rectangles in three dimensions. Discontinuities that could be built using rectangles were the only type considered. Then, for each wavelength, a second code computed on the order of N squared coupling coefficients, where N is the total number of unknowns. The last code "solved" the overdetermined system of equations using a variant on the ART algorithm. These Fortran programs are documented in Appendix C. They were all developed on the Gould 9080 computer system at the University of Utah College of Engineering.

Writing a code capable of specifying more arbitrary shapes is certainly possible but, from an implementation point of view, this job of specifying and organizing the surface geometry is the most difficult. Once the surface shapes are specified and ordered by some accounting system, the calculus follows mechanically from the equations of Section III. The purpose here was only to find out if, and then how well, a formulation of this type could be expected to work. So while a code limited to only discontinuities that could be built from rectangles is of no practical value, it was entirely adequate for answering many basic questions concerning the performance of the algorithm.

Data are presented for three test cases. The first two are very simple, a straight section of uniform waveguide as a two-port junction

and a straight section of uniform waveguide with a perfect short on one side. These test cases, although trivial, are important because exact analytic results are available for comparison. They were used initially to debug the Fortran codes, and later to help answer such questions as how many surface patches are needed along the length and across the width of the waveguide at a given operating wavelength in order to retain an accurate solution without using an excessive number of patches. For the straight section of uniform waveguide, results are given with and without the conservation of power constraint enforced.

An asymmetric iris in a straight section of uniform waveguide was considered as the third test case. These results were compared with analytic results found using approximate susceptance data available in the Microwave Engineers' Handbook,¹⁴ Vol. 1, p. 81. For the reflected phase and amplitude, measured data were also obtained using a slotted line for further comparison. Data are also given in this case showing the equation consistency (average residual error per equation) as a function of wavelength for the solutions presented.

Case 1. Straight Shot Data

In this case, the TE_{10} mode in a rectangular waveguide is numerically "propagated" for a distance of one tenth of the free space cutoff wavelength, λ_c , along the guide axis. In Fig. 3, the cross-section dimensions were $a = 0.5 \lambda_c$ and $b = 0.2 \lambda_c$, or equivalently by 0.5 by 0.2 cutoff units.

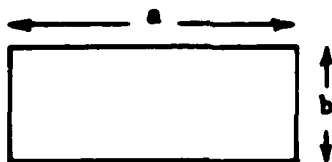


Fig. 3. Waveguide cross section for straight shot data.

The surface patches were such that we had 12 subdivisions along a , 2 along b , and 2 more along the axis. Each field boundary was tested at 24 uniformly distributed points. This resulted in 168 surface patch unknowns and 2 field boundary unknowns. Since the field boundaries were tested so many times, we had 312 complex equations.

Two sets of results are given, Figs. 4 and 5 represent the solution obtained without using the conservation of power constraint, and Figs. 6 and 7 show the solution with the constraint enforced, as outlined in Section III. On all the plots, the abscissa gives the operating wavelength in cutoff wavelength units. The upper curve in Figs. 4 and 6 represents the computed transmission amplitudes, while the lower curves on these plots are the computed reflection amplitudes. Clearly, the exact results should be unity transmission and zero reflection amplitudes. Figures 5 and 7 are the transmission phase with respect to the input plane for the unconstrained and constrained cases, respectively. On these plots, the marked curves are from the computed data, while the solid unmarked curves represent the exact phase given by

$$\phi = -36^\circ \frac{\sqrt{1 - R^2}}{R}$$

where the operating wavelength is given by $\lambda = R \lambda_c$.

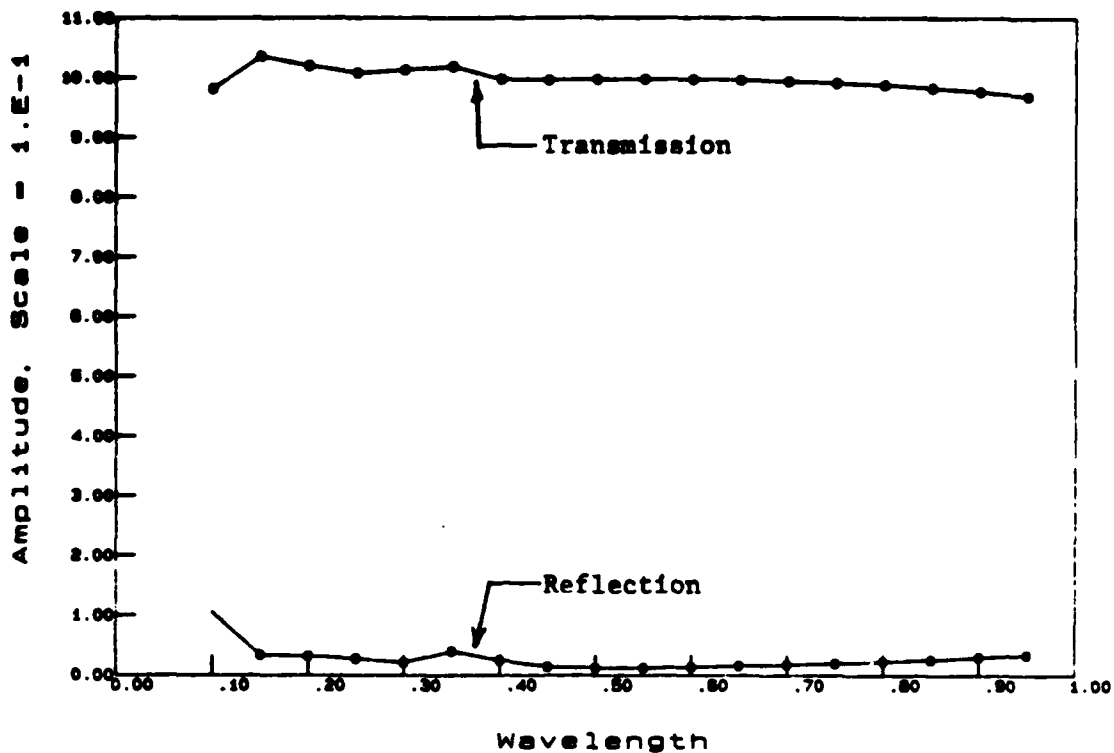


Fig. 4. Computed transmission and reflection amplitudes for the straight shot case without the conservation of power constraint enforced.

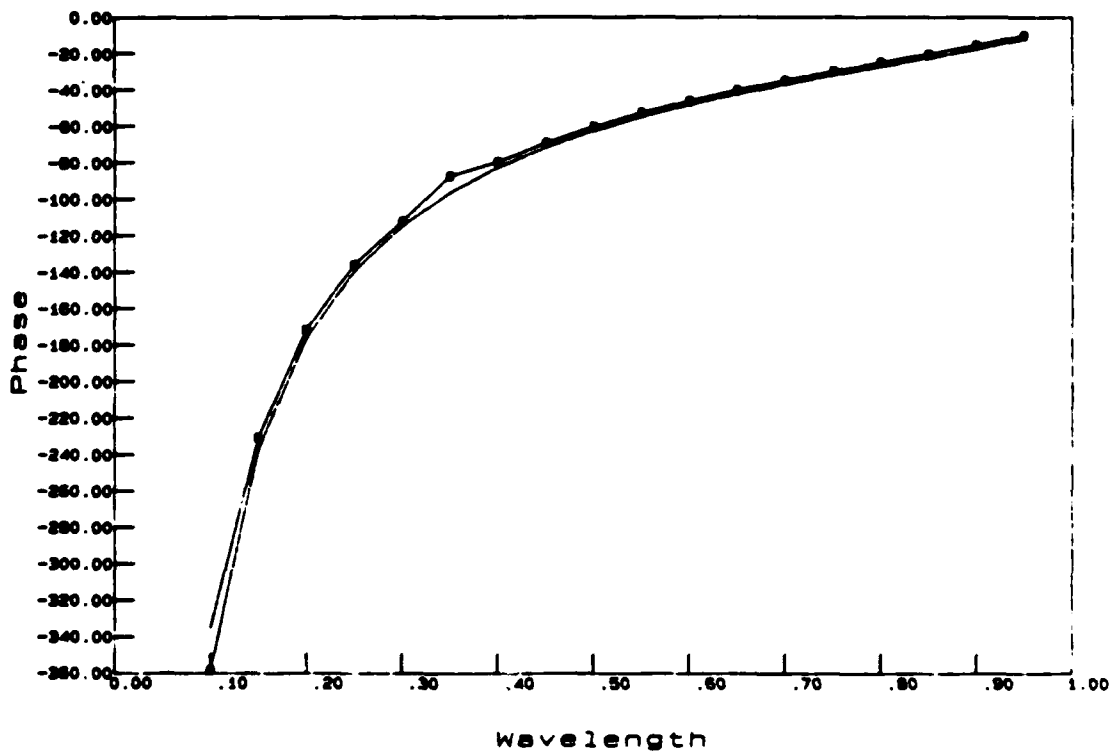


Fig. 5. Computed versus theoretically exact phase for the transmission data in Fig. 4.

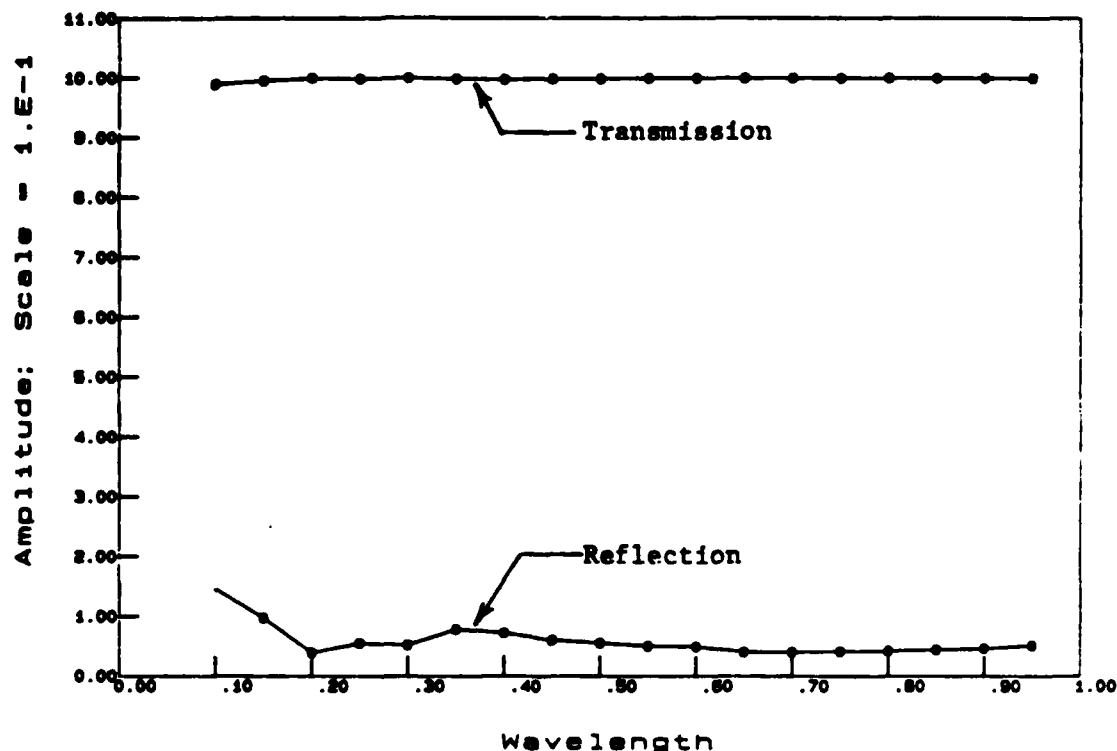


Fig. 6. Computed transmission and reflection amplitudes for the straight shot case with the conservation of power constraint enforced.

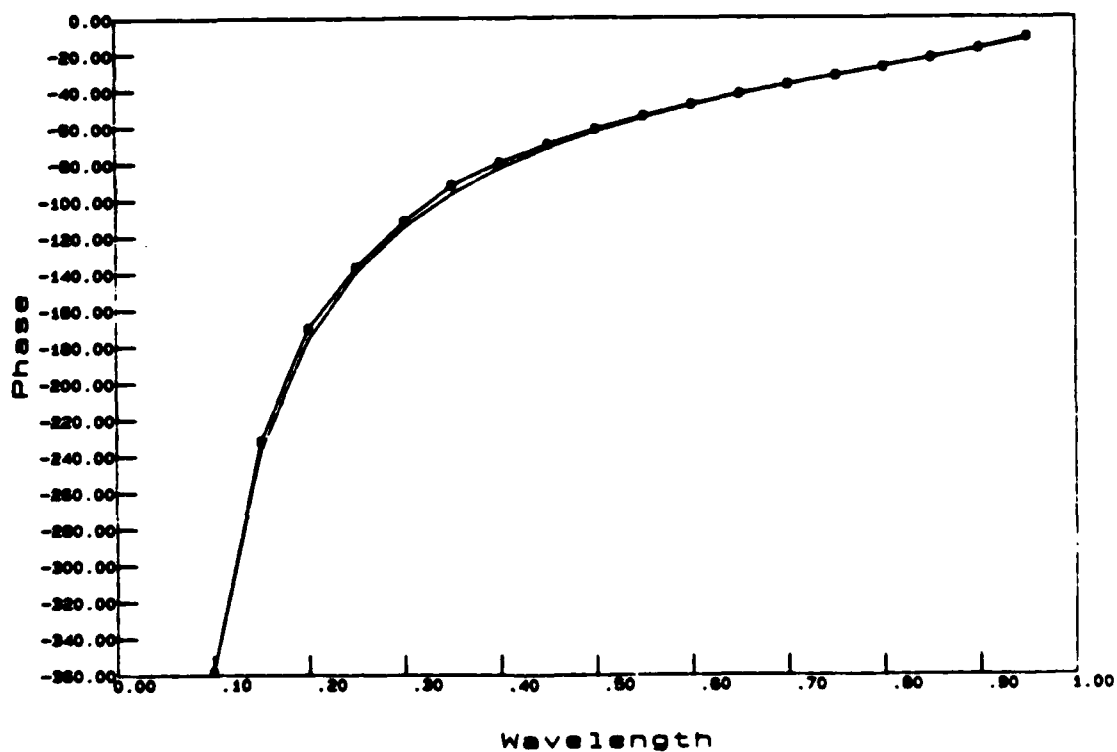


Fig. 7. Computed versus theoretically exact phase for the transmission data in Fig. 6.

From these data, it can be seen that the conservation constraint made very little difference in the solution. The transmission phase and amplitude are slightly more accurate with the constraint enforced, but the reflection amplitude was a bit more accurate without the constraint. Note, the overall accuracy in the unconstrained solution indicates that the system of equations used has retained most of the physics despite the expansion function approximations. One of the most remarkable aspects of this solution was the ability of the numerical algorithm to accurately track the phase all the way from the low frequency end, where the guide wavelength is much longer than the free space wavelength, up to the high frequency end, where the guide wavelength and the free space wavelength are not much different.

Case 2. Straight Shot with Perfect Short

For this case, the TE_{10} mode in a rectangular waveguide is numerically reflected off a perfect shoring plane $0.1 \lambda_c$ away from the reference field plane. The guide cross section was the same as for Case 1. Again, 12 subdivisions were used along a , 2 along b , and 2 along the axis. The shorting rectangle was subdivided into 3 by 12 patches along its short and long axes, respectively. The field boundary was tested at 36 uniformly distributed points. This results in 276 surface patch unknowns and 1 field boundary unknown, for a total of 277 complex unknowns. The number of complex equations as a result of testing is 384.

Again, two sets of results are given. Figures 8 and 9 give the amplitude and phase for the unconstrained solution, and Fig. 10 gives the phase for the constrained solution. In the case of the constrained solution, the amplitude, as a result of the constraint, is exactly unity, and so was not plotted. Both phase plots indicate the numerically predicted solutions with marks; the unmarked curves are the exact phase results which are given by

$$\phi = -180^\circ - 72^\circ \frac{\sqrt{1 - R^2}}{R}$$

where, as before, R is the abscissa value on these plots. The abscissa value is related to the physical wavelength by $\lambda = R \lambda_c$.

While the phase tracking is not bad for the unconstrained solution, the reflection amplitude (Fig. 8) deviates away from unity by as much as 10 percent at the short wavelength end, $R = 0.4$. In the region where the data overlap the phase accuracy of the constrained solution is a bit better than for the unconstrained solution, and is not bad all the way down to $R = 0.2$. Of course, as was previously mentioned, the reflected amplitude is perfectly unity for the constrained solution. Although the unconstrained solution mimics the exact solution rather well, the value of the constraint condition as a means of improving the solution is much clearer here than in Case 1.

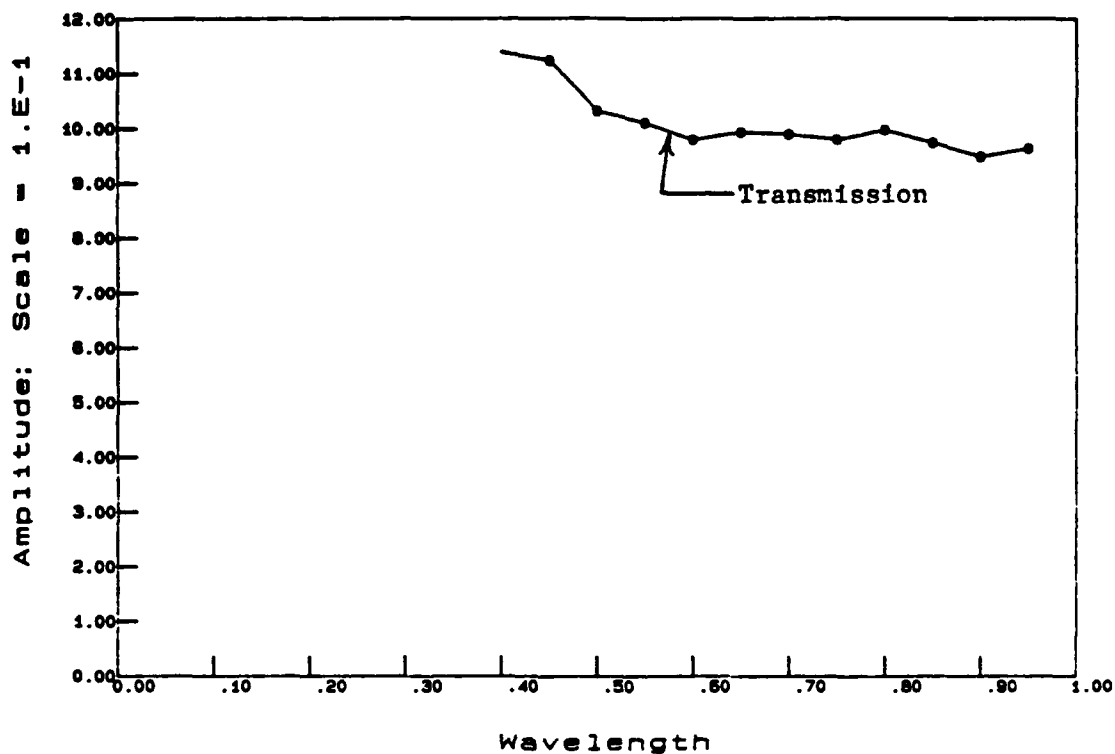


Fig. 8. Reflection amplitude for the straight shot case with a perfect shorting plane on one side and also without enforcing the conservation of power constraint.

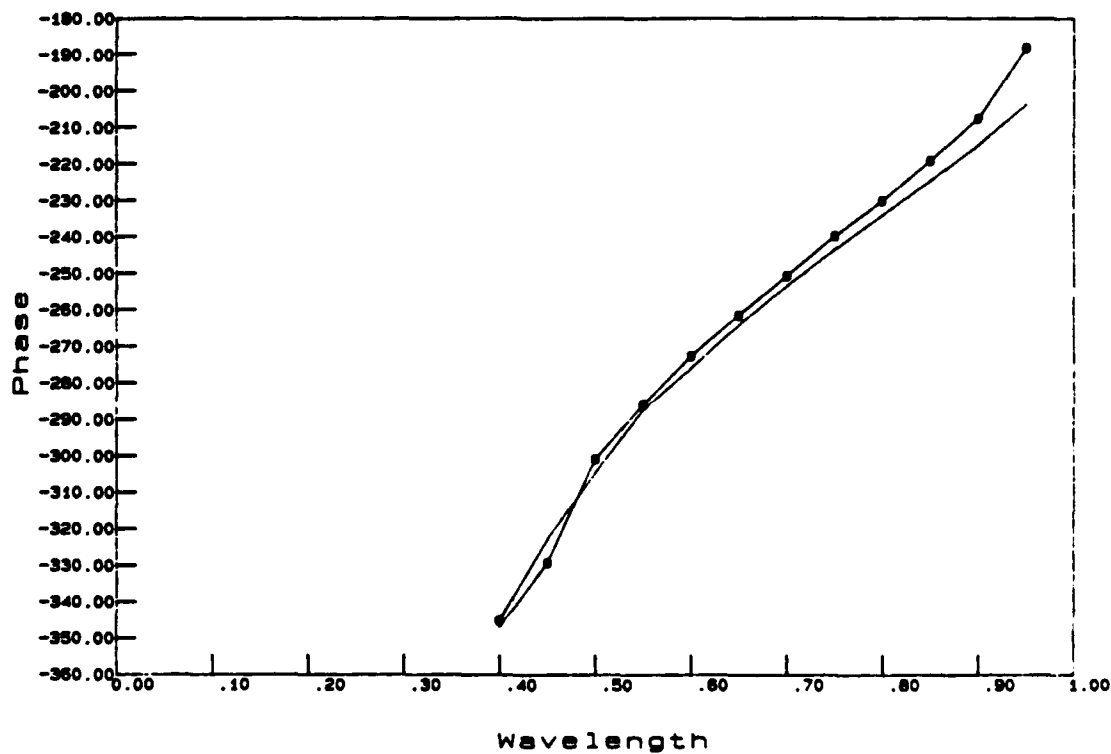


Fig. 9. Computed versus theoretically exact phase for the reflection data in Fig. 8.

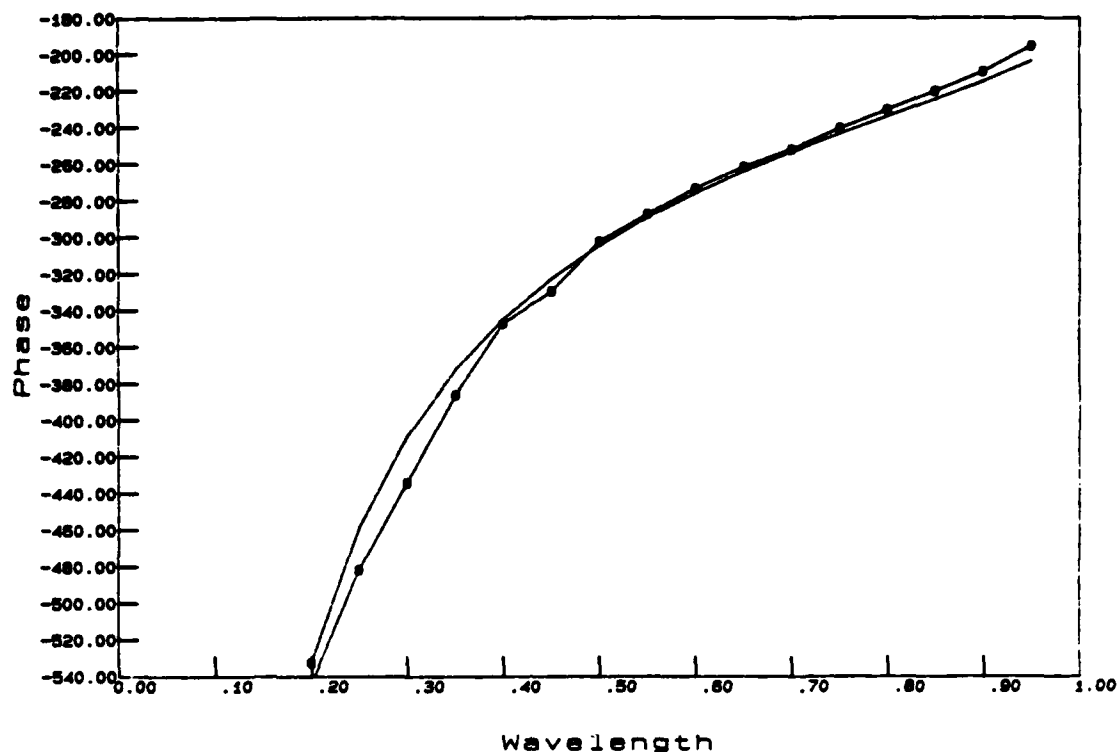


Fig. 10. Computed versus theoretically exact phase for the straight shot with shorting plane case with the conservation of power constraint enforced.

Case 3. Straight Shot with Asymmetric Iris

In this case, an asymmetric iris is placed $0.1 \lambda_c$ away from the input port, and the output port is $0.1 \lambda_c$ on the other side of the iris. The iris is assumed to be perfectly conducting and to have zero thickness (see Fig. 11). The waveguide cross section is the same as for Case 1. This geometry was subdivided using 16 divisions along a , 2 along b , and 4 along the axis from port to port. The iris was subdivided using 3 by 4 patches for $\delta/a = 0.75$, and 3 by 6 patches for $\delta/a = 0.625$. Note that there are two iris rectangles, one for each side.

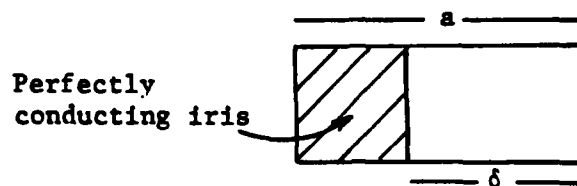


Fig. 11. Geometry for asymmetric iris.

The dominant TE_{10} mode was used to drive the system, while the first 8 TE_{M0} modes were considered in the field expansions at each field plane or port. Each field boundary was tested at 32 uniformly distributed points. Taken all together, this resulted in 504 surface patch unknowns and 16 field boundary unknowns, for a total of 520 complex unknowns. The number of complex equations generated by testing was 696. This is for the case $\delta/a = 0.75$. In the case $\delta/a = 0.625$, we had 556 complex unknowns and 732 complex equations. Computation times were on the order of 25 minutes per wavelength to generate the coupling coefficients, and 10 minutes per wavelength to solve the resulting system of equations. These times are for the Gould 9080 computer system when there are no other users on board.

The conservation of power constraint was enforced for all of the computed curves for Case 3. The numerically computed solutions are marked by "o" for TE_{10} results, and by "*" for TE_{20} results. Reflection phase measurements were made using a slotted line, and these results are denoted using an "I" symbol on the reflection plots. The solid unmarked curves are results predicted using published susceptance data, they are for TE_{10} comparison.

Maracuvitz¹⁵ indicates that susceptance data of the type used here have been derived by the "equivalent static method," which uses the static aperture field for the two lowest modes. In terms of these data, the complex reflection amplitude is given by

$$a^-(0) = \frac{-1 e^{-2j\beta L}}{2(Y_0/B) - j}$$

and the complex transmission amplitude by

$$a^+(2L) = \frac{2(Y_0/B) e^{-2j\beta L}}{2(Y_0/B) - j}$$

where the ratio Y_0/B is found on page 81 of the Microwave Engineers' Handbook,¹⁴ and $\beta = 2\pi/\lambda_g$. For all the data in this section, $L = 0.1 \lambda_c$. In the limiting case of a perfect short, Y_0/B goes to 0, and the equation for the reflection amplitude gives $a^-(0) = e^{-2j\beta L}$, which is the correct amplitude, but the phase is wrong by 180° . One can simply multiply the equation by -1 to correct the deficiency, but rechecking the derivation did not warrant this. The reflection phase curves were all plotted with this 180° discrepancy removed. Shifting these curves 180° brings them into closer agreement with both the measured and numerically predicted phase curves.

Figures 12-15 are the computed solutions for $\delta/a = 0.75$. In this case, agreement between the numerical solution and the solution found using susceptance data is really quite good, especially for the transmission data. Figures 16-19 give solutions for $\delta/a = 0.625$. Here the agreement is less striking. Keep in mind that the susceptance data are

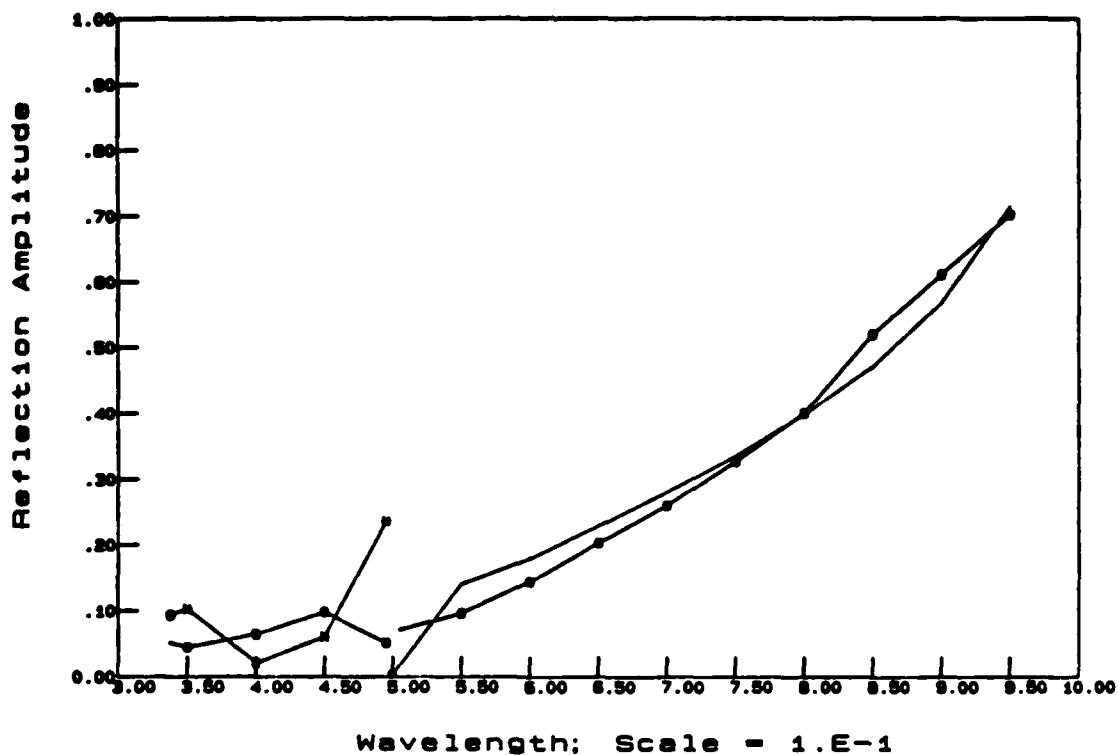


Fig. 12. Reflective amplitude, case $\delta/a = 0.75$.

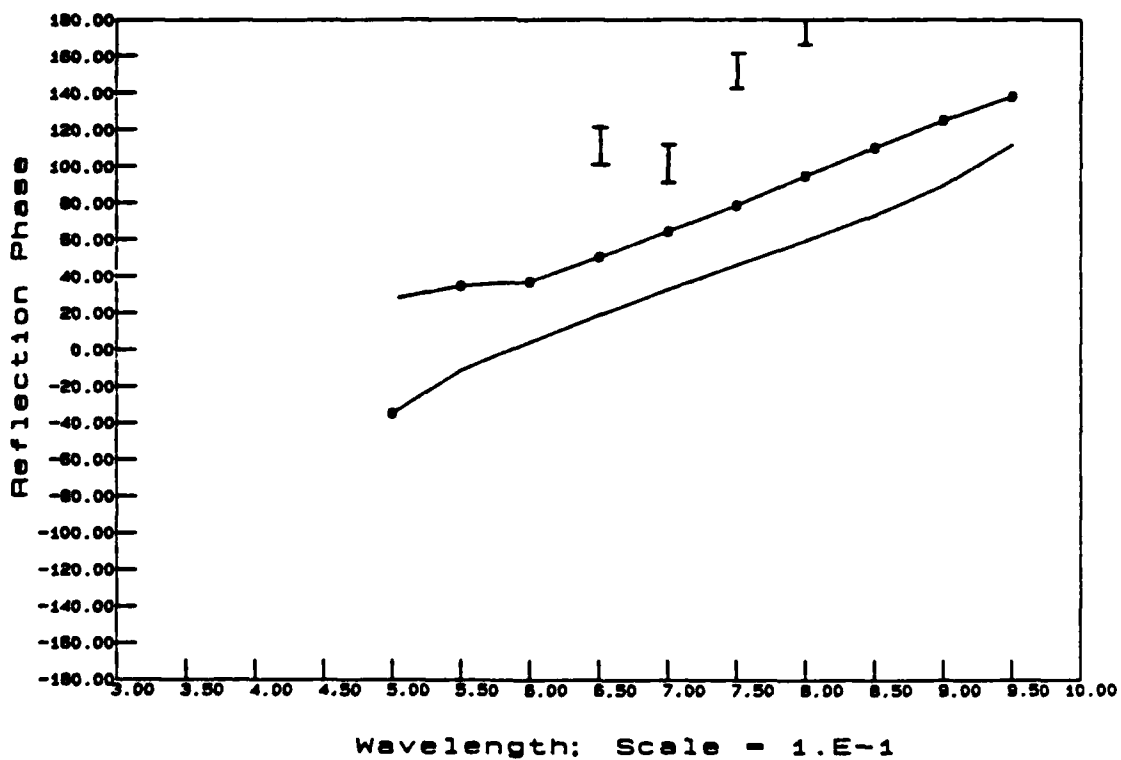


Fig. 13. Reflection phase, case $\delta/a = 0.75$.

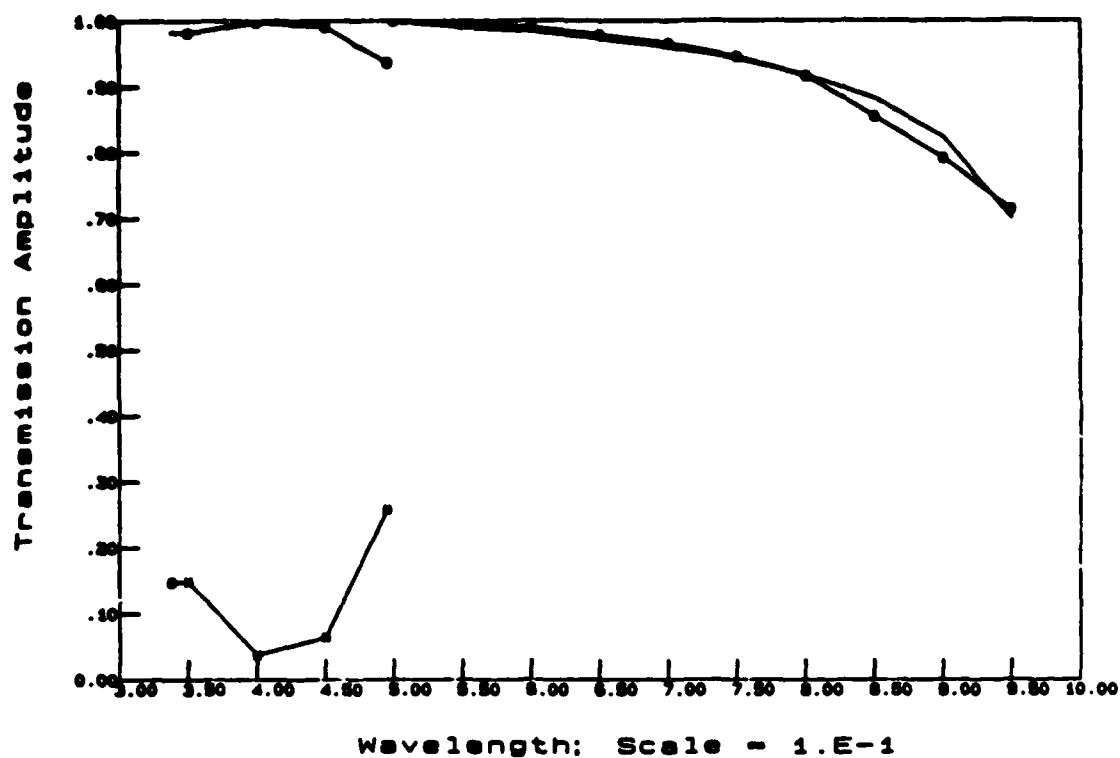


Fig. 14. Transmission amplitude, case $\delta/a = 0.75$.

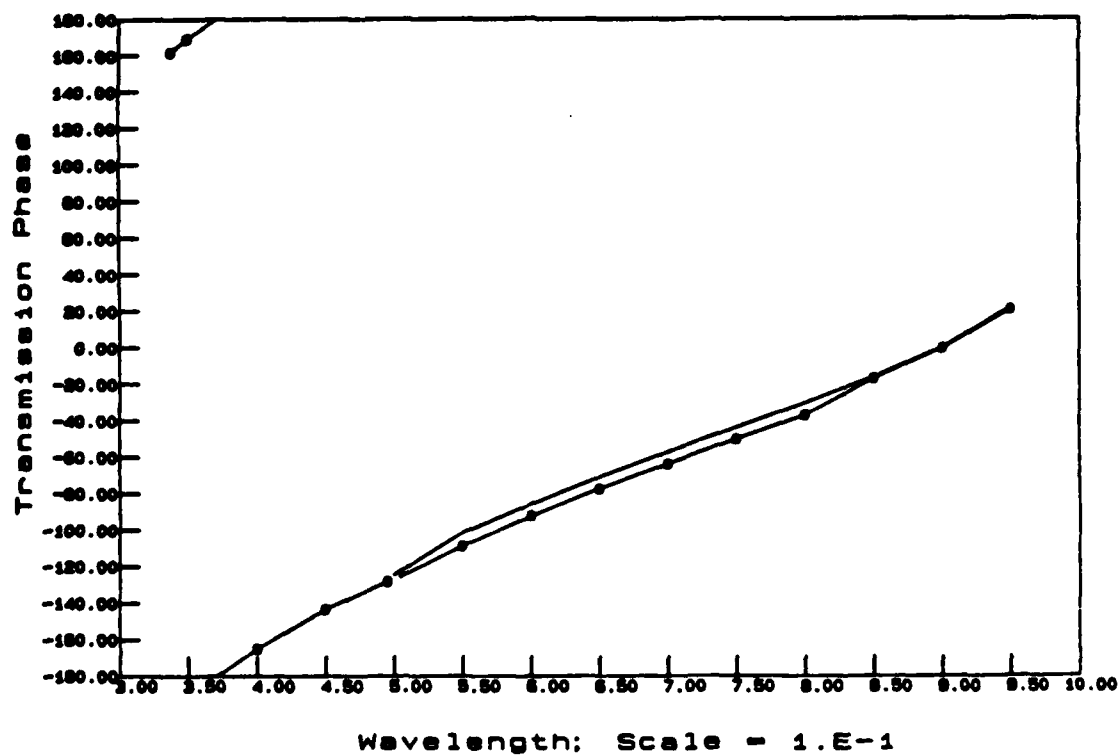


Fig. 15. Transmission phase, case $\delta/a = 0.75$.

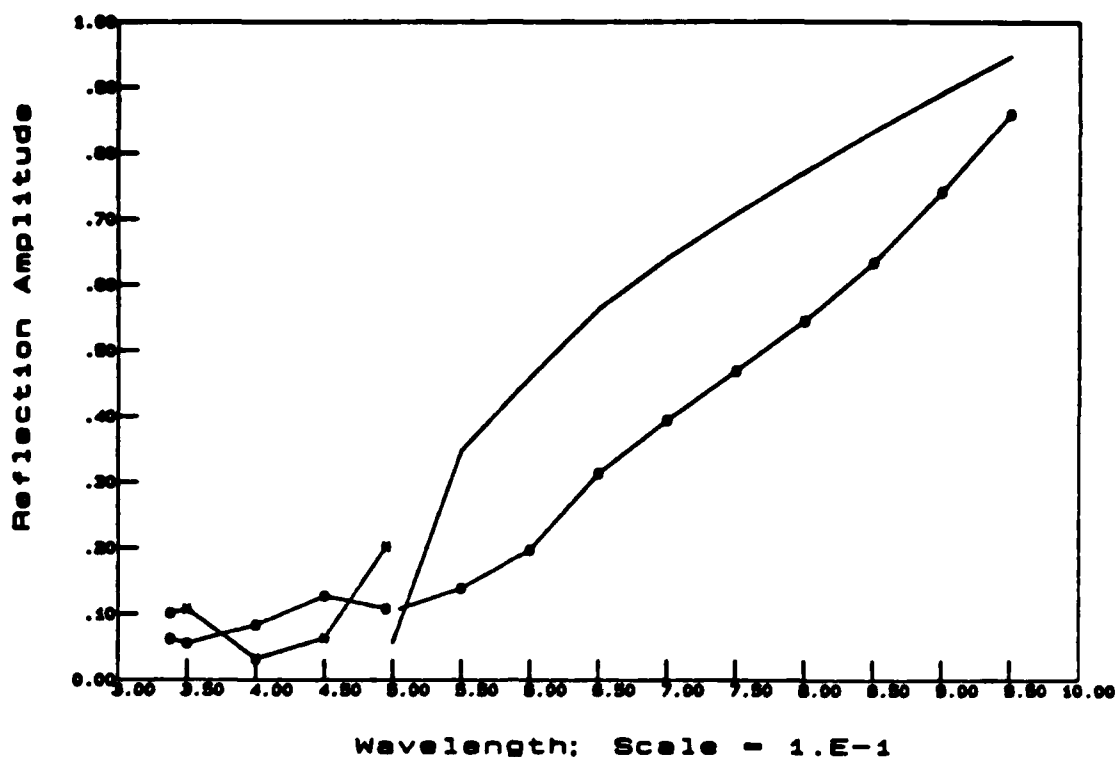


Fig. 16. Reflection amplitude, case $\delta/a = 0.625$.

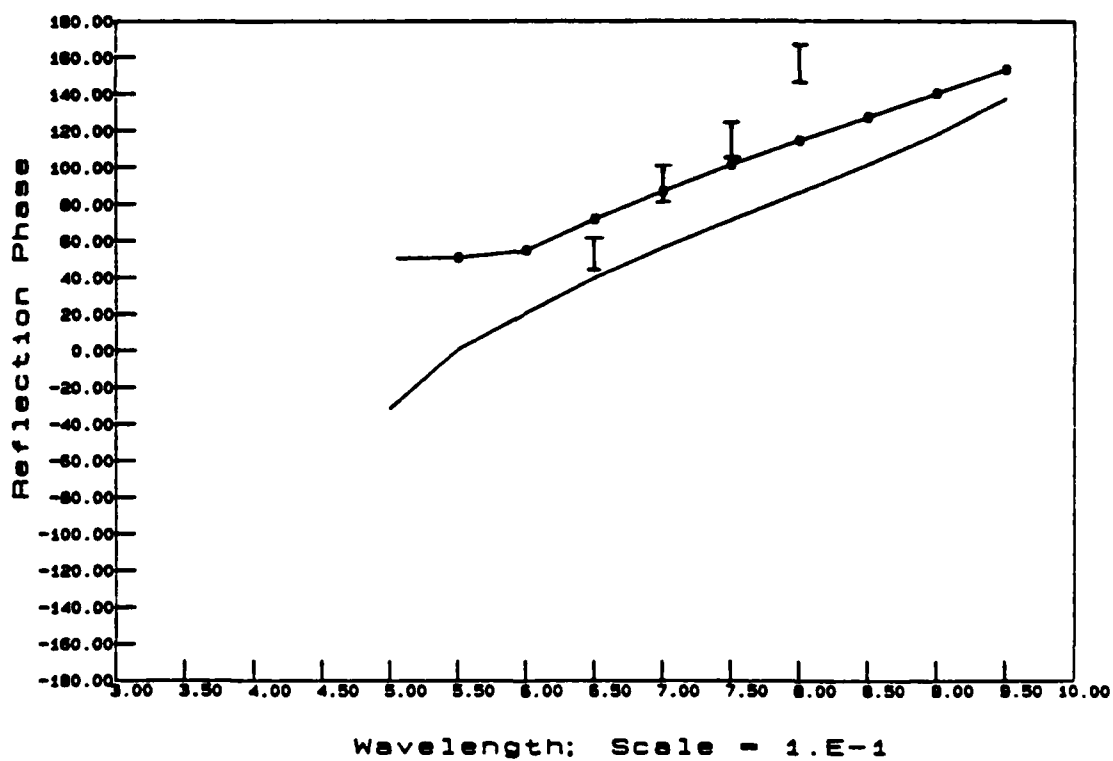


Fig. 17. Reflection phase, case $\delta/a = 0.625$.

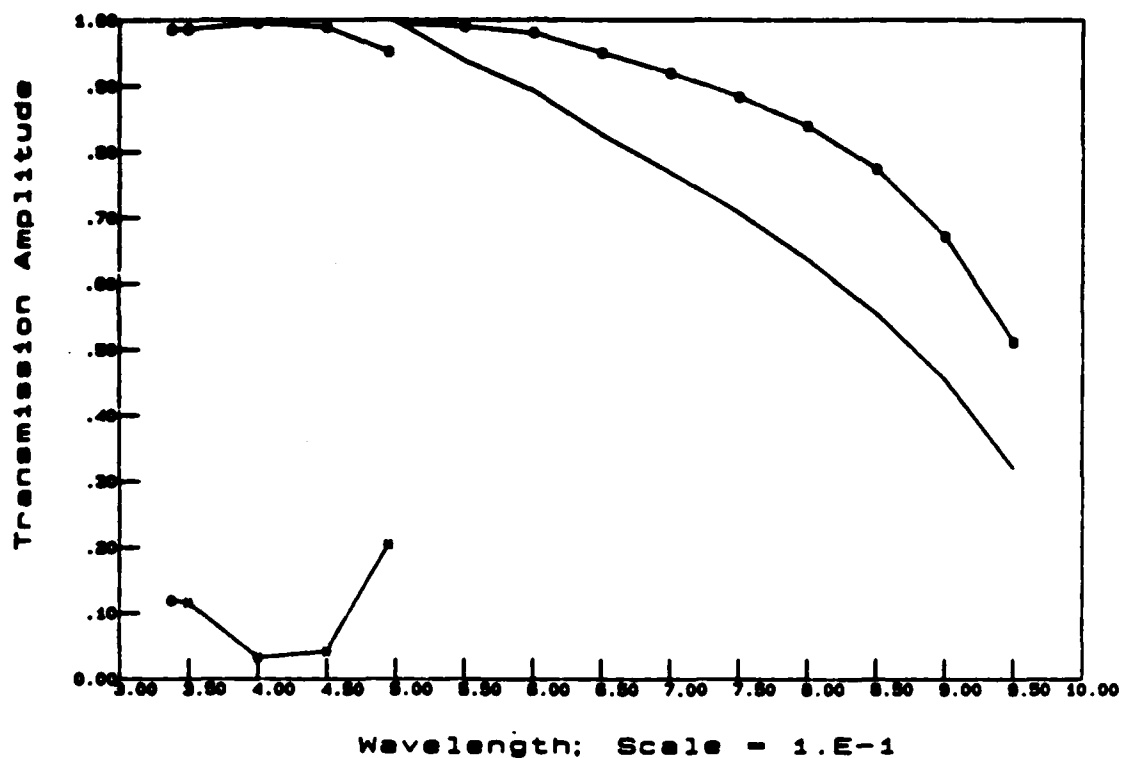


Fig. 18. Transmission amplitude, case $\delta/a = 0.625$.

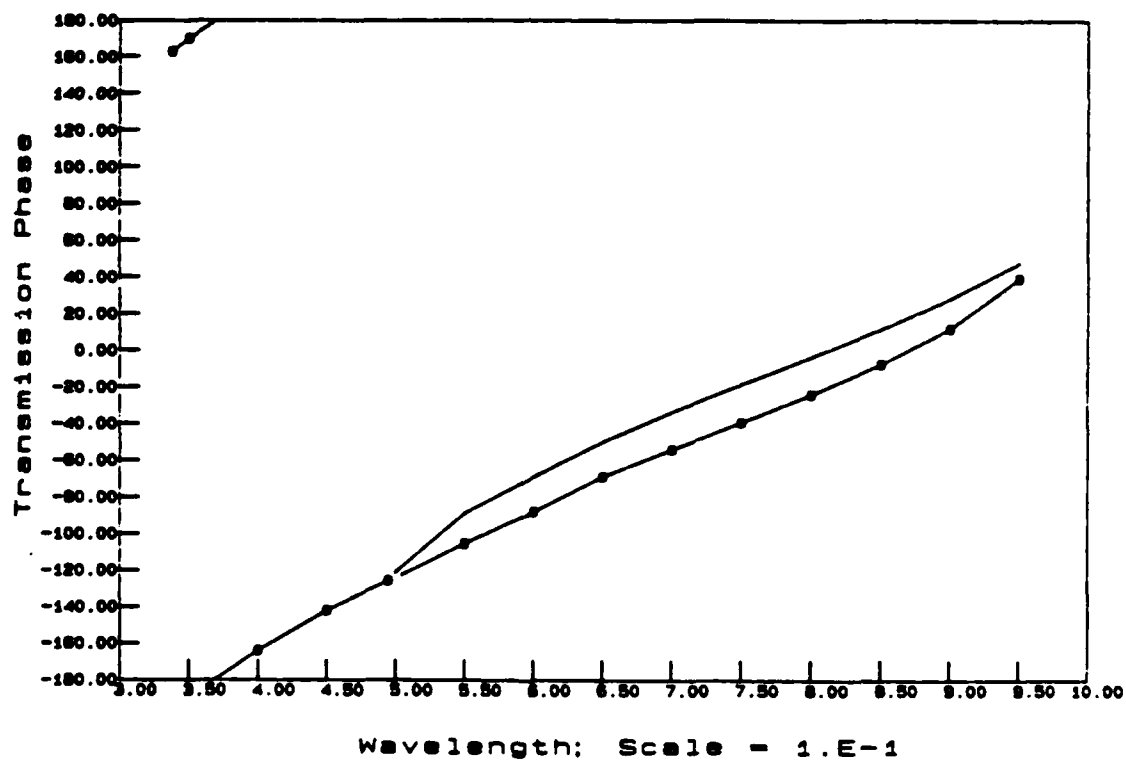


Fig. 19. Transmission phase, case $\delta/a = 0.625$.

not exact. Compare the reflection amplitude curves in Figs. 12 and 16. Note that the susceptance data solutions have changed quite markedly, while the numerically predicted solutions differ more modestly. The numerically predicted solutions also appear to vary in a more realistic way near the transition region, where $R = 1/2$. Although not conclusive, the measured reflection phase angle is closer to the numerically predicted curves than to the susceptance data curves, even when these curves are corrected by 180° , as previously mentioned.

Figures 20 and 21 give a measure of the consistency in the equation set at that point in the iterative process where the solution was taken. The ordinate on these graphs gives the average error per equation. The reason for this inconsistency was detailed in Section III; it is primarily due to the pulse basis approximation. These graphs show that the equation sets are more consistent at the high frequency end than at the low frequency end. Although this was not expected a priori, one postulation is that at the low frequency end, the guide wavelength is much different from the free space wavelength, and the effect of the waveguide walls is much stronger making the surface patch approximation more important. On the other hand, at the high frequency end, the guide wavelength and the free space wavelength are nearly the same, so the effect of the guiding walls is less critical.

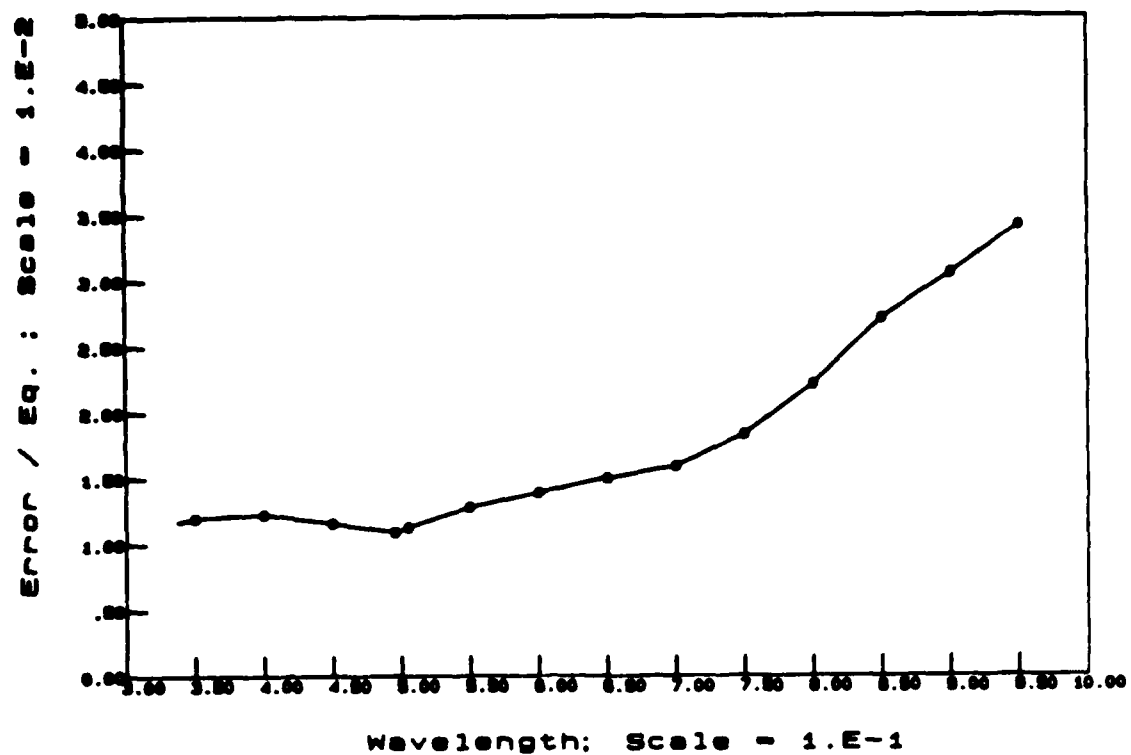


Fig. 20. Average error per equation, case $\delta/a = 0.75$.

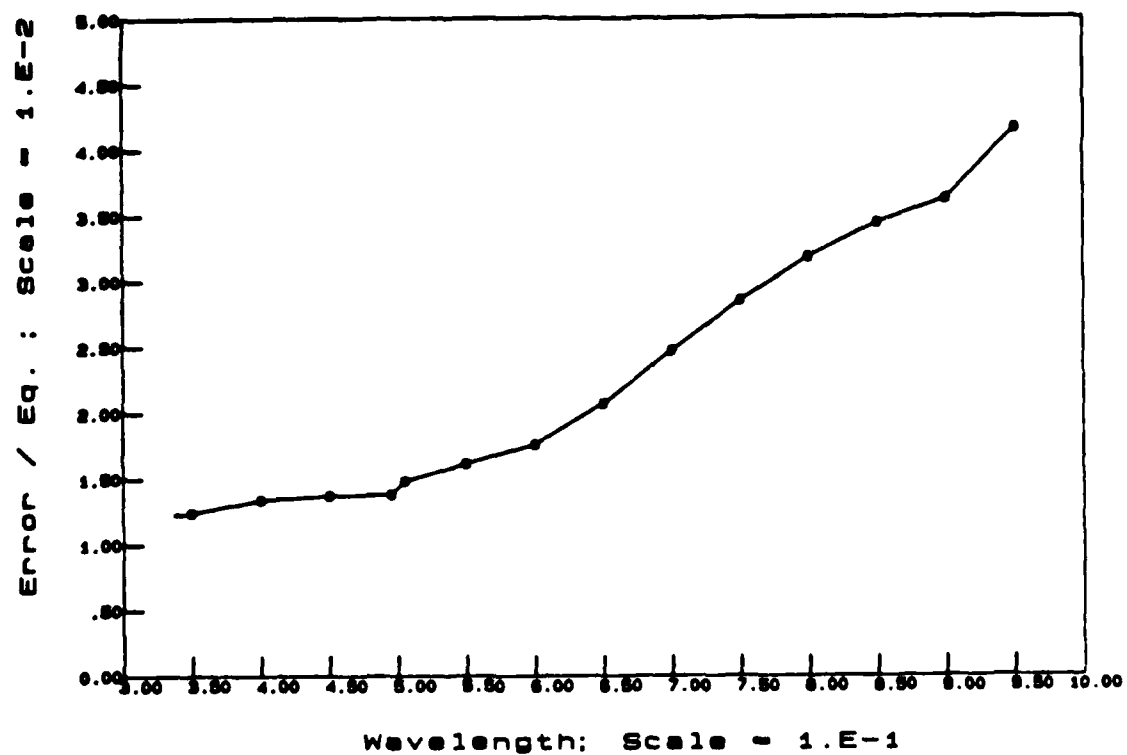


Fig. 21. Average error per equation, case $\delta/a = 0.625$.

V. CONCLUSIONS AND RECOMMENDATIONS

This paper has presented a three-dimensional multiport formulation for waveguide scattering. Simple test computations were made to verify the validity of the formulation. Test data were found to be in good agreement with results obtained using other methods for TE_{10} comparison. In the case of the asymmetric iris, results were obtained for higher order TE modes as well; these results are of dubious accuracy, as no other data were available for comparison. It is suspected that the higher mode accuracy will diminish rapidly because the patch size becomes comparable to the rate of variation in these modes.

Although the formulation presented here is quite generally valid, it may be more expensive, both in terms of storage cost and in terms of computational intensity, than a corresponding three-dimensional finite element approach. Computationally, one must compute on the order of N squared numerical integrations at each new wavelength. Because the solution algorithm is iterative, the same coefficients are needed several times, and hence all N squared complex numbers must be saved. The recomputation cost is simply too high to do otherwise. This represents a storage cost proportional to the surface area of the enclosed volume squared, or equivalently proportional to the volume to the four-thirds power. Finite element methods would require storage proportional to the volume. So a formulation of this type may not be competitive with the finite element method for internal scattering problems. The formulation of this paper may, however, be valuable for determining the scattering characteristics of slots in waveguides; that is, junctions

that involve radiation into free space. Here, boundary conditions external to the waveguide must also be taken into account. Unbounded problems pose more difficulties for finite element formulations than for surface integral equation approaches.

The formulation in this paper could be pushed into a category of useful mathematical software if some of the following problems could be solved:

1. A generalized surface generation routine could be developed that would allow the engineer to rapidly define new surface geometries. This might be done using a triangular net boundary. Each net point could then be moved about in three dimensions to form a wide variety of surface shapes.
2. Because the integral operator is a "smoothing" operator, we got away with using a pulse basis expansion on the conducting boundaries for the simple test cases of this paper. However, it is suggested for more complex geometries and increased solution accuracy that a smooth C^0 basis set be used. For the triangular net above, the rooftop functions described in Section III would form an appropriate basis set.
3. For the asymmetric iris data, the bulk of the computation time was tied up in computing the coupling coefficients and not in solving the resulting system of equations. Algorithms capable of rapidly and accurately approximating these coefficients would be valuable.
4. Many important problems involve at least one plane of symmetry; if this symmetry could be exploited, the number of

unknowns would be reduced by one half and the matrix would have only one quarter as many elements.

APPENDIX A
NORMALIZED MODE FUNCTIONS AND EXPANSIONS

Part 1

General relations for EM waves on a uniform cylindrical waveguiding system with care taken to preserve sign changes as a result of assumed propagation direction.

In a uniform waveguiding system, that is, a system having no physical or material variation along the z axis, it turns out that we have simplified relations for the transverse components of any mode supported by the structure in terms only of the axial, $\hat{z}$ directed field components.

To this end, expand the field vectors and the del operator into their transverse and axial components,

$$\bar{\mathbf{A}} \equiv \bar{\mathbf{A}}_t + \bar{\mathbf{A}}_z$$

$$\nabla \equiv \nabla_t + \nabla_z$$

With these definitions, note that

$$\nabla \times \bar{\mathbf{A}} = \nabla_t \times \bar{\mathbf{A}}_t + \nabla_t \times \bar{\mathbf{A}}_z + \nabla_z \times \bar{\mathbf{A}}_t$$

Using this identity in Maxwell's equations, Eqs. 10 and 11, and identifying component directions, we obtain two relations for the transverse components of the equations,

$$\nabla_t \times \bar{E}_z + \nabla_z \times \bar{E}_t = -j\omega\mu\bar{H}_t \quad (A.1)$$

$$\nabla_t \times \bar{H}_z + \nabla_z \times \bar{H}_t = j\omega\epsilon\bar{E}_t \quad (A.2)$$

Since the structure is uniform along z , we can assume that mode m will propagate with spatial variation that looks like

$$\bar{A}(x,y,z) = \bar{A}(x,y) e^{\mp j\beta_m z}$$

where the $-$ sign is for propagation in the $+\hat{z}$ direction and the $+$ sign for propagation along the $-\hat{z}$ direction. With this understanding, note that

$$\nabla_z \times \bar{A}_{m_t} = \mp j\beta_m (\hat{z} \times \bar{A}_{m_t})$$

Using this result in Eqs. A.1 and A.2,

$$-j\omega\mu\bar{H}_{m_t} = \nabla_t \times \bar{E}_{m_z} \mp j\beta_m (\hat{z} \times \bar{E}_{m_t}) \quad (A.3)$$

$$j\omega\epsilon\bar{E}_{m_t} = \nabla_t \times \bar{H}_{m_z} \mp j\beta_m (\hat{z} \times \bar{H}_{m_t}) \quad (A.4)$$

Substituting Eq. A.4 into Eq. A.3 to eliminate the transverse component of E and using the identities

$$\hat{z} \times (\nabla_t \times \bar{A}_z) = \nabla_t A_z$$

$$\hat{z} \times (\hat{z} \times \bar{A}_t) = -\bar{A}_t$$

we obtain

$$(k^2 - \beta_m^2) \bar{H}_{m_t} = j\omega\epsilon \nabla_t \times \bar{E}_{m_z} \mp j\beta_m \nabla_t H_{m_z} \quad (A.5)$$

Similarly, we could eliminate the transverse components of H by substituting Eq. A.3 into Eq. A.4 to obtain

$$(k^2 - \beta_m^2) \bar{E}_{m_t} = -j\omega\mu \nabla_t \times \bar{H}_{m_z} \mp j\beta_m \nabla_t E_{m_z} \quad (A.6)$$

Here, $k^2 \equiv \omega^2 \mu \epsilon$. Equations A.5 and A.6 are standard results except that the $\mp$ signs are not often included; usually, just the $-$ sign is used for propagation in the $+\hat{z}$ direction. This distinction is important for mode expansions involving propagation in both directions.

In the case of TE modes, $E_{m_z} = 0$ and Eqs. A.5 and A.6 reduce to

$$\begin{aligned} \bar{H}_{m_t} &= \frac{\mp j\beta_m}{k^2 - \beta_m^2} \nabla_t H_{m_z} \\ \bar{E}_{m_t} &= \frac{-j\omega\mu}{k^2 - \beta_m^2} \left[\nabla_t H_{m_z} \times \hat{z} \right] \end{aligned} \quad (A.7)$$

For TM modes, $H_{m_z} = 0$ and Eqs. A.5 and A.6 reduce to

$$\begin{aligned}\bar{H}_{m_t} &= \frac{j\omega\epsilon}{k^2 - \beta_m^2} \left[\nabla_t E_{m_z} \times \hat{z} \right] \\ \bar{E}_{m_t} &= \frac{\mp j\beta_m}{k^2 - \beta_m^2} \nabla_t E_{m_z}\end{aligned}\quad (A.8)$$

These equations show that the axial field component in a cylindrical waveguide plays a role similar to that of a scalar potential in general EM field theory. The fields in a cylindrical system are completely determined once the axial fields are known.

Since the electric and magnetic fields are solutions to vector wave equations, Eqs. 14 and 15, the axial fields must be solutions to

$$(\nabla_t^2 + \gamma_m^2) \begin{bmatrix} H_{m_z} \\ E_{m_z} \end{bmatrix} = 0 \quad \begin{matrix} \text{TE} \\ \text{TM} \end{matrix} \quad (A.9)$$

where $\gamma_m^2 \equiv k^2 - \beta_m^2$. Solutions to Eq. A.9 are subject to the following boundary conditions:

$$\hat{n} \cdot \nabla H_{m_z} = 0 \quad \text{for TE}$$

$$E_{m_z} = 0 \quad \text{for TM}$$

where $\hat{n}$ is a unit normal to the conducting surface of the cylindrical waveguide.

Part 2

Normalized mode functions. In Part 1, the basic relationships between field components in a cylindrical system were outlined. The electric and magnetic mode vectors were denoted by capital letters E and H, respectively. In this section, a normalization condition is given. The normalized mode vectors will be denoted using lower case e and h. The normalization condition used here is that each mode carry 1 watt real power above cutoff, and 1 watt reactive (imaginary) power below cutoff. Explicitly, integrate the Poynting vector over the guide cross section S and enforce the following condition:

$$\frac{1}{2} \int_S \vec{e}_{m_t} \times \vec{h}_{m_t} \cdot d\vec{s} = \begin{cases} 1 \text{ [W] for both TE and TM modes above cutoff} \\ j \text{ [W] for TE modes below cutoff} \\ -j \text{ [W] for TM modes below cutoff} \end{cases} \quad (\text{A.10})$$

The change of sign on the reactive power below cutoff reflects the fact that nonpropagating TE modes appear inductive, while nonpropagating TM modes appear capacitive.

Using the relations from Part 1, the condition indicated by Eq. A.10 can be enforced in terms of the axial fields only. For TE modes, use Eqs. A.7 to obtain

$$\frac{1}{2} \sqrt{\frac{\mu}{\epsilon}} \frac{k\beta_m^*}{\gamma_m^2} \int_S h_{m_z} h_{m_z}^* ds = \begin{cases} 1 \text{ [W] above cutoff} \\ 1 \text{ [W] below cutoff} \end{cases} \quad (\text{A.11})$$

For TM modes, use Eqs. A.8 yielding

$$\frac{1}{2} \sqrt{\frac{\epsilon}{\mu}} \frac{k\beta_m}{\gamma_m^2} \int_S \mathbf{e}_m \cdot \mathbf{e}_m^* dz = \begin{cases} 1 [W] & \text{above cutoff} \\ -j [W] & \text{below cutoff} \end{cases} \quad (\text{A.12})$$

Once the axial modes have been normalized, the corresponding transverse field components may be found using Eqs. A.7 and A.8.

Arbitrary fields in a cylindrical waveguide can now be written in terms of the normalized mode functions,

$$\bar{\mathbf{H}} = \sum_m (a_m \hat{\mathbf{h}}_m + b_m \hat{\mathbf{h}}_m^*)$$

where m ranges over all modes, both TE and TM, above and below cutoff. The forward-backward arrows indicate propagation in the $+\hat{z}$ and $-\hat{z}$ directions, respectively.

Part 3

Explicit mode functions for rectangular waveguides. In a rectangular waveguide with cross-section dimensions a and b , the mode functions that satisfy Eq. A.9 with the appropriate boundary condition for TE modes is

$$h_{z_{mn}} = A_{mn} \cos \frac{m\pi x}{a} \cos \frac{n\pi y}{b}$$

with

$$\gamma_{mn}^2 = \left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2$$

$m, n = 0, 1, 2, \dots$; but not both zero

Using Eq. A.12 for normalization,

$$A_{mn} = \left[\frac{2v\gamma_{mn}^2 \eta_{mn}}{k\beta_{mn}^* ab} \sqrt{\frac{\epsilon}{\mu}} \right]^{1/2}$$

where

$$v = \begin{cases} 1 & [W] \text{ above cutoff} \\ j & [W] \text{ below cutoff} \end{cases}$$

$$\eta_{mn} = \begin{cases} 2 & \text{if } m \text{ or } n \text{ is } 0 \\ 4 & \text{otherwise} \end{cases}$$

For TM modes, the mode function that satisfies Eq. A.9 with $e_z = 0$ on the boundary is

$$e_{z_{mn}} = A_{mn} \sin \frac{\pi mx}{a} \sin \frac{\pi ny}{b}$$

with

$$\gamma_{mn}^2 = \left(\frac{m\pi}{a} \right)^2 + \left(\frac{n\pi}{b} \right)^2$$

$$m, n = 1, 2, \dots$$

Using Eq. A.13 for normalization,

$$A_{mn} = \left[\frac{8v\gamma_{mn}^2}{k\beta_{mn}^* ab} \sqrt{\frac{\mu}{\epsilon}} \right]^{1/2}$$

where

$$v = \begin{cases} 1 [W] \text{ above cutoff} \\ -j [W] \text{ below cutoff} \end{cases}$$

APPENDIX B

COMPLEX ART

Consider a linear system of equations,

$$A\underline{x} = \underline{b}$$

Let $\underline{y}^{(k)}$ be the k th approximation to the solution vector $\underline{x}$ and note that the error or residual in the i th equation in the system can be written as

$$r_i^{(k)} = \underline{a}_i \underline{y}^{(k)} - b_i$$

where $\underline{a}_i$ is the i th row vector from the matrix A , and b_i the i th element in the column vector $\underline{b}$. The gradient of the residual, steepest descent direction, can be computed with respect to a change in the approximation vector $\underline{y}^{(k)}$,

$$\nabla r_i^{(k)} = \underline{a}_i^T$$

This says that the gradient of the residual for the i th equation in the system is the transpose of the i th row vector in the matrix A . Now, adjust the approximate solution along the steepest descent direction using the update form

$$\underline{y}^{(k+1)} = \underline{y}^{(k)} - \alpha_k \underline{a}_i^T$$

The scalar α_k is chosen to make r_i zero. This gives the condition

$$0 = \underline{a}_i \left[\underline{y}^{(k)} - \alpha_k \underline{a}_i^T \right] - b_i$$

Solving for α_k , we have

$$\alpha_k = \frac{r_i^{(k)}}{\underline{a}_i^T \underline{a}_i}$$

The basic ART update for real systems of equations is then

$$\underline{y}^{(k+1)} = \underline{y}^{(k)} - \frac{r_i^{(k)} \underline{a}_i^T}{\underline{a}_i^T \underline{a}_i} \quad (\text{B.1})$$

Note that ART adjusts every element in the solution vector for each equation. When Eq. B.1 has been enforced for every equation in the system, a complete update has been accomplished. Usually, several updates are required, but if the system has a unique solution, then

$$\lim_{k \rightarrow \infty} \left(\underline{y}^{(k)} - \underline{x} \right) = 0$$

Any n by m complex system of equations can be written as a $2n$ by $2m$ real system to which Eq. B.1 can be applied directly. Alternatively, update formulae equivalent to Eq. B.1 can be derived which apply directly to complex systems. For $\underline{A} \underline{x} = \underline{b}$ a complex system, decompose the complex residual into its real and imaginary parts,

$$r_i = \alpha_i + j\beta_i \quad \alpha, \beta \text{ real valued}$$

Then the update that makes α_i zero is

$$y^{(k+1)} = y^{(k)} - \frac{\alpha_i^{(k)} \underline{a}_i^+}{\underline{a}_i \underline{a}_i^+} \quad (\text{B.2})$$

The update making β_i zero is

$$y^{(k+1)} = y^{(k)} - \frac{\beta_i^{(k)} \underline{a}_i^T}{\underline{a}_i \underline{a}_i^+} \quad (\text{B.3})$$

where the following notation applies:

$\underline{a}_i^T$: the transpose of $\underline{a}_i$

$\underline{a}_i^+$: the conjugate transpose of $\underline{a}_i$

Equations B.2 and B.3 are applied by computing the real part of the residual for equation i , α_i , then enforcing Eq. B.2. Next compute the imaginary part of the residual for equation i , β_i , and enforce Eq. B.3.

A variant on this algorithm that I call new ART or NART applies to systems where the diagonal element is larger in magnitude than other elements in the same row. This update changes only the i th component of y when applied to the i th equation, rather than all the elements in y . The NART update formulas corresponding to Eqs. B.2 and B.3 are

$$y_1^{(k+1)} = y_1^{(k)} - \frac{\alpha_1^{(k)} a_{11}}{a_1 + a_1}$$

$$y_1^{(k+1)} = y_1^{(k)} - \frac{\beta_1^{(k)} a_{11}}{a_1 + a_1} \quad (\text{B.4})$$

where the elements of A are $[a_{ij}]$. Equations B.4 no longer reduce the i th residual to zero at each update, but in many cases this update scheme converges in fewer iterations and it costs less per iteration. NART is the update scheme used by subroutines ARTA and ARTB in this report.

APPENDIX C

FORTRAN FOREVER

This appendix gives documentation for the three Fortran codes used to test the formulation presented in Section III. These codes are of no immediate value, but they are included here to demonstrate work successfully completed and also because they may be useful for showing how such computations may be organized.

Program rspec.f

This program allows for the specification of arbitrary rectangles in three dimensions. First, the plane of the rectangle is specified by a point in the plane, $\bar{x}_1$, and by the direction of the "outward" normal to the plane, $\hat{n}$. Here, the term outward means outward with respect to the volume that is enclosed by the rectangles. The equation of the plane is given by all points $\bar{x}$ such that $\bar{x} \cdot \hat{n} = \bar{x}_1 \cdot \hat{n}$, or more explicitly by

$$n_1x + n_2y + n_3z = \hat{n} \cdot \bar{x}_1 \quad (C.1)$$

where $\hat{n} = n_1\hat{x} + n_2\hat{y} + n_3\hat{z}$. The second vertex of the rectangle $\bar{x}_2$ can now be specified using only two of its three components. Since specifying two of the three variables in Eq. C.1 completely determines the third variable, we can write the triplet of points as:

$$\begin{aligned}
&[(\hat{n} \cdot \bar{x}_1 - n_2 y_2 - n_3 z_2)/n_1, y_2, z_2] && \text{for } n_1 > n_2, n_3 \\
&[x_2 (\hat{n} \cdot \bar{x}_1 - n_1 x_2 - n_3 z_2)/n_2, z_2] && \text{for } n_2 > n_1, n_3 \\
&[x_2, y_2, (\hat{n} \cdot \bar{x}_1 - n_1 x_2 - n_2 y_2)/n_3] && \text{for } n_3 > n_1, n_2 \quad (C.2)
\end{aligned}$$

The three forms above are entirely equivalent except when n_1 , n_2 , or n_3 are zero. To avoid division by zero, use the appropriate version of Eq. C.2; note that at least one component of $\hat{n}$ is not 0, since $|\hat{n}| = 1$. The surface primary direction can now be defined using

$$\bar{p} = (x_1 - x_1)\hat{x} + (y_2 - y_1)\hat{y} + (z_2 - z_1)\hat{z} \quad (C.3)$$

The third vertex $\bar{x}_3$ can now be specified by entering just one of its three components, since it must lie in the plane and on a line perpendicular to $\bar{p}$, i.e., $\bar{s} \cdot \bar{p} = 0$, where $\bar{s}$ is the secondary direction given by

$$\bar{s} = (x_3 - x_1)\hat{x} + (y_3 - y_1)\hat{y} + (z_3 - z_1)\hat{z} \quad (C.4)$$

Enforcing this condition, we have for each of the three cases in Eq. C.2:

Case 1, $n_1 > n_2, n_3$

$$ay_3 + bz_3 = c$$

where

$$a = y_2 - y_1 - n_2(x_2 - x_1)/n_1$$

$$b = z_2 - z_1 - n_3(x_2 - x_1)/n_1$$

$$c = [x_1 - \hat{n} \cdot \bar{x}_1/n_1](x_2 - x_1) + y_1(y_2 - y_1) + z_1(z_2 - z_1)$$

If $|a| > |b|$, use $y_3 = (c/a) - (b/a)z_3$, enter z_3 , evaluate y_3 , then use Eq. C.1 to find x_3 , otherwise use $z_3 = (c/b) - (a/b)y_3$, enter y_3 , evaluate z_3 , then use Eq. C.1 to find x_3 .

Case 2, $n_2 > n_1, n_3$

$$ax_3 + bz_3 = c$$

where

$$a = x_2 - x_1 - n_1(y_2 - y_1)/n_2$$

$$b = z_2 - z_1 - n_3(y_2 - y_1)/n_2$$

$$c = x_1(x_2 - x_1) + [y_1 - \hat{n} \cdot \bar{x}_1/n_2](y_2 - y_1) + z_1(z_2 - z_1)$$

If $|a| > |b|$, use $x_3 = (c/a) - (b/a)z_3$, enter z_3 , evaluate x_3 , then use Eq. C.1 to find y_3 , otherwise use $z_3 = (c/b) - (a/b)x_3$, enter x_3 , evaluate z_3 , then use Eq. C.1 to find y_3 .

Case 3, $n_3 > n_1, n_2$

$$ax_3 + by_3 = c$$

where

$$a = x_2 - x_1 - n_1(z_2 - z_1)/n_3$$

$$b = y_2 - y_1 - n_2(z_2 - z_1)/n_3$$

$$c = x_1(x_2 - x_1) + y_1(y_2 - y_1) + [z_1 - \hat{n} \cdot \bar{x}_1/n_3](z_2 - z_1)$$

If $|a| > |b|$, use $x_3 = (c/a) - (b/a)y_3$, enter y_3 , evaluate x_3 , then use Eq. C.1 to find z_3 , otherwise use $y_3 = (c/b) - (a/b)x_3$, enter x_3 , evaluate y_3 , then use Eq. C.1 to find z_3 .

This method of rectangle specification requires a minimum amount of input from the user, thereby avoiding specification errors.

Note that the rectangle vertex information should be ordered so that $\hat{p} \times \hat{s} = \hat{n}$, where $\hat{p}$ and $\hat{s}$ are unit vectors in the directions $\bar{p}$ and $\bar{s}$ from Eqs. C.3 and C.4. Any point on the rectangular surface can now be described by the local coordinates (p,s) . The transformation from the local system to the absolute point $\bar{x}$ is accomplished by

$$\bar{x} = \bar{x}_1 + p\hat{p} + s\hat{s} \quad \text{for} \quad \begin{array}{l} p \in [0, D_p] \\ s \in [0, D_s] \end{array} \quad (C.5)$$

where

$$D_p = |\bar{x}_2 - \bar{x}_1|, \quad D_s = |\bar{x}_3 - \bar{x}_1| \quad (C.6)$$

For the final output record `rspec.f` stores the following information about each rectangle:

$$\bar{x}_1, \hat{n}, \hat{p}, \hat{s}, D_p, N_p, N_s$$

where D_p and D_s are the primary and secondary edge lengths from Eqs. C.6, and N_p and N_s are the number of primary and secondary subdivisions along each edge.

PROGRAM RSPEC

C This program allows you to specify arbitrary rectangles in three
C dimensions. First the plane of the rectangle is defined by the
C "outward" normal direction and the coordinates of the first vertex,
C V1. Next the program will ask you to give the two coordinates need-
C ed to specify the second vertex, V2, on the defined plane. Finally
C you will be ask to enter the single remaning coordinate required
C to specify the third vertex, V3, along the normal to V1-V2, through
C V1, and on the defined plane. From this data the primary and second-
C ary unit directions are computed for later use by surface integra-
C tion routines.

```
REAL V1(10,3),UN(10,3),UP(10,3),US(10,3),DP(10),DS(10)
INTEGER NP(10),NS(10)
```

```
WRITE(6,*) 'Enter the number of rectangles to be specified.'
READ(5,*) NR
WRITE(6,*) 'Enter the number of conducting rectangles out of total.'
READ(5,*) NRC
```

```
WRITE(6,*) 'Specify the conducting rectangles first.'
DO 100 I=1,NR
5 WRITE(6,*) ' '
WRITE(6,*) 'Data entery for rectangle number ',I,'.'
WRITE(6,*) 'Enter normal direction, Nx,Ny,Nz.'
READ(5,*) X,Y,Z
D=SQRT(X**2+Y**2+Z**2)
UN(I,1)=X/D
UN(I,2)=Y/D
UN(I,3)=Z/D
WRITE(6,*) 'Enter first vertex, X1,Y1,Z1.'
READ(5,*) X1,Y1,Z1
D=X1*UN(I,1)+Y1*UN(I,2)+Z1*UN(I,3)
```

C Test for free variables.

```
XN=ABS(UN(I,1))
YN=ABS(UN(I,2))
ZN=ABS(UN(I,3))
IF((XN.GE.YN).AND.(XN.GE.ZN)) GOTO 60
IF((YN.GE.XN).AND.(YN.GE.ZN)) GOTO 30
```

C Case 1. N3 > N1 and N2 .

```
WRITE(6,*) 'Enter X2,Y2.'
READ(5,*) X2,Y2
Z2=(D-UN(I,1)*X2-UN(I,2)*Y2)/UN(I,3)
UP(I,1)=X2-X1
UP(I,2)=Y2-Y1
UP(I,3)=Z2-Z1
A=UP(I,1)-UN(I,1)*UP(I,3)/UN(I,3)
B=UP(I,2)-UN(I,2)*UP(I,3)/UN(I,3)
C=X1*UP(I,1)+Y1*UP(I,2)+(Z1-D/UN(I,3))*UP(I,3)
IF(ABS(A).GT.ABS(B)) GOTO 10

WRITE(6,*) 'Enter X3.'
READ(5,*) X3
Y3=C/B-(A/B)*X3
```

```

GOTO 20

10  WRITE(6,*) 'Enter Y3.'
    READ(5,*) Y3
    X3=C/A-(B/A)*Y3

20  Z3=(D-UN(I,1)*X3-UN(I,2)*Y3)/UN(I,3)
    GOTO 90

C   Case 2. N2 > N1 and N3 .
30  WRITE(6,*) 'Enter X2,Z2.'
    READ(5,*) X2,Z2
    Y2=(D-UN(I,1)*X2-UN(I,3)*Z2)/UN(I,2)
    UP(I,1)=X2-X1
    UP(I,2)=Y2-Y1
    UP(I,3)=Z2-Z1
    A=UP(I,1)-UN(I,1)*UP(I,2)/UN(I,2)
    B=UP(I,3)-UN(I,3)*UP(I,2)/UN(I,2)
    C=X1*UP(I,1)+Z1*UP(I,3)+(Y1-D/UN(I,2))*UP(I,2)
    IF(ABS(A).GT.ABS(B)) GOTO 40

    WRITE(6,*) 'Enter X3.'
    READ(5,*) X3
    Z3=C/B-(A/B)*X3
    GOTO 50

40  WRITE(6,*) 'Enter Z3.'
    READ(5,*) Z3
    X3=C/A-(B/A)*Z3

50  Y3=(D-UN(I,1)*X3-UN(I,3)*Z3)/UN(I,2)
    GOTO 90

C   Case 3. N1 > N2 and N3 .
60  WRITE(6,*) 'Enter Y2,Z2.'
    READ(5,*) Y2,Z2
    X2=(D-UN(I,2)*Y2-UN(I,3)*Z2)/UN(I,1)
    UP(I,1)=X2-X1
    UP(I,2)=Y2-Y1
    UP(I,3)=Z2-Z1
    A=UP(I,2)-UN(I,2)*UP(I,1)/UN(I,1)
    B=UP(I,3)-UN(I,3)*UP(I,1)/UN(I,1)
    C=Y1*UP(I,2)+Z1*UP(I,3)+(X1-D/UN(I,1))*UP(I,1)
    IF(ABS(A).GT.ABS(B)) GOTO 70

    WRITE(6,*) 'Enter Y3.'
    READ(5,*) Y3
    Z3=C/B-(A/B)*Y3
    GOTO 80

70  WRITE(6,*) 'Enter Z3.'
    READ(5,*) Z3
    Y3=C/A-(B/A)*Z3

80  X3=(D-UN(I,2)*Y3-UN(I,3)*Z3)/UN(I,1)

```

```

C      Compute unit directions and distances.
90     DP(I)=SQRT(UP(I,1)**2+UP(I,2)**2+UP(I,3)**2)
        UP(I,1)=UP(I,1)/DP(I)
        UP(I,2)=UP(I,2)/DP(I)
        UP(I,3)=UP(I,3)/DP(I)
        US(I,1)=X3-X1
        US(I,2)=Y3-Y1
        US(I,3)=Z3-Z1
        DS(I)=SQRT(US(I,1)**2+US(I,2)**2+US(I,3)**2)
        US(I,1)=US(I,1)/DS(I)
        US(I,2)=US(I,2)/DS(I)
        US(I,3)=US(I,3)/DS(I)
        V1(I,1)=X1
        V1(I,2)=Y1
        V1(I,3)=Z1

        WRITE(6,*) 'Enter the number of primary subdivisions, NP.'
        READ(5,*) NP(I)
        WRITE(6,*) 'Enter the number of secondary subdivisions, NS.'
        READ(5,*) NS(I)

C      Error allowance, in case of incorrect data entry.
        WRITE(6,*) 'If data entry was incorrect type "0", otherwise "1".'
        READ(5,*) NANS
        IF(NANS.EQ.0) GOTO 5
100    CONTINUE

C      Store point data.
        OPEN(10,FILE='rec.dat',STATUS='unknown')
        REWIND(10)
        WRITE(10,*) NR,NRC
        DO 110 I=1,NR
            WRITE(10,*) (V1(I,J),J=1,3)
            WRITE(10,*) (UN(I,J),J=1,3)
            WRITE(10,*) (UP(I,J),J=1,3)
            WRITE(10,*) (US(I,J),J=1,3)
            WRITE(10,*) DP(I),DS(I),NP(I),NS(I)
110    CONTINUE
        CLOSE(10)

C      Test, compute vertices from data as a check.
        DO 120 I=1,NR
            WRITE(6,*) I
            WRITE(6,*) (V1(I,J),J=1,3)
            X2=V1(I,1)+DP(I)*UP(I,1)
            Y2=V1(I,2)+DP(I)*UP(I,2)
            Z2=V1(I,3)+DP(I)*UP(I,3)
            WRITE(6,*) X2,Y2,Z2
            X3=V1(I,1)+DS(I)*US(I,1)
            Y3=V1(I,2)+DS(I)*US(I,2)
            Z3=V1(I,3)+DS(I)*US(I,3)
            WRITE(6,*) X3,Y3,Z3
120    CONTINUE

        STOP 'End of run.'
        END

```

Program DoJob.f

This program reads the geometrical data from file rec.dat as set up by program rspec.f and generates the patch coupling coefficients needed to form a linear system of equations. The numerical integrations are performed using Gaussian quadrature rules. Two-dimensional integrations are estimated by combining two one-dimensional integration schemes, one embedded inside the other. The arbitrary interval formula, Eq. 25.4.30 from Abramowitz and Stegun¹⁶ was used.

The primary program variable definitions are:

A	Test point, loop index. Integer value. Used for counting and ordering rectangles.
C	Evaluation loop index. Integer value. Used for ordering either conducting or field boundary rectangles, but not both in a given Do loop.
V1(A,I)	Absolute (x,y,z) position of vertex 1 on rectangle A. I = 1,2,3 for x,y,z components, respectively.
UN(A,I), UP(A,I) US(A,I)	Unit vector directions n, p, and s, respectively.
DP(A), DS(A)	Distance primary and secondary, D_p and D_s , for rectangle A.
NP(A), NS(A)	Number of primary and secondary subdivisions of rectangle A.
MODES(C)	Number of modes on field boundary C.

$XT(I), XI(I)$ Test position and evaluation position variables. $I = 1, 2, 3$ for x, y, z components of position. Position is computed in terms of local surface coordinates using Eq. C.5.

$TSI(I), W(I)$ Test point and weight values for 8 point Gaussian quadrature. Used for numerical approximation of self term coupling coefficients. $I = 1, 2, \dots, 8$.

$TS(I), V(I)$ Test point and weight values for 3 point Gaussian quadrature. Used for computing all coupling coefficients except the self term value. $I = 1, 2, 3$.

PHI Complex scalar. Numerical value for $\phi(\bar{x}, \bar{x}')$ from Section III. Computed by subroutine GREEN and depends only on wavelength and distance between $\bar{x}$ and $\bar{x}'$.

$GPHI(I)$ Complex vector. Numerical value for $\nabla\phi(\bar{x}, \bar{x}')$ from Section III. Computed by subroutine GREEN. $I = 1, 2, 3$ for x, y, z components, respectively.

$H(I)$ Complex vector. Numerical value for $\vec{h}[i, n]$ or $\vec{h}[i, n]$ from Section III. Computed by subroutine TEMO for TE_{MO} modes in a rectangular waveguide. Depends on local field boundary position p, s , the mode number M , and on $N = 1, 2$ for $+, \rightarrow$ directions, respectively. $I = 1, 2, 3$ for p, s, n direction components.

ZH(I) Complex vector. Same as H(I) except converted to x,y,z directions by the transformation:

$$\begin{aligned} \text{ZH(I)} &= \text{H(1)}\text{UP(I)} + \text{H(2)}\text{US(I)} + \text{H(3)}\text{UN(I)}; \\ \text{I} &= 1,2,3 \end{aligned}$$

HN(I) Complex vector. Numerical value for $\partial/\partial n \vec{h}[i,n]$ from Section III. Also computed by subroutine TEMO.

ZHN(I) Complex vector. Related to HN(I) in the same way ZH(I) is related to H(I).

ZG Complex scalar. Used to compute and store the numerical value for

$$\int_{[c,k1]} \phi[a,1j] \, da$$

ZGN Complex scalar. Used to compute and store the numerical value for

$$\int_{[c,k1]} [\hat{n} \cdot \nabla \phi[a,1j]] \, da$$

ZF(I) Complex vector. Used to compute and store the numerical value for

$$\int_{S_c} \left\{ \phi[a,1j] \frac{\partial}{\partial n} \vec{h}[c,n] - [\hat{n} \cdot \nabla \phi[a,1j]] \hat{h}[c,n] \right\} da$$

```

Program DoJob
IMPLICIT COMPLEX (Z)
INTEGER A,C,NP(10),NS(10),MODES(2)
REAL V1(10,3),UN(10,3),UP(10,3),US(10,3),DP(10),DS(10)
REAL XT(3),XI(3),W(8),TSI(8),V(3),TS(3)
DIMENSION ZF(3),ZFI(3),ZFO(3),ZH(3),ZHN(3)
COMPLEX GPHI(3),PHI,H(3),HN(3)

```

```

COMMON AA,BB,AK
DATA TPI / 6.283185308 /
DATA SMALL / 2.5E-07 /

```

C Initialize W and TSI arrays for 8 point Gaussian quadrature.

```

TSI(1)=-0.9602898565
TSI(2)=-0.7966664774
TSI(3)=-0.5255324099
TSI(4)=-0.1834346425
TSI(5)=-TSI(4)
TSI(6)=-TSI(3)
TSI(7)=-TSI(2)
TSI(8)=-TSI(1)

```

```

W(1)=0.1012285363
W(2)=0.2223810345
W(3)=0.3137066459
W(4)=0.3626837834
W(5)=W(4)
W(6)=W(3)
W(7)=W(2)
W(8)=W(1)

```

C Initialize TS array for 3 point Gaussian quadrature.

```

TS(1)=-0.7745966692
TS(2)=0.0
TS(3)=-TS(1)

```

```

V(1)=0.5555555556
V(2)=0.8888888889
V(3)=V(1)

```

C Read rectangle specification data as generated by rspec.f .

```

OPEN(10,FILE='rec.dat',STATUS='old')
REWIND(10)
READ(10,*) NR,NRC
DO 10 I=1,NR
READ(10,*) (V1(I,J),J=1,3)
READ(10,*) (UN(I,J),J=1,3)
READ(10,*) (UP(I,J),J=1,3)
READ(10,*) (US(I,J),J=1,3)
READ(10,*) DP(I),DS(I),NP(I),NS(I)

```

10 CONTINUE
CLOSE(10)

C Open file for vector coupling coefficients.

```

OPEN(10,FILE='green.dat',STATUS='unknown')
REWIND(10)

```

```

C      Enter run data.
      WRITE(6,*) 'Enter ratio: working wavelength / cutoff wavelength.'
      READ(5,*) RATIO
      AK=TPI/RATIO
      WRITE(10,*) RATIO

C      Enter number of TE-M0 modes at each field plane.
      NFB=NR-NRC
      WRITE(10,*) NFB
      DO 15 I=1,NFB
      WRITE(6,*) ' '
      WRITE(6,*) 'For field plane number',I,', '
      WRITE(6,*) 'Enter the number of TE modes.'
      READ(5,*) MODES(I)
15     CONTINUE
      WRITE(10,*) (MODES(I),I=1,NFB)

C      Test point loop over rectangular boundaries.
      DO 530 A=1,NR
      WRITE(6,*) 'Doing work for rectangle number',A,'.'
      HXX=DP(A)/FLOAT(NP(A))
      HYY=DS(A)/FLOAT(NS(A))

C      Test point loops over patches on rectangle A.
      DO 530 I=1,NP(A)
      XX=HXX*(FLOAT(I)-0.5)
      DO 530 J=1,NS(A)
      YY=HYY*(FLOAT(J)-0.5)

C      Compute position of test point.
      DO 20 II=1,3
      XT(II)=V1(A,II)+XX*UP(A,II)+YY*US(A,II)
20     CONTINUE

C      Evaluation loop over conducting rectangles.
      DO 170 C=1,NRC
      HX=DP(C)/FLOAT(NP(C))
      HY=DS(C)/FLOAT(NS(C))
      F=HX*HY/4.0

C      Evaluation loops over patches on rectangle C.
      DO 170 K=1,NP(C)
      X=HX*(FLOAT(K)-0.5)
      DO 170 L=1,NS(C)
      Y=HY*(FLOAT(L)-0.5)

C      Initialize integration vector.
      ZG=CMPLX(0.0,0.0)
      ZGN=CMPLX(0.0,0.0)

C      Test for self term; or common patches.
      SUM=0.0
      DO 40 II=1,3
      XI(II)=V1(C,II)+X*UP(C,II)+Y*US(C,II)
      SUM=SUM+(XT(II)-XI(II))**2
40     CONTINUE
      IF(((A.EQ.C).AND.((K.EQ.I).AND.(L.EQ.J))).OR.(SUM.LT.SMALL)) THEN

```

```

C      Do self term integration using 8 by 8 point Gaussian scheme.
      DO 90 M=1,8
      AX=HX*TSI(M)/2.0+X
      ZGI=CMPLX(0.0,0.0)

      DO 70 N=1,8
      AY=HY*TSI(N)/2.0+Y
      DO 50 II=1,3
      XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
50      CONTINUE
      CALL GREEN(XT,XI,PHI)
      ZGI=ZGI+W(N)*PHI
70      CONTINUE

      ZG=ZG+W(M)*ZGI
90      CONTINUE

ELSE
C      Do integration using 3 by 3 point Gaussian scheme.
      DO 150 M=1,3
      AX=HX*TS(M)/2.0+X
      ZGI=CMPLX(0.0,0.0)
      ZGNI=CMPLX(0.0,0.0)

      DO 130 N=1,3
      AY=HY*TS(N)/2.0+Y
      DO 110 II=1,3
      XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
110      CONTINUE
      CALL GREEN(XT,XI,PHI)
      CALL GGREEN(XT,XI,GPHI)
      ZGI=ZGI+V(N)*PHI
      Z=CMPLX(0.0,0.0)
      DO 120 II=1,3
      Z=Z+UN(C,II)*GPHI(II)
120      CONTINUE
      ZGNI=ZGNI+V(N)*Z
130      CONTINUE

      ZG=ZG+V(M)*ZGI
      ZGN=ZGN+V(M)*ZGNI
150      CONTINUE
      ENDIF

      ZG=F*ZG
      ZGN=F*ZGN
      WRITE(10,*) ZG,ZGN
170      CONTINUE

C      Evaluation loop over field boundary rectangles.
      DO 350 C=(NRC+1),NR
      AA=DP(C)
      BB=DS(C)
      HX=AA/FLOAT(NP(C))
      HY=BB/FLOAT(NS(C))
      F=HX*HY/4.0

```

```

C      Evaluation loop over modes.
      IC=C-NRC
      DO 350 MM=1,MODES(IC)

C      Initialize integration sum.
      DO 190 II=1,3
      ZF(II)=CMPLX(0.0,0.0)
190    CONTINUE

C      Start summing loops over patches on field boundary C.
      DO 340 IA=1,NP(C)
      X=HX*(FLOAT(IA)-0.5)
      DO 340 JA=1,NS(C)
      Y=HY*(FLOAT(JA)-0.5)

C      Initialize patch integration.
      DO 200 II=1,3
      ZFO(II)=CMPLX(0.0,0.0)
200    CONTINUE

C      Test for self patch.
      IF((A.EQ.C).AND.((I.EQ.IA).AND.(J.EQ.JA))) THEN

C      Do self term integration using 8 by 8 point Gaussian scheme.
      DO 260 M=1,8
      AX=HX*TSI(M)/2.0+X
      DO 210 II=1,3
      ZFI(II)=CMPLX(0.0,0.0)
210    CONTINUE

      DO 240 N=1,8
      AY=HY*TSI(N)/2.0+Y
      DO 220 II=1,3
      XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
220    CONTINUE

C      Evaluate the integrand.
      CALL GREEN(XT,XI,PHI)
      CALL GGREEN(XT,XI,GPHI)
      CALL TEMO(AX,AY,AA,BB,RATIO,2,MM,H,HN)
      Z=CMPLX(0.0,0.0)
      DO 230 II=1,3
      Z=Z+UN(C,II)*GPHI(II)
      ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
      ZHN(II)=HN(1)*UP(C,II)+HN(2)*US(C,II)+HN(3)*UN(C,II)
230    CONTINUE
      DO 235 II=1,3
      ZFI(II)=ZFI(II)+W(N)*(PHI*ZHN(II)-Z*ZH(II))
235    CONTINUE
240    CONTINUE

      DO 250 II=1,3
      ZFO(II)=ZFO(II)+W(M)*ZFI(II)
250    CONTINUE
260    CONTINUE

```

```

ELSE
C      Do integration using 3 by 3 point Gaussian scheme.
      DO 320 M=1,3
      AX=HX*TS(M)/2.0+X
      DO 270 II=1,3
      ZFI(II)=CMPLX(0.0,0.0)
270      CONTINUE

      DO 300 N=1,3
      AY=HY*TS(N)/2.0+Y
      DO 280 II=1,3
      XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
280      CONTINUE

C      Evaluate the integrand.
      CALL GREEN(XT,XI,PHI)
      CALL GGREEN(XT,XI,GPHI)
      CALL TEMO(AX,AY,AA,BB,RATIO,2,MM,H,HN)
      Z=CMPLX(0.0,0.0)
      DO 290 II=1,3
      Z=Z+UN(C,II)*GPHI(II)
      ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
      ZHN(II)=HN(1)*UP(C,II)+HN(2)*US(C,II)+HN(3)*UN(C,II)
290      CONTINUE
      DO 295 II=1,3
      ZFI(II)=ZFI(II)+V(N)*(PHI*ZHN(II)-Z*ZH(II))
295      CONTINUE
300      CONTINUE

      DO 310 II=1,3
      ZFO(II)=ZFO(II)+V(M)*ZFI(II)
310      CONTINUE
320      CONTINUE
ENDIF

C      Sum up values from each patch integration.
      DO 330 II=1,3
      ZF(II)=ZF(II)+F*ZFO(II)
330      CONTINUE
340      CONTINUE

      WRITE(10,*) (ZF(II),II=1,3)
350      CONTINUE

C      Compute forcing term for each test point.
      C=NRC+1
      AA=DP(C)
      BB=DS(C)
      HX=AA/FLOAT(NP(C))
      HY=BB/FLOAT(NS(C))
      F=HX*HY/4.0

C      Initialize integration sum.
      DO 370 II=1,3
      ZF(II)=CMPLX(0.0,0.0)
370      CONTINUE

```

```

C      Start summing loops over patches on field boundary C.
      DO 520 IA=1, NP(C)
      X=HX*(FLOAT(IA)-0.5)
      DO 520 JA=1, NS(C)
      Y=HY*(FLOAT(JA)-0.5)

C      Initialize patch integration.
      DO 380 II=1,3
      ZFO(II)=CMPLX(0.0,0.0)
380    CONTINUE

C      Test for self patch.
      IF((A.EQ.C).AND.((I.EQ.IA).AND.(J.EQ.JA))) THEN

C          Do self term integration using 8 by 8 point Gaussian scheme.
          DO 440 M=1,8
          AX=HX*TSI(M)/2.0+X
          DO 390 II=1,3
          ZFI(II)=CMPLX(0.0,0.0)
390    CONTINUE

          DO 420 N=1,8
          AY=HY*TSI(N)/2.0+Y
          DO 400 II=1,3
          XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
400    CONTINUE

C          Evaluate the integrand.
          CALL GREEN(XT,XI,PHI)
          CALL GGREEN(XT,XI,GPHI)
          CALL TEMO(AX,AY,AA,BB,RATIO,1,1,H,HN)
          Z=CMPLX(0.0,0.0)
          DO 410 II=1,3
          Z=Z+UN(C,II)*GPHI(II)
          ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
          ZHN(II)=HN(1)*UP(C,II)+HN(2)*US(C,II)+HN(3)*UN(C,II)
410    CONTINUE
          DO 415 II=1,3
          ZFI(II)=ZFI(II)+W(N)*(PHI*ZHN(II)-Z*ZH(II))
415    CONTINUE
420    CONTINUE

          DO 430 II=1,3
          ZFO(II)=ZFO(II)+W(M)*ZFI(II)
430    CONTINUE
440    CONTINUE

      ELSE

C          Do integration using 3 by 3 point Gaussian scheme.
          DO 500 M=1,3
          AX=HX*TS(M)/2.0+X
          DO 450 II=1,3
          ZFI(II)=CMPLX(0.0,0.0)
450    CONTINUE

          DO 480 N=1,3
          AY=HY*TS(N)/2.0+Y

```

```

DO 460 II=1,3
  XI(II)=V1(C,II)+AX*UP(C,II)+AY*US(C,II)
460  CONTINUE

C    Evaluate the integrand.
    CALL GREEN(XT,XI,PHI)
    CALL GGREEN(XT,XI,GPHI)
    CALL TEMO(AX,AY,AA,BB,RATIO,1,1,H,HN)
      Z=CMPLX(0.0,0.0)
      DO 470 II=1,3
        Z=Z+UN(C,II)*GPHI(II)
        ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
        ZHN(II)=HN(1)*UP(C,II)+HN(2)*US(C,II)+HN(3)*UN(C,II)
470  CONTINUE
      DO 475 II=1,3
        ZFI(II)=ZFI(II)+V(N)*(PHI*ZHN(II)-Z*ZH(II))
475  CONTINUE
480  CONTINUE

      DO 490 II=1,3
        ZFO(II)=ZFO(II)+V(M)*ZFI(II)
490  CONTINUE
500  CONTINUE
    ENDIF

C    Sum up values of patch integrations.
    DO 510 II=1,3
      ZF(II)=ZF(II)+F*ZFO(II)
510  CONTINUE
520  CONTINUE

    WRITE(10,*) (ZF(II),II=1,3)
530  CONTINUE

C    Compute and store magnetic field variation on field rectangle C.
    DO 550 C=(NRC+1),NR
      AA=DP(C)
      BB=DS(C)
      HX=AA/FLOAT(NP(C))
      HY=BB/FLOAT(NS(C))

C    Loop over modes.
      IC=C-NRC
      DO 550 MM=1,MODES(IC)

C    Loops over points on rectangle C.
      DO 550 IA=1,NP(C)
        X=HX*(FLOAT(IA)-0.5)
        DO 550 JA=1,NS(C)
          Y=HY*(FLOAT(JA)-0.5)

          CALL TEMO(X,Y,AA,BB,RATIO,2,MM,H,HN)
          DO 540 II=1,3
            ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
540  CONTINUE
          WRITE(10,*) (ZH(II),II=1,3)
550  CONTINUE

```

```

C      Compute and store incident magnetic field variation.
      C=NRC+1
      AA=DP(C)
      BB=DS(C)
      HX=AA/FLOAT(NP(C))
      HY=BB/FLOAT(NS(C))

C      Loops over points on rectangle C.
      DO 570 IA=1, NP(C)
      X=HX*(FLOAT(IA)-0.5)
      DO 570 JA=1, NS(C)
      Y=HY*(FLOAT(JA)-0.5)

      CALL TEM0(X,Y,AA,BB,RATIO,1,1,H,HN)
      DO 560 II=1,3
      ZH(II)=H(1)*UP(C,II)+H(2)*US(C,II)+H(3)*UN(C,II)
560    CONTINUE
      WRITE(10,*) (ZH(II),II=1,3)
570    CONTINUE

      CLOSE(10)
      STOP 'End of Run.'
      END

```

```

SUBROUTINE GREEN(XT,XI,PHI)
REAL XT(3),XI(3)
COMPLEX PHI,Z
COMMON A,B,AK
DATA FPI /12.56637062/
Z=-CMPLX(0.0,1.0)
SPAN=0.0
DO 10 I=1,3
SPAN=SPAN+(XT(I)-XI(I))**2
10 CONTINUE
SPAN=SQRT(SPAN)
Z=Z*AK*SPAN
Z=CEXP(Z)
PHI=Z/(FPI*SPAN)
RETURN
END

```

```

SUBROUTINE GGREEN(XT,XI,GPHI)
REAL XT(3),XI(3),D(3)
COMPLEX GPHI(3),Z,ZJ
COMMON A,B,AK
DATA FPI /12.56637062/
ZJ=CMPLX(0.0,1.0)
SPSQ=0.0
DO 10 I=1,3
D(I)=XT(I)-XI(I)
SPSQ=SPSQ+D(I)**2
10 CONTINUE
SPAN=SQRT(SPSQ)
Z=-ZJ*AK*SPAN
Z=CEXP(Z)
Z=Z/(SPSQ*FPI)
SPAN=1.0/SPAN
ZJ=CMPLX(SPAN,AK)
Z=Z*ZJ
DO 20 I=1,3
GPHI(I)=Z*D(I)
20 CONTINUE
RETURN
END

```

SUBROUTINE TEMO(X,Y,A,B,WL,N,M,H,HN)

This subroutine computes the magnetic field components for TE-M0 modes in a rectangular waveguide. The mode functions are normalized so that each mode carries one watt of power above cutoff and one watt reactive (inductive) below cutoff. The derivative of the magnetic field components in the direction of the outward normal are also computed.

IMPLICIT COMPLEX (Z)
COMPLEX H(3),HN(3)
DATA PI,RMOE / 3.141592654, 376.730 /
ZJ=CMPLX(0.0,1.0)

AK=2.0*PI/WL
AKSQ=AK**2
GM=PI*FLOAT(M)/A
GMSQ=GM**2
WLCO=2.0*A/FLOAT(M)

IF(WL.LT.WLCO) THEN
ZNEW=1.0
ARG=AKSQ-GMSQ
ZB=SQRT(ARG)
ELSE
ZNEW=ZJ
ARG=GMSQ-AKSQ
ZB=-ZJ*SQRT(ARG)
ENDIF

ZBC=CONJG(ZB)
ZA=4.0*ZNEW*GMSQ/(AK*ZBC*A*B*RMOE)
AM0=REAL(CSQRT(ZA))
GMX=GM*X

H(3)=AM0*COS(GMX)
HN(3)=-ZJ*ZB*H(3)
IF(N.EQ.1) HN(3)=-HN(3)

H(2)=CMPLX(0.0,0.0)
HN(2)=CMPLX(0.0,0.0)

H(1)=ZJ*ZB*AM0*SIN(GMX)/GM
HN(1)=-ZJ*ZB*H(1)
IF(N.EQ.1) H(1)=-H(1)

RETURN
END

Program DoMore.f

This program reads the geometrical data from file rec.dat as set up by program rspec.f and the patch coupling data from file green.dat as generated by dojob.f. The overdetermined system of equations is solved in two coupled parts. The type A equations, equations due to testing on the conducting surface, are partially solved for the pulse function amplitudes while holding the mode amplitudes fixed. That is, a square system of equations is built from Eqs. 23, 24, and 25 of the form $A_1 x_1 = b_1$, where b_1 is a linear function of mode amplitudes. Next, the type B equations, equations due to testing on the field boundaries, Eq. 26, are partially solved for the mode amplitudes while holding the pulse function amplitudes fixed. Here we have $A_2 x_2 = b_2$, where b_2 is a linear function of the pulse function amplitudes. The overall iteration takes the following form:

1. Hold the mode amplitudes fixed.
2. Make one ART update on the type A system, $A_1 x_1 = b_1$.
3. Hold the pulse amplitudes fixed.
4. Make one ART update on the type B system, $A_2 x_2 = b_2$.
5. Enforce the conservation of power constraint on the mode amplitudes, Eq. 28.
6. Repeat until the solution vector from the type B equations changes by less than 2 percent.

The ART update on the type A equations is done using subroutine ARTA. This subroutine uses a relaxation factor R that is between 0 and 1. For the asymmetric iris data in Section IV, $R = 1, 1/2, 1/4, 1/8$, with 10

iterations at each value starting with 1. The update on the type B equations was done using subroutine ARTB with no relaxation; i.e., $R = 1$.

1. Conservation of power was enforced by subroutine ECPMO.

The primary program variable definitions are:

A, C	Test point and evaluation point indices, respectively.
V1(A,I)	Absolute (x,y,z) position of first vertex on rectangle A. $I = 1,2,3$ for x,y,z components, respectively.
UN(A,I), UP(A,I) US(A,I)	Unit vector directions n, p, and s.
DP(A), DS(A)	Distance primary and secondary for rectangle A.
NP(A), NS(A)	Number of primary and secondary subdivisions on rectangle A.
MODES(C)	Number of modes on field boundary C.
NR	Total number of rectangles enclosing the volume.
NRC	Number of conducting rectangles out of the total.
NFB	Number of field boundary rectangles.
RATIO	Real value. Ratio of the working wavelength to the cutoff wavelength of the dominant mode on field boundary number 1.
LA, LC	Integer values. Linear index, used for counting and ordering patches on rectangles A, C. Computed from a primary index I and a secondary index J by the following sequence:

```

DO @ I=1,NP(A)
M=NS(A)*(I-1)
DO @ J=1,NS(A)
LA=J+M

```

```

.
.

```

@ CONTINUE

H(C,M,LC,I)	Complex array. Magnetic field distribution due to mode M at point LC on field boundary rectangle C. I = 1,2,3 for x,y,z components. Computed by dojob.f under variable name ZH(I).
HI(L1,I)	Complex array. Magnetic field distribution due to the incident mode at point L1 on field boundary rectangle number 1. I = 1,2,3 for x,y,z components. Computed by dojob.f under variable name ZH(I).
ZG(A,LA,C,LC)	Complex array. Coupling from patch LC on rectangle C to point LA on rectangle A. Its value is computed by dojob.f under the variable name ZG.
ZGN(A,LA,C,LC)	Complex array. Coupling from patch LC to point LA. Value computed by dojob.f under variable name ZGN.
ZF(A,LA,C,M,I)	Complex array. Coupling from mode M on field rectangle C to point LA on rectangle A. I = 1,2,3 for x,y,z components. Computed by dojob.f under variable name ZF(I).

ZFI(A,LA,I) Complex array. Coupling from the incident or driving mode in field rectangle number 1 to point LA on rectangle A. $I = 1,2,3$ for x,y,z components. Computed by dojob.f under variable name ZF(I).

ZH(C,LC,I) Complex array. Solution vector for pulse function amplitudes. Value for patch LC on rectangle C, $I = 1,2,3$ for H_p , H_s , and H'_n .

ZA(C,M) Complex array. Solution vector for mode amplitudes, mode M on field boundary C.

ZHC(C,LC,I) Complex array. Work space, coupling to equation LA from the pulse amplitude coefficient LC on rectangle C, $I = 1,2,3$ for coupling H_p , H_s , and H'_n , respectively.

ZAC(C,M) Complex array. Work space, coupling to equation LA from the amplitude of mode M on field boundary C.

DA(C,LC,I) Real array. Contains denominator terms needed by the ART algorithm for the type A equations. Used to update ZH(C,LC,I). Computed by first access of subroutine ARTA.

DB(C,LC,I) Real array. Contains denominator terms for type B equations. Used to update ZA(C,M). Computed by first access of subroutine ARTB.

Program DoMore

```
IMPLICIT COMPLEX (Z)
INTEGER A,C
REAL V1(10,3),UP(10,3),US(10,3),UN(10,3),DP(10),DS(10)
DIMENSION ZF(10,40,2,8,3),ZFI(10,40,3),ZOLD(2,8)
COMPLEX H(2,8,40,3),HI(40,3)
COMMON / B1 / ZG(10,40,8,40),ZGN(10,40,8,40)
COMMON / B2 / ZH(8,40,3),ZHC(8,40,3),DA(8,40,3)
COMMON / B3 / ZA(2,8),ZAC(2,8),DB(2,40,3)
COMMON / B4 / NP(10),NS(10),NRC,NFB,MODES(2)
```

C Read in rectangle specification data as generated by rspec.f .
OPEN(10,FILE='rec.dat',STATUS='old')
REWIND(10)

```
READ(10,*) NR,NRC
DO 10 I=1,NR
READ(10,*) (V1(I,J),J=1,3)
READ(10,*) (UN(I,J),J=1,3)
READ(10,*) (UP(I,J),J=1,3)
READ(10,*) (US(I,J),J=1,3)
READ(10,*) DP(I),DS(I),NP(I),NS(I)
```

10 CONTINUE
CLOSE(10)

C Read greens function data as generated by dojob.f .
OPEN(10,FILE='green.dat',STATUS='old')

```
REWIND(10)
READ(10,*) RATIO
READ(10,*) NFB
READ(10,*) (MODES(I),I=1,NFB)
```

```
DO 50 A=1,NR
DO 50 I=1,NP(A)
MM=NS(A)*(I-1)
DO 50 J=1,NS(A)
LA=J+MM
```

```
DO 30 C=1,NRC
DO 30 K=1,NP(C)
M=NS(C)*(K-1)
DO 30 L=1,NS(C)
LC=L+M
READ(10,*) ZG(A,LA,C,LC),ZGN(A,LA,C,LC)
```

30 CONTINUE

```
DO 40 C=1,NFB
DO 40 M=1,MODES(C)
READ(10,*) (ZF(A,LA,C,M,II),II=1,3)
```

40 CONTINUE

```
READ(10,*) (ZFI(A,LA,II),II=1,3)
```

50 CONTINUE

```
DO 60 C=1,NFB
IC=C+NRC
DO 60 N=1,MODES(C)
```

```

DO 60 I=1,NP(IC)
M=NS(IC)*(I-1)
DO 60 J=1,NS(IC)
LC=J+M
READ(10,*) (H(C,N,LC,II),II=1,3)
60 CONTINUE

IC=NRC+1
DO 70 I=1,NP(IC)
M=NS(IC)*(I-1)
DO 70 J=1,NS(IC)
LC=J+M
READ(10,*) (HI(LC,II),II=1,3)
70 CONTINUE

CLOSE(10)
WRITE(6,*) 'Done reading data files.'

C Initialize unknown vectors.
DO 80 C=1,NRC
DO 80 I=1,NP(C)
M=NS(C)*(I-1)
DO 80 J=1,NS(C)
LC=J+M
DO 80 II=1,3
ZH(C,LC,II)=CMPLX(0.0,0.0)
80 CONTINUE

DO 90 C=1,NFB
DO 90 M=1,MODES(C)
ZA(C,M)=CMPLX(0.0,0.0)
ZOLD(C,M)=CMPLX(0.0,0.0)
90 CONTINUE

C Initialize iteration control.
ITMAX=10
RESMAX=5.0E-06
NEQ1=0
DO 95 A=1,NRC
NEQ1=NEQ1+NP(A)*NS(A)
95 CONTINUE
NEQ2=0
DO 100 A=(NRC+1),NR
NEQ2=NEQ2+NP(A)*NS(A)
100 CONTINUE
WRITE(6,*) 'Number of complex vector equations =',NEQ1,'+',NEQ2

110 IT=1
WRITE(6,*) 'Enter relaxation factor.'
READ(5,*) RLAX
120 RSUM=0.0

C Start ART iteration for type A equations.
C Test loops over patches on conducting rectangles.
DO 220 A=1,NRC
DO 220 I=1,NP(A)
MM=NS(A)*(I-1)

```

DO 220 J=1,NS(A)
LA=J+MM

C Compute primary component LHS.

ZLHS=CMPLX(0.0,0.0)

DO 130 II=1,3

ZLHS=ZLHS+UP(A,II)*ZFI(A,LA,II)

DO 130 C=1,NFB

DO 130 NM=1,MODES(C)

ZLHS=ZLHS+ZA(C,NM)*UP(A,II)*ZF(A,LA,C,NM,II)

130 CONTINUE

C Compute coefficients coupling to the primary term.

DO 150 C=1,NRC

F1=0.0

F2=0.0

F3=0.0

DO 140 II=1,3

F1=F1+UP(A,II)*UP(C,II)

F2=F2+UP(A,II)*US(C,II)

F3=F3+UP(A,II)*UN(C,II)

140 CONTINUE

DO 150 K=1,NP(C)

M=NS(C)*(K-1)

DO 150 L=1,NS(C)

LC=L+M

ZHC(C,LC,1)=F1*ZGN(A,LA,C,LC)

ZHC(C,LC,2)=F2*ZGN(A,LA,C,LC)

ZHC(C,LC,3)=-F3*ZG(A,LA,C,LC)

150 CONTINUE

ZHC(A,LA,1)=ZHC(A,LA,1)+CMPLX(0.5,0.0)

C Do update to vector ZH.

CALL ARTA(A,LA,1,IT,RSUM,RLAX,ZLHS)

C Compute secondary component LHS.

ZLHS=CMPLX(0.0,0.0)

DO 160 II=1,3

ZLHS=ZLHS+US(A,II)*ZFI(A,LA,II)

DO 160 C=1,NFB

DO 160 NM=1,MODES(C)

ZLHS=ZLHS+ZA(C,NM)*US(A,II)*ZF(A,LA,C,NM,II)

160 CONTINUE

C Compute coefficients coupling to the secondary term.

DO 180 C=1,NRC

F1=0.0

F2=0.0

F3=0.0

DO 170 II=1,3

F1=F1+US(A,II)*UP(C,II)

F2=F2+US(A,II)*US(C,II)

F3=F3+US(A,II)*UN(C,II)

170 CONTINUE

DO 180 K=1,NP(C)

M=NS(C)*(K-1)

DO 180 L=1,NS(C)

```

      LC=L+M
      ZHC(C,LC,1)=F1*ZGN(A,LA,C,LC)
      ZHC(C,LC,2)=F2*ZGN(A,LA,C,LC)
      ZHC(C,LC,3)=-F3*ZG(A,LA,C,LC)
180  CONTINUE
      ZHC(A,LA,2)=ZHC(A,LA,2)+CMPLX(0.5,0.0)

C    Do update to vector ZH.
      CALL ARTA(A,LA,2,IT,RSUM,RLAX,ZLHS)

C    Compute normal component LHS.
      ZLHS=CMPLX(0.0,0.0)
      DO 190 II=1,3
        ZLHS=ZLHS+UN(A,II)*ZFI(A,LA,II)
      DO 190 C=1,NFB
        DO 190 NM=1,MODES(C)
          ZLHS=ZLHS+ZA(C,NM)*UN(A,II)*ZF(A,LA,C,NM,II)
190  CONTINUE

C    Compute coefficients coupling to the normal term.
      DO 210 C=1,NRC
        F1=0.0
        F2=0.0
        F3=0.0
        DO 200 II=1,3
          F1=F1+UN(A,II)*UP(C,II)
          F2=F2+UN(A,II)*US(C,II)
          F3=F3+UN(A,II)*UN(C,II)
200  CONTINUE
        DO 210 K=1,NP(C)
          M=NS(C)*(K-1)
          DO 210 L=1,NS(C)
            LC=L+M
            ZHC(C,LC,1)=F1*ZGN(A,LA,C,LC)
            ZHC(C,LC,2)=F2*ZGN(A,LA,C,LC)
            ZHC(C,LC,3)=-F3*ZG(A,LA,C,LC)
210  CONTINUE

C    Do update to vector ZH.
      CALL ARTA(A,LA,3,IT,RSUM,RLAX,ZLHS)

220  CONTINUE
      RA1=RSUM/FLOAT(3*NEQ1)

C    Start ART iteration for type B equations.
C    Test loops over patches on field boundaries.
230  RSUM=0.0
      DO 300 A=(NRC+1),NR
        IA=A-NRC
        DO 300 I=1,NP(A)
          MM=NS(A)*(I-1)
          DO 300 J=1,NS(A)
            LA=J+MM

C    Compute X component LHS.
      ZLHS=ZFI(A,LA,1)
      IF(IA.EQ.1) ZLHS=ZLHS-(0.5)*HI(LA,1)

```

```

DO 240 C=1,NRC
DO 240 K=1,NP(C)
M=NS(C)*(K-1)
DO 240 L=1,NS(C)
LC=L+M
ZLHS=ZLHS+UN(C,1)*ZH(C,LC,3)*ZG(A,LA,C,LC)
ZLHS=ZLHS-UP(C,1)*ZH(C,LC,1)*ZGN(A,LA,C,LC)
ZLHS=ZLHS-US(C,1)*ZH(C,LC,2)*ZGN(A,LA,C,LC)
240 CONTINUE

C   Compute coefficients coupling to the X component.
DO 250 C=1,NFB
DO 250 NM=1,MODES(C)
ZAC(C,NM)=-ZF(A,LA,C,NM,1)
250 CONTINUE
DO 255 NM=1,MODES(IA)
ZAC(IA,NM)=ZAC(IA,NM)+(0.5)*H(IA,NM,LA,1)
255 CONTINUE

C   Do update to vector ZA.
CALL ARTB(IA,LA,1,IT,RSUM,ZLHS)

C   Compute Y component LHS.
ZLHS=ZFI(A,LA,2)
IF(IA.EQ.1) ZLHS=ZLHS-(0.5)*HI(LA,2)
DO 260 C=1,NRC
DO 260 K=1,NP(C)
M=NS(C)*(K-1)
DO 260 L=1,NS(C)
LC=L+M
ZLHS=ZLHS+UN(C,2)*ZH(C,LC,3)*ZG(A,LA,C,LC)
ZLHS=ZLHS-UP(C,2)*ZH(C,LC,1)*ZGN(A,LA,C,LC)
ZLHS=ZLHS-US(C,2)*ZH(C,LC,2)*ZGN(A,LA,C,LC)
260 CONTINUE

C   Compute coefficients coupling to the Y component.
DO 270 C=1,NFB
DO 270 NM=1,MODES(C)
ZAC(C,NM)=-ZF(A,LA,C,NM,2)
270 CONTINUE
DO 275 NM=1,MODES(IA)
ZAC(IA,NM)=ZAC(IA,NM)+(0.5)*H(IA,NM,LA,2)
275 CONTINUE

C   Do update to vector ZA.
CALL ARTB(IA,LA,2,IT,RSUM,ZLHS)

C   Compute Z component LHS.
ZLHS=ZFI(A,LA,3)
IF(IA.EQ.1) ZLHS=ZLHS-(0.5)*HI(LA,3)
DO 280 C=1,NRC
DO 280 K=1,NP(C)
M=NS(C)*(K-1)
DO 280 L=1,NS(C)
LC=L+M
ZLHS=ZLHS+UN(C,3)*ZH(C,LC,3)*ZG(A,LA,C,LC)
ZLHS=ZLHS-UP(C,3)*ZH(C,LC,1)*ZGN(A,LA,C,LC)

```

```

      ZLHS=ZLHS-US(C,3)*ZH(C,LC,2)*ZGN(A,LA,C,LC)
280  CONTINUE

C    Compute coefficients coupling to the Z component.
      DO 290 C=1,NFB
        DO 290 NM=1,MODES(C)
          ZAC(C,NM)=-ZF(A,LA,C,NM,3)
290  CONTINUE
        DO 295 NM=1,MODES(IA)
          ZAC(IA,NM)=ZAC(IA,NM)+(0.5)*H(IA,NM,LA,3)
295  CONTINUE

C    Do update to vector ZA.
      CALL ARTB(IA,LA,3,IT,RSUM,ZLHS)

300  CONTINUE
      RA2=RSUM/FLOAT(3*NEQ2)
      RA3=(RA1+RA2)/2.0

C    Enforce conservation of power.
      CALL ECPH0(RATIO,DP)

C    Compute fraction change in solution.
      SNUM=0.0
      SDEN=0.0
      DO 305 C=1,NFB
        DO 305 I=1,MODES(C)
          SDEN=SDEN+CABS(ZA(C,I))
          ZDIFF=ZOLD(C,I)-ZA(C,I)
          SNUM=SNUM+CABS(ZDIFF)
          ZOLD(C,I)=ZA(C,I)
305  CONTINUE
      SNUM=SNUM/SDEN
      WRITE(6,*) IT,': Ave error =',RA3,' Soln Chng =',SNUM

      IF(RA1.LE.RESMAX) GOTO 320
      IF(IT.GE.ITMAX) GOTO 310
      IT=IT+1
      GOTO 120

310  WRITE(6,*) 'Hit ITMAX limit; for more its type "1", otherwise "0".'
      READ(5,*) NANS
      IF(NANS.EQ.1) GOTO 110

320  OPEN(10,FILE='domore.dat',STATUS='unknown')
      WRITE(10,*) ' '
      WRITE(10,*) RATIO
      WRITE(10,*) 'EA =',RA1,' EB =',RA2
      DO 330 I=1,NFB
        DO 330 J=1,MODES(I)
          WRITE(10,*) I,J,ZA(I,J)
330  CONTINUE
      CLOSE(10)

      STOP 'End of Run.'
      END

```

```

SUBROUTINE ARTA(A,LA,N,ITEST,RSUM,FAC,ZLHS)
IMPLICIT COMPLEX (Z)
INTEGER A,C
COMMON / B2 / ZH(8,40,3),ZHC(8,40,3),DA(8,40,3)
COMMON / B4 / NP(10),NS(10),NRC,NFB,MODES(2)

```

```

C      Check to see if this is a first pass access.
      IF(ITEST.GE.2) GOTO 20

```

```

C      Compute denominator term.
      DA(A,LA,N)=0.0
      DO 10 C=1,NRC
      DO 10 K=1,NP(C)
      M=NS(C)*(K-1)
      DO 10 L=1,NS(C)
      LC=L+M
      DO 10 II=1,3
      Z=ZHC(C,LC,II)
      DA(A,LA,N)=DA(A,LA,N)+REAL(Z*CONJG(Z))
10     CONTINUE

```

```

C      Compute the real part of the residual.
20     ALFA=0.0
      DO 30 C=1,NRC
      DO 30 K=1,NP(C)
      M=NS(C)*(K-1)
      DO 30 L=1,NS(C)
      LC=L+M
      DO 30 II=1,3
      ZA=ZHC(C,LC,II)
      ZY=ZH(C,LC,II)
      RA=REAL(ZA)
      AA=AIMAG(ZA)
      RY=REAL(ZY)
      AY=AIMAG(ZY)
      ALFA=ALFA+RA*RY-AA*AY
30     CONTINUE
      ALFA=ALFA-REAL(ZLHS)
      RSUM=RSUM+ABS(ALFA)

```

```

C      Update the vector ZH.
      ALFA=FAC*ALFA/DA(A,LA,N)
      ZA=ZHC(A,LA,N)
      ZY=ZH(A,LA,N)
      RA=REAL(ZA)
      AA=AIMAG(ZA)
      RY=REAL(ZY)
      AY=AIMAG(ZY)
      RY=RY-ALFA*RA
      AY=AY+ALFA*AA
      ZH(A,LA,N)=CMPLX(RY,AY)

```

```

C      Compute the imaginary part of the residual.
      BETA=0.0
      DO 50 C=1,NRC
      DO 50 K=1,NP(C)
      M=NS(C)*(K-1)

```

AD-A182 414

A THREE-DIMENSIONAL VECTOR BOUNDARY ELEMENT FORMULATION
FOR MULTI-PORT SCA. (U) UTAH UNIV SALT LAKE CITY DEPT OF
ELECTRICAL ENGINEERING M L TRACY FEB 87 UTEC-MD-86-867
RADC-TR-87-24 F30602-82-C-0161

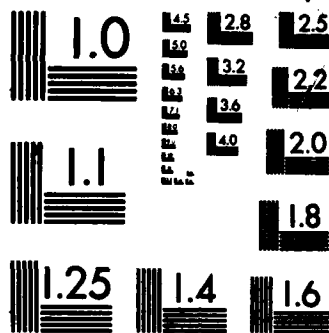
2/2

UNCLASSIFIED

F/G 20/3

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```

DO 50 L=1,NS(C)
LC=L+M
DO 50 II=1,3
ZA=ZHC(C,LC,II)
ZY=ZH(C,LC,II)
RA=REAL(ZA)
AA=AIMAG(ZA)
RY=REAL(ZY)
AY=AIMAG(ZY)
BETA=BETA+AA*RY+RA*AY
50 CONTINUE
BETA=BETA-AIMAG(ZLHS)
RSUM=RSUM+ABS(BETA)

C Update the vector ZH.
BETA=FAC*BETA/DA(A,LA,N)
ZA=ZHC(A,LA,N)
ZY=ZH(A,LA,N)
RA=REAL(ZA)
AA=AIMAG(ZA)
RY=REAL(ZY)
AY=AIMAG(ZY)
RY=RY-BETA*AA
AY=AY-BETA*RA
ZH(A,LA,N)=CMPLX(RY,AY)

RETURN
END

```

```

SUBROUTINE ARTB(IA,LA,N,ITEST,RSUM,ZLHS)
IMPLICIT COMPLEX (Z)
INTEGER C
COMMON / B3 / ZA(2,8),ZAC(2,8),DB(2,40,3)
COMMON / B4 / NP(10),NS(10),NRC,NFB,MODES(2)
DATA FAC / 1.0 /

```

```

C      Check to see if this is a first pass access.
      IF(ITEST.GE.2) GOTO 20

```

```

C      Compute denominator term.
      DB(IA,LA,N)=0.0
      DO 10 C=1,NFB
      DO 10 NM=1,MODES(C)
      Z=ZAC(C,NM)
      DB(IA,LA,N)=DB(IA,LA,N)+REAL(Z*CONJG(Z))
10     CONTINUE

```

```

C      Compute the real part of the residual.
20     ALFA=0.0
      DO 30 C=1,NFB
      DO 30 NM=1,MODES(C)
      ZB=ZAC(C,NM)
      ZY=ZA(C,NM)
      RA=REAL(ZB)
      AA=AIMAG(ZB)
      RY=REAL(ZY)
      AY=AIMAG(ZY)
      ALFA=ALFA+RA*RY-AA*AY
30     CONTINUE
      ALFA=ALFA-REAL(ZLHS)
      RSUM=RSUM+ABS(ALFA)

```

```

C      Update the vector ZA.
      ALFA=FAC*ALFA/DB(IA,LA,N)
      C=IA
      DO 40 NM=1,MODES(C)
      ZB=ZAC(C,NM)
      ZY=ZA(C,NM)
      RA=REAL(ZB)
      AA=AIMAG(ZB)
      RY=REAL(ZY)
      AY=AIMAG(ZY)
      RY=RY-ALFA*RA
      AY=AY+ALFA*AA
      ZA(C,NM)=CMPLX(RY,AY)
40     CONTINUE

```

```

C      Compute the imaginary part of the residual.
      BETA=0.0
      DO 50 C=1,NFB
      DO 50 NM=1,MODES(C)
      ZB=ZAC(C,NM)
      ZY=ZA(C,NM)
      RA=REAL(ZB)
      AA=AIMAG(ZB)
      RY=REAL(ZY)

```

```

    AY=AIMAG(ZY)
    BETA=BETA+AA*RY+RA*AY
50  CONTINUE
    BETA=BETA-AIMAG(ZLHS)
    RSUM=RSUM+ABS(BETA)

C    Update the vector ZA.
    BETA=FAC*BETA/DB(IA,LA,N)
    C=IA
    DO 60 NM=1,MODES(C)
    ZB=ZAC(C,NM)
    ZY=ZA(C,NM)
    RA=REAL(ZB)
    AA=AIMAG(ZB)
    RY=REAL(ZY)
    AY=AIMAG(ZY)
    RY=RY-BETA*AA
    AY=AY-BETA*RA
    ZA(C,NM)=CMPLX(RY,AY)
60  CONTINUE

    RETURN
    END

```

```

SUBROUTINE ECPH0(WL,DP)
IMPLICIT COMPLEX (Z)
DIMENSION DP(10),NM(2)
COMMON / B3 / ZA(2,8),ZAC(2,8),DB(2,40,3)
COMMON / B4 / NP(10),NS(10),NRC,NFB,MODES(2)

```

C Find out how many modes can propagate at each field plane.

```

DO 10 I=1,NFB
IA=NRC+I
A=DP(IA)
NM(I)=IFIX(2.0*A/WL)
IF(NM(I).GT.MODES(I)) NM(I)=MODES(I)

```

10 CONTINUE

C Compute the adjustment factor.

```

FAC=0.0
DO 20 I=1,NFB
IF(NM(I).EQ.0) GOTO 20
DO 20 J=1,NM(I)
Z=ZA(I,J)
FAC=FAC+REAL(Z*CONJG(Z))

```

20 CONTINUE

```

FAC=SQRT(FAC)

```

C Adjust the solution vector to force conservation of power.

```

DO 30 I=1,NFB
IF(NM(I).EQ.0) GOTO 30
DO 30 J=1,NM(I)
ZA(I,J)=ZA(I,J)/FAC

```

30 CONTINUE

```

RETURN
END

```

REFERENCES

1. S. Kagami and I. Fukai, "Application of Boundary-Element Method to Electromagnetic Field Problems," IEEE Transactions on Microwave Theory and Techniques, Vol. 32, No. 4, April 1984, pp. 455-461.
2. E. Tonye and H. Baudrand, "Multimode S-Parameters of Planar Multiport Junctions by Boundary Element Method," Electronics Letters, Vol. 20, No. 19, September 1984, pp. 799-802.
3. C. A. Brebbia, The Boundary Element Method for Engineers, Chapter 3, Pentech Press, London, England, 1978.
4. T. A. Mielke, "Solution of the Generalized Waveguide Discontinuity Problem," AFTER Thesis, Stanford University, Stanford, California, October 1983.
5. J. M. Neilson, "Field Theory Analysis of the Pill Box Window," AFTER Thesis, University of Utah, Salt Lake City, Utah, February 1985.
6. B. G. Braatz, "Computer Analysis of Arbitrarily Tapered Waveguides with Double-Ridged Cross Sections," AFTER Thesis, University of Utah, Salt Lake City, Utah, 1985.
7. L. Solymar, "Spurious Mode Generation in Nonuniform Waveguide," IRE Transactions on Microwave Theory and Techniques, Vol. 7, July 1959, pp. 379-383.
8. I. B. Bernstein, L. K. Divringi, and T. M. Smith, "The Theory of Irregular Waveguides and Open Resonators," International Journal of Infrared and Millimeter Waves, vol. 4, No. 1, 1983.
9. R. F. Harrington, Field Computation by Moment Methods, Syracuse University, Syracuse, New York, 1968.
10. G. Strang and G. J. Fix, An Analysis of the Finite Element Method, Prentice Hall, Inc., Englewood Cliffs, New Jersey, Section 1.9, 1973.
11. A. Ralston and P. Rabinowitz, A First Course in Numerical Analysis, McGraw-Hill Book Co., Inc., New York, Section 9.7, 1978.
12. D. M. Kerns, "Analysis of Symmetrical Waveguide Junctions," Journal of Research of the National Bureau of Standards, Vol. 46, No. 4, April 1951, pp. 267-282.
13. B. E. Spielman, "Waveguides of Arbitrary Cross Section by Solution of a Nonlinear Integral Eigenvalue Equation," Technical Report TR-71-1, Syracuse University, Syracuse, New York, January 1971.

14. Microwave Engineers' Handbook, compiled and edited by T. S. Saad, Artech House, Inc., Dedham, Massachusetts, Vol. 1, 1971, p. 81.
15. N. Maracuvits, Waveguide Handbook, McGraw-Hill Book Company, Inc., New York, 1951.
16. Handbook of Mathematical Functions, edited by M. Abramowitz and I. A. Stegun, National Bureau of Standards, Applied Mathematics Series 55, June 1964.

*MISSION
of
Rome Air Development Center*

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control, Communications and Intelligence (C³I) activities. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C³I systems. The areas of technical competence include communications, command and control, battle management, information processing, surveillance sensors, intelligence data collection and handling, solid state sciences, electromagnetics, and propagation, and electronic, maintainability, and compatibility.

END

8-87

DTIC