RL-857

SIMPLE INTEGRAL EQUATIONS FOR TWO-DIMENSIONAL SCATTERING WITH FURTHER REDUCTION IN UNKNOWNS AND SCATTERING CODE USER'S MANUAL

M.A. Ricoy, L.C. Kempel And J.L. Volakis

Radiation Laboratory
Department of Electrical Engineering
and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109-2122

Original publication December 1988

Revised June 1990

RL-857

Simple Integral Equations for Two-Dimensional Scattering with Further Reduction in Unknowns and Scattering Reduction in Unknowns

M.A. Ricoy, L.C. Kempel and J.L. Volakis

Original Publication December 1988
Revised June 1990

SIMPLE INTEGRAL EQUATIONS FOR TWO-DIMENSIONAL SCATTERING WITH FURTHER REDUCTION IN UNKNOWNS

M.A. Ricoy, L.C. Kempel and J.L. Volakis

Radiation Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109-2122

Abstract- A simple set of integral equations with reduced unknowns and kernel singularity are derived for simulating arbitrarily-shaped two-dimensional inhomogeneous composite scatterers. By utilizing a known equivalence between electric and magnetic currents, new equivalent currents are introduced with the resultant integral formulation exhibiting a volume integral in addition to a surface integral, each in terms of a single equivalent current component. A pulse basis-point matching moment method implementation of the reduced unknown integral equations is presented. Scattering patterns computed with the numerical code are compared with results obtained via alternate analytical techniques.

TABLE OF CONTENTS

I. INTRODUCTION

The computation of the scattered electromagnetic field from two-dimensional inhomogeneous dielectric/magnetic structures has traditionally been accomplished via a numerical solution of the appropriate integral equations. Over the past twenty-five years, a variety of volume and surface integral formulations have been developed and applied to cylinders of arbitrary cross-section and composition [1] - [5], as well as specialized cylindrical configurations [6] - [7]. A point of commonality among these techniques is the necessity of solving for three independent current components within a material body having a nontrivial permittivity and permeability profile. Thus, for a total of N cells comprising the discretized composite structure, the resultant total number of system unknowns is 3N. As is well known, the expenditure of computer resources (CPU time and memory allocation) for numerical codes is proportional to M^{α}, where M denotes the total number of unknowns and $\alpha > 1$. For example, the required CPU time varies as M^3 in the case of non-optimized moment method formulations, and thus any reduction in M will produce a significant improvement in the solution algorithm's computational efficiency and economy.

Recently, Ricoy and Volakis [8] presented an integral formulation for modeling inhomogeneous composite cylinders of arbitrary cross-section utilizing only two current components across the volume of the scatterer. In this paper, a simple integral equation set which accomplishes a further reduction in unknowns is developed. By replacing the traditional polarization currents with new equivalent currents, the resultant integral equation exhibits a volume integral in addition to a surface integral, each in terms of a single independent equivalent current component. Consequently, this implies a substantially reduced system with only $N^v + N^s$ unknowns, where N^v is the number of volumetric sampling points and N^S is the number of surface sampling points. In contrast to the development in [8], this

reduction of unknowns is obtained without an increase in the singularity or complexity of the derived integral equations.

To illustrate the validity and applicability of the volume/surface integral formulation, a general numerical code was developed using a pulse basis-point matching moment method procedure. A variety of scattering patterns computed using this code are presented and compared to results obtained by alternate analytical techniques.

II. FORMULATION

From Stratton [9], the electric and magnetic scattered fields can be expressed as (an $e^{-i\omega t}$ time dependence is assumed and suppressed)

$$\bar{E}^s = \nabla\nabla\cdot\bar{\Pi}^e + k_o^2\bar{\Pi}^e + ik_oZ_o\nabla\times\bar{\Pi}^m \tag{1}$$

$$\bar{H}^s = \nabla\nabla\cdot\bar{\Pi}^m + k_o^2\bar{\Pi}^m - ik_oY_o\nabla\times\bar{\Pi}^e \tag{2}$$

with

$$\bar{\Pi}^e = \frac{iZ_o}{k_o}\int\int_{A'}\bar{J}(\bar{r}')\,G(\bar{r},\bar{r}')\,dA' \tag{3a}$$

$$\bar{\Pi}^m = \frac{iY_o}{k_o}\int\int_{A'}\bar{M}(\bar{r}')\,G(\bar{r},\bar{r}')\,dA' \tag{3b}$$

In the above equations, $\bar{\Pi}^e$ and $\bar{\Pi}^m$ are the Hertz potentials, $\bar{J}$ and $\bar{M}$ denote the electric and magnetic currents generating the scattered fields, k_o is the free space wave number and $Z_o = \frac{1}{Y_o}$ is the free space intrinsic impedance. Additionally,

$$G(\bar{r},\bar{r}') = \frac{i}{4}H_o^{(1)}(k_o\rho) \tag{4}$$

is the two-dimensional Green's function with

$$\rho = |\bar{r} - \bar{r}'|, \tag{5}$$

where $\bar{r}$ and $\bar{r}'$ are the source and observation position vectors as shown in Figure 1.

It is further assumed that the infinite dimension of the scatterer is along the z-direction and thus the direction of the ∇ operator is transverse to $\hat{z}$.

If we are interested in the scattered field by a composite structure due to a given excitation, then $\bar{J}$ and $\bar{M}$ appearing in (3) become the traditional polarization currents [10]

$$\bar{J} = -ik_o Y_o (\varepsilon_r - 1)\, \bar{E} \tag{6a}$$

$$\bar{M} = -ik_o Z_o (\mu_r - 1)\, \bar{H} \tag{6b}$$

where $(\bar{E}, \bar{H})$ represent the total fields within the scatterer. The relative permittivity and permeability of the composite medium are denoted by ε_r and μ_r, respectively, and here on these will refer to the observation point only. The integral equations for determining $\bar{J}$ and $\bar{M}$ can now be generated by employing the relations

$$\bar{E}^s + \bar{E}^i = \bar{E} \tag{7a}$$

$$\bar{H}^s + \bar{H}^i = \bar{H} \tag{7b}$$

within the scatterer, where $(\bar{E}^i, \bar{H}^i)$ denote the source or incident fields.

Since we are concerned with a two-dimensional problem we may now assume H_z-incidence (TE incidence) without loss of generality for generating the integral equations implied by (7). The corresponding integral expressions for E_z-incidence will then be obtained via duality.

For the case of H_z-incidence, the traditional polarization currents take the form

$$\bar{J} = \hat{x} J_x + \hat{y} J_y , \quad \bar{M} = \hat{z} M_z \tag{8}$$

Thus, by substituting (1), (2) and (8) into (7a) we obtain

$$\bar{E}^i(\bar{r}) = Z_o \frac{j}{k_o}\left[\frac{\varepsilon_r}{\varepsilon_r - 1}\right]\bar{J}(\bar{r}) - \nabla \times \left\{ \frac{iZ_o}{k_o} \iint_{A'} \nabla \times \left[\bar{J}(\bar{r}')\, G(\bar{r}, \bar{r}')\right] dA' \right\}$$

$$+ \nabla \times \iint_{A'} \hat{z}\, M_z(\bar{r}')\, G(\bar{r}, \bar{r}')\, dA' \tag{9}$$

where we have also employed the identities

$$\nabla \times \nabla \times \bar{B}(\bar{r}) = \nabla\nabla \cdot \bar{B}(\bar{r}) - \nabla^2 \bar{B}(\bar{r}) \tag{10}$$

$$(\nabla^2 + k_o^2)\, G(\bar{r}, \bar{r}') = -\delta(\bar{r} - \bar{r}'), \quad \bar{r} \in A' \tag{11}$$

in the given sequence. In the above, $\delta(\bar{r})$ denotes the usual Dirac delta function.

Proceeding with similar substitutions into (7b) we obtain

$$-\frac{i}{k_o}\nabla \times \bar{E}^i = \frac{i}{k_o(\mu_r - 1)}\bar{M} - ik_o \int\int_{A'} \bar{M}(\bar{r}')\, G(\bar{r}, \bar{r}')\, dA' - Z_o \nabla \times \int\int_{A'} \bar{J}(\bar{r}')\, G(\bar{r}, \bar{r}')\, dA' \tag{12}$$

The coupled equations (9) and (12) represent one form of the standard integral equations employed in the solution of the scattered field by an inhomogeneous scatterer. As seen, in the case of a general two-dimensional composite scatterer they involve three unknown current components per scatterer location. Clearly, any reduction in the number of unknowns is very desirable when considering their numerical implementation. Also, of equal importance is the singularity associated with the kernel of the integrals to be evaluated in such an implementation. The purpose in the sections to follow is to address both of the above issues.

III. REDUCTIONS IN UNKNOWNS

In this section we develop a new set of integral equations involving fewer unknown current components per scatterer location than those quoted above. This is accomplished via a manipulation of the coupled set of integral equations to allow the introduction of a new equivalent current component to replace two of those now appearing in (9) and (12). In proceeding to do so, let us first concentrate on (8).

By twice employing the identity

$$\nabla \times (\bar{B}f) = \nabla f \times \bar{B} + f\nabla \times \bar{B} \tag{13}$$

in the first integral of (9) we obtain

$$\bar{E}^{i}(\bar{r}) = Z_o \frac{i}{k_o}\left(\frac{\varepsilon_r}{\varepsilon_r - 1}\right)\bar{J}(\bar{r}) + \nabla \times \left\{ \frac{iZ_o}{k_o} \int\int_{A'} \nabla' \times \left[\bar{J}(\bar{r}')\, G(\bar{r}, \bar{r}')\right] dA' \right\}$$

$$+ \int\int_{A'} \nabla \times \left\{ \left[\hat{z}\, M_z - \frac{iZ_o}{k_o} \nabla' \times \bar{J}(\bar{r}') \right] G(\bar{r}, \bar{r}') \right\} dA' \qquad (14)$$

It is now instructive to define the new equivalent magnetic current [11]

$$\tilde{M}_z(\bar{r}') = M_z(\bar{r}') - \frac{iZ_o}{k_o} \hat{z} \bullet \nabla' \times \bar{J}(\bar{r}') \qquad (15)$$

and seek a modification of (14) and (12) for a more global use of (15). With this in mind, we multiply (14) by $-\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right)$ and take its curl to obtain

$$-\nabla \times \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \bar{E}^{i}(\bar{r}) \right) = -\frac{iZ_o}{k_o} \nabla \times \bar{J}(\bar{r}) - \frac{iZ_o}{k_o} \nabla \times \int_{C'} \frac{\varepsilon_r - 1}{\varepsilon_r} \nabla \times \left[\hat{n}' \times \bar{J}(\bar{r}')\, G(\bar{r}, \bar{r}')\right] dl'$$

$$- \nabla \times \int\int_{A'} \frac{\varepsilon_r - 1}{\varepsilon_r} \nabla \times \left[\hat{z}\, \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}')\right] dA' \qquad (16)$$

after also applying the vector Stoke's theorem [12] in the first integral of (14). Accordingly, C' is the contour enclosing the area A' and $\hat{n}'$ denotes the outward unit normal to C' (see figure 1).

To eliminate the presence of $\nabla \times \bar{J}$ in (16), we now return to (12). By employing identity (13) in the second integral of that equation and invoking the definition (15) we find

$$-\frac{i}{k_o} \nabla \times \bar{E}^{i}(\bar{r}) = \hat{z} \frac{i}{k_o(\mu_r - 1)} M_z(\bar{r}) - \hat{z}\, ik_o \int\int_{A'} \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}')\, dA' + Z_o \int\int_{A'} \nabla' \times \left[\bar{J}(\bar{r})\, G(\bar{r}, \bar{r}')\right] dA' \qquad (17)$$

Further, applying as before the vector Stoke's theorem in the last integral of (17) we deduce

$$- (\mu_r - 1)\, \nabla \times \bar{E}^{\,i}(\bar{r}) = \hat{z}\, M_z(\bar{r}) - \hat{z}\, k_o^2 (\mu_r - 1) \int\int_{A'} \tilde{M}_z(\bar{r}\,')\, G(\bar{r}, \bar{r}')\, dA'$$

$$- i Z_o k_o (\mu_r - 1) \int_{C'} \left[n' \times \bar{J}(\bar{r}') \right] G(\bar{r}, \bar{r}')\, dl' \qquad (18)$$

By adding (16) and (18) we finally obtain the scalar integral equation

$$\left[- \nabla \times \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \bar{E}^{\,i}(\bar{r}) \right) - (\mu_r - 1)\, \nabla \times \bar{E}^{\,i}(\bar{r}) \right] \bullet \hat{z} = \tilde{M}_z(\bar{r})$$

$$- k_o^2 \int\int_{A'} \hat{z} \bullet \left[(\mu_r - 1) + \frac{1}{k_o^2} \nabla \times \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \nabla \times \right) \right] \hat{z}\, \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}')\, dA'$$

$$- i k_o \int_{C'} \hat{z} \bullet \left[(\mu_r - 1) + \frac{1}{k_o^2} \nabla \times \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \nabla \times \right) \right] \hat{z}\, \tilde{J}_z(\bar{r}')\, G(\bar{r}, \bar{r}')\, dl' \qquad (19)$$

where we have set

$$\tilde{J}_z(\bar{r}\,') = Z_o\, \hat{z} \bullet \left[\hat{n}' \times \bar{J}(\bar{r}\,') \right] \qquad (20)$$

in accordance with (8).

We observe that (19) involves only one unknown magnetic current density component ($\tilde{M}_z$) across the cross-section of the scatterer and another electric current density component ($\tilde{J}_z$) just over the boundary of the inhomogeneous scatterer. A similar conclusion could also be reached by examining the integral equation derived recently by Tai [13] using a different procedure. It should be emphasized, though that $\tilde{J}_z$ is not a surface current sheet, but simply the associated current density at the locations coinciding with the scatterer's boundary. From (7a), (18) and (20), the scattered field in terms of these new unknowns is given by

$$H_z^s = + i k_o Y_o \int\int_{A'} \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}\,')\, dA' - Y_o \int_{C'} \tilde{J}_z(\bar{r}')\, G(\bar{r}, \bar{r}')\, dl' \qquad (21)$$

Clearly, the above formulation implies a substantial reduction in unknowns when compared with the coupled set (9) and (12). However, the reduction in unknowns appears to have been achieved at the expense of some complexity. As will be shown though in the next section, the kernel of the integrals in (19) can be substantially simplified leading to one of lower singularity and thus more attractive for numerical implementation.

IV. REDUCTION OF KERNEL SINGULARITY

The integral equation (19) is attractive because it involves less unknowns. However, its numerical implementation could indeed be difficult due to the complexity and high singularity associated with the kernels of the relevant integrals. The integrals in question are

$$\bar{I}_1 = \frac{-i}{k_o} \int_{C'} \nabla x \left\{ \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \right) \nabla x \left[\hat{z}\, \tilde{J}_z(\bar{r}')\, G(\bar{r}, \bar{r}') \right] \right\} dl' \tag{22}$$

and

$$\bar{I}_2 = - \int\int_{A'} \nabla x \left\{ \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \right) \nabla x \left[\hat{z}\, \tilde{M}_z(\bar{r}, \bar{r}') \right] \right\} dA' \tag{23}$$

and because of their apparent similarity, it will suffice to consider the simplification of $\bar{I}_2$ only.

By employing (13), $\bar{I}_2$ can be written as

$$\bar{I}_2 = - \iint_{A'} \nabla \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \right) x \nabla x \left[\hat{z}\, \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}) \right] dA' - \frac{(\varepsilon_r - 1)}{\varepsilon_r} \iint_{A'} \nabla x \nabla x \left[\hat{z}\, \tilde{M}_z(\bar{r}')\, G(\bar{r}, \bar{r}') \right] dA' \tag{24}$$

Since ∇ is transverse to $\hat{z}$, the integrand in the first integral of (24) can be simplied as

$$\nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \times \nabla \times \left[\hat{z}\,\tilde{M}_z(\bar{r}')\,G(\bar{r},\bar{r}')\right] = \nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \times \nabla G(\bar{r},\bar{r}') \times \hat{z}\,\tilde{M}_z(\bar{r}')$$

$$= -\left[\nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \bullet \nabla G(\bar{r},\bar{r}')\right] \hat{z}\,\tilde{M}_z(\bar{r}') \qquad (25)$$

after first employing (13) followed by the use of a standard vector identity. In case of the second integral in (24) we find that

$$\nabla \times \nabla \times \left[\hat{z}\,\tilde{M}_z(\bar{r}')\,G(\bar{r},\bar{r}')\right] = -\nabla^2\left[\hat{z}\,\tilde{M}_z(\bar{r}')\,G(\bar{r},\bar{r}')\right]$$

$$= \hat{z}\,\tilde{M}_z(\bar{r}')\left[k_o^2\,G(\bar{r},\bar{r}') + \delta(\bar{r} - \bar{r}')\right] \qquad (26)$$

after first employing (10) followed by the use of (11). Substituting, (25) and (26) into (24) we have

$$\bar{I}_2 = -\hat{z}\frac{(\varepsilon_r - 1)}{\varepsilon_r}\tilde{M}_z(\bar{r}) + \hat{z}\iint_{A'} \tilde{M}_z(\bar{r}')\left[\nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \bullet \nabla G(\bar{r},\bar{r}')\right] dA' - \frac{\hat{z}\,k_o^2(\varepsilon_r - 1)}{\varepsilon_r}\iint_{A} \tilde{M}_z(\bar{r}')\,G(\bar{r},\bar{r}')dA' \qquad (27)$$

Similarly, $\bar{I}_1$ can be rewritten as

$$\bar{I}_1 = \hat{z}\frac{i}{k_o}\int_{C'} \tilde{J}_z(\bar{r}')\left[\nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \bullet \nabla G(\bar{r},\bar{r}')\right] dl' - \frac{\hat{z}\,ik_o(\varepsilon_r - 1)}{\varepsilon_r}\int_{C'} \tilde{J}_z(\bar{r}')\,G(\bar{r},\bar{r}')\,dl' \qquad (28)$$

since $\nabla^2 G(\bar{r},\bar{r}') = -k_o^2 G(\bar{r},\bar{r}')$ on C'.

Using (27) and (28), the integral equation (19) can be written as

$$\left[-\nabla\left(\frac{\varepsilon_r - 1}{\varepsilon_r}\right) \times \bar{E}^i(\bar{r}) - \left(\frac{\varepsilon_r\mu_r - 1}{\varepsilon_r}\right)\nabla \times \bar{E}^i(\bar{r})\right] \bullet \hat{z} = \left(\frac{1}{\varepsilon_r}\right)\tilde{M}_z(\bar{r})$$

$$-ik_o \iint_{A'} \tilde{M}_z(\bar{r}')\,K(\bar{r},\bar{r}')\,dA' + \int_{C'} \tilde{J}_z(\bar{r}')\,K(\bar{r},\bar{r}')\,dl' \qquad (29)$$

with

$$K(\bar{r},\bar{r}') = \left[\frac{-ik_o(\varepsilon_r \mu_r - 1)}{\varepsilon_r} + \frac{i}{k_o} \nabla \left(\frac{\varepsilon_r - 1}{\varepsilon_r} \right) \bullet \nabla \right] G(\bar{r},\bar{r}') \qquad (30)$$

As seen, the singularity of $K(\bar{r},\bar{r}')$ is only $O\left(\frac{1}{\rho}\right)$. In comparison, the singularity of the kernels associated with (19) was $O\left(\frac{1}{\rho^2}\right)$.

The numerical implementation of (29) is now a straightforward task. However, the condition of the resulting system matrix may be further improved by utilizing an auxiliary integral equation when testing on C', i.e. the surface of the scatterer. One such integral equation can be obtained from (16) by removing the indicated curl operation. This gives

$$\bar{E}^i(r) = \frac{i\varepsilon_r Z_o}{k_o(\varepsilon_r - 1)} \bar{J}(\bar{r}) + \frac{i}{k_o} \int_{C'} \nabla \times \left[\hat{z}\, \tilde{J}_z(\bar{r}')\, G(\bar{r},\bar{r}') \right] dl' + \iint_{A'} \nabla \times \left[\hat{z}\, \tilde{M}_z(\bar{r}')\, G(\bar{r},\bar{r}') \right] dA'$$

Alternatively, we may write

$$\hat{z} \bullet [\hat{n} \times \bar{E}^i(\bar{r})] = \frac{i\varepsilon_r}{k_o(\varepsilon_r - 1)} \tilde{J}_z(\bar{r}) + \frac{i}{k_o} \int_{C'} \tilde{J}_z(\bar{r}') \left\{ \hat{z} \bullet \hat{n} \times \left[\nabla G(\bar{r},\bar{r}') \times \hat{z} \right] \right\} dl'$$

$$+ \iint_{A'} \tilde{M}_z(\bar{r}') \left\{ \hat{z} \bullet \hat{n} \times \left[\nabla G(\bar{r},\bar{r}') \times \hat{z} \right] \right\} dA' \qquad (31)$$

after employing (13) and the implied vector products.

The above integral equation can now be employed in place of (29) for generating the system equations corresponding to the testing points on the surface of the scatterer.

V. NUMERICAL IMPLEMENTATION

In proceeding with a moment method solution of the integral equations (29) and (31), the cross-section of the two-dimensional cylindrical scatterer is subdivided into a discrete number of rectangular and non-rectangular cells (see figure 2) totaling N^v for the entire material body. Additionally, the exterior contour of the scatterer and

any interior boundries, where step-discontinuities in permittivity or permeability occur, are discretized in order to generate the required contour segments totaling N^s. Since neither integral expression involves derivatives of the unknown equivalent magnetic or electric currents, a pulse basis-point matching procedure is implemented by assuming the appropriate current density to be constant within each cell/segment and by subsequently satisfying the derived integral equations at the centroid/center of each volume cell/contour segment. The generated system of N^v+ N^s equations can be represented in matrix form as

$$\begin{bmatrix} V_\ell^v \\ V_\ell^s \end{bmatrix} = \begin{bmatrix} \left[F_{\ell m}^1\right] & \left[F_{\ell m}^2\right] \\ \left[F_{\ell m}^3\right] & \left[F_{\ell m}^4\right] \end{bmatrix} \begin{bmatrix} \tilde{M}_{z\ell} \\ \tilde{J}_{z\ell} \end{bmatrix} \tag{32}$$

where $\tilde{M}_{z\ell}$ $\ell = 1,2,....,N^v$ denotes the value of the equivalent magnetic current density at the ℓ^{th} volume cell and $\tilde{J}_{z\ell}$ $\ell = N^v + 1, ..., N^v + N^s$ denotes the equivalent electric current density at the ℓ^{th} contour segment. The excitation vector elements of the system (32) are given by

$$V_\ell^v = -ik_o\left(\mu_r - \frac{1}{\varepsilon_r}\right) Z H_z^i(\bar{r}_\ell) + \left(\frac{\partial}{\partial s}\frac{1}{\varepsilon_r}\right) E_n^i(\bar{r}) - \left(\frac{\partial}{\partial n}\frac{1}{\varepsilon_r}\right) E_s^i(\bar{r}_\ell)$$

for $\ell = 1, 2....N^v$

$$V_\ell^s = -E_s^i(\bar{r}_\ell)$$

for $\ell = N^v + 1, ..., N^v + N^s$. (33)

where $\bar{r}_\ell$ denotes the position vector associated with the centroid/center of the observation element. Finally, the four impedance matrix elements can be expressed as

$$F^1_{\ell m} = \frac{1}{\varepsilon_r}\,\delta_{\ell m} - k_o^2 \iint_{A_m} \left\{ \left(\mu_r - \frac{1}{\varepsilon_r}\right) + \frac{1}{k_o^2}\left[\left(\frac{\partial}{\partial s}\,\frac{1}{\varepsilon_r}\right)\frac{\partial}{\partial s} + \left(\frac{\partial}{\partial n}\,\frac{1}{\varepsilon_r}\right)\frac{\partial}{\partial n} \right] \right\} G(\bar{r}_\ell, \bar{r}')\, ds'dn'$$

$$F^2_{\ell m} = -ik_o \int_{C_m} \left\{ \left(\mu_r - \frac{1}{\varepsilon_r}\right) + \frac{1}{k_o^2}\left[\left(\frac{\partial}{\partial s}\,\frac{1}{\varepsilon_r}\right)\frac{\partial}{\partial s} + \left(\frac{\partial}{\partial n}\,\frac{1}{\varepsilon_r}\right)\frac{\partial}{\partial n} \right] \right\} G(\bar{r}_\ell, \bar{r}')\, ds'$$

$$F^3_{\ell m} = -\iint_{A_m} \frac{\partial}{\partial n} G(\bar{r}_\ell, \bar{r}')\, ds'\, dn'$$

$$F^4_{\ell m} = \frac{i}{k_o}\left(\frac{\varepsilon_r}{\varepsilon_r - 1}\right)\delta_{\ell m} - \frac{i}{k_o}\int_{C_m} \frac{\partial}{\partial n} G(\bar{r}_\ell, \bar{r}')\, ds' \qquad (34)$$

where

$$\delta_{\ell m} = \begin{cases} 1 = \ell = m \\ 0 = \text{otherwise} \end{cases}$$

and $(\hat{s}, \hat{n}, \hat{z})$ represent a general rectilinear coordinate system oriented at the observation cell/segment as illustrated in figure 3.

To simplify the rather complex numerical evaluation of the smaller non-rectangular cells, we have chosen to replace them with square cells of equal area and perform an evaluation similar to that given by Richmond [2] under the premise that the generated inaccuracy is minimal due to the comparatively small size and number of these cells. The remaining integrals may be accurately evaluated by simple numerical means when their integrands are not singular or near-singular over the range of integration. When the latter is true, the impedance matrix element must then be evaluated analytically.

An analytical evaluation of the integrals (34) can be obtained for the singular condition by invoking the small argument approximation of the Hankel function and subsequently integrating each of the resulting terms. This procedure results in a highly accurate evaluation of the diagonal and near-diagonal impedance matrix elements in the case of both the line and area integrals occurring in (34). To explicitly evaluate the analytical form of the impedance elements we first refer to

figure 4. With the testing point P, located at the centroid of the ℓ^{th} volume cell or alternately at the midpoint of the ℓ^{th} contour segment, the impedance elements can be expressed in terms of three generic integrals, defined as

$$U_o(\alpha,\beta) = \lim_{\varepsilon\to 0}\int_{\varepsilon}^{a}\int_{\varepsilon}^{\beta} H_o^{(1)}\left(k_o\sqrt{t^2+u^2}\right)dtdu$$

$$U_1(\alpha,\beta) = \lim_{\varepsilon\to 0}\int_{\varepsilon}^{a} H_o^{(1)}\left(k_o\sqrt{t^2+\beta^2}\right)dt$$

$$U_2(\alpha,\beta) = \lim_{\varepsilon\to 0}\frac{\partial}{\partial\beta}\int_{\varepsilon}^{a} H_o^{(1)}\left(k_o\sqrt{t^2+\beta^2}\right)dt \qquad (35)$$

For this specific implementation an $O(\rho^4, \rho^4 \ln\rho)$ small argument expansion for $H_o^{(1)}(\rho)$ was employed and the reader is referred to [8], where explicit expressions are given for $U_i(\alpha,\beta)$ when α and β are small.

Since the details for the evaluation of the above $F^p_{\ell m}$ are rather lengthy, we will simply state the results. With the introduction of the additional definitions

$$r_1 = \sqrt{x_1^2 + y_1^2}$$

$$r_2 = \sqrt{x_2^2 + y_2^2}$$

$$r_3 = \sqrt{x_3^2 + y_3^2}$$

$$r_4 = \sqrt{x_4^2 + y_4^2} \qquad (36)$$

where (x_i, y_i) are defined in figure 3, , the analytic expressions of the impedance elements take the form

$$F^1_{\ell m} = \frac{1}{\varepsilon_r}\delta_{\ell m} - \frac{ik_o^2}{4}\left(\mu_r - \frac{1}{\varepsilon_r}\right)\left[U_o(x_3,y_3) - U_o(x_2,y_2) - U_o(x_4,y_4) + U_o(x_1,y_1)\right]$$

$$+\frac{i}{4}\left\{(\hat{s}_\ell \bullet \hat{n}_m)\left(\frac{\partial}{\partial s}\frac{1}{\varepsilon_r}\right) + (\hat{n}_\ell \bullet \hat{n}_m)\left(\frac{\partial}{\partial n}\frac{1}{\varepsilon_r}\right)\right\}\left[U_1(x_3,y_3) - U_1(x_2,y_2) - U_1(x_4,x_4) + U_1(x_1,y_1)\right.$$

$$+\frac{i}{4}\left\{(\hat{s}_\ell \bullet \hat{s}_m)\left(\frac{\partial}{\partial s}\frac{1}{\varepsilon_r}\right) + (\hat{n}_1 \bullet \hat{s}_m)\left(\frac{\partial}{\partial n}\frac{1}{\varepsilon_r}\right)\right\}\left[U_1(y_3,x_3) - U_1(y_2,x_2) - U_1(y_4,x_4) - U_1(y_1,x_1)\right.$$

$$F^2_{\ell m} = \frac{k_o}{4}\left(\mu_r - \frac{1}{\varepsilon_r}\right)\left[U_1(x_2,y_2) - U_1(x_1,y_1)\right]$$

$$-\frac{1}{4k_o}\left\{(\hat{s}_\ell \bullet \hat{s}_m)\left(\frac{\partial}{\partial s}\frac{1}{\varepsilon_r}\right) + (\hat{n}_\ell \bullet \hat{s}_m)\left(\frac{\partial}{\partial n}\frac{1}{\varepsilon_r}\right)\right\}\left[H_o^{(1)}(k_o r_2) - H_o^{(1)}(k_o r_1)\right]$$

$$-\frac{1}{4k_o}\left\{(\hat{s}_\ell \bullet \hat{n}_m)\left(\frac{\partial}{\partial s}\frac{1}{\varepsilon_r}\right) + (\hat{n}_\ell \bullet \hat{n}_m)\left(\frac{\partial}{\partial n}\frac{1}{\varepsilon_r}\right)\right\}\left[U_2(x_2,y_2) - U_2(x_1,y_1)\right]$$

$$F^3_{\ell m} = \frac{i}{4}(\hat{n}_\ell \bullet \hat{n}_m)\left[U_1(x_3,y_3) - U_1(x_2,y_2) - U_1(x_4,y_4) + U_1(x_1,y_1)\right]$$

$$+\frac{i}{4}(\hat{n}_\ell \bullet \hat{s}_m)\left[U_1(y_3,x_3) - U_1(y_2,x_2) - U_1(y_4,x_4) + U_1(y_1,x_1)\right]$$

$$F^4_{\ell m} = \frac{i}{k_o}\left(\frac{\varepsilon_r}{\varepsilon_r - 1}\right)\delta_{\ell m} - \frac{\hat{n}_\ell \bullet \hat{s}_m}{4k_o}\left[H_o^{(1)}(k_o r_2) - H_o^{(1)}(k_o r_1)\right]$$

$$-\frac{\hat{n}_\ell \bullet \hat{n}_m}{4k_o}\left[U_2(x_2,y_2) - U_2(x_1,y_1)\right]. \qquad (37)$$

VI. NUMERICAL RESULTS

In this section, we present scattering patterns for a variety of two-dimensional geometries that have been modeled using the reduced unknown integral equations. Results obtained via alternate methods are additionally presented to verify the validity of the presented formulation. The given selection of material geometries and constitutive parameters attest to the versatility of the implemented computer code.

Figures 4 to 11 present the backscatter echo width patterns corresponding to a variety of composite configurations which include perfectly conducting and inhomogeneous dielectric rectangular cylinders. The specific geometrical parameters associated with each of these configurations are noted in the corresponding figure. For each presented case, the agreement between the results obtained with the reduced unknown integral equations and those via alternate techniques [2], [3], [8] is excellent. In modeling the illustrated test configurations a sampling interval of 0.1λ was employed throughout the structure except near the corners (within 0.1λ) of the perfect conductors where a higher sampling rate is required to accurately describe the currents in this region.

VII. SUMMARY

A computationally efficient and simple set of integral equations was presented for modeling two-dimensional scatterers of arbitrary composition and geometrical cross-section. The derived integral equations employ only one current component over the scatterer's cross section and another over the surface of the scatterer. In comparison to the three current components generally required with traditional formulations, this highly significant reduction in unknowns was obtained without the expense of computational complexity or increased singularity of the resulting equations. To verify the validity of the derived integral equations a pulse basis-point matching moment method implementation was considered. This procedure involved a highly accurate analytical evaluation of the diagonal and near-diagonal impedance matrix elements. A representative selection of scattering patterns produced by the developed computer code implementation was also presented. In all cases, these were in excellent agreement with corresponding results obtained via alternate methods.

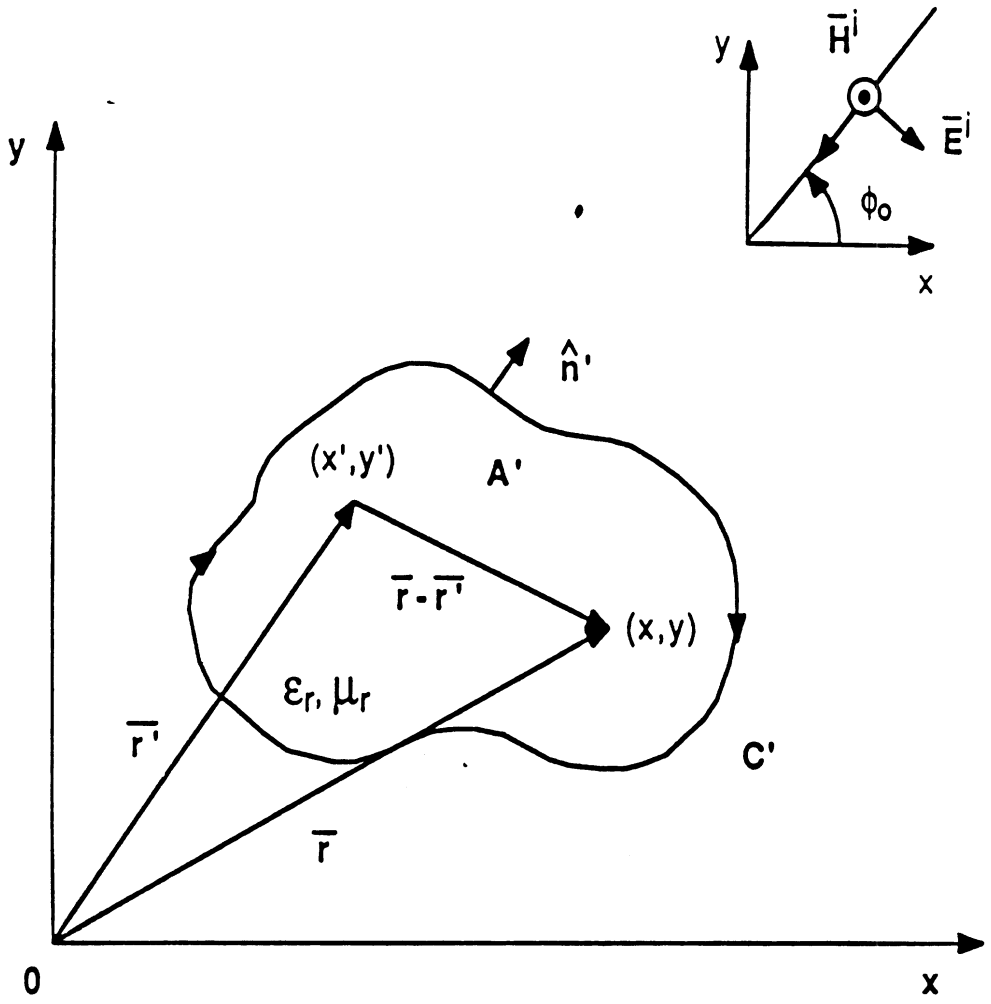


Figure 1: Arbitrary cylindrical structure in free space.

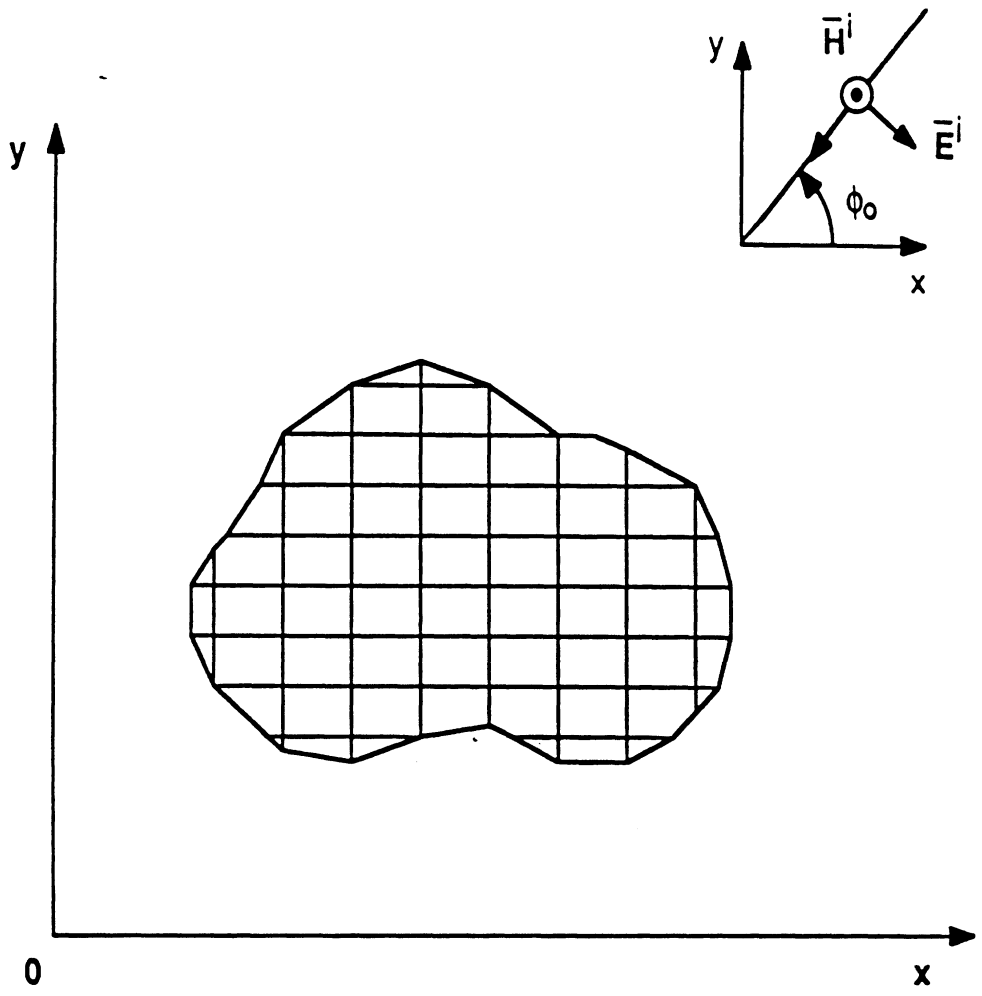


Figure 2: Discretization of scatterer into rectangular and non-rectangular volume cells and linear contour segments.

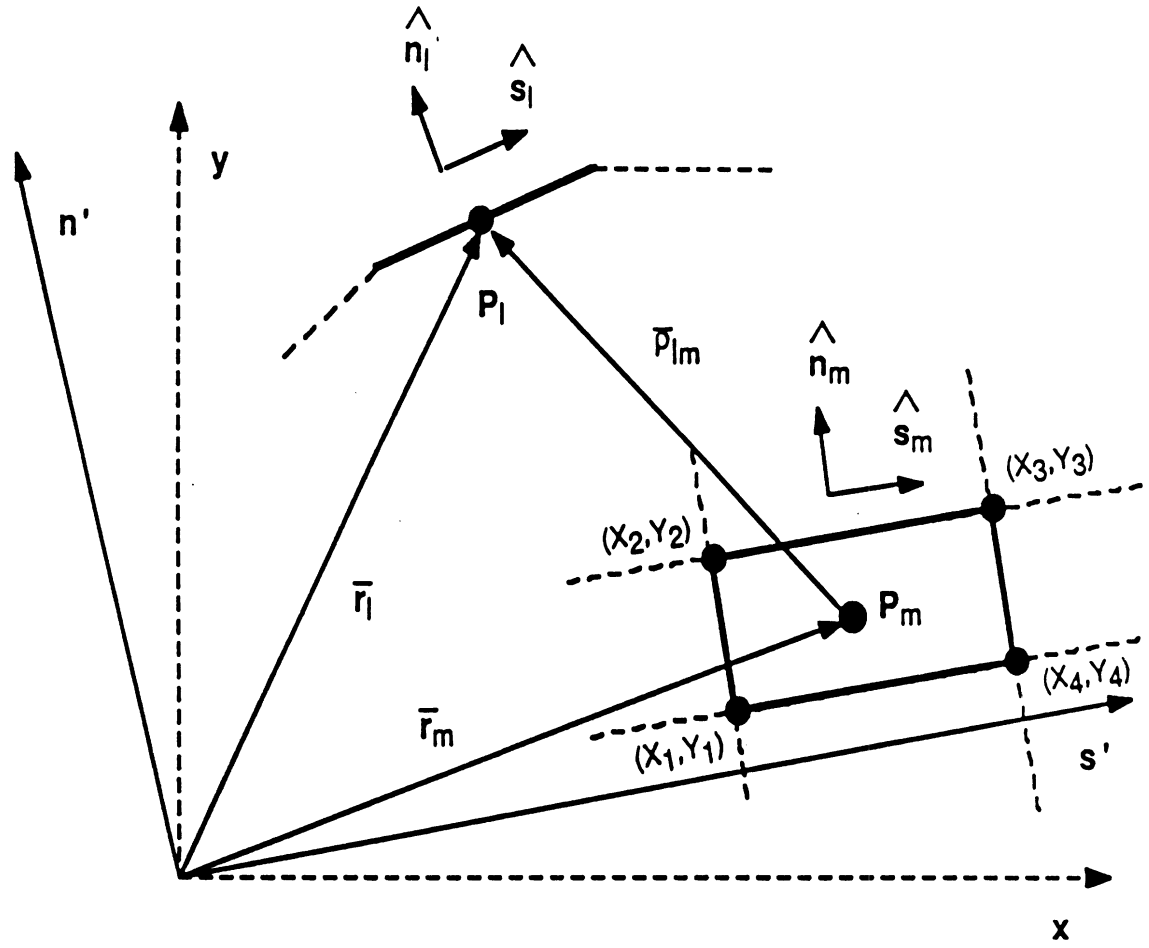


Figure 3: Coordinate frame used in the evaluation of impedance elements requiring integration over a rectangular cell.

HOMOGENEOUS
DIELECTRIC STRIP

E-POLARIZATION

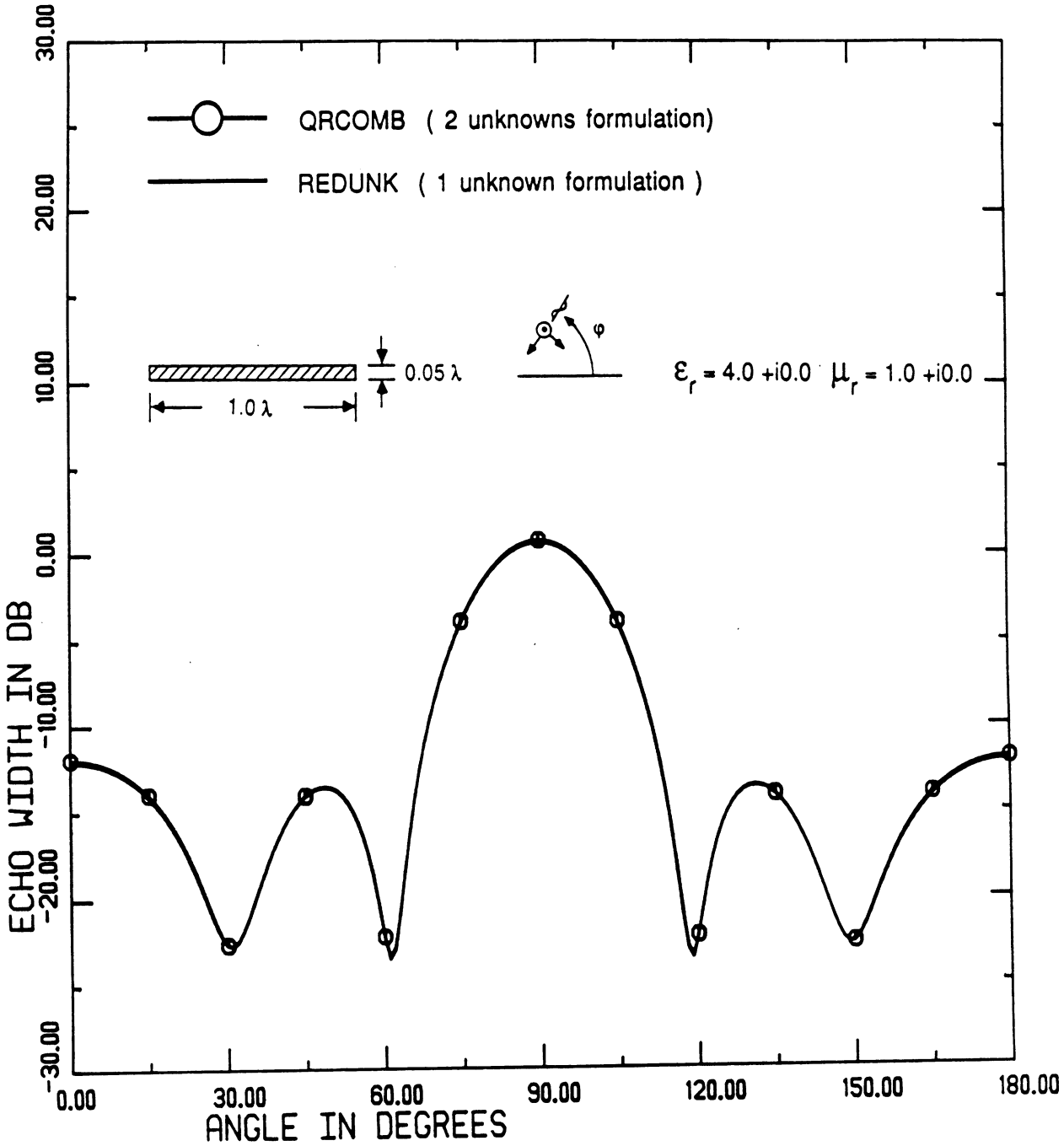


Figure 4

HOMOGENEOUS
DIELECTRIC STRIP

H-POLARIZATION

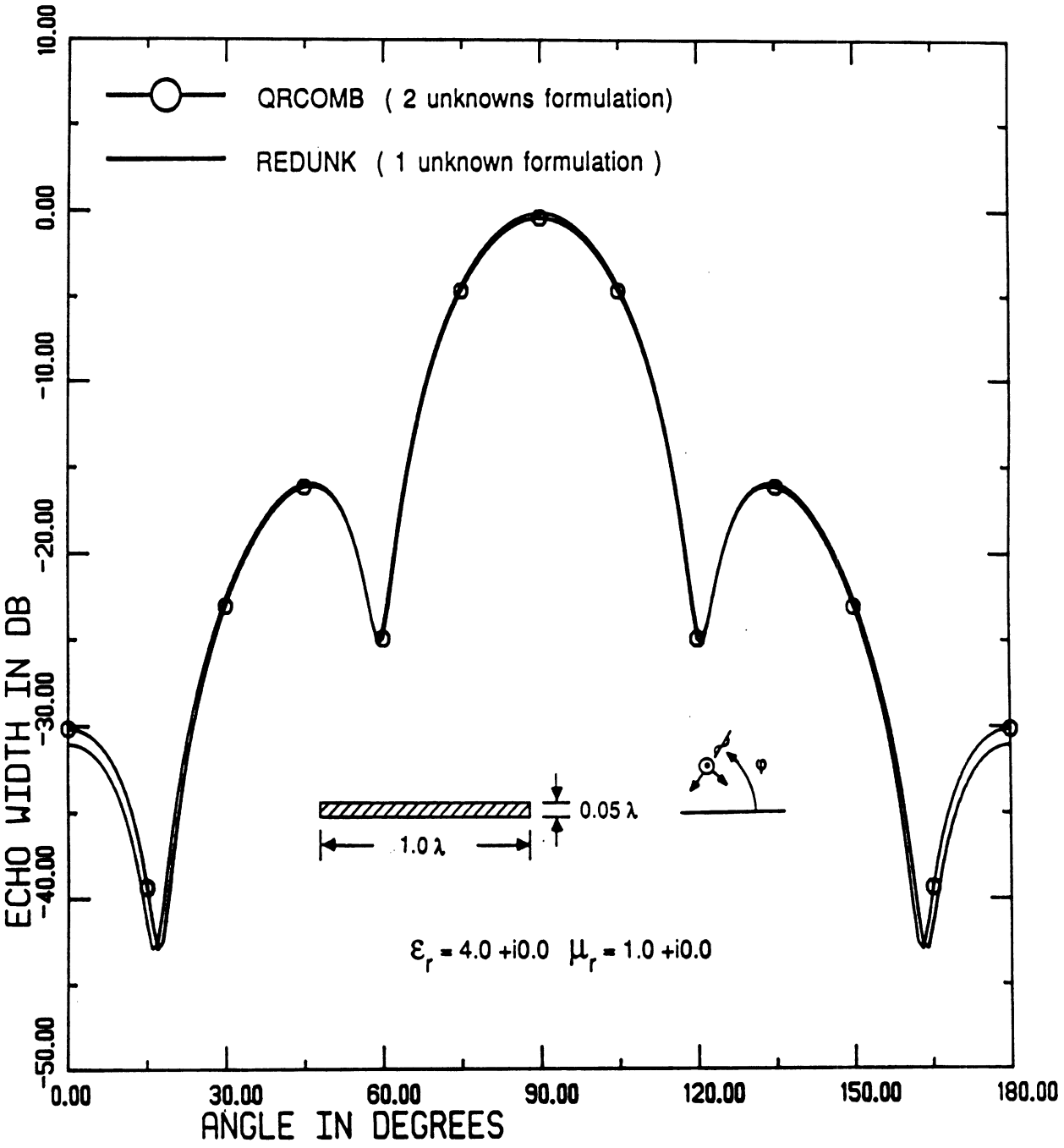


Figure 5

PERFECTLY CONDUCTING RECTANGULAR CYLINDER

E-POLARIZATION

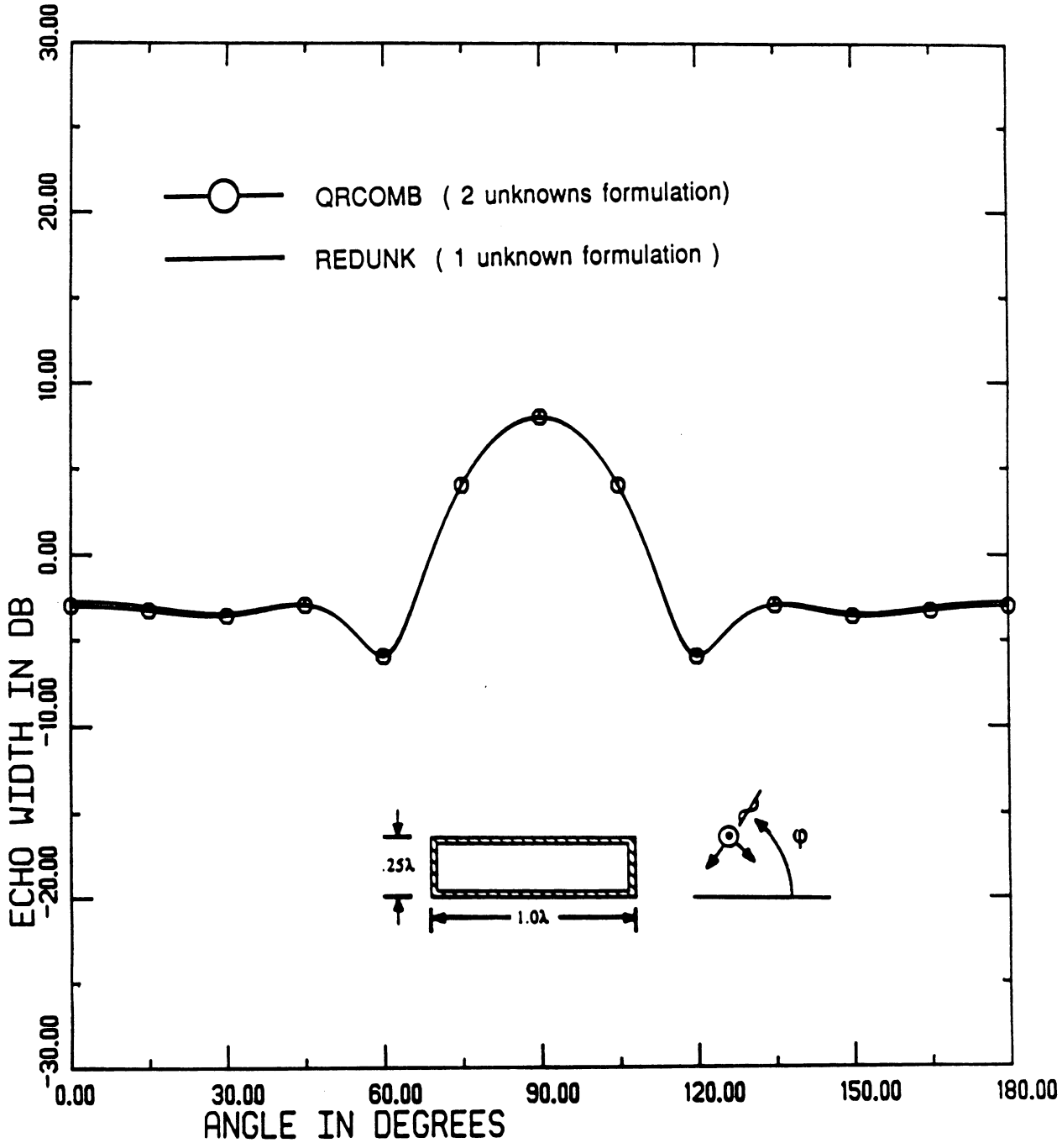


Figure 6

PERFECTLY CONDUCTING RECTANGULAR CYLINDER

H-POLARIZATION

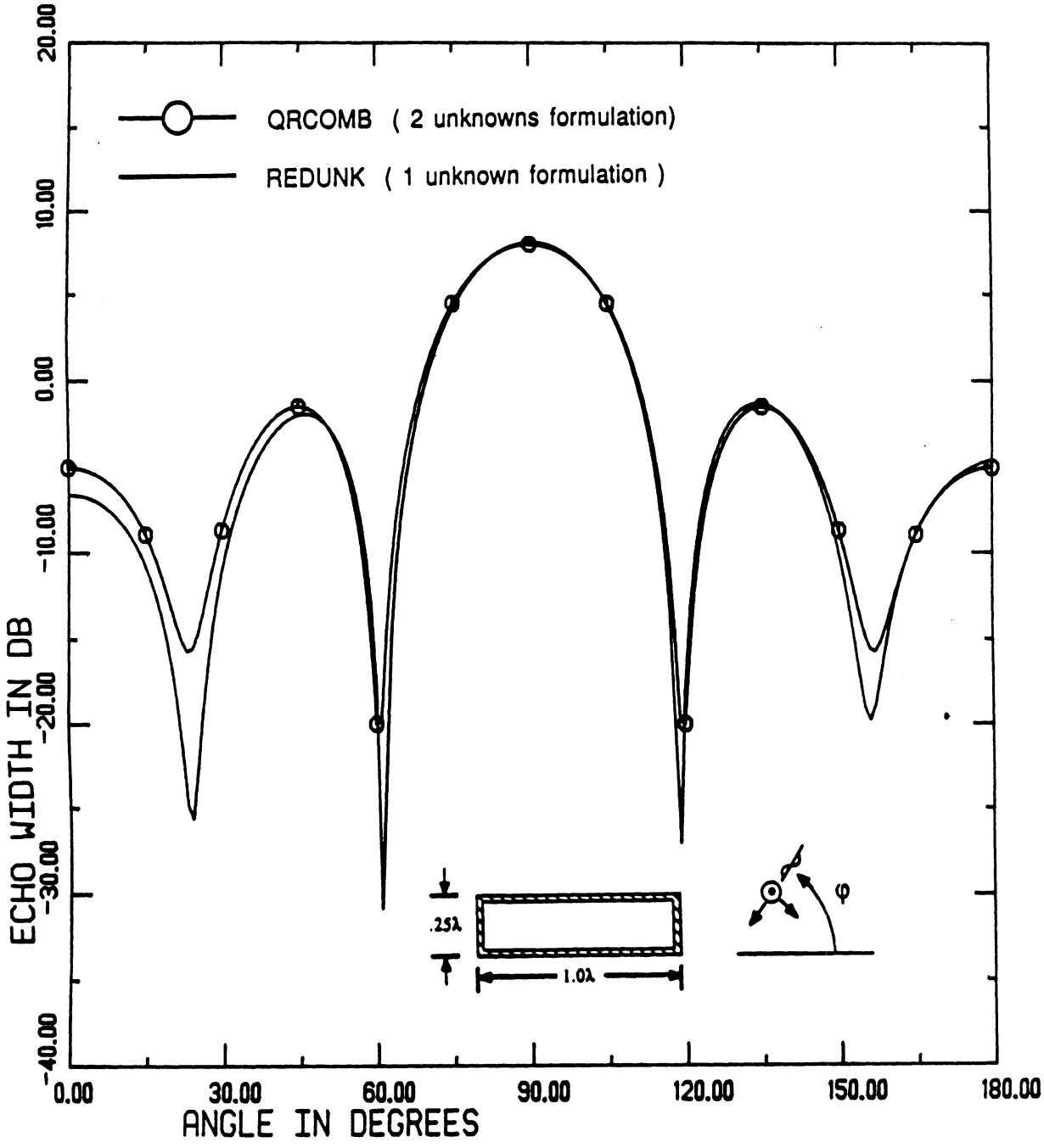


Figure 7

HOMOGENEOUS DIELECTRIC SQUARE CYLINDER

H-POLARIZATION

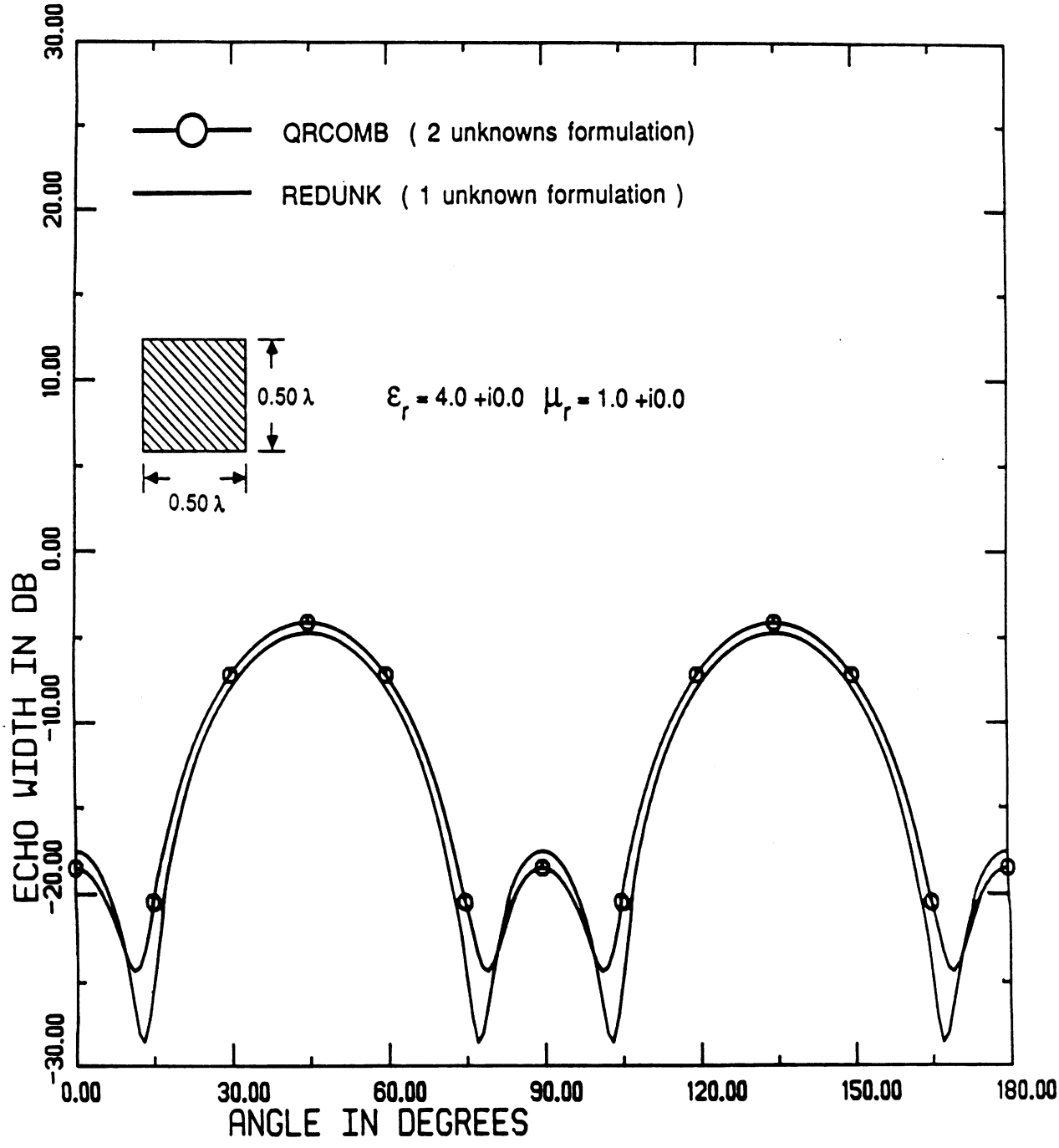


Figure 8

INHOMOGENEOUS DIELECTRIC STRIP

H-POLARIZATION

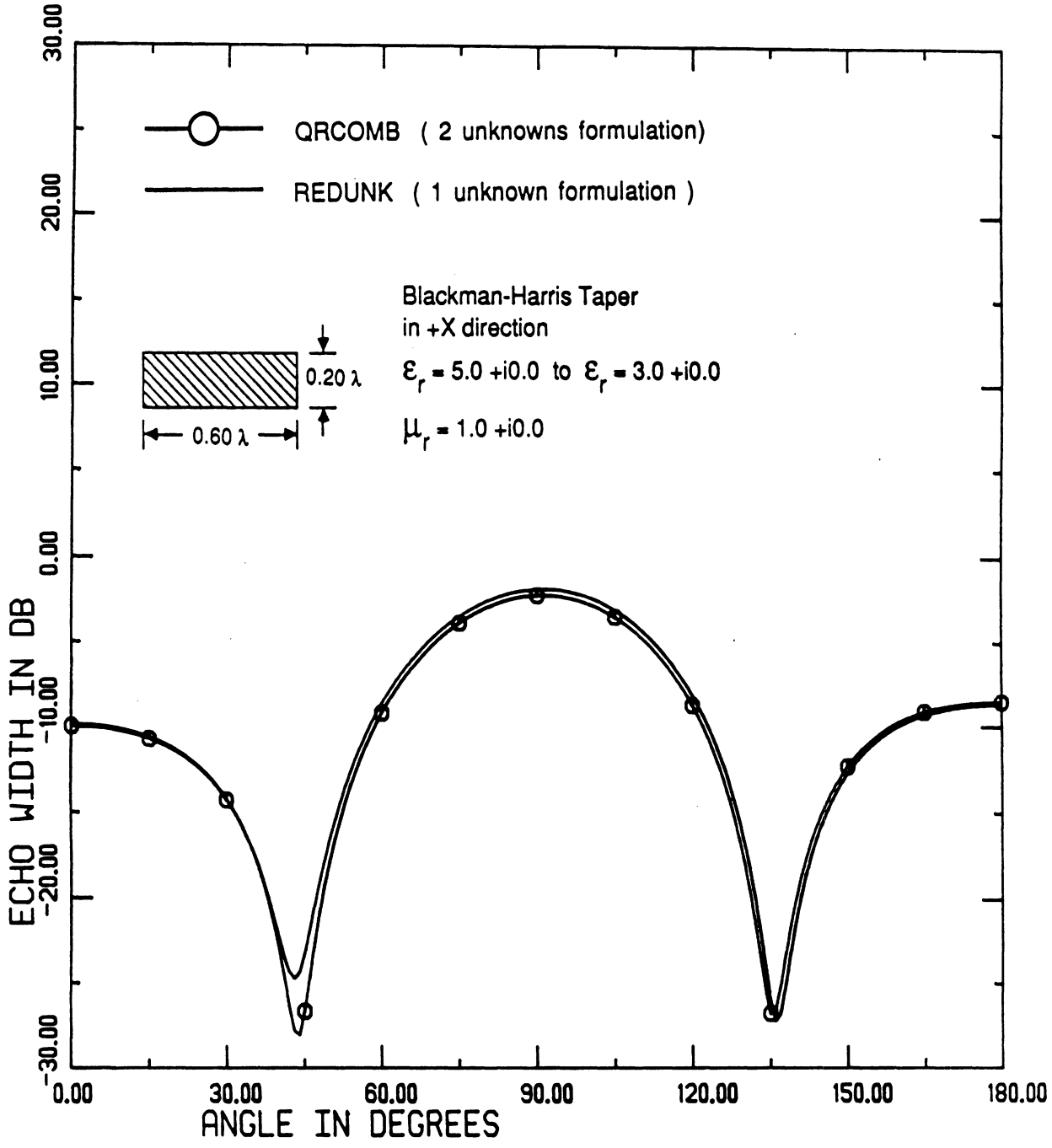


Figure 9

INHOMOGENEOUS DIELECTRIC STRIP

H-POLARIZATION

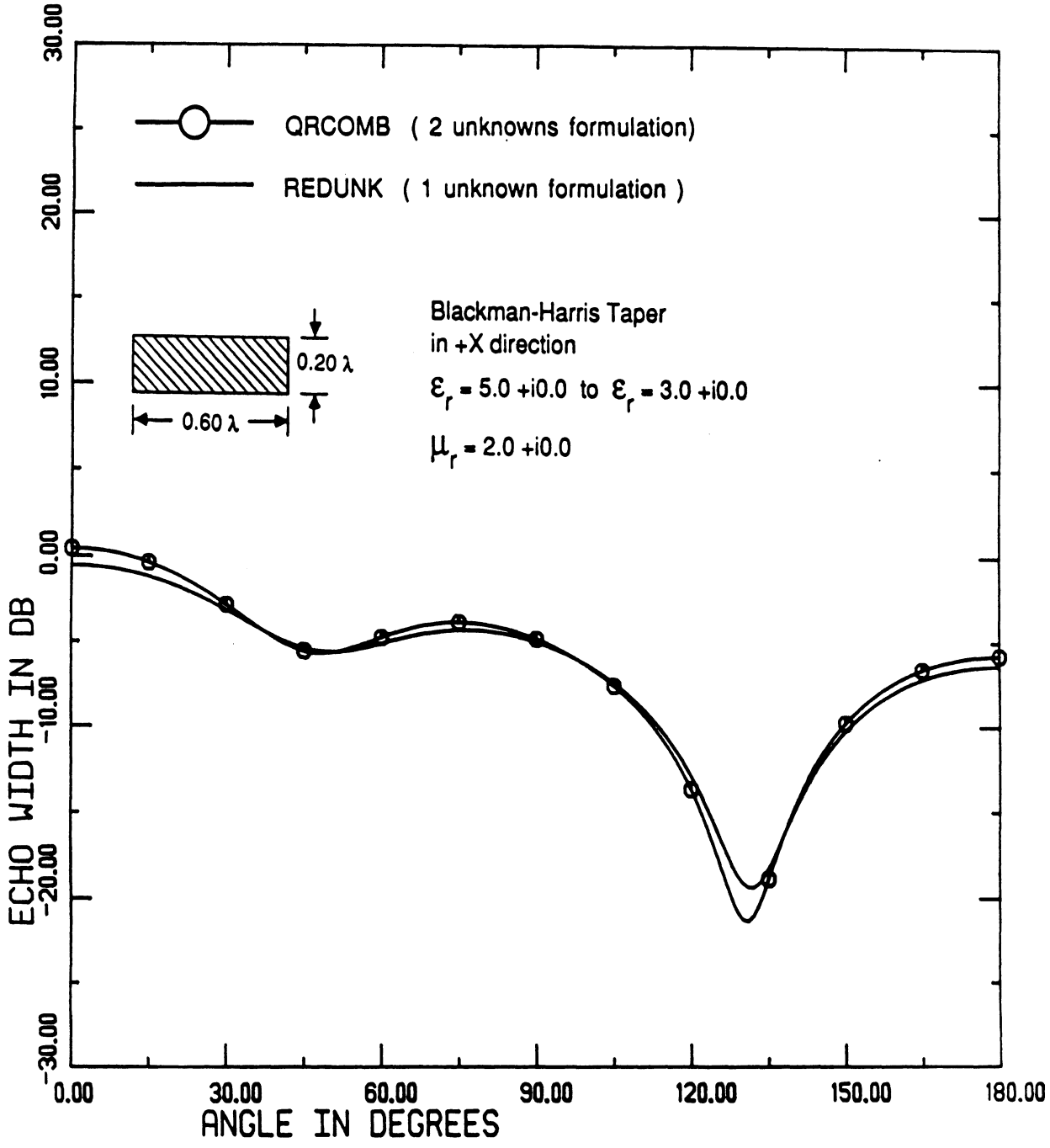


Figure 10

INHOMOGENEOUS DIELECTRIC SQUARE CYLINDER

H-POLARIZATION

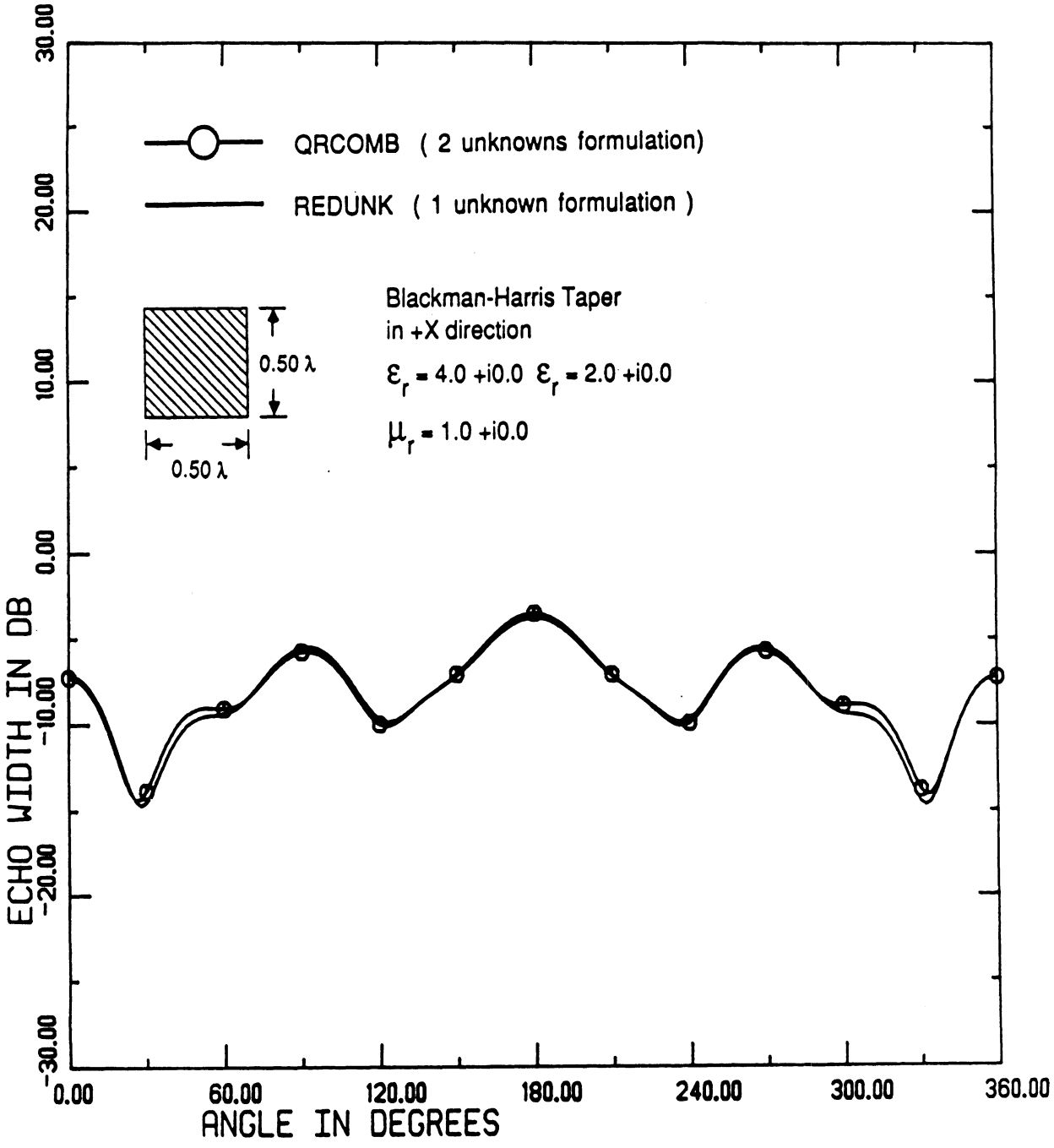


Figure 11

VIII. INTERPRETATION OF INPUT DATA FILE

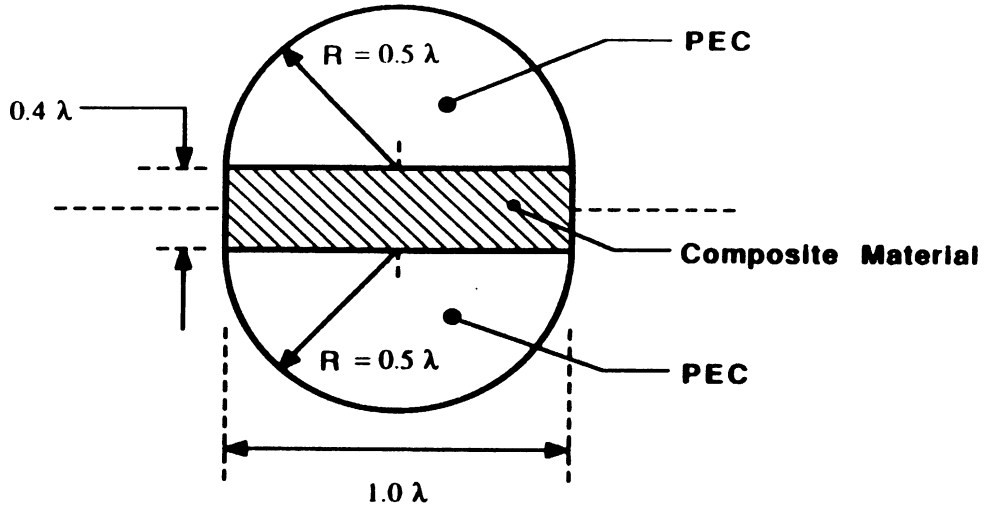
In order to determine the radiation pattern of a two-dimensional scatterer of arbitrary composition and geometrical cross-section, the execution of the scattering code **REDUNK** requires a format-specific, block input file. As shown in Figure 12, the input file consists of six sections which serve to specify the desired output of the code and to model the geometric and material properties of the scatterer. The selected input format permits complex geometries and material parameters to be described in a relatively simple form and is well suited for studies on the RCS effect of a material's impedance properties. While the format of each input section is discussed below, it is important to note that the volume geometry data and contour geometry data coordinates must be entered in a positive X-direction and clockwise direction, respectively, in order to preserve the surface normal direction of the scatterer. Furthermore, with the exception of the initial **Output Format Parameter** and **Contour Geometry** section, each of the remaining data blocks must be terminated by a required input of (000) zeroes.

A. Output Format Parameter

The **Output Format Parameter** section allows the user to select the desired scattering pattern, incident polarization, observation range and additional outputs of interest. The specific function of each variable in the data block is defined in Figure 13. While many of the input requests are self-explanatory, the following varibles are specifically defined as:

Pattern Type	=0	Bistatic scattering pattern
	=1	Backscatter scattering pattern
Scaling Ratio		used to convert the geometrical data coordinates in terms of wavelength
Polarization	=1	E-polarization
	=2	H-polarization

REDUNK SAMPLE INPUT FILE




```
COMMENTS TO BE STORED WITH OUTPUT FILE
    1   1.00000    2OUTFILE
  00.0      360.00000   3.00000  00.00000 2 3   0.0
001          1.000          0.000          1.000          0.000
002          5.000          0.000          5.000          2.000
003          3.000          0.000          3.000          0.000
005          1.000  9999999.000          1.000  9999999.000
006          1.000  9999999.000          1.000          0.000
000
001 00    0.00000    0.00000    0.00000    0.00000    0.00000
002 11   -0.10000    0.10000    0.00000    0.00000    1.00000
003 21    1.00000   -1.00000    0.00000    0.00000    1.50000
004 31    1.00000   -1.00000    0.00000    0.00000    2.00000
005 41   -0.25000    0.25000    0.00000    0.00000    0.00000
000
   4   10  -0.50000    0.00000    0.50000    0.00000    0.40000 23 11 11
00
   0   10  -0.50000    0.20000    0.50000    0.20000 2513 1111 11
   0    5   0.50000    0.20000    0.50000   -0.20000 2111 1111 11
   0   10   0.50000   -0.20000   -0.50000   -0.20000 2111 1111 11
   0    5  -0.50000   -0.20000   -0.50000    0.20000 2513 1111 11
 180   40  -0.50000    0.20000    0.50000    0.20000 5111 1111 11
 180   40   0.50000   -0.20000   -0.50000   -0.20000 5111 1111 11
999
```

Output Format Parameters

Material Impedance Properties

Material Taper Specifications

Volume Geometry Data

Contour Geometry Data

Figure 12

OUTPUT FORMAT PARAMETERS

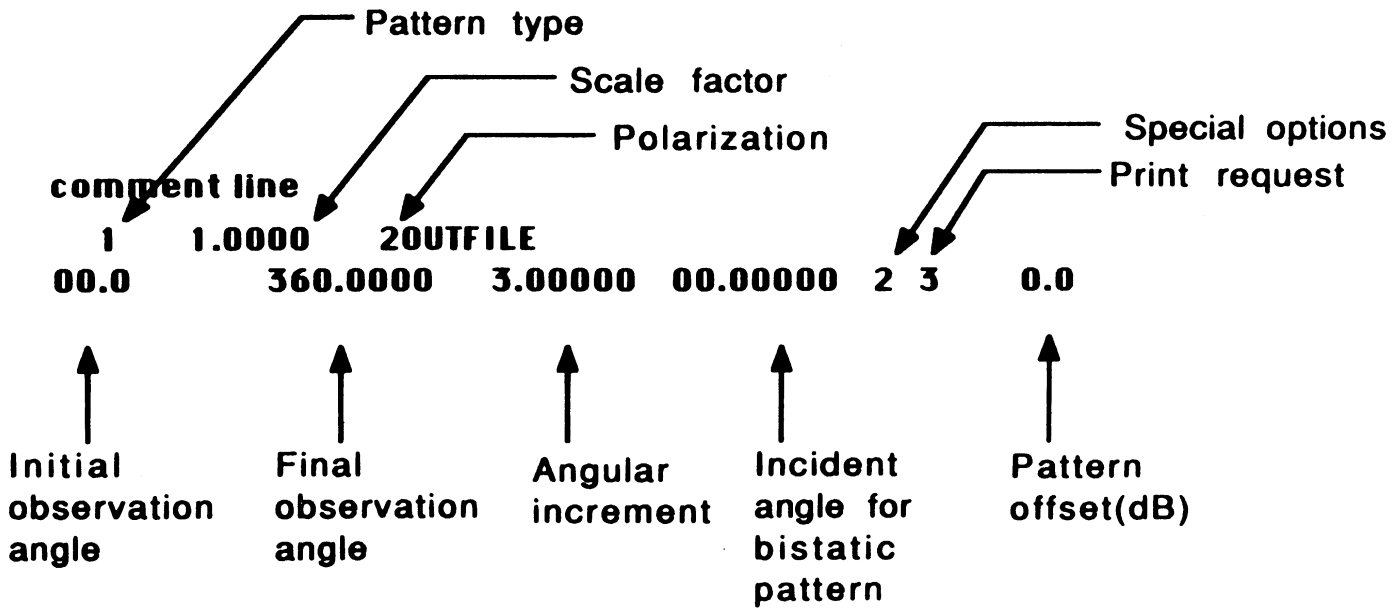


Figure 13

MATERIAL IMPEDANCE SPECIFICATIONS

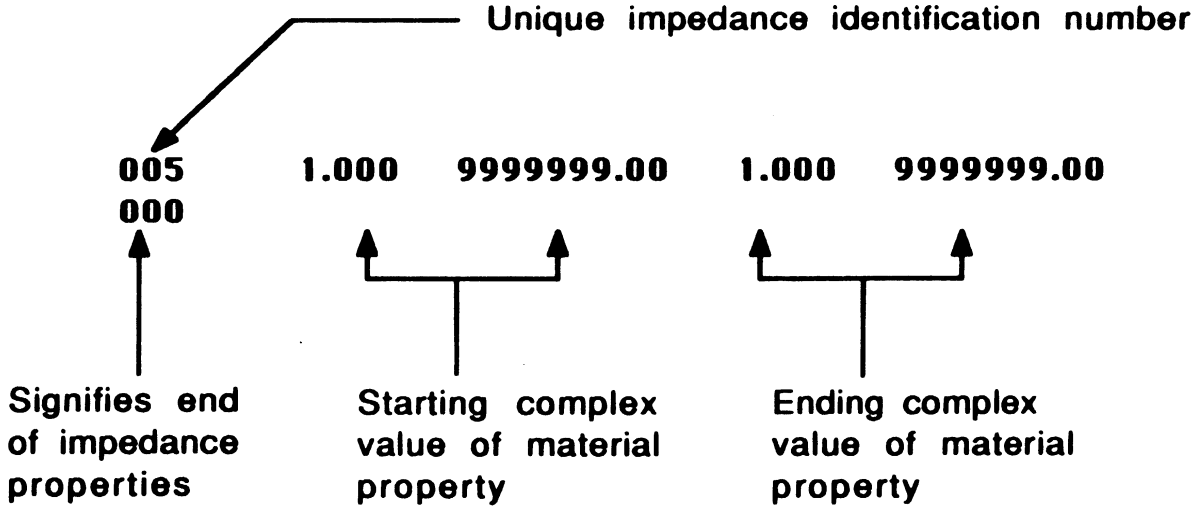


Figure 14

Output Request 1	=1	generates gap impedance/near-field computation
	=2	includes image wave so that the structure is modeled over an infinite ground plane
Output Request 2	=3	prints material specifications and volume and surface currents
	=4	prints impedance matrix elements (real)
	=5	prints impedance matrix elements (complex)
	=6	creates input geometry plotting file

B. Material Impedance Specifications

The second data section of the input file is used as an impedance reference table from which the material properties of the scatterer are later specified. Up to 50 material impedance values can be contained in the table, irregardless of their actual use in describing the material properties of a specific scattering body. As illustrated in Figure 14, each input line consists of a unique impedance specification number, starting complex impedance value, and ending complex impedance value. In this manner, each impedance characteristic is assigned a unique specification number that is subsequently used for identifying the associated permittivity and permeability of an entered geometry component. Since the $e^{-i\omega t}$ time convention is used in the scattering code, all imaginary values of impedance are entered as positive numbers.

C. Material Taper Specifications

The third data section of the input file is used to specify the material taper characteristics of the scatterer. In a manner similar to the **Material Impedance** section, each tapering specification is assigned a unique number so that it can later be identified with an entered geometry component. As such, the combination of the material impedance specification number and material taper specification number can be used to completely model any desired material property of the structure.

Again, up to 50 material taper values can be contained in the table, irregardless of their actual use in describing the material tapering properties of the scattering body. A typical taper specification input line consists of a unique specification number, taper type specification number, taper direction number, taper geometry coordinates and an exponential arguement as shown in Figure15. The actual tapering characteristic employed is determined by the following identification scheme:

Type of material taper	=1	Linear
	=2	Gaussian
	=3	Cos^n (Hanning)
	=4	Blackman-Harris

In combination with the type of material taper employed, the associated tapering direction is specified by the following identification scheme:

Direction of material taper	=0	no taper
	=1	1-sided x-direction
	=2	2-sided x-direction
	=3	1-sided y-direction
	=4	2-sided y-direction
	=5	radial taper with respect to a point

The region of impedance tapering is specified by the entered geometry coordinates and is a function of the taper direction specification. The required coordinate values for each taper direction can be found in the function **Taper** on page 19-20 of the REDUNK program listing.

D. Volume Geometry Data

While the three previous sections have been concerned with the program's output requests and creating reference tables of material and tapering properties, the remaining geometry data sections are used to completely describe the geometry and material characteristics of the scatterer. In modeling the scatterer's volumetric

MATERIAL TAPER SPECIFICATIONS

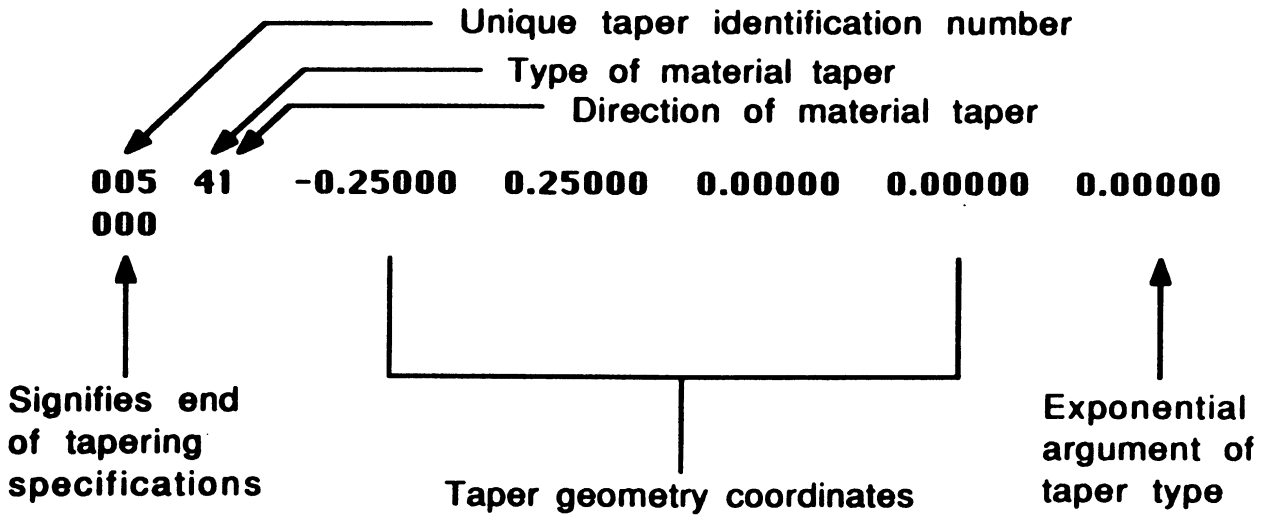


Figure 15

cross-section, the internal volume is subdivided into successively smaller rectangular volume cells, until the remaining non-rectangular volumes are approximately less than .01 λ^2. As depicted in Figure 16, the entire volume of the scatterer is thus comprised of rectangular and non-rectangular cells. Each type of volumetric cell requires a unique input format . In returning to the format of the volume geometry section, a single input line is used to enter the geometrical coordinates and material properties of the volume cell, as shown in Figure 17. As mentioned earlier, the volume geometry coordinate inputs must be entered in a positive X-direction, in order to provide the correct surface normal of the scatterer. In the case of a rectangular volume cell, the starting and ending perimeter midpoints are entered along with the volume's total thickness. Additionally, the number of sampling layers within the specific volume and the number of sampling points per layer are also entered as illustrated in Figure 17. The remaining sequence of numbers is used to describe the impedance characteristics of the specific volume cell. The first digit of the sequence is used to identify the complex permittivity (ε_r) of the volume and is specified from a unique impedance specification number found in the material reference table (i.e **Material Impedance Specification** section). The second digit of the sequence is used to identify the complex permeability (μ_r) of the volume and is specified from a unique impedance specification number which is also found in the material reference table (i.e **Material Impedance Specification** section). As shown, the remaining digits are concerned with specifying the tapering of the material's impedance properties and with the subsequent tapering of the computed volume currents. These specific tapering values are identified by their association with the unique specification numbers found in the **Material Taper Specification** section.

As shown in Figure 18, an entry of **1** for the number of layers **and 1** for the sampling points per layer is required to signify the input of a non-rectangular cell. For such a cell type, the coordinates of the centroid of the cell and total cell area are entered as illustrated. The remaining material properties and tapering characteristics of the cell are entered in a manner identical to that of the rectangular

Volume Geometry Discretization

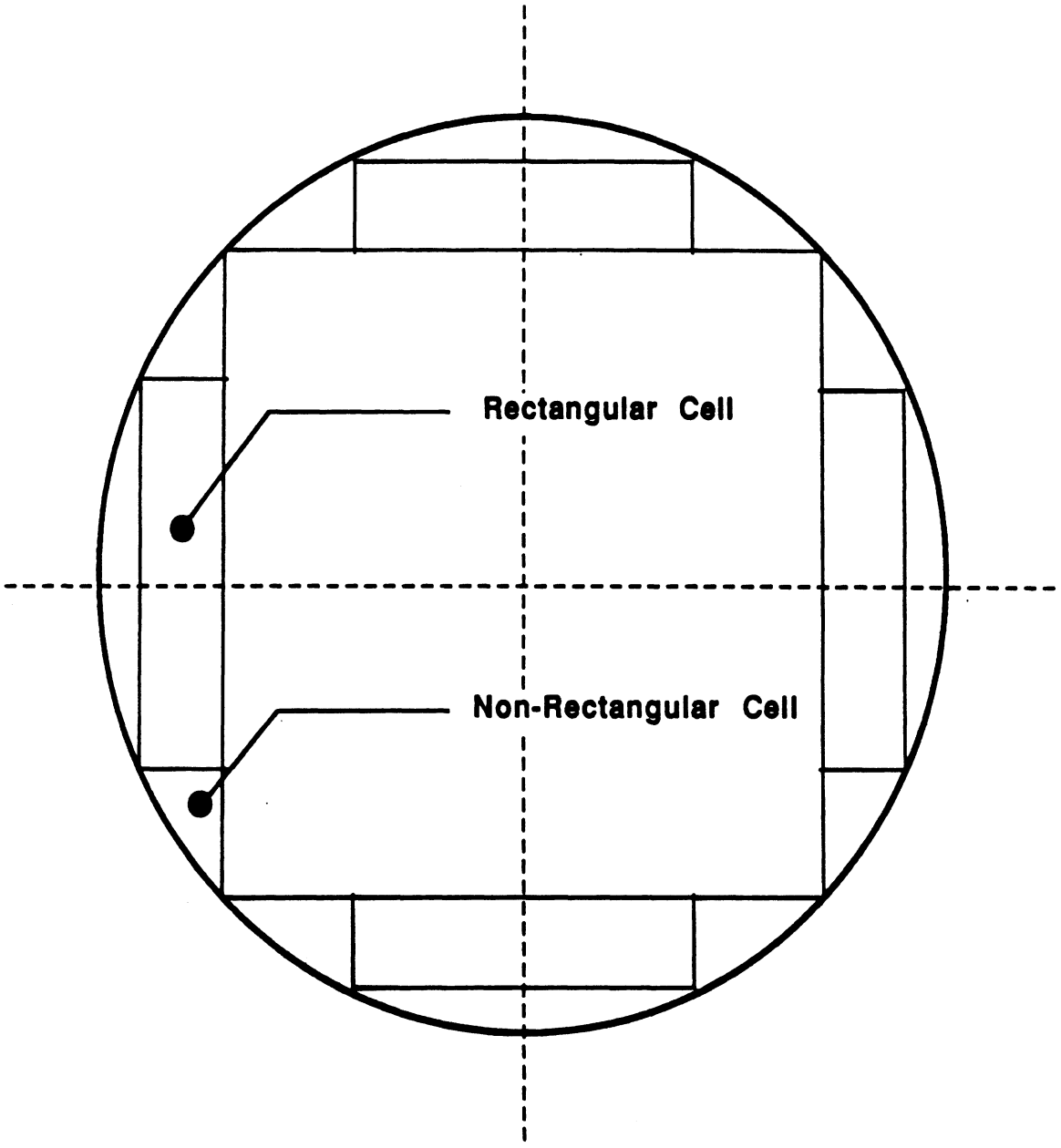


Figure 16

VOLUME GEOMETRY DATA

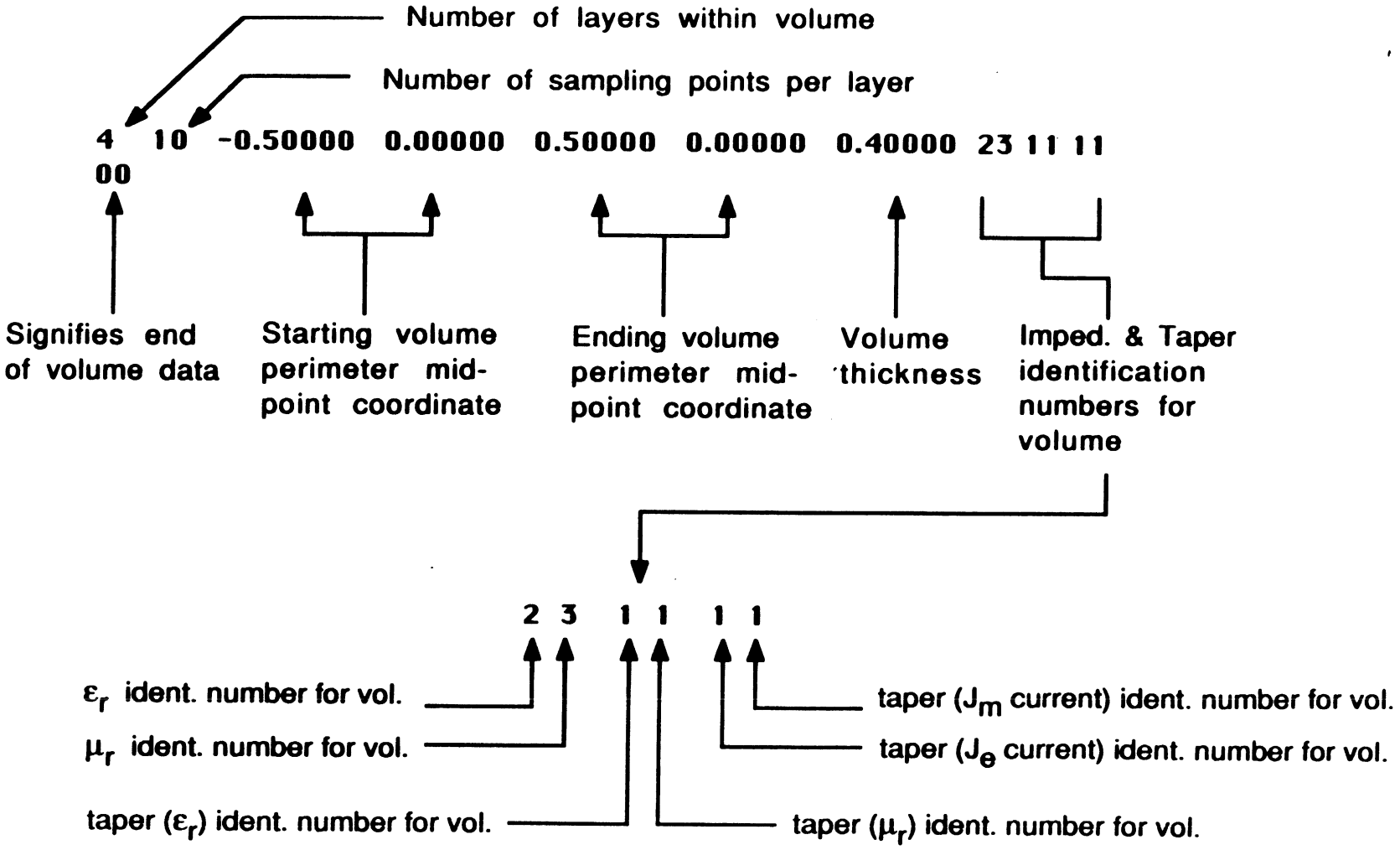


Figure 17

VOLUME GEOMETRY DATA

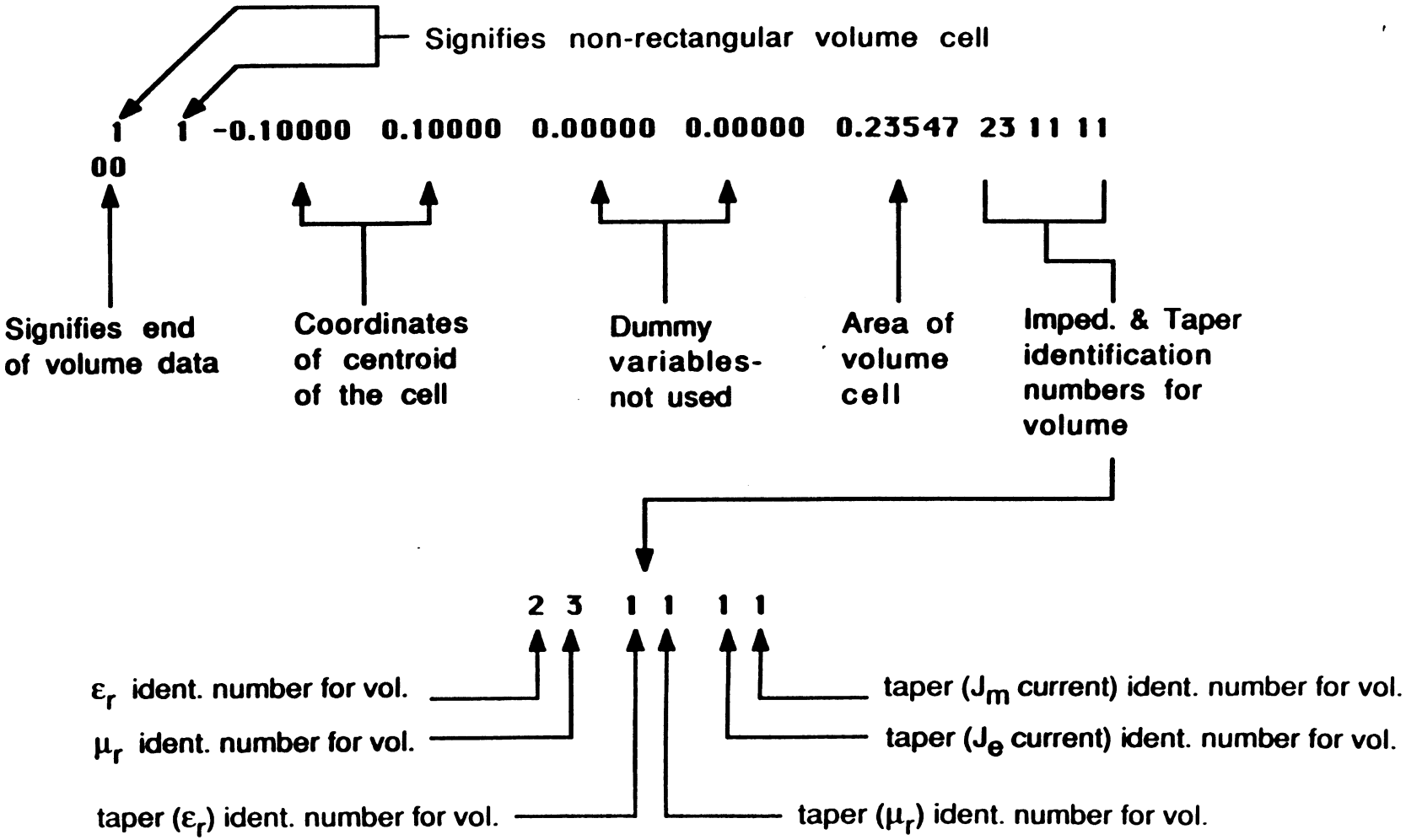


Figure 18

CONTOUR GEOMETRY DATA

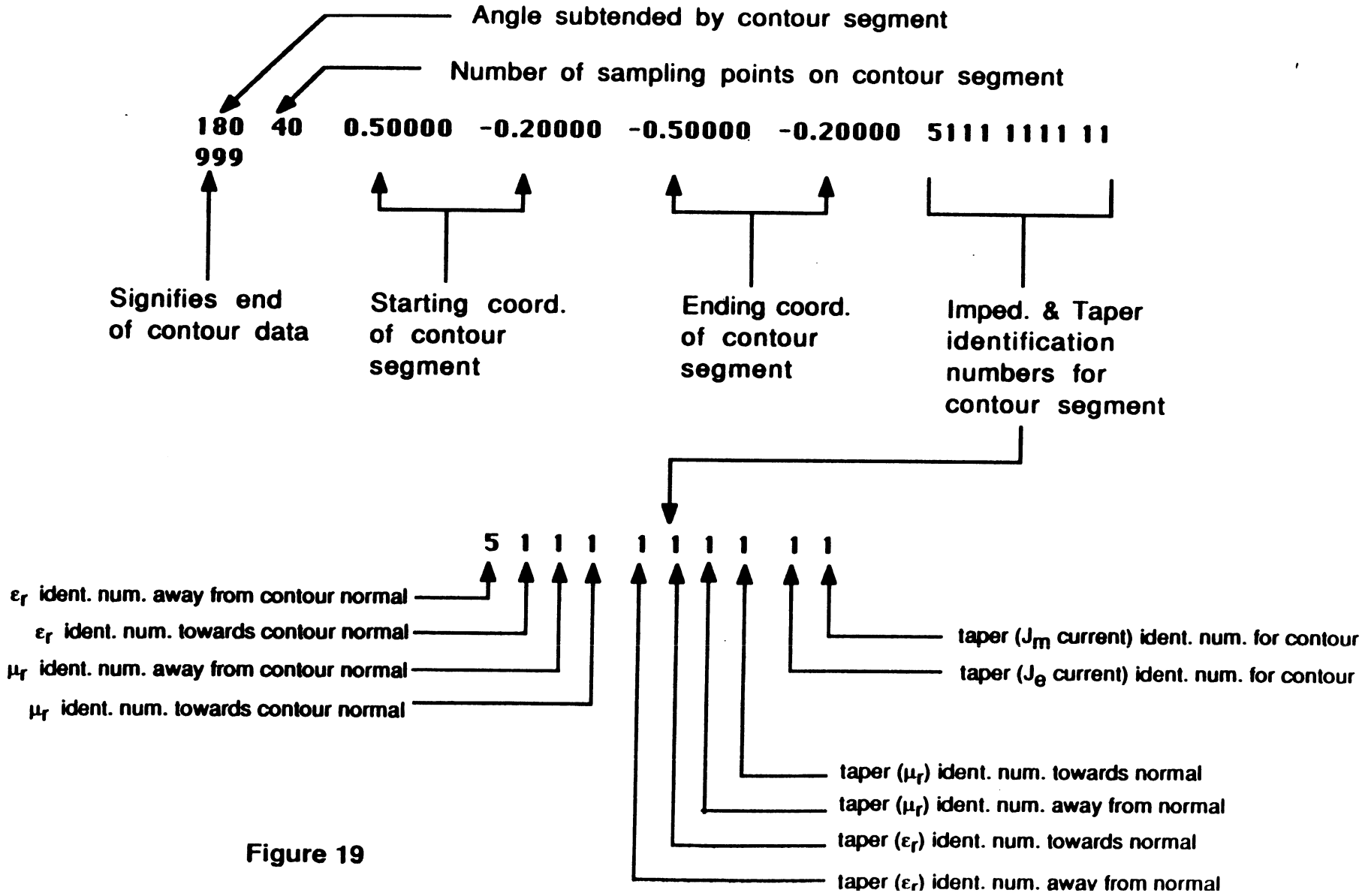


Figure 19

cells.

Once the set of volume cells has been inputed, a sequence of zeroes (000) is used to terminate the volume geometry section.

E. Contour Geometry Data

In the numerical execution of **REDUNK**, the surface contour of any arbitrary scatterer is modeled through the use of straight and/or curved contour segments. Thus, the contour of the scattering body is described by set of piecewise continuous segments which must be entered in a clockwise direction in order to maintain an outward surface normal direction. In addition to the exterior surface, any internal boundries where a step discontinuity in impedance exists must also be entered. The specific input format for the contour geometry data is shown in Figure 19. As illustrated, each contour segment is described by a set of starting and ending coordinates, the angle subtended by the coordinate pair (i.e. ranging from 0 = straight line to 180 = semicircle), the number of sampling points on the specific segment, and a sequence of digits which describes the associated impedance and tapering characteristics contour segment. The first group of four digits is used to specify the permittivity and permeability on both sides of the associated contour segment. In a manner identical to that of the volume geometry section, the impedance value on each side of the contour is specified by an impedance identification number from the **Material Impedance Specification** section of the input. The second group of four digits is used to specify the material tapering characteristics on both sides of the associated contour segment. As before, the material tapering characteristic on each side of the contour is specified by an tapering identification number from the **Material Tapering Specification** section of the input. The remaining digits in the contour segment characteristics sequence is used to taper the computed contour surface currents and are identified with specification numbers from the **Material Tapering Specification** section of the input. Once the set of contour segments has been inputed, a sequence of nines (999) is used to terminate the contour geometry section.

IX. MODELING CONSIDERATIONS

As with most multi-purpose scattering codes, the proper modeling of the scattering structure is critical to producing an accurate radiation pattern. Although the scattering output of **REDUNK** is somewhat insensitive to sampling densities, the following considerations are discussed in order to aid the user in the proper modeling of the scattering body.

Purely Dielectric Structures

In the specific case of E_z-incidence (TM-incidence), the equivalent surface currents on any purely dielectric structure (including perfectly conducting) are identically zero. As such, the contour segments which are used to evaluate these surface currents are not required and thus, they **are not included** in the input file. However, the sequence of nines (i.e. 999) which signifies the end of the contour data section are still included. Figure 20, illustrates the above condition and subsequent input file.

Perfect Electrical Conductors

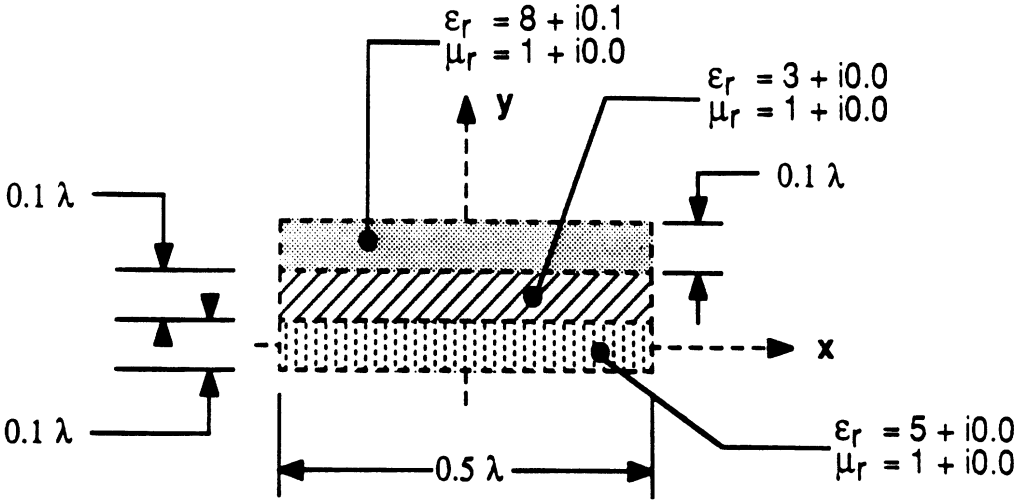
The proper modeling of perfectly conducting surfaces is directly dependent upon the polarization of interest. Thus special care must be taken when computing the E-plane and H-plane radiation pattern of such a structure, as each polarization requires a different modeling configuration. By exploiting the existence of only a single current component for each polarization, the user can significantly reduce the number of required unknowns and resultant computer storage and CPU time requirements.

For the case of H_z-incidence (TE-incidence) on a perfect electrical conductor, no equivalent volumetric currents exist within the interior of the scatterer. As such, the inclusion of the volume geometry data for the perfectly conducting structure is not required. Figure 21, illustrates the above condition and subsequent input file.

For the case of E_z-incidence (TM-incidence) on a perfect electrical conductor, no equivalent surface currents exist on the contour of the scatterer. As

Modeling Considerations - Sample Inputs

(Purely Dielectric Structure)

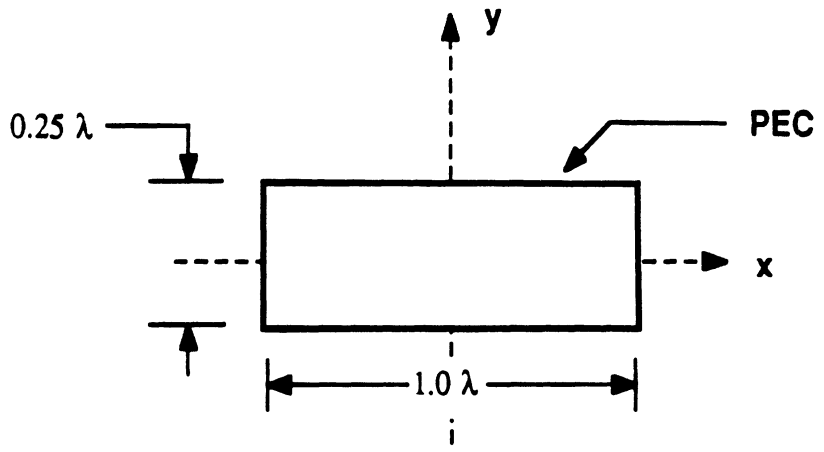



```
3 layer Dielectric Strip, E-Pol, Back
    1   1.00000     1OUTFILE
   0.000  360.000      3.0000     0.0000  0 0  0.0
001          1.000         0.000          1.000          0.000
002          5.000         0.000          5.000          0.000
003          3.000         0.000          3.000          0.000
004          8.000         0.100          8.000          0.100
005          1.000   9999999.000          1.000          0.000
000
001 00    0.00000    0.00000    0.00000    0.00000    0.00000
002 11   -0.10000    0.10000    0.00000    0.00000    1.00000
003 21    1.00000   -1.00000    0.00000    0.00000    1.50000
004 31    1.00000   -1.00000    0.00000    0.00000    2.00000
005 41   -0.25000    0.25000    0.00000    0.00000    0.00000
000
  2   10  -0.50000    0.00000    0.50000    0.00000    0.10000 21 11 11
  2   10  -0.50000    0.05000    0.50000    0.05000    0.10000 31 11 11
  2   10  -0.50000    0.15000    0.50000    0.15000    0.10000 41 11 11
000
999
 1   20  -1.00000    3.00000    1.00000    3.00000
00
```

Figure 20

Modeling Considerations - Sample Inputs

(Perfect Conductor - H Polarization)




```
1.0 lam X 0.25 lam PC, H-pol, Bistatic - 45 deg
    0   1.00000    2OUTFILE
   0.000  360.000      3.0000   45.0000  0 0  0.0
001         1.000         0.000         1.000         0.000
002         5.000         0.000         5.000         0.000
003         3.000         0.000         3.000         0.000
004         1.000   9999999.000         1.000   9999999.000
000
001 00   0.00000   0.00000   0.00000   0.00000   0.00000
002 11  -0.10000   0.10000   0.00000   0.00000   1.00000
003 21   1.00000  -1.00000   0.00000   0.00000   1.50000
004 31   1.00000  -1.00000   0.00000   0.00000   2.00000
000
000
  0   10  -0.50000   0.12500  -0.40000   0.12500 4111 1111 11
  0   15  -0.40000   0.12500   0.40000   0.12500 4111 1111 11
  0   10   0.40000   0.12500   0.50000   0.12500 4111 1111 11
  0   25   0.50000   0.12500   0.50000  -0.12500 4111 1111 11
  0   10   0.50000  -0.12500   0.40000  -0.12500 4111 1111 11
  0   15   0.40000  -0.12500  -0.40000  -0.12500 4111 1111 11
  0   10  -0.40000  -0.12500  -0.50000  -0.12500 4111 1111 11
  0   25  -0.50000  -0.12500  -0.50000   0.12500 4111 1111 11
999
 1   20  -1.00000   3.00000   1.00000   3.00000
00
```

Figure 21

such, the inclusion of the contour geometry data for the perfectly conducting structure is not required. However, in order to correctly simulate the scattering characteristic of the structure, thin volumetric segments (i.e. 1 layer) are used to model the conducting surface as illustrated in Figure 22. A minimal thickness of .01λ (which requires a permittivity of ε_r = 1.0 + i999999.0) can be employed for such a configuration.

Additionally, for either polarization, increased sampling (i.e. on the order 1 sample per 1/100 λ) is required within 0.1 λ of the corners on a perfectly conducting structure. Such high sampling densities are required in order to accurately describe the currents which exist in this region.

Perfect Magnetic Conductors

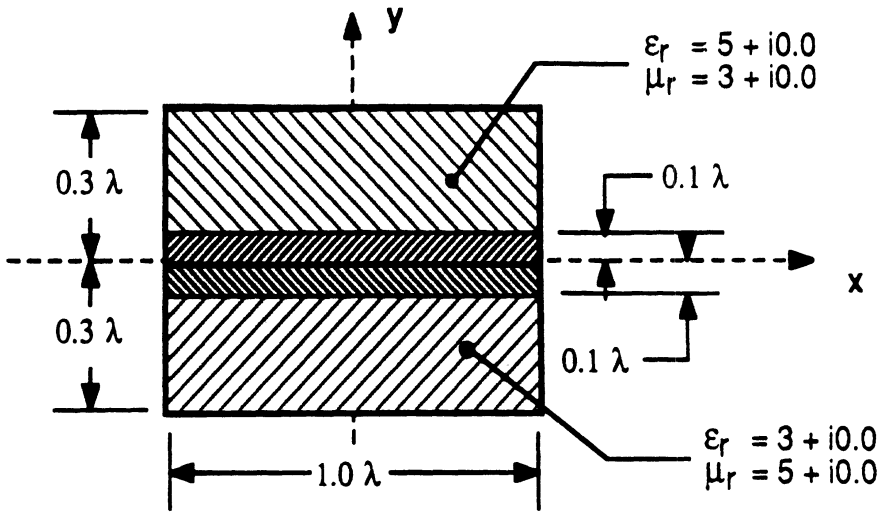
By the application of duality, the modeling of perfect magnetic conductors is obtained in a reciprocal fashion to that of the perfect electrical conductor case. Thus, for the case of E_z-incidence on a perfect magnetic conductor, only contour segments are inputed and for the case of H_z-incidence, only thin volumetric cells are used.

Material Boundries

At the boundry between two components of differing material properties, an increased sampling density is required for the immediate volume cells on both sides of the discontinuity. Such geometric configurations typically require 2 sampling layers of .05 λ thickness in order to accurately describe the volumetric current variation in the immediate region of the material junction. The above modeling consideration is illustrated in Figure 23.

Modeling Considerations - Sample Inputs

(Material Boundries)

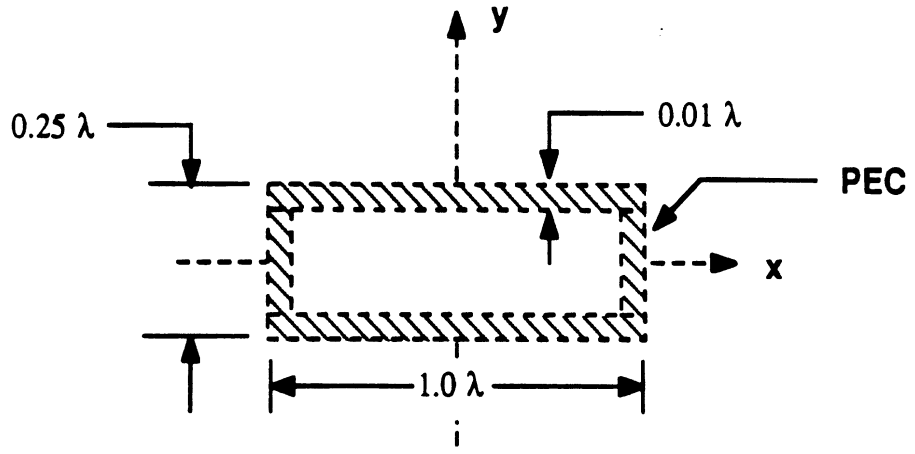



```
TWO RECTANGULAR COMPOSITE SLABS,H-POL,BACK
    1   1.00000     2OUTFILE
   0.000   360.000       3.0000     0.0000  0 0  0.0
001          1.000         0.000          1.000          0.000
002          5.000         0.000          5.000          0.000
003          3.000         0.000          3.000          0.000
005          1.000   9999999.000          1.000    9999999.000
006          1.000   9999999.000          1.000          0.000
000
001 00    0.00000    0.00000    0.00000    0.00000    0.00000
002 11   -0.10000    0.10000    0.00000    0.00000    1.00000
003 21    1.00000   -1.00000    0.00000    0.00000    1.50000
004 31    1.00000   -1.00000    0.00000    0.00000    2.00000
005 41   -0.25000    0.25000    0.00000    0.00000    0.00000
000
  2    10  -0.50000    0.20000    0.50000    0.20000    0.20000 23 11 11
  2    10  -0.50000    0.05000    0.50000    0.05000    0.10000 23 11 11
  2    10  -0.50000   -0.05000    0.50000   -0.05000    0.10000 32 11 11
  2    10  -0.50000   -0.20000    0.50000   -0.20000    0.20000 32 11 11
000
  0    15  -0.50000    0.00000   -0.50000    0.30000 2131 1111 11
  0    40  -0.50000    0.30000    0.50000    0.30000 2131 1111 11
  0    15   0.50000    0.30000    0.50000    0.00000 2131 1111 11
  0    15   0.50000    0.00000    0.50000   -0.30000 3121 1111 11
  0    40   0.50000   -0.30000   -0.50000   -0.30000 3121 1111 11
  0    15  -0.50000   -0.30000   -0.50000    0.00000 3121 1111 11
  0    40  -0.50000    0.00000    0.50000    0.00000 3223 1111 11
999
 1   20  -1.00000    3.00000    1.00000    3.00000
00
```

Figure 23

Modeling Considerations - Sample Inputs

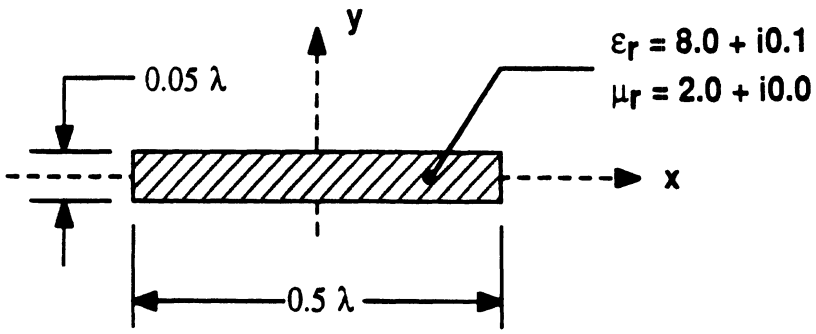
(Perfect Conductor - E Polarization)




```
1.0 lam X 0.25 lam PC, E-pol, Bistatic - 45 deg
    0   1.00000     1OUTFILE
   0.000  360.000      3.0000     0.0000  0 0  0.0
001         1.000         0.000         1.000         0.000
002         5.000         0.000         5.000         0.000
003         3.000         0.000         3.000         0.000
005         1.000  9999999.000         1.000  9999999.000
006         1.000  9999999.000         1.000         0.000
000
001 00    0.00000    0.00000    0.00000    0.00000    0.00000
002 11   -0.10000    0.10000    0.00000    0.00000    1.00000
003 21    1.00000   -1.00000    0.00000    0.00000    1.50000
004 31    1.00000   -1.00000    0.00000    0.00000    2.00000
005 41   -0.25000    0.25000    0.00000    0.00000    0.00000
000
  1   10  -0.50000    0.12500   -0.40000    0.12500    0.01000 51 11 11
  1   20  -0.40000    0.12500    0.40000    0.12500    0.01000 51 11 11
  1   10   0.40000    0.12500    0.50000    0.12500    0.01000 51 11 11
  1   10  -0.50000   -0.12500   -0.40000   -0.12500    0.01000 51 11 11
  1   20  -0.40000   -0.12500    0.40000   -0.12500    0.01000 51 11 11
  1   10   0.40000   -0.12500    0.50000   -0.12500    0.01000 51 11 11
  1   20  -0.50000   -0.12000   -0.50000    0.12000    0.01000 51 11 11
  1   20   0.50000    0.12000    0.50000   -0.12000    0.01000 51 11 11
000
999
 1   20  -1.00000    3.00000    1.00000    3.00000
00
```

Figure 22

X Sample Program Input/Output




```
COMMENTS TO BE STORED WITH OUTPUT FILE
    0   1.00000    2OUTFILE
  0.000  180.000     15.0000   90.0000  0 3  0.0
001         1.000        0.000        1.000        0.000
002         8.000        0.100        8.000        0.100
003         2.000        0.000        2.000        0.000
005         1.000  9999999.000        1.000  9999999.000
006         1.000  9999999.000        1.000        0.000
000
001 00   0.00000   0.00000   0.00000   0.00000   0.00000
002 11  -0.10000   0.10000   0.00000   0.00000   1.00000
003 21   1.00000  -1.00000   0.00000   0.00000   1.50000
004 31   1.00000  -1.00000   0.00000   0.00000   2.00000
005 41  -0.25000   0.25000   0.00000   0.00000   0.00000
000
  1   10  -0.25000   0.00000   0.25000   0.00000   0.05000 23 11 11
000
  0   10  -0.25000   0.02500   0.25000   0.02500 2131 1111 11
  0    4   0.25000   0.02500   0.25000  -0.02500 2131 1111 11
  0   10   0.25000  -0.02500  -0.25000  -0.02500 2131 1111 11
  0    4  -0.25000  -0.02500  -0.25000   0.02500 2131 1111 11
999
 1   10  -0.50000   0.12600   0.50000   0.12600
00
```

Figure 24

THE UNIVERSITY OF MICHIGAN RADIATION LABORATORY

COMMENTS TO BE STORED WITH OUTPUT FILE

1% PROGRAM REDUNK

INPUT GEOMETRY LISTING

TYPE	VOL NUM	LAY NUM	NUM PTS	ENDPOINTS OF THE SEGMENT XA	YA	XB	YB
VOLUME SEGMENT:	1	1	10	-0.25000	0.0	0.25000	0.0
CONTOUR SEGMENT:	0		10	-0.25000	0.02500	0.25000	0.02500
CONTOUR SEGMENT:	0		4	0.25000	0.02500	0.25000	-0.02500
CONTOUR SEGMENT:	0		10	0.25000	-0.02500	-0.25000	-0.02500
CONTOUR SEGMENT:	0		4	-0.25000	-0.02500	-0.25000	0.02500

TOTAL NUMBER OF SAMPLING POINTS = 38
MATRIX ELEMENTS GENERATED
DECOMPOSING MATRIX

KEY PARAMETERS

INCIDENT POLARIZATION	H
TOTAL NUMBER OF INTERIOR VOLUME CELLS	10
TOTAL NUMBER OF CONTOUR SEGMENTS USED	28
NUMBER OF INCIDENT FIELD DIRECTIONS	1
NUMBER OF BISTATIC DIRECTIONS	13
WAVELENGTH	1.00000
RECIPROCAL CONDITION NUMBER	0.2704928E-01

THE UNIVERSITY OF MICHIGAN RADIATION LABORATORY

COMMENTS TO BE STORED WITH OUTPUT FILE

1% PROGRAM REDUNK

ABSORBER SURFACE; INCIDENT FIELD DIRECTION = 90.00

VOLUME OUTPUT

I	SEG	X	Y	S	DSQ	-- EPS --		-- MU --		MOD(Jm)	ARG
1	1	-0.2250	0.0	0.0250	0.0500	8.000	0.100	2.000	0.0	0.2684	-81
2	1	-0.1750	0.0	0.0750	0.0500	8.000	0.100	2.000	0.0	0.3008	-77
3	1	-0.1250	0.0	0.1250	0.0500	8.000	0.100	2.000	0.0	0.3160	-74
4	1	-0.0750	0.0	0.1750	0.0500	8.000	0.100	2.000	0.0	0.3256	-72
5	1	-0.0250	0.0	0.2250	0.0500	8.000	0.100	2.000	0.0	0.3305	-71
6	1	0.0250	0.0	0.2750	0.0500	8.000	0.100	2.000	0.0	0.3305	-71
7	1	0.0750	0.0	0.3250	0.0500	8.000	0.100	2.000	0.0	0.3256	-72
8	1	0.1250	0.0	0.3750	0.0500	8.000	0.100	2.000	0.0	0.3160	-74
9	1	0.1750	0.0	0.4250	0.0500	8.000	0.100	2.000	0.0	0.3008	-77
10	1	0.2250	0.0	0.4750	0.0500	8.000	0.100	2.000	0.0	0.2684	-81

I	SEG	X	Y	S	DSQ	EPS in		EPS out		MU in		MU out		MOD(Je)	ARG(Je)
11	2	-0.2250	0.0250	0.0250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0552	-99.939
12	2	-0.1750	0.0250	0.0750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0716	-99.033
13	2	-0.1250	0.0250	0.1250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0811	-95.712
14	2	-0.0750	0.0250	0.1750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0871	-93.696
15	2	-0.0250	0.0250	0.2250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0900	-92.768
16	2	0.0250	0.0250	0.2750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0900	-92.767
17	2	0.0750	0.0250	0.3250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0871	-93.695
18	2	0.1250	0.0250	0.3750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0811	-95.711
19	2	0.1750	0.0250	0.4250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0716	-99.033
20	2	0.2250	0.0250	0.4750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0552	-99.939
21	2	0.2500	0.0188	0.0062	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0321	145.856
22	2	0.2500	0.0063	0.0187	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0295	-175.066
23	2	0.2500	-0.0062	0.0312	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0277	-144.130
24	2	0.2500	-0.0187	0.0437	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0332	-91.921
25	2	0.2250	-0.0250	0.0250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0722	140.962
26	2	0.1750	-0.0250	0.0750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0894	144.340
27	2	0.1250	-0.0250	0.1250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0980	144.758
28	2	0.0750	-0.0250	0.1750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.1029	145.082
29	2	0.0250	-0.0250	0.2250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.1052	145.281
30	2	-0.0250	-0.0250	0.2750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.1052	145.281
31	2	-0.0750	-0.0250	0.3250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.1029	145.082
32	2	-0.1250	-0.0250	0.3750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0980	144.758
33	2	-0.1750	-0.0250	0.4250	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0894	144.339
34	2	-0.2250	-0.0250	0.4750	0.0500	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0722	140.962
35	2	-0.2500	-0.0188	0.0062	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0332	-91.922
36	2	-0.2500	-0.0063	0.0187	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0277	-144.130
37	2	-0.2500	0.0062	0.0312	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0295	-175.066
38	2	-0.2500	0.0187	0.0437	0.0125	8.00	0.10	1.00	0.0	2.00	0.0	1.00	0.0	0.0321	145.856

THE UNIVERSITY OF MICHIGAN RADIATION LABORATORY

COMMENTS TO BE STORED WITH OUTPUT FILE

** PROGRAM REDUNK

BISTATIC SCATTERING CROSS SECTION
10*LOG(SIGMA/LAMBDA)
FOR INCIDENT FIELD DIRECTION = 90.00
(H-POLARIZATION)

THETA	MOD(P)	ARG(P)	DB
0.0	0.4283E-01	-136.62	-13.68
15.00	0.3125E-02	-175.99	-25.05
30.00	0.1348E-01	-94.95	-18.70
45.00	0.8184E-01	-109.79	-10.87
60.00	0.2019E+00	-113.33	-6.95
75.00	0.3264E+00	-114.65	-4.86
90.00	0.3806E+00	-115.01	-4.19
105.00	0.3264E+00	-114.65	-4.86
120.00	0.2019E+00	-113.34	-6.95
135.00	0.8184E-01	-109.79	-10.87
150.00	0.1347E-01	-94.95	-18.70
165.00	0.3126E-02	-175.97	-25.05
180.00	0.4283E-01	-136.62	-13.68

XI. REFERENCES

[1] A. W. Glisson, "An integral equation for electromagnetic scattering from homogeneous dielectric bodies," IEEE Trans. Antennas Propagat., vol. AP-32, pp. 173-175, Feb. 1984.

[2] J. H. Richmond, "Scattering by a dielectric cylinder of arbitrary cross section shape," IEEE Trans. Antennas Propagat., vol. AP-13, pp. 334-341, May 1965.

[3] J. H. Richmond, "TE-wave scattering by a dielectric cylinder of arbitrary cross section shape," IEEE Trans. Antennas Propagat., vol. AP-14, pp. 460-464, July 1966.

[4] J. Z. Izadian, L. Peters, Jr., and J. H. Richmond, "Computation of scattering from penetrable cylinders with improved numerical efficiency," IEEE Trans. Geosci. Remote Sensing, vol. GE-22, pp. 52-61, Jan. 1984.

[5] D. K. Langan and D. R. Wilton, "Numerical solution of TE scattering by inhomogeneous two-dimensional composite dielectric/metallic bodies of arbitrary cross section," Abstracts of 1986 Nat. Radio Sci. Meet., Philadelphia, PA, June 1986.

[6] E. H. Newman, "TM scattering by a dielectric cylinder in the presence of a half plane," IEEE Trans. on Antennas Proagat. vol. AP-33, pp. 773-782, July 1985.

[7] E. H. Newman, "TM and TE scattering by a dielectric/ferrite cylinder in the presence of a half plane," IEEE Trans. on Antennas Proagat. vol. AP-34, pp. 804-813, June 1986.

[8] M. A. Ricoy and J. L. Volakis, "Integral equations with reduced unknowns for the simulation of two-dimensional composite structures," accepted in IEEE Trans. Antennas and Propagat.

[9] J. A. Stratton, Electromagnetic Theory. New York: McGraw Hill Co., 1941, p.394.

[10] R. F. Harrington, Time-Harmonic Electromagnetic Fields. New York: McGraw Hill Co., 1961, p. 127.

[11] P. E. Mayes, "The equivalence of electric and magnetic sources," IEEE Trans. Antennas Propagat., vol AP-6, pp. 295-296, July 1958.

[12] C.-T. Tai, Dyadic Green's Functions in Electromagnetic Theory. San Francisco: International Textbook Co., 1971, p. 5.

[13] C.-T. Tai, "A note on the integral equations for the scattering of a plane wave by an electromagnetically permeable body," Electromagnetics, vol. 5, pp. 79-88, Jan. 1985.

ADDENDUM A

Revisions For Inclusion of Second Order Quadrilaterals (June 1990)

I. Introduction

This is an addendum to the University of Michigan report "Simple Integral Equations for Two-Dimensional Scattering with Further Reduction in Unknowns." It describes improvements in the implementation of the associated computer code referred to as RUFCODE (Reduced Unknowns Formulation) and applies to the May 1990 version of this code. In particular this addendum describes the modification resulting from the addition of other types of volume elements which are better suited for non-rectangular scatterer cross-sections.

Earlier versions of the RUFCODE relied primarily on discretizations in terms of rectangular cells/elements. Although, this type of discretization leads to a simpler formulation, it has not been found convenient for discretizing penetrable structures with piecewise cylindrical boundaries as is the case with material coated circular cylinders or ogives. In these cases, a non-rectangular volume cell forming an annular sector (see Fig. 1) is a more suitable choice. This type of a cell provides a better fitting with the actual layer or coating contours leading to a more accurate simulation. In addition, any possible overlap between the outer contour and the volume elements is eliminated. Also, in many situations, particularly at corners, neither a rectangular nor an annular sector volume cell are conformal to the actual geometry. Although, only very few volume cells of the structure usually fit in this category, it is important that corners be modeled accurately because of high field intensities at those locations. A more arbitrary or flexible volume cell such as that shown in Figure 2 is, therefore, required for modeling corners formed at the meeting of non-rectangular boundaries(a typical example is the tip of an ogive). The cell in Figure 2 is a quadrilateral whose sides are second order curvilinear segments.

Below we describe the evaluation of the impedance elements attributed to the quadrilaterals illustrated in Figures 1 and 2. Certain additions/modifications to the input format of the code are also discussed along with a few examples which illustrate the use of the new input format. Finally some scattering patterns are included for validation purposes.

II. Impedance Element Evaluation for Second Order Quadrilaterals

In evaluating the impedance elements F^1 and F^3 as defined in (34) of the original report, the integrations must be performed over the section of the cell/element. This can be done analytically for rectangular elements and the results are given in (37) of the original report. However, when the elements are quadrilaterals of more arbitrary shape such are those shown in Figures 1 and 2, the integrations must be performed numerically. A convenient way to do this is via parametric transformation to map the arbitrarily shaped quadrilateral to a square (see for example. [1]). Gaussian quadrature can then be employed to perform the integrations in a standard manner.

On the assumption of quadrilateral elements whose sides are at most second order(quadratic), the required one-to-one parametric transformation is shown in Figure 3 and given by:

$$X(u,v) = \sum_{i=1}^{8} N_i(u,v)\, x_i, \quad Y(u,v) = \sum_{i=1}^{8} N_i(u,v)\, x_i$$

where $N_i(u,v)$ are the shape functions defined as

$$N_1 = \frac{1}{4}(1-u)(1-v)(u+v+1) \qquad N_2 = \frac{1}{4}(1+u)(1-v)(u-v-1)$$

$$N_3 = \frac{1}{4}(1+u)(1+v)(u+v-1) \qquad N_4 = \frac{1}{4}(1-u)(1+v)(v-u-1)$$

$$N_5 = \frac{1}{2}(1-u^2)(1-v) \qquad N_6 = \frac{1}{2}(1+u)(1-v^2)$$

$$N_7 = \frac{1}{2}(1-u^2)(1+v) \qquad N_8 = \frac{1}{2}(1-u)(1-v^2).$$

Clearly, since the annular sector element given in Figure 1 can be considered as a special case of that in Figure 2, there is no need for a separate treatment of these elements. However, the annular sector elements are of particular importance because a sequence of these elements can be used to simulate a circular material layer of constant thickness. The discretization of the layer in small annular sectors can then be programmed into the code, thus, avoiding the time consuming process of entering each of the small annular sectors individually. An internal generation of the more general quadrilaterals would require a rather sophisticated discretization process not available with the code. This type of discretization is generally done with finite element mesh generation packages and a

number of them are available commercially. Such packages can, of course, be integrated with this code without any modification in the structure of the code or its input format. (However,some cosmetic changes may be desirable.)

Incorporating the above parametric transformations into the expressions for F^1 and F^3 yields:

$$\bar{\bar{F}}^1_{lm} = \frac{1}{\varepsilon_r}\delta_{lm} - \frac{ik_o^2}{4}\left(u_r - \frac{1}{\varepsilon_r}\right)\int_{-1}^{1}\int_{-1}^{1} H_o^{(1)}[k_o\,\rho(u,v)]\,J(u,v)\,du\,dv$$

$$+ ik_o \nabla\frac{1}{\varepsilon_r} \cdot \left\{ \hat{x}\int_{-1}^{1}\int_{-1}^{1} X(u,v)/\rho(u,v)\, H_o^{(1)}[k_o\rho(u,v)]\,J(u,v)\,du\,dv \right.$$

$$\left. + \hat{y}\int_{-1}^{1}\int_{-1}^{1} Y(u,v)/\rho(u,v)\, H_o^{(1)}[k_o\rho(u,v)]\,J(u,v)\,du\,dv \right\}$$

$$\bar{\bar{F}}^3_{lm} = \frac{ik_o\hat{n}}{4} \cdot \left\{ \hat{x}\int_{-1}^{1}\int_{-1}^{1} X(u,v)/\rho(u,v)\, H_o^{(1)}[k_o\rho(u,v)]\,J(u,v)\,du\,dv + \right.$$

$$\left. \hat{y}\int_{-1}^{1}\int_{-1}^{1} Y(u,v)/\rho(u,v)\, H_o^{(1)}[k_o\rho(u,v)]\,J(u,v)\,du\,dv \right\}$$

In these equations J(u,v) denotes the Jacobian of the transformation given by

$$J(u,v) = \left| \frac{\partial X(u,v)}{\partial u}\frac{\partial Y(u,v)}{\partial v} - \frac{\partial X(u,v)}{\partial v}\frac{\partial Y(u,v)}{\partial u} \right|$$

and

$$\rho(u,v) = \sqrt{X(u,v)^2 + Y(u,v)^2}\ .$$

The integrations can now be performed via Gaussian quadrature in a straightforward manner without any special considerations due to the integrand singularities at $\rho = 0$. As can be easily seen, the integrand becomes non-singular when the variable transformation $(u,v) \rightarrow (\rho,\theta)$ is introduced and there is no need to consider the principal value of these area integrals.

III. Input Format Modification

As a consequence of the quadrilateral volume/area elements, the input format in this new version has been modified to allow the input of these elements. More importantly, the incorporation of the annular sector elements has now permitted the specification of the circular constant thickness layers in a very simple manner. As noted above, this can be segmented into a sequence of small annular sectors and this segmentation is now done within the code upon specification of the circular contour transversing through the center of the circular layer. Also, new cards were added for an external specification of the more arbitrary second order quadrilaterals. Below we describe these input card modifications in some detail. Some additional capabilities are also described.

The volume geometry card was modified with an additional entry to allow the specification of rectangular as well as annular volume sectors. The new card is now illustrated in Figure 4 and has entries for the following information:

1. The number of layers to be used in subdividing the annular or rectangular sector
2. The number of cells to be used in subdividing each layer
3. The points specifying the midpoints at the start and end angles of the annular sector(enter clockwise for convex layers)
4. The thickness of the annular sector(since the sector can be subdivided into several layers, this can be large)
5. The angle subtended by the sector(zero, if a rectangular sector is to be specified)
6. The material/taper specifications for the sector.

As seen the only modification from the previous card is the addition of the subtended angle by the sector.

The specification of the quadratic quadrilaterals requires three new input cards to be read in sequence for each of these cells as illustrated in Figure 5. The first of these cards carries the descriptor (11) and the Cartesian coordinates of the four nodes where the four sides of the quadrilateral meet. The second card provides the midpoints of the four sides forming the quadrilateral and the center

coordinate of the cell. These eight nodes must be entered in the order shown in Figure 2. It should also be noted that the midpoint nodes must lie in the middle third of the cells in order to assure a one-to-one mapping. Finally, the third card contains the outward normal direction specified in Cartesian coordinates and the material/taper specification identifiers. As usual the end of this input specification section is signified with the 000 card entered below the third card.

Another feature of the code is the generation of output data files to be used for plotting the scatterer's geometry. In particular, two files are generated one for the contour and another for the volume data. Currently up to ten contour segments can be handled and each segment is specified by two columns containing the Cartesian coordinates of the centers of all linear elements comprising the contour segment. The volume segments/ layers are specified by the coordinates of the inner and outer contours. This, of course, requires four columns, two for each contour. Up to five volume layers can be handled totalling again ten contour segments. All data are saved in ASCII and can be read using free Format. Once read, the user can employ a local plotting routine for displaying the contours. Note that at the moment, the outer contours of the isolated quadrilaterals are not saved.

For better file management, all output files have the same first six characters which are found after the polarization code in the input cards. The various output files will have one of the following extensions:

Contents	Extension
Echo Width/Angle	.plt
Parametric Cells	.cells
Contour Segments	.con
Volume Segments	.vol .

The different files created are specified by the user by indicating the appropriate print request as follows:

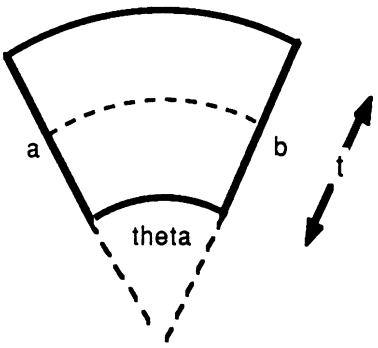
```
Print Request  = 3 prints material specifications
                  and volume/surface currents
               = 4 prints impedance matrix (real)
               = 5 prints impedance matrix (cmplx)
               = 6 creates geometry plotting files
               = 7 creates parametric cell file
               = 8 both 6 and 7
               = 9 both 3 and 8 .
```

IV. Examples

The following pages illustrate the use of RUFCODE's new volume elements. The first figure is the bistatic echo width of a coated cylinder with incidence angle of zero degrees. As shown, the parametric elements yields an improved pattern compared to the former modeling method. This comparison is made with respect to the exact solution. The input file is included to illustrate the modeling technique. The second figure is the H-pol, backscatter echo width for a coated ogive. The coating is homogeneous and of constant thickness. The final figure is the E-pol pattern for the same coated ogive. For both runs the input file is included below the plot. The modeling of the tips of the ogive is illustrated in Figure 6.

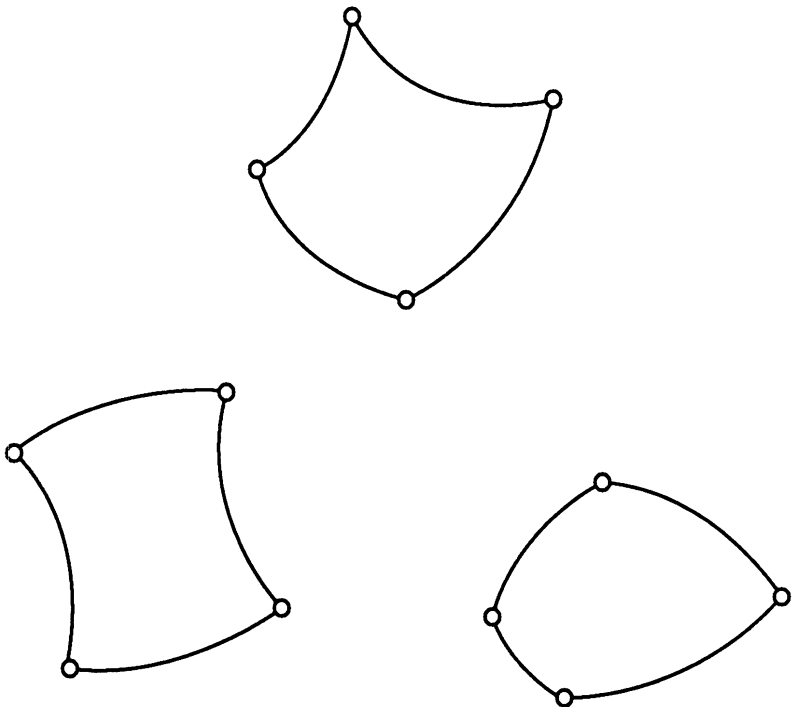
V. Reference

[1] J. Jin, J.L. Volakis, and V.V. Liepa, "A moment method solution of a volume-surface integral equation using isoparametric elements and point matching (TE scattering),", IEEE Trans. Micro. Theory and Tech., vol. MTT-37, pp. 1641-5, Oct. 1989.



Sector of an Annulus

Figure 1



Arbitrary Volume Cells

Figure 2

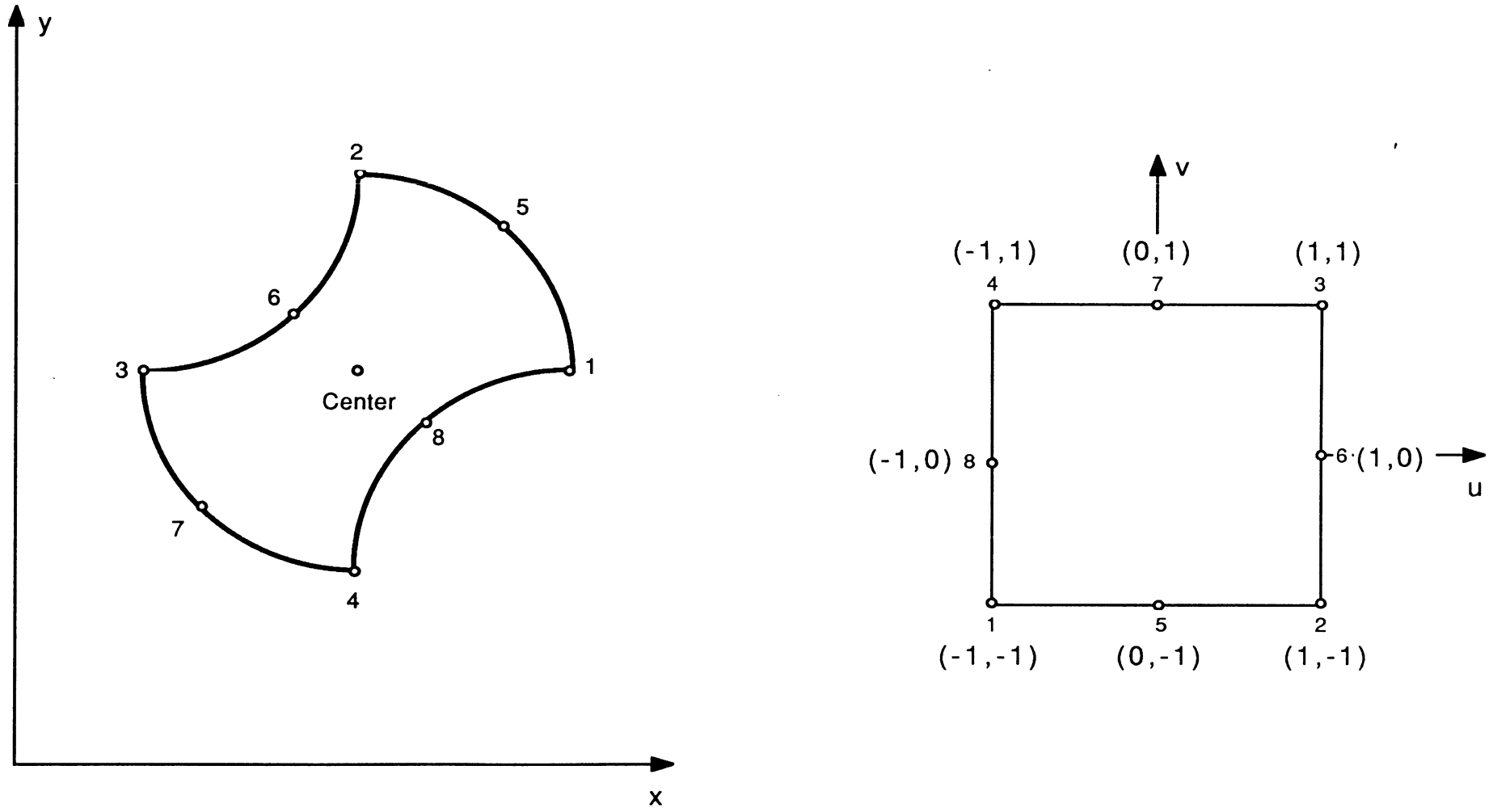


Illustration of Parametric Cell Mapping

Figure 3

Addendum Figure

Layered Volume Geometry Card

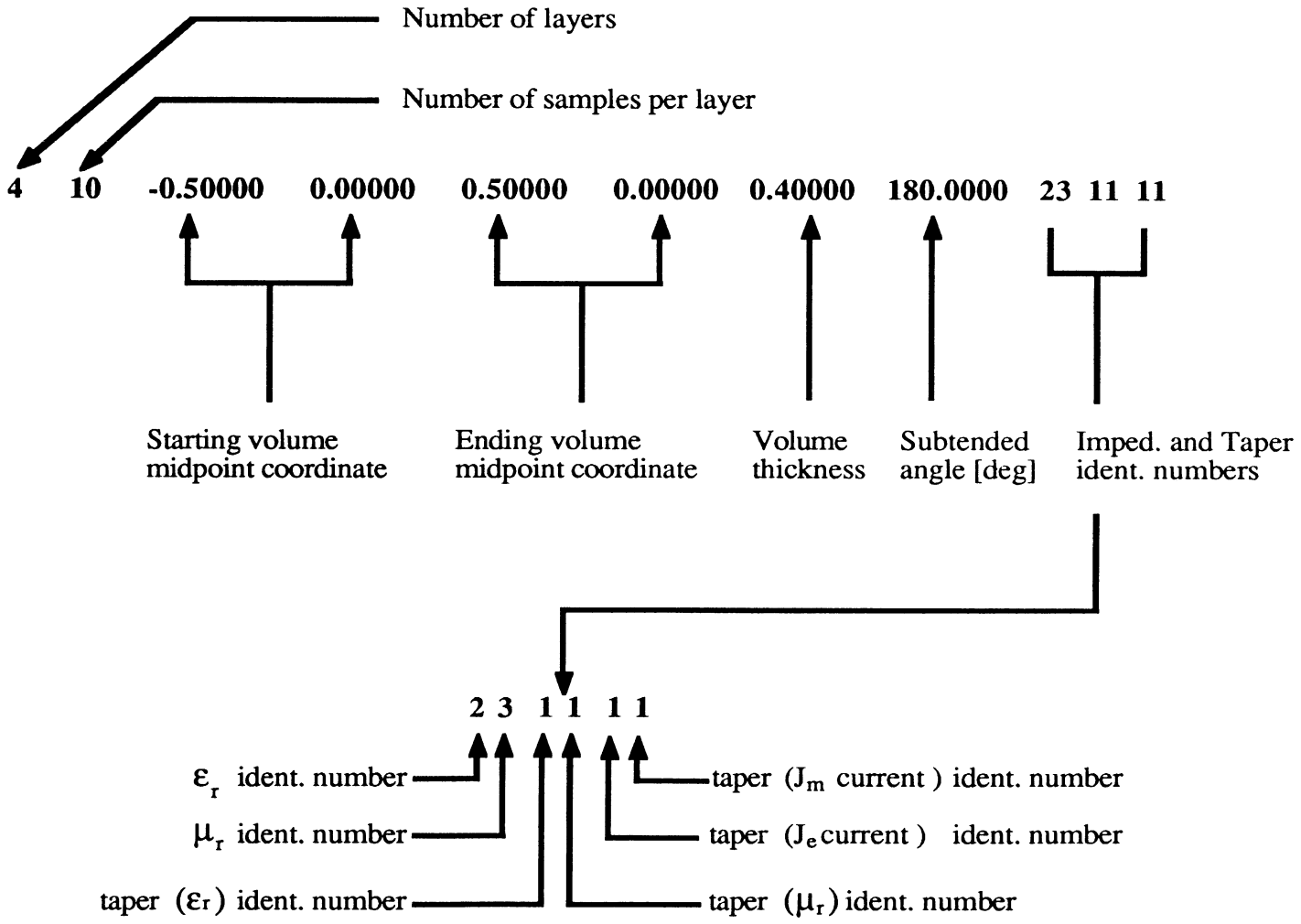


Figure 4

Arbitrary Quadrilateral Card

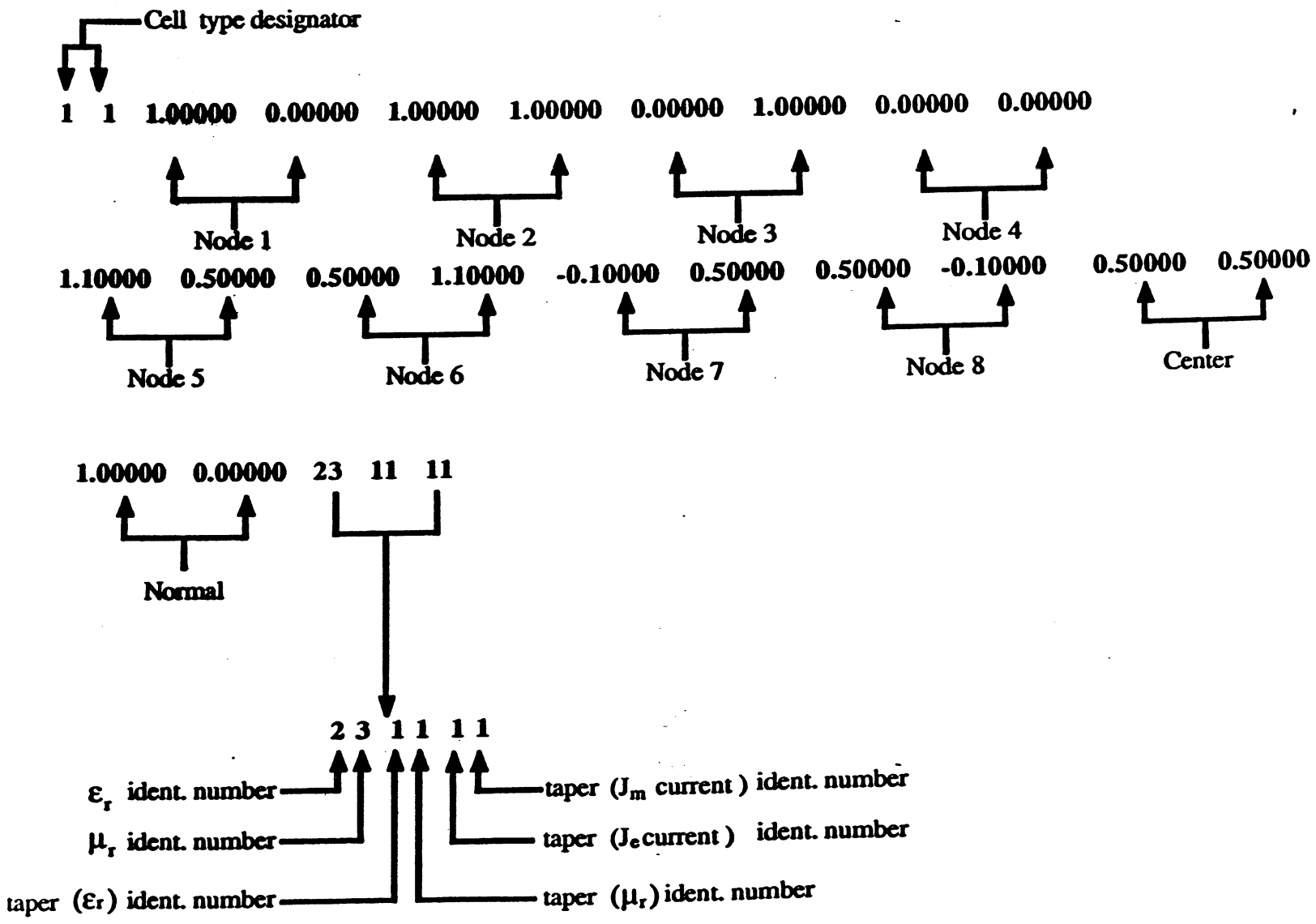
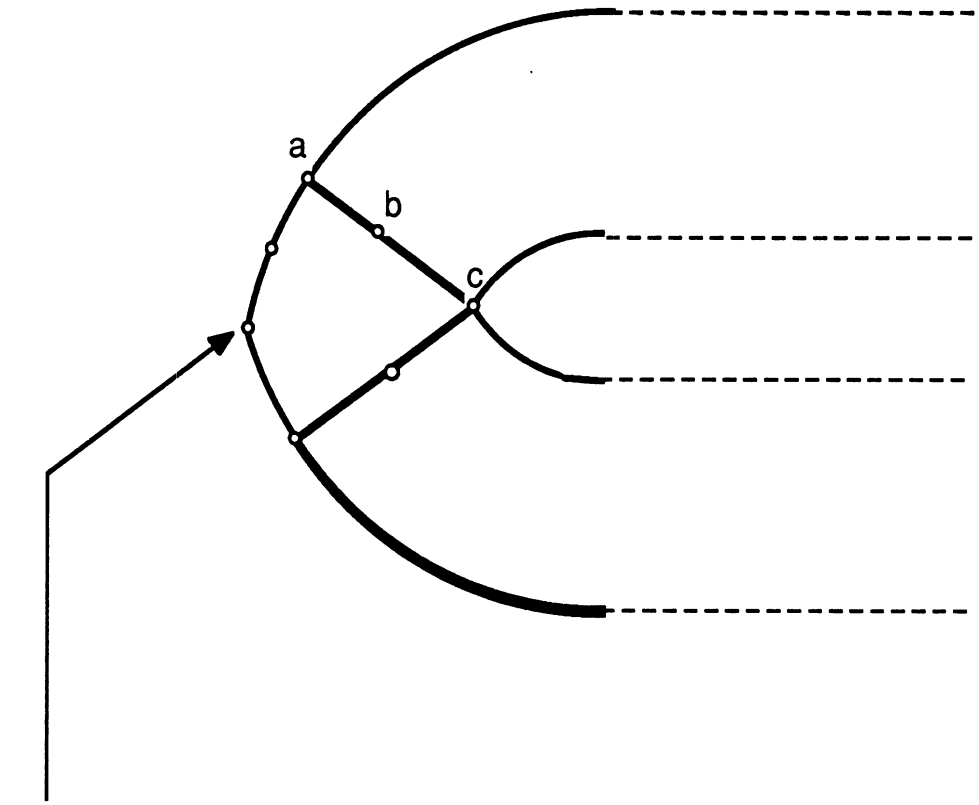


Figure 5

Tip Modeling



Tip of Ogive
Modeled with
an Arbitrary Volume Cell

Figure 6

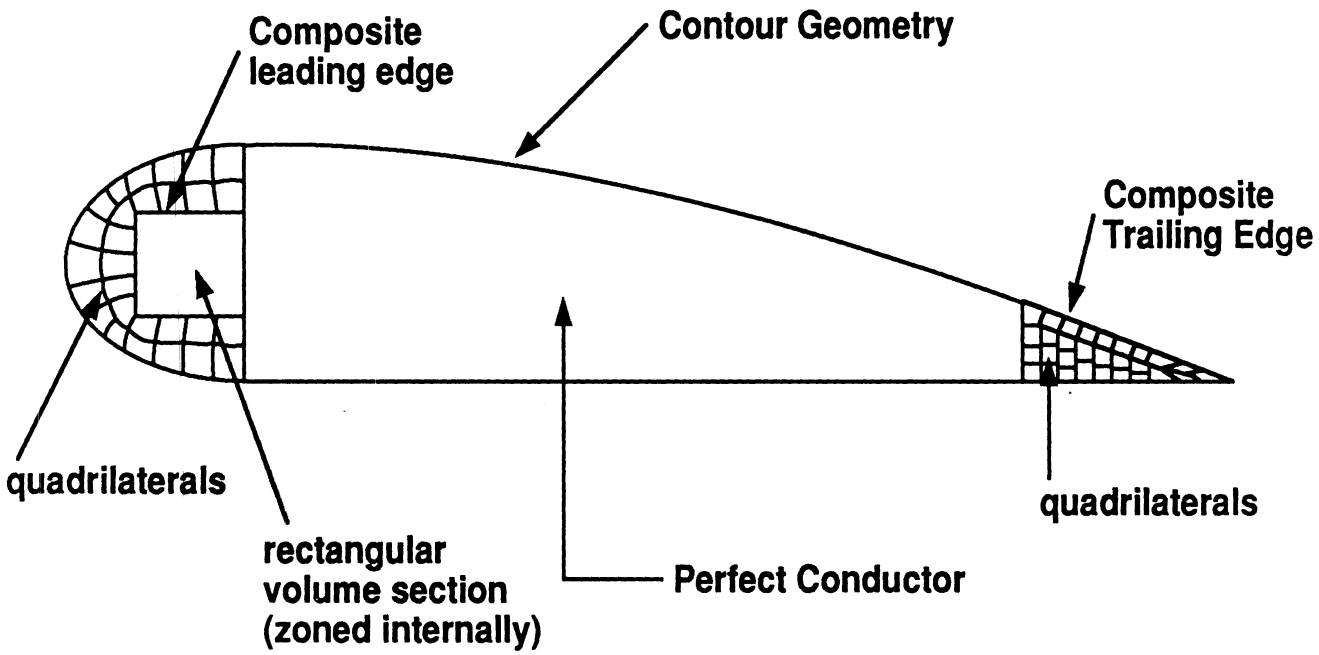
RUFCODE MODELING GEOMETRY

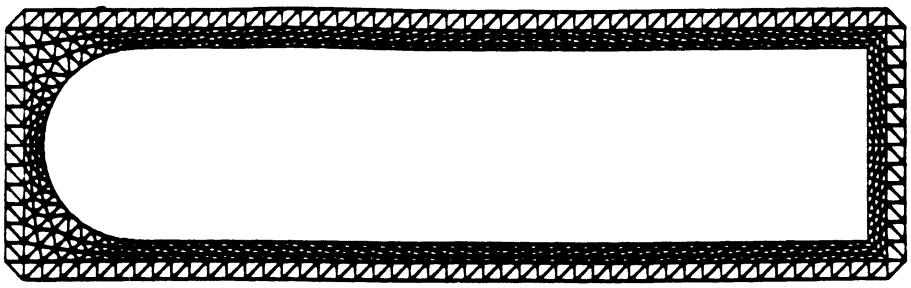
Geometry Components

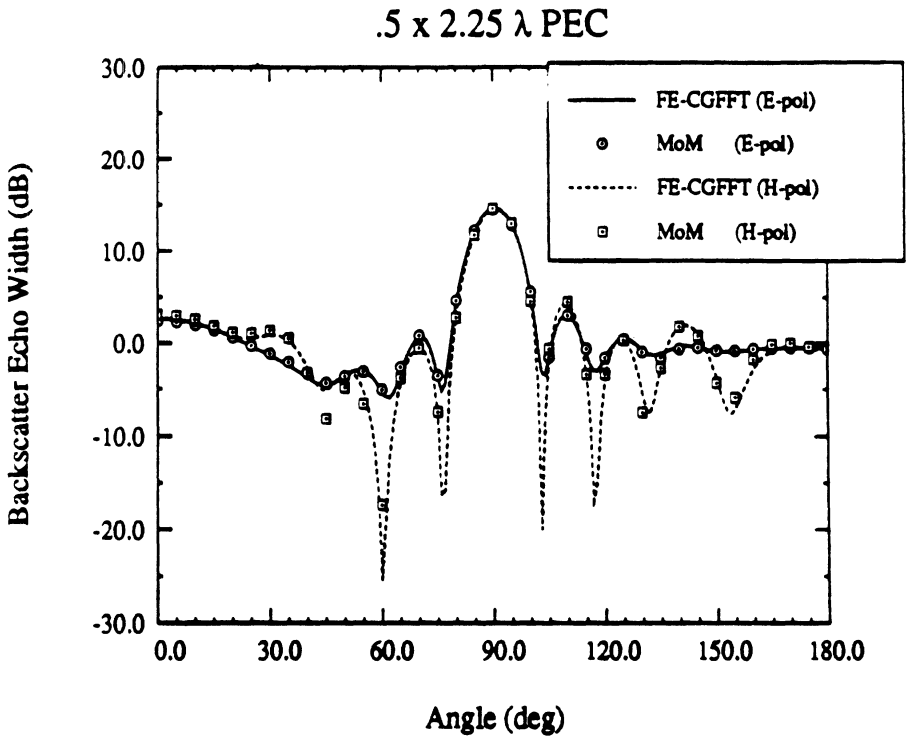
(1) Contour Geometry
- (a) Linear Segments
- (b) Arc Segments

(2) Volume Geometry
- (a) Rectangular Cells
- (b) Triangle Cells
- (c) 1st and 2nd Order Quadrilaterals

Example Structure

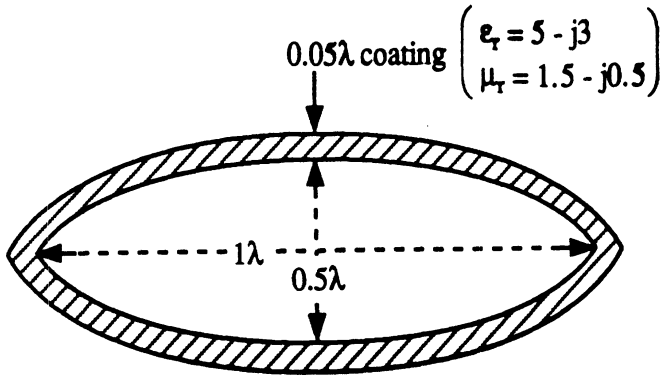


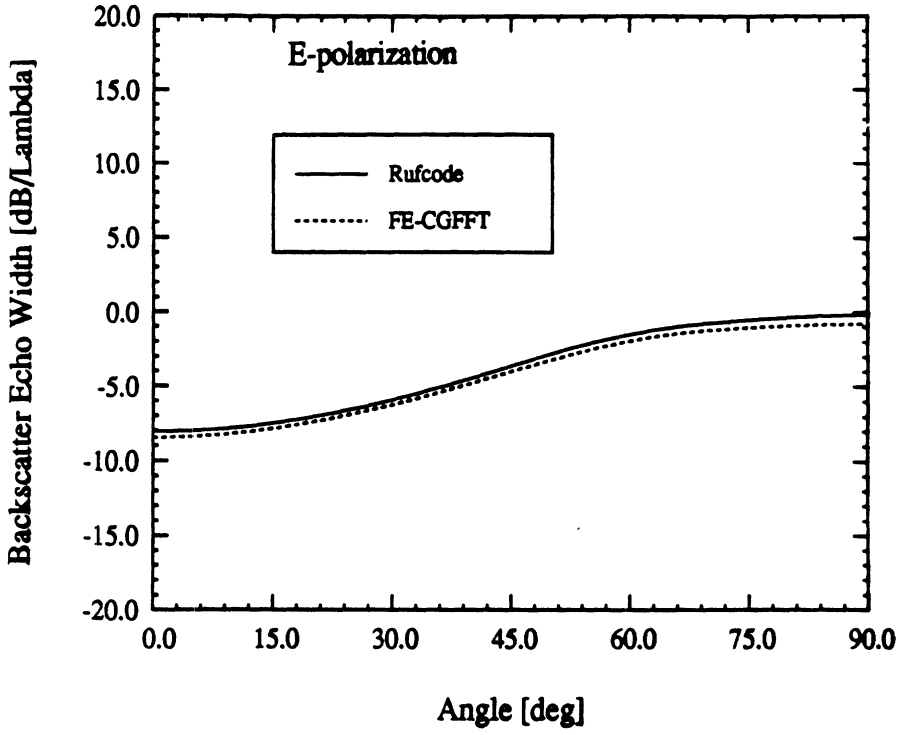


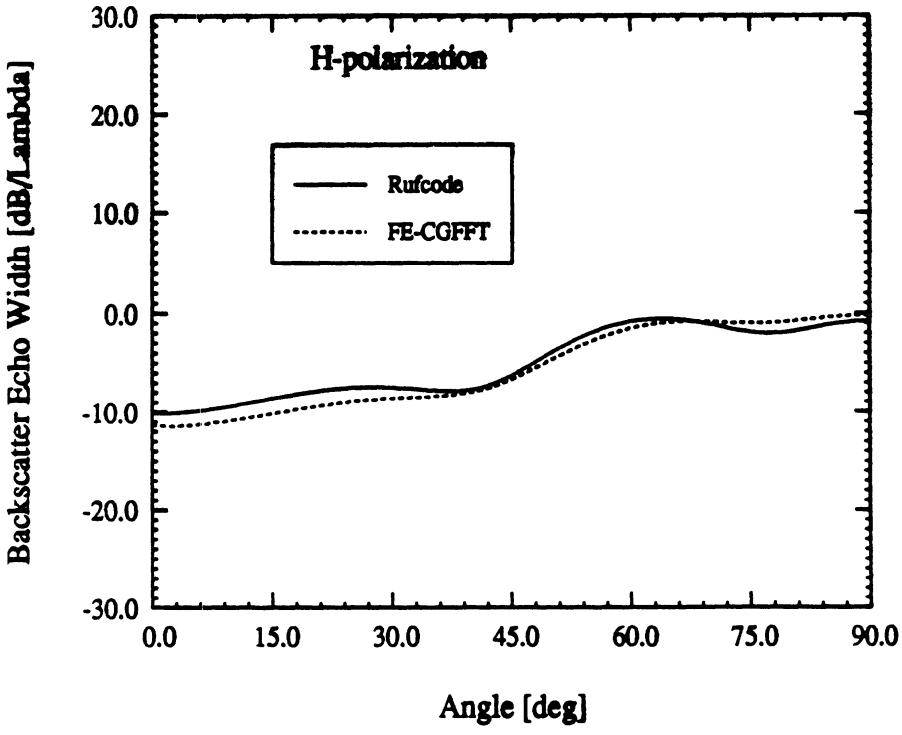


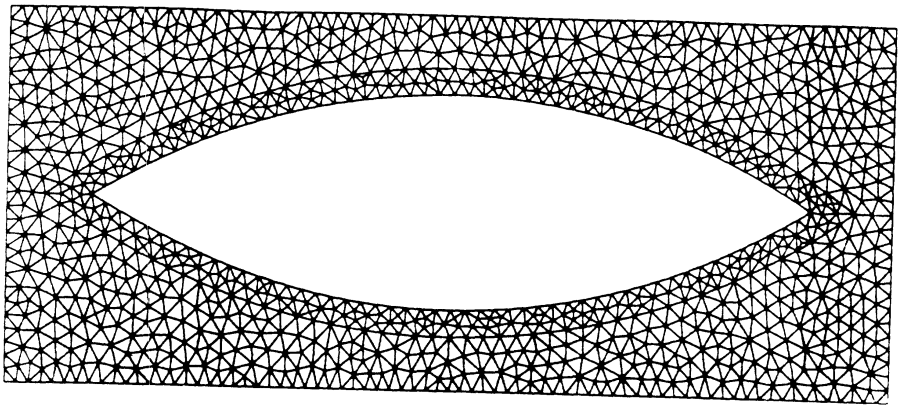
E_z and H_z backscatter echowidth patterns for the illustrated perfectly conducting cylinder.

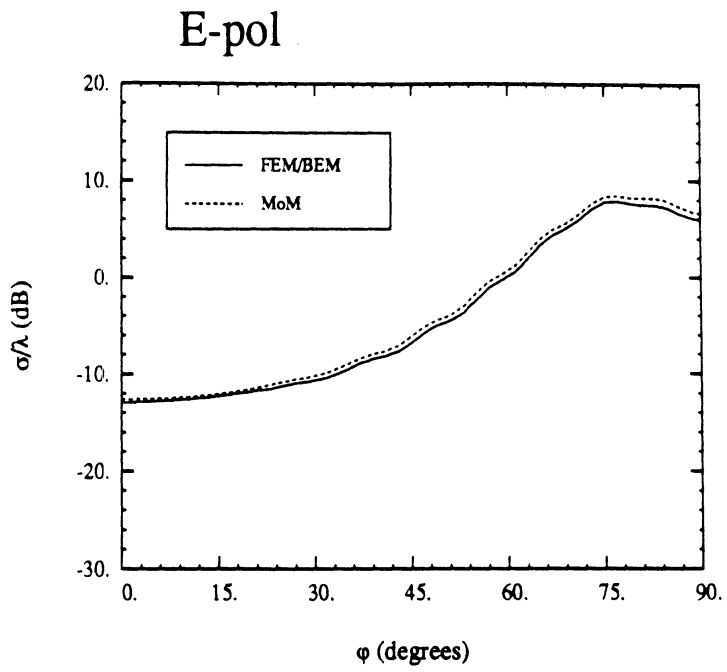
COATED OGIVE



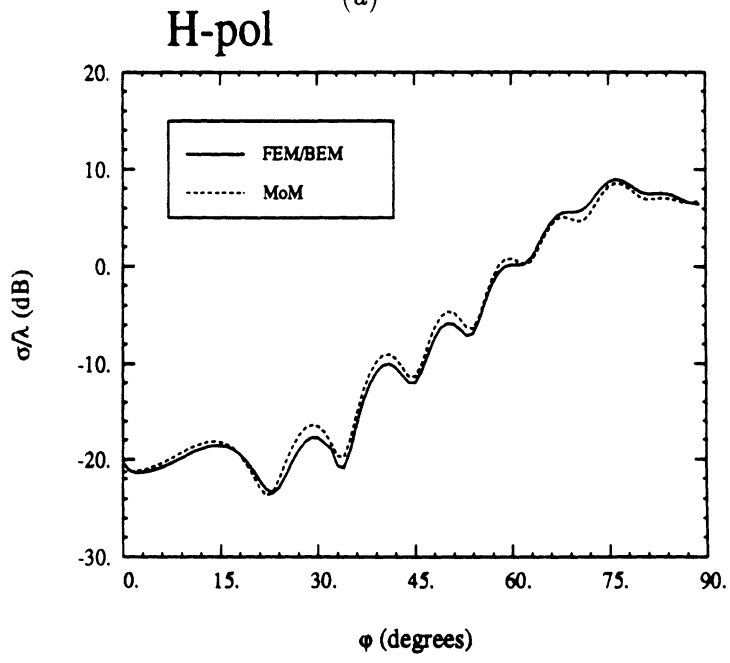








(a)



(b)

Coated Ogive
$4\lambda \times 1\lambda$
0.05λ coating
$\varepsilon_r = 3 - j5$
$\mu_r = 1.5 - j0.5$

//alek/users/+research/kempel.dir/parametric/src/rufcode.f **11/20/90 6:05 PM**

```
C*****************************************************************C
C*****************************************************************C
C                                                                 C
C        RUFCODE        -- VERSION 4 --(May   1990)-- M.A. Ricoy  C
C                                                     L.C. Kempel C
C                                                     J.L. Volakis C
C                                                                 C
C                       -- Parametric Cell Implementation         C
C                                                                 C
C*****************************************************************C
C  COPYRIGHT: THE UNIVERSITY OF MICHIGAN RADIATION LABORATORY 1988 C
C*****************************************************************C
C     NOTE: This program calls common subroutines in REDSUBS      C
C*****************************************************************C
C                                                                 C
C   Program RUFCODE computes the equivalent electric surface and  C
C   magnetic volume currents of a dielectric and/or magnetic two- C
C   dimensional body illuminated by a plane wave.  It also finds  C
C   the bistatic or backscattering cross sections of the body     C
C   Either H-polarized or E-polarized incident plane waves        C
C   can be specified.  The time convention used is exp(-iwt).     C
C   Body geometry coordinates should be entered in a CW direction. C
C                                                                 C
C*****************************************************************C
C                                                                 C
C  INPUT FORMAT                                                   C
C                                                                 C
C  CARD 1  FORMAT (18A4)       TITLE CARD; USE UP TO 72 COLUMNS   C
C                                                                 C
C  CARD 2  FORMAT (I5,F10.5,I5,A6)   KODE,WAVE,IPOL,FILE         C
C          KODE=0        COMPUTES BISTATIC SCATTERING PATTERN     C
C          KODE=1        COMPUTES BACKSCATTERING PATTERN          C
C          WAVE          WAVELENGTH                               C
C          IPOL=1        E-POLARIZATION                           C
C          IPOL=2        H-POLARIZATION                           C
C          FILE          PLOT FILE FOR OUTPUT DATA                C
C                                                                 C
C  CARD 3  FORMAT (4F10.5,2I2,F8.3)   FIRST,LAST,INK,ANGINC,      C
C                                     IMAG,IPRINT,OFFDB           C
C          FIRST         INITIAL INCIDENT ANGLE                   C
C          LAST          FINAL ANGLE                              C
C          INK           ANGULAR INCREMENT                        C
C          ANGINC        INCIDENT ANGLE FOR BISTATIC COMPUTATIONS C
C          IMAG          1=COMPUTE GAP IMPEDANCE                  C
C                        2=INCLUDE IMAGE WAVE                     C
C          IPRINT        3=PRINT ABSORBER STATISTICS              C
C                        4=PRINT MATRIX ELEMENTS (MAGNITUDE)      C
C                        5=PRINT MATRIX ELEMENTS (COMPLEX)        C
c                        6=print geometry plot file               c
c                        7=PRINT Parametric Cell Principle Nodes  C
c                        8= print both geometry and Parametric Cellsc
c                        9= 3 & 8
C          OFFDB         OFFSET(DB)                               C
C                                                                 C
C ********************** MATERIAL SPECIFICATIONS ****************** C
C                                                                 C
C  CARD 4  FORMAT (I3,4F13.3) SPECZ,ZA,ZB                         C
C          SPECZ         IDENTIFICATION NUMBER FOR DESIGNATED     C
C                        IMPEDANCE PROPERTIES (=000 FOR END)      C
C          ZA            COMPLEX PERMITTIVITY OR PERMEABILITY(START)C
C          ZB            COMPLEX PERMITTIVITY OR PERMEABILITY (END) C
C                                                                 C
C ********************** TAPER SPECIFICATIONS ********************* C
C                                                                 C
C  CARD 5  FORMAT (I3,I3,5F10) SPECT,TINFO,C,P                    C
C          SPECT         IDENTIFICATION NUMBER FOR DESIGNATED     C
C                        TAPER SPECIFICATION (=000 FOR END)       C
C          TINFO         TYPE OF TAPERING EMPLOYED: SEE SUB. TAPER C
C          C             COORDINATE ARGUEMENTS: SEE SUB. TAPER    C
C          P             EXPON. ARGUEMENT: SEE SUB. TAPER         C
C                                                                 C
C ************************ BODY VOLUME DATA ********************** C
C                                                                 C
C  CARD 6  FORMAT (I3,I5,5F10.5,3I3) NL,N,XA,YA,XB,YB,THIC,       C
C                                    INFZ,INFZT,INFI              C
C          NL            NUMBER OF LAYERS WITHIN VOLUME           C
C                        (=000 FOR END)                           C
C          N             NUMBER OF SAMPLING POINTS PER LAYER      C
C          XA,YA,XB,YB   VOLUME ENDPOINTS                         C
C          THIC          TOTAL THICKNESS OF VOLUME                C
C          INFZ          PERMITTIVITY & PERMEABILITY IDENTIF. NUM. C
C          INFZT         MATERIAL TAPERING INDENTIF. NUMBER       C
C          INFI          CURRENT TAPERING INDENTIF. NUMBER        C
C                                                                 C
C ************************ BODY CONTOUR DATA ********************** C
```

```
C  CARD 7  FORMAT (F5.1,I3,4F10.5,2I5,I3) CANGLE,N,CXA,CYA,CXB,     C
C                                         CYB,INFZ,INFZT,INFI       C
C          CANGLE       ANGLE SUBTENDED BY CONTOUR SEGMENT         C
C                       (=999 FOR END)                             C
C          N            NUMBER OF SAMPLING POINTS PER SEGMENT      C
C          CXA,CYA      STARTING POINT OF CONTOUR SEGMENT          C
C          CXB,CYB      ENDING POINT OF CONTOUR SEGMENT            C
C          INFZ         PERMITTIVITY & PERMEABILITY IDENTIF. NUM.  C
C                       ON EACH SIDE OF CONTOUR                    C
C          INFZT        MATERIAL TAPERING IDENTIF. NUMBER          C
C                       ON EACH SIDE OF CONTOUR                    C
C          INFI         CURRENT TAPERING INDENTIF. NUMBER          C
C                                                                  C
C******************************************************************C
C                                                                  C
C    IO SPECIFICATIONS                                             C
C    5: INPUT DATA;  6: OUTPUT(PRINTER);                           C
C                                                                  C
C******************************************************************C
      PARAMETER (MDIM=3000,MDIM2=3000)
      COMPLEX F(MDIM2,MDIM2),PHI(MDIM2),SV(MDIM2)
      COMPLEX YE(MDIM,7),YM(MDIM,7),RYE,RYM,RYEI,RYEO
      COMPLEX SUM,FINC(MDIM),DUM1,DUM2,SIGMA,RYMI,RYMO
      COMPLEX YIMP(6,MDIM),DEL,DUM3,CI
      REAL LAST,INK,EMUL(MDIM),HMUL(MDIM),FNORM(MDIM2)
      REAL X(MDIM,5,5),Y(MDIM,5,5),XN(MDIM,5,5),YN(MDIM,5,5)
      REAL XAA(MDIM),YAA(MDIM),XBB(MDIM),YBB(MDIM)
      REAL CXAA(MDIM),CYAA(MDIM),XPL(MDIM),YPL(MDIM)
      REAL TAUS(MDIM,5),DSQ(MDIM,5),DOT1(MDIM),DOT2(MDIM)
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      DIMENSION XI(MDIM),YI(MDIM),XNI(MDIM),YNI(MDIM),S(MDIM,5)
      DIMENSION NSEG(MDIM,3),KPVT(MDIM2),FILE(4)
      DIMENSION THEA(361),SCATA(361),PHASEA(361),fprint(mdim,40)
      dimension fvprint(mdim,40)
      CHARACTER*4 IPP(2)
      integer ident(mdim)
      character*6 pfilei
      character*20 pfile,gvfile,cfile,gcfile
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
      DATA IPP/'EEEE','HHHH'/
C
C****** READ INPUT DATA AND GENERATE BODY PROFILE

CCC       OPEN(5,FILE='RUFCODE_INPUT')
    5  READ(5,140,END=999) ID
       READ (5,200) KODE,WAVE,IPOL,pFILEi
       READ (5,210) FIRST,LAST,INK,ANGINC,IMAG,IPRINT,OFFDB
       WRITE (6,150) ID
       WRITE (6,160)
       gvfile = pfilei//'.vol'
       gcfile = pfilei//'.con'
       cfile = pfilei//'.cells'
       pfile = pfilei//'.plt'
       NINC=1
       NBIT=0
       NANGLE=1+IFIX((LAST-FIRST)/INK)
       IF (KODE.EQ.0) NBIT=NANGLE
       IF (KODE.EQ.1) NINC=NANGLE
       CALL GEOVOL(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,
      1          EMUL,HMUL,WAVE,TAUS,XAA,YAA,XBB,YBB,id,ident,nrmcell)
       WRITE (6,161)
       CALL abnormal(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,
      1          EMUL,HMUL,WAVE,TAUS,XAA,YAA,XBB,YBB,id,ident)
       WRITE(6,170)
       CALL GEOCON(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,L,
      1          EMUL,HMUL,WAVE,TAUS,CXAA,CYAA,id,ident)
       MMK = L
       MTOT = L
       XK=TWOPI
C
C***** GENERATE MATRIX ELEMENTS
C
       CALL MTXRED(MDIM,MDIM2,MTOT,M,IPOL,XK,X,Y,XN,YN,DSQ,NSEG,
      &          YE,YM,F,TAUS,IPRINT,nrmcell)
       PRINT *,'MATRIX ELEMENTS GENERATED'
C
C*************** TAKE INTO ACCOUNT POLARIZATION ***************
C                                                               *
C     IMPEDANCE         E-POLARIZATION          H-POLARIZATION  *
C       YIMP1               1/Ur                    1/Er        *
C       YIMP2             Er - 1/Ur               Ur - 1/Er     *
C       YIMP3             d/ds(1/Ur)              d/ds(1/Er)    *
C       YIMP4             d/dn(1/Ur)              d/dn(1/Er)    *
C       YIMP5             (Ur-1)/Ur               (Er-1)/Er     *
C       YIMP6              1/Ur out                1/Er out     *
C                                                               *
C***************************************************************
```

```
       IF(IPOL .EQ. 2) THEN
         IMUL=1
          ELSE
           IMUL=-1
       ENDIF
       DO 15 I=1,MTOT
          IF(IPOL .EQ. 2) GO TO 10
          IF(NSEG(I,3) .EQ. 1) THEN
            YIMP(1,I)=YM(I,1)
            YIMP(2,I)=YE(I,2)
            YIMP(3,I)=YM(I,3)
            YIMP(4,I)=YM(I,4)
            YIMP(5,I)=0.
            YIMP(6,I)=0.
              ELSE
            YIMP(1,I)=YM(I,1)
            YIMP(2,I)=0.
            YIMP(3,I)=0.
            YIMP(4,I)=0.
            YIMP(5,I)=YM(I,5)
            YIMP(6,I)=YM(I,6)
          ENDIF
          GO TO 15
10        IF(NSEG(I,3) .EQ. 1) THEN
            YIMP(1,I)=YE(I,1)
            YIMP(2,I)=YM(I,2)
            YIMP(3,I)=YE(I,3)
            YIMP(4,I)=YE(I,4)
            YIMP(5,I)=0.
            YIMP(6,I)=0.
              ELSE
            YIMP(1,I)=YE(I,1)
            YIMP(2,I)=0.
            YIMP(3,I)=0.
            YIMP(4,I)=0.
            YIMP(5,I)=YE(I,5)
            YIMP(6,I)=YE(I,6)
          ENDIF
15     CONTINUE
C
C***** COMPUTE INCIDENT FIELD
C
20     TETA=RED*ANGINC
       CT1=COS(TETA)
       ST1=SIN(TETA)
       CT2=COS(TWOPI-TETA)
       ST2=SIN(TWOPI-TETA)
C
       DO 50 I=1,MTOT
            IF(NSEG(I,3) .EQ. 1) THEN
              XNI(I)=XN(I,3,3)
              YNI(I)=YN(I,3,3)
              XI(I)=X(I,3,3)
              YI(I)=Y(I,3,3)
              XPL(I) = X(I,3,3)
              YPL(I) = Y(I,3,3)
            ELSE
              XNI(I)=XN(I,3,1)
              YNI(I)=YN(I,3,1)
              XI(I)=X(I,3,1)
              YI(I)=Y(I,3,1)
              XPL(I) = X(I,3,1)
              YPL(I) = Y(I,3,1)
            ENDIF
          FINC(I)=CEXP(-CI*XK*(XI(I)*CT1+YI(I)*ST1))
          DOT1(I)=XNI(I)*CT1+YNI(I)*ST1
          DOT2(I)=XNI(I)*ST1-YNI(I)*CT1
          FNORM(I)=CABS(F(I,I))
          IF(NSEG(I,3).EQ.1) THEN
            PHI(I)=FINC(I)*(YIMP(3,I)*DOT2(I)-YIMP(4,I)*DOT1(I)
     &           -CI*XK*YIMP(2,I))
          ELSE
            PHI(I)=-CI*XK*YIMP(5,I)*FINC(I)*DOT1(I)
          ENDIF
          IF(IMAG .NE. 2) GO TO 40
C *add image currents*
             DUM1=CEXP(-CI*XK*(XI(I)*CT2+YI(I)*ST2))
             DUM2=XNI(I)*CT2+YNI(I)*ST2
             DUM3=XNI(I)*ST2-YNI(I)*CT2
          IF(NSEG(I,3).EQ.1) THEN
             PHI(I)=PHI(I)+IMUL*DUM1*(YIMP(3,I)*DUM3-YIMP(4,I)*DUM2
     &                  -CI*XK*YIMP(2,I))
               ELSE
              PHI(I)=PHI(I)+IMUL*(-CI*XK*YIMP(5,I)*DUM1*DUM2)
          ENDIF
40        PHI(I)=PHI(I)/FNORM(I)
          DO 45 J=1,MTOT
```

```
              F(I,J)=F(I,J)/FNORM(I)
45          CONTINUE
50      CONTINUE
C
C********* L-U DECOMPOSE MATRIX [A] AND SOLVE FOR CURRENTS
C
          PRINT *,'DECOMPOSING MATRIX'
        CALL CGECO(F,MDIM2,MMK,KPVT,RC,SV)
        CALL CGESL(F,MDIM2,MMK,KPVT,PHI,0)
C
C***** APPLY CURRENT TAPERING
C
        DO 65 I=1,MTOT
           IF(NSEG(I,3).NE.1) THEN
           PHI(I)= EMUL(I)*PHI(I)
             ELSE
                PHI(I)= HMUL(I)*PHI(I)
           ENDIF
65      CONTINUE
C
C***** PRINT OUT STATISTICS ON ABSORBER SURFACE
C
70      IT=MTOT-M
        WRITE(6,400) IPP(IPOL),M,IT,NINC,NBIT,WAVE,RC
        IF(IPRINT .EQ. 3.OR.iprint.eq.9) THEN
           WRITE (6,150) ID
           WRITE (6,350)ANGINC
C
           IF(IPOL .EQ. 1) THEN
               WRITE(6,376)
                  ELSE
                      WRITE(6,375)
           ENDIF
         endif
71       DO 75 I=1,MTOT
           IF(IPRINT .EQ. 3.OR.iprint.eq.9) THEN
             IF(I .EQ. M+1 .AND. IPOL .EQ. 1) WRITE(6,378)
             IF(I .EQ. M+1 .AND. IPOL .EQ. 2) WRITE(6,377)
           ENDIF
           AMPM=0.
           PHASEM=0.
           AMPE=0.
           PHASEM=0.
           AMP=0.
           PHASE=0.
           IF(NSEG(I,3).NE.1) GO TO 73
           DEL=PHI(I)
           DELR=REAL(DEL)
           DELI=AIMAG(DEL)
           AMPM=SQRT(DELR*DELR+DELI*DELI)
           AMPM=AMPM/Z0
           IF(ABS(DELR) .GT. 1.E-8)  THEN
             PHASEM=DIG*ATAN3(DELI,DELR)
               ELSE
                 PHASEM=0.
            ENDIF
           AMP=AMPM
           PHASE=PHASEM
           RYE = 1./YE(I,1)
           RYM = 1./YM(I,1)
           IF(IPRINT .EQ. 3) WRITE(6,395) I,NSEG(I,3),XI(I),YI(I),
     &                                      S(I,3),DSQ(I,3),RYE,RYM,
     &                                      AMP,PHASE
           IF(IPRINT .EQ. 9) WRITE(6,395) I,NSEG(I,3),XI(I),YI(I),
     &                                      S(I,3),DSQ(I,3),RYE,RYM,
     &                                      AMP,PHASE
           GO TO 75
73         DEL=PHI(I)
           DELR=REAL(DEL)
           DELI=AIMAG(DEL)
           AMPE=SQRT(DELR*DELR+DELI*DELI)
           AMPE=AMPE/Z0
           PHASEE=DIG*ATAN3(DELI,DELR)
           AMP=AMPE
           PHASE=PHASEE
           RYEI= 1./YE(I,1)
           IF(AIMAG(RYEI) .GT. 1.E5) RYEI=(1.0, 9999.99)
           RYEO= 1./YE(I,6)
           IF(AIMAG(RYEO) .GT. 1.E5) RYEO=(1.0, 9999.99)
           RYMI= 1./YM(I,1)
           IF(AIMAG(RYMI) .GT. 1.E5) RYMI=(1.0, 9999.99)
           RYMO= 1./YM(I,6)
           IF(AIMAG(RYMO) .GT. 1.E5) RYMO=(1.0, 9999.99)
           IF(IPRINT .EQ. 3) WRITE(6,396) I,NSEG(I,3),XI(I),YI(I),
     &                         S(I,3),DSQ(I,3),RYEI,RYEO,RYMI,RYMO,
     &                         AMP,PHASE
           IF(IPRINT .EQ. 9) WRITE(6,396) I,NSEG(I,3),XI(I),YI(I),
```

```
     &                     S(I,3),DSQ(I,3),RYEI,RYEO,RYMI,RYMO,
     &                     AMP,PHASE
75    CONTINUE
      WRITE (6,150) ID
      IF (KODE.EQ.1) WRITE(6,600) IPP(IPOL)
      IF (KODE.EQ.0) WRITE(6,800) ANGINC,IPP(IPOL)
      WRITE(6,840)
      print*,pfile
      open(unit=8,file=pfile,status='unknown')
C
C***** COMPUTE SCATTERING CROSS SECTIONS
C
      DO 100 INDX=1,NANGLE
         TETA=RED*(FIRST+(INDX-1)*(LAST-FIRST)/(NANGLE-1))
         CT1=COS(TETA)
         ST1=SIN(TETA)
         CT2=COS(TWOPI-TETA)
         ST2=SIN(TWOPI-TETA)
         DO 80 I=1,MTOT
            FINC(I)=CEXP(-CI*XK*(XI(I)*CT1+YI(I)*ST1))
            DOT1(I)=XNI(I)*CT1+YNI(I)*ST1
            DOT2(I)=XNI(I)*ST1-YNI(I)*CT1
80       CONTINUE
         IF (KODE.EQ.0) GOTO 92
C
C***** COMPUTE NEW CURRENTS FOR BACKSCATTERING
C
         DO 90 I=1,MTOT
            IF(NSEG(I,3).EQ.1) THEN
              PHI(I)=FINC(I)*(YIMP(3,I)*DOT2(I)-YIMP(4,I)*DOT1(I)
     &              -CI*XK*YIMP(2,I))
                ELSE
              PHI(I)=-CI*XK*YIMP(5,I)*FINC(I)*DOT1(I)
            ENDIF
86          IF(IMAG .NE. 2) GO TO 87
C *add image currents*
            DUM1=CEXP(-CI*XK*(XI(I)*CT2+YI(I)*ST2))
            DUM2=XNI(I)*CT2+YNI(I)*ST2
            DUM3=XNI(I)*ST2-YNI(I)*CT2
         IF(NSEG(I,3).EQ.1) THEN
            PHI(I)=PHI(I)+IMUL*DUM1*(YIMP(3,I)*DUM3-YIMP(4,I)*DUM2
     &                  -CI*XK*YIMP(2,I))
              ELSE
             PHI(I)=PHI(I)+IMUL*(-CI*XK*YIMP(5,I)*DUM1*DUM2)
         ENDIF
87       PHI(I)=PHI(I)/FNORM(I)
90       CONTINUE
C
56       CALL CGESL(F,MDIM2,MMK,KPVT,PHI,0)
C
C***** APPLY CURRENT TAPERING
C
          DO 91 I=1,MTOT
               IF(NSEG(I,3).NE.1) THEN
               PHI(I)=EMUL(I)*PHI(I)
               ELSE
               PHI(I)=HMUL(I)*PHI(I)
               ENDIF
91       CONTINUE
C
C***** ADD UP THE CURRENTS FOR FAR FIELD
C
92       SUM=CMPLX(0.,0.)
         DO 98 I=1,MTOT
            ARG1=XK*DSQ(I,3)/2.*DOT2(I)
            ARG2=XK*TAUS(I,3)/2.*DOT1(I)
            IF(ABS(ARG1) .LT. 1.E-4) THEN
              SINC1=1
                ELSE
              SINC1=SIN(ARG1)/ARG1
            ENDIF
            IF(ABS(ARG2) .LT. 1.E-4) THEN
              SINC2=1
                ELSE
              SINC2=SIN(ARG2)/ARG2
            ENDIF
            FACTOR=SINC1*SINC2
            IF(NSEG(I,3).EQ.1) THEN
               SUM=SUM+(DSQ(I,3)*TAUS(I,3)*FINC(I)*
     &          PHI(I)*FACTOR)
                 ELSE
              SUM=SUM-(CI/XK*PHI(I)*SINC1*DSQ(I,3)*FINC(I))
            ENDIF
98       CONTINUE
         SIGMA=XK*SUM*SUM/4.*10.**(OFFDB/10.)
         SIGMA=SIGMA
         AMPR= REAL(SIGMA)
```

```
         AMPI=AIMAG(SIGMA)
         AMP=  CABS(SIGMA)
         IF(AMP.EQ.0) THEN
           SCATA(INDX)=-500.
           GO TO 99
         ENDIF
          SCATA(INDX)=10.*ALOG10(AMP)
99         THEA(INDX)=TETA/RED
         PHASEA(INDX)=DIG*ATAN3(AMPI,AMPR)
         WRITE(6,920) THEA(INDX),AMP,PHASEA(INDX),SCATA(INDX)
         write(8,950) thea(indx),scata(indx),phasea(indx)
100   CONTINUE
C
C***** COMPUTE GAP IMPEDANCE
C
       IF(IMAG .EQ. 1) THEN
         CALL ZIMP(MDIM,MTOT,XK,X,Y,XN,YN,WAVE,
     &             IPOL,DSQ,TAUS,NSEG,PHI,ANGINC)
       ENDIF
C
C***** FIND MAX AND MIN VALUES OF SCAT AND PHASE
C
      SCAMIN=SCATA(1)
      SCAMAX=SCATA(1)
      FASMIN=PHASEA(1)
      FASMAX=PHASEA(1)
      DO 333 I=2,NANGLE
         IF( SCATA(I) .LT. SCAMIN) SCAMIN= SCATA(I)
         IF( SCATA(I) .GT. SCAMAX) SCAMAX= SCATA(I)
         IF(PHASEA(I) .LT. FASMIN) FASMIN=PHASEA(I)
         IF(PHASEA(I) .GT. FASMAX) FASMAX=PHASEA(I)
 333  CONTINUE
C
C
C****** CREATE Parametric Cell File
       IF(IPRINT .EQ. 7 .OR. iprint .eq. 8 .OR. iprint .eq. 9) THEN
          open(unit = 2, file=cfile, status='unknown')
          write(2,*)'Cell  Left Side        Right Side'
          do 957 lk = 1,mtot
             if(nseg(lk,3).eq.1.AND.nseg(lk,2).eq.1) then
                 WRITE(2,960)lk,x(lk,5,1),y(lk,5,1),x(lk,5,5),y(lk,5,5)
                 write(2,961)x(lk,3,1),y(lk,3,1),x(lk,3,5),y(lk,3,5)
                 write(2,961)x(lk,1,1),y(lk,1,1),x(lk,1,5),y(lk,1,5)
                 write(2,*)
             endif
 957       continue
       ENDIF
C****** CREATE Geometry plot File
       IF(IPRINT .EQ. 6 .OR. iprint .eq.8 .OR. iprint.eq.9) THEN
          smdel = 0.001
          open(unit = 3, file=gvfile, status='unknown')
          open(unit = 4, file=gcfile, status='unknown')
          nvol = -1
          nvvol = -3
          ncmax = 0
          nvmax = 0
          do 958 iid = 1,id
           nvol = nvol + 2
           nvvol = nvvol + 4
           ncon = 0
           nvcon = 0
           jvol = 0
           do 956 lk = 1,mtot
            if(ident(lk).eq.iid) then
               if(nseg(lk,3).eq.1) then
c Volume Element
                 if(ident(lk).ne.1) then
c Test if same contour
                  testx =abs(x(lk,5,1))
                  testy = abs(y(lk,5,1))
                  xnb = abs(fvprint(nvold,nvvol-2))
                  ynb = abs(fvprint(nvold,nvvol-1))
                  if(xnb-smdel.LT.testx.AND.xnb+smdel.GT.testx) then
                   if(ynb-smdel.LT.testy.AND.ynb+smdel.GT.testy) then
                    nvvol = nvvol - 4
                    nvcon = nvold
                   endif
                  endif
                 endif
c
                  jvol = 1
                  nvcon = nvcon + 1
                  fvprint(nvcon,nvvol) = x(lk,1,1)
                  fvprint(nvcon,nvvol+1) = y(lk,1,1)
                  fvprint(nvcon,nvvol+2) = x(lk,5,1)
                  fvprint(nvcon,nvvol+3) = y(lk,5,1)
                  nvcon = nvcon + 1
```

```
                fvprint(nvcon,nvvol) = x(lk,1,5)
                fvprint(nvcon,nvvol+1) = y(lk,1,5)
                fvprint(nvcon,nvvol+2) = x(lk,5,5)
                fvprint(nvcon,nvvol+3) = y(lk,5,5)
              else
c Test if same contour
               if(nvol.ne.1) then
                testx = abs(x(lk,1,1))
                testy = abs(y(lk,1,1))
                xnb = abs(fprint(nold,nvol-2))
                ynb = abs(fprint(nold,nvol-1))
                if(xnb-smdel.LT.testx.AND.xnb+smdel.GT.testx) then
                 if(ynb-smdel.LT.testy.AND.ynb+smdel.GT.testy) then
                  nvol = nvol - 2
                  ncon = nold
                 endif
                endif
               endif
c
                ncon = ncon + 1
                fprint(ncon,nvol) = x(lk,1,1)
                fprint(ncon,nvol+1) = y(lk,1,1)
                ncon = ncon + 1
                fprint(ncon,nvol) = x(lk,5,1)
                fprint(ncon,nvol+1) = y(lk,5,1)
              endif
            endif
 956        continue
            if(jvol.eq.1) then
               nvol = nvol - 2
            else
               nvvol = nvvol - 4
            endif
            if(ncon.gt.ncmax) ncmax = ncon
            if(nvcon.gt.nvmax) nvmax = nvcon
            nvold = nvcon
            nold = ncon
 958      continue
c Print it
c
          do 959 lk = 1,ncmax
                write(4,962)(fprint(lk,kl),kl = 1,nvol+1)
 959        continue
          do 954 lk = 1,nvmax
                write(3,962)(fvprint(lk,kl),kl = 1,nvvol+3)
 954        continue

       ENDIF
C
C      GO TO 5
C
C***** STOP WHEN INPUT DATA RUNS OUT
C
  999 WRITE(6,150)
      CLOSE(2)
      CLOSE(3)
C
C***** FORMATS
C
 140  FORMAT (18A4)
 150  FORMAT('1',35X,'THE UNIVERSITY OF MICHIGAN RADIATION LABORATORY'
     &        ,///,12X,18A4,5X,'** PROGRAM RUFCODE')
 160  FORMAT('0',27X,'INPUT GEOMETRY LISTING',//,6X,'TYPE',9X,
     &  'VOL',1X,'LAY',2X,'NUM',12X,'ENDPOINTS OF THE SEGMENT',/,19X,
     &  'NUM',1X,'NUM',2X,'PTS',7X,'XA',8X,'YA',8X,'XB',8X,'YB',8X,
     &  'Thick',5X,'Angle')
 161  FORMAT(/,6X,'TYPE',11X,
     &  'NUM',5X,'Center Point of Cell(X,Y)',9X,
     &  'Normal Direction(X,Y)')
 170  FORMAT(/,6X,'TYPE',9X,
     &  'SEGMENT',2X,'NUM',12X,'ENDPOINTS OF THE SEGMENT',/,19X,
     &  'ANGLE',4X,'PTS',7X,'XA',8X,'YA',8X,'XB',8X,'YB')
 180  FORMAT(I4,5X,I4,5X,F10.5,F10.5)
 200  FORMAT (I5,F10.5,I5,A6)
 210  FORMAT (4F10.5,2I2,F8.3)
 350  FORMAT('0',30X,'ABSORBER SURFACE; INCIDENT FIELD DIRECTION = ',
     &          F7.2/'0',50X,'VOLUME OUTPUT')
 375  FORMAT(8H0  I SEG,4X,1HX,7X,1HY,7X,1HS,6X,3HDSQ,
     &14X,11H--  EPS  --,24X,10H--  MU  --,12X,'MOD(Jm)',4X,'ARG(Jm)',
     &2X)
 376  FORMAT(8H0  I SEG,4X,1HX,7X,1HY,7X,1HS,6X,3HDSQ,
     &14X,11H--  EPS  --,24X,10H--  MU  --,10X,'MOD(Je/Zo)',2X
     &,'ARG(Je/Zo)',4X)
 377  FORMAT('0',50X,'CONTOUR OUTPUT'/8H0  I SEG,4X,1HX,7X,1HY,7X,1HS,
     &6X,3HDSQ,8X,'EPS in',11X,'EPS out',10X,'MU in',12X,'MU out',6X,
     &'MOD(Je)',4X,'ARG(Je)',2X)
 378  FORMAT('0',50X,'CONTOUR OUTPUT'/8H0  I SEG,4X,1HX,7X,1HY,7X,1HS,
```

```
     &6X,3HDSQ,8X,'EPS in',11X,'EPS out',10X,'MU in',12X,'MU out',4X,
     &'MOD(Jm/Zo)',2X,'ARG(Jm/Zo)',4X)
 395  FORMAT(1H ,2I3,4F8.4,5X,2F11.3,12X,2F11.3,7X,F9.4,F11.3)
 396  FORMAT(1H ,2I3,4F8.4,4(1X,2F8.2),F9.4,F11.3)
 400  FORMAT(///32X,'KEY PARAMETERS'//
     &       17X,21HINCIDENT POLARIZATION,22X,1A1/
     &       17X,'TOTAL NUMBER OF INTERIOR VOLUME CELLS',I7/
     &       17X,37HTOTAL NUMBER OF CONTOUR SEGMENTS USED,I7/
     &       17X,35HNUMBER OF INCIDENT FIELD DIRECTIONS,I9/
     &       17X,29HNUMBER OF BISTATIC DIRECTIONS,I15/
     &       17X,10HWAVELENGTH,F34.5,/
     &       17X,'RECIPROCAL CONDITION NUMBER',E17.7)
 600  FORMAT(///,45X,28HBACKSCATTERING CROSS SECTION,/,
     &       49X,20H10*LOG(SIGMA/LAMBDA),/,
     &       51X,1H(,1A1,14H-POLARIZATION),///)
 800  FORMAT(///,43X,33HBISTATIC SCATTERING CROSS SECTION,/,
     &       49X,20H10*LOG(SIGMA/LAMBDA),/,
     &       41X,'FOR INCIDENT FIELD DIRECTION = ',F7.2,/,
     &       51X,1H(,1A1,14H-POLARIZATION),///)
 840  FORMAT(4X,'THETA',14X,'MOD(P)',9X,'ARG(P)',14X,'DB',/' ')
 900  FORMAT(4X,F7.2,5(1X,2F8.2))
 920  FORMAT(2X,F7.2,8X,E14.4,7X,F7.2,10X,F7.2)
 951  FORMAT(4A4,/,18A4,/,4F8.3,2F8.2,I5)
 950  FORMAT(2F8.3,F8.2)
 955  FORMAT(F8.3,'*',F8.3,'*',F8.2)
 960  FORMAT(i3,4f10.5)
 961  FORMAT(3x,4f10.5)
 962  FORMAT(40(f6.3,1x))
 963  FORMAT(2f10.5)
 964  FORMAT(20X,4f10.5)
1000  END
C*********************************************************************
      COMPLEX FUNCTION FONE(X,Y)
      COMPLEX YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,CI,SING
      REAL XK,NDNP,NDSP,SDNP,SDSP,XNI,YNI,XNJ,YNJ,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      DATA CI /(0.,1.)/
C
      NDNP=  XNI*XNJ+YNI*YNJ
      NDSP=-(YNI*XNJ-XNI*YNJ)
      SDSP=  XNI*XNJ+YNI*YNJ
      SDNP=  YNI*XNJ-XNI*YNJ
C
      FONE= -YIMP2*CI*XK*XK*.25*SING(0,X,Y,XK)
     &      +YIMP3*CI*SDSP *.25*SING(1,Y,X,XK)
     &      +YIMP3*CI*SDNP *.25*SING(1,X,Y,XK)
     &      +YIMP4*CI*NDSP *.25*SING(1,Y,X,XK)
     &      +YIMP4*CI*NDNP *.25*SING(1,X,Y,XK)
      RETURN
      END
C*********************************************************************
      COMPLEX FUNCTION FTWO(X,Y)
      COMPLEX YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,HZ,H1,CI,SING
      REAL XK,NDNP,NDSP,SDNP,SDSP,XNI,YNI,XNJ,YNJ,X,Y,RHO
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      DATA CI /(0.,1.)/
C
      NDNP=  XNI*XNJ+YNI*YNJ
      NDSP=-(YNI*XNJ-XNI*YNJ)
      SDNP=  YNI*XNJ-XNI*YNJ
      SDSP=  XNI*XNJ+YNI*YNJ
      RHO =  SQRT(X*X+Y*Y)
      ARG =  XK*RHO
      CALL HANKZ1(ARG,0,HZ,H1)
C
      FTWO=-YIMP2*XK       *.25*SING(1,X,Y,XK)
     &     +YIMP3*(SDSP/XK)*.25*HZ
     &     +YIMP3*(SDNP/XK)*.25*SING(2,X,Y,XK)
     &     +YIMP4*(NDSP/XK)*.25*HZ
     &     +YIMP4*(NDNP/XK)*.25*SING(2,X,Y,XK)
      RETURN
      END
C*********************************************************************
      COMPLEX FUNCTION FTHREE(X,Y,N)
      COMPLEX YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,CI,SING
      REAL XK,NDNP,NDSP,XNI,YNI,XNJ,YNJ,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      DATA CI /(0.,1.)/
C
      IF(N .EQ. 0) THEN
           FTHREE = (0.,0.)
           GO TO 100
      ENDIF
      NDNP=XNI*XNJ+YNI*YNJ
```

```
      NDSP=-(YNI*XNJ-XNI*YNJ)
C
      FTHREE= -XK*YIMP5*NDSP*.25*SING(1,Y,X,XK)
     &        -XK*YIMP5*NDNP*.25*SING(1,X,Y,XK)
100   RETURN
      END
C*****************************************************************
      COMPLEX FUNCTION FFOUR(X,Y,N,IN)
      COMPLEX YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,HZ,H1,CI,SING
      REAL XK,NDNP,NDSP,XNI,YNI,XNJ,YNJ,RHO,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      DATA CI /(0.,1.)/
C
      IF( N .EQ. 0) THEN
          FFOUR = (0.,0.)
          GO TO 100
      ENDIF
      NDSP=-(YNI*XNJ-XNI*YNJ)
      NDNP=XNI*XNJ+YNI*YNJ
      RHO=SQRT(X*X+Y*Y)
      ARG=XK*RHO
      CALL HANKZ1(ARG,0,HZ,H1)
      FFOUR=   CI*NDSP*.25*HZ
     &        +CI*NDNP*.25*SING(2,X,Y,XK)
       IF(IN .EQ. 0) THEN
           FFOUR = FFOUR*YIMP5
              ELSE
           FFOUR = FFOUR*YIMP7
       ENDIF
C     PRINT *, FFOUR,X,Y
100   RETURN
      END
C*****************************************************************
      COMPLEX FUNCTION SING(ISING,ALF,BTA,XK)
      COMPLEX CI,EXPR
      REAL ALF,BTA,A,B,XK,ARG,ATAN4
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      A=ALF*XK
      B=BTA*XK
C     PRINT *,'XK,A,B,CI:',XK,A,B,CI
      ARG=A*A+B*B
      EXPR=GAMA-1.5-ALOG(2.0)-.5*CI*PI+.5*ALOG(ARG)
      IF(ISING.EQ.0) THEN
        GO TO 10
          ELSE IF(ISING.EQ.1) THEN
            GO TO 20
              ELSE IF(ISING.EQ.2) THEN
                GO TO 30
                  ELSE IF(ISING.EQ.3) THEN
                    GO TO 40
                      ELSE
                        GO TO 70
      ENDIF
10    SING=A*B*( EXPR*(2.-ARG/6.)+ARG/18.)
     &     +  B*B*(1.-B*B/12.)*ATAN4(A,B)
     &     +  A*A*(1.-A*A/12.)*ATAN4(B,A)
      SING=SING*CI/(PI*XK*XK)
      GO TO 100
20    SING=B*(2.-B*B/3.+B**4/30.)*ATAN4(A,B)
     &+A*(4.-A*A/6.-A**4/1600.+A*A*B*B/240.+11*B**4/320.)/3.
     &+A*(EXPR-1./6.)
     &*(2.-A*A/6.-B*B/2.+A**4/160.+A*A*B*B/48.+B**4/32.)
      SING=SING*CI/(XK*PI)
      GO TO 100
30    SING=A*B/8.*(EXPR*(A*A/3.+B*B-8.)-(A*A/3.+5.*B*B)/12.)
     &        +(2.-B*B+B**4/12.)*ATAN4(A,B)
      SING=SING*CI/PI
C     PRINT *,'ALF,BTA,SING:',ALF,BTA,SING
      GO TO 100
40    SING=A/8.*(EXPR*(A*A/3.+3.*B*B-8.)
     &       -(A*A/3.+11.*B*B)/12.-16./ARG)
     &       -B*(2.-B*B/3.)*ATAN4(A,B)
      SING=SING*CI*XK/PI
      GO TO 100
70    PRINT *,'CANNOT HAVE ISING=',ISING
100   RETURN
      END
C*****************************************************************
      SUBROUTINE COORDS(XOBS,YOBS,X1,Y1,X2,Y2,X3,Y3,X4,Y4,JTYPE,I,J)
      REAL XOBS,YOBS,X1,Y1,X2,Y2,X3,Y3,X4,Y4
      REAL X11,Y11,X22,Y22,X33,Y33,X44,Y44
      REAL HATSX,HATSY,HATNX,HATNY
C
C     Indexing:
```

```
C
C     1 - (Sint-Del/2, Nint-Tau/2)
C     2 - (Sint-Del/2, Nint+Tau/2)
C     3 - (Sint+Del/2, Nint+Tau/2)
C     4 - (Sint+Del/2, Nint-Tau/2)
C
      IF(JTYPE.EQ.2) GO TO 200
C
C***** JTYPE=1 (VOLUMETRIC CELL)
C
      HATNX=(X2+X3-X1-X4)/SQRT((X2+X3-X1-X4)**2+(Y2+Y3-Y1-Y4)**2)
      HATNY=(Y2+Y3-Y1-Y4)/SQRT((X2+X3-X1-X4)**2+(Y2+Y3-Y1-Y4)**2)
      HATSX=(X4-X1)/SQRT((X4-X1)*(X4-X1)+(Y4-Y1)*(Y4-Y1))
      HATSY=(Y4-Y1)/SQRT((X4-X1)*(X4-X1)+(Y4-Y1)*(Y4-Y1))
C
      X11=(X1-XOBS)*HATSX+(Y1-YOBS)*HATSY
      Y11=(X1-XOBS)*HATNX+(Y1-YOBS)*HATNY
      X22=(X2-XOBS)*HATSX+(Y2-YOBS)*HATSY
      Y22=(X2-XOBS)*HATNX+(Y2-YOBS)*HATNY
      X33=(X3-XOBS)*HATSX+(Y3-YOBS)*HATSY
      Y33=(X3-XOBS)*HATNX+(Y3-YOBS)*HATNY
      X44=(X4-XOBS)*HATSX+(Y4-YOBS)*HATSY
      Y44=(X4-XOBS)*HATNX+(Y4-YOBS)*HATNY
C
      X1=(X11+X22)*.5D0
      Y1=Y11
      X2=(X11+X22)*.5D0
      Y2=Y22
      X3=(X33+X44)*.5D0
      Y3=Y33
      X4=(X33+X44)*.5D0
      Y4=Y44
      GO TO 410
C
C **** JTYPE=2 (CONTOUR ELEMENT)
C
200   HATSX=(X2-X1)/SQRT((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))
      HATSY=(Y2-Y1)/SQRT((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))
      HATNX=-HATSY
      HATNY=HATSX
      IF( I .EQ. J ) THEN
         XOBS = (X2 + X1)/2.0
         YOBS = (Y2 + Y1)/2.0
      ENDIF
C
      X11=(X1-XOBS)*HATSX+(Y1-YOBS)*HATSY
      Y11=(X1-XOBS)*HATNX+(Y1-YOBS)*HATNY
      X22=(X2-XOBS)*HATSX+(Y2-YOBS)*HATSY
      Y22=(X2-XOBS)*HATNX+(Y2-YOBS)*HATNY
      IF (I .EQ. J) THEN
          IF(ABS(Y11) .LT. 1.E-6) Y11 = 0.0
          IF(ABS(Y22) .LT. 1.E-6) Y22 = 0.0
      ENDIF
      X1=X11
      Y1=Y11
      X2=X22
      Y2=Y22
      X3 = 0.
      Y3 = 0.
      X4 = 0.
      Y4 = 0.
C     PRINT *,X1,Y1,X2,Y2
410   RETURN
      END
C*********************************************************************
      REAL FUNCTION DIST(XA,YA,XB,YB)
      REAL XA,YA,XB,YB
      DIST=SQRT((XB-XA)*(XB-XA)+(YB-YA)*(YB-YA))
      RETURN
      END
C*********************************************************************
      FUNCTION ATAN4(ARG1,ARG2)
      REAL ARG1,ARG2
C
C  Range: pi/2. to -pi/2.
C
      IF(ARG2 .EQ. 0.) THEN
        IF(ARG1.GE.0.) ATAN4= 3.141592654/2.
        IF(ARG1.LT.0.) ATAN4=-3.141592654/2.
          ELSE
            ATAN4=ATAN(ARG1/ARG2)
      ENDIF
      RETURN
      END
C*********************************************************************C
C
      SUBROUTINE CHMTX(MDIM2,NTOT,NVOL,A,IEL,IPRINT)
```

```
C                                                       VER JUNE 1988       C
C*************************************************************************C
C     This subroutine is used to print the matrix [A].                    C
C     Only 40 columns of the matrix are printed. The program is           C
C     intended for debugging purposes only and is controlled by           C
C     activating call statement in RUFCODE.                               C
C*************************************************************************C
C                                                                         C
      COMPLEX A(MDIM2,MDIM2)
      DIMENSION B(40)
      IF(IPRINT.NE.4.AND.IPRINT.NE.5) GO TO 50
      IF(IEL.EQ.0) THEN
        IST=1
        IEND=NTOT
        JST=1
        JEND=NTOT
      ELSE IF(IEL.EQ.1) THEN
        IST=1
        IEND=NVOL
        JST=1
        JEND=NVOL
      ELSE IF(IEL.EQ.2) THEN
        IST=1
        IEND=NVOL
        JST=NVOL+1
        JEND=NTOT
      ELSE IF(IEL.EQ.3) THEN
        IST=1+NVOL
        IEND=NTOT
        JST=1
        JEND=NVOL
      ELSE IF(IEL.EQ.4) THEN
        IST=NVOL+1
        IEND=NTOT
        JST=1+NVOL
        JEND=NTOT
      ENDIF
C
      JJEND=JST+39
      IF(JEND.LT.JJEND) JJEND=JEND
      IF(IEL.NE.0) WRITE(6,40) IEL
      WRITE (6,41)
      DO 35 I=IST,IEND
         DO 25 J=JST,JJEND
              B(J-JST+1)=CABS(A(I,J))
 25      CONTINUE
         IF(IPRINT.EQ.4) WRITE (6,42) (B(J-JST+1),J=JST,JJEND)
         IF(IPRINT.EQ.5) WRITE (6,43) (A(I,J),J=JST,JJEND)
 35   CONTINUE
 40   FORMAT('0',1HF,I1,2Hij,' MATRIX ELEMENTS')
 41   FORMAT('0',1X,'A(1,1) A(1,2) A(1,3) A(1,4) A(1,5) A(1,6) A(1,7)'
     &,' A(1,8)',6X,'A(1,10)',7X,'A(1,12)',7X,'A(1,14)',7X,'A(1,16)',
     & 7X,'A(1,18)')
 42   FORMAT(40E10.3)
 43   FORMAT(40( '(' , F7.4 , ',' , F7.4 , ') '))
 50   RETURN
      END
C*************************************************************C
C                                                             C
      SUBROUTINE ZIMP(MDIM,MTOT,XK,X,Y,XN,YN,WAVE,
     &                IPOL,DSQ,TAUS,NSEG,PHI,ANGINC)
C                                                             C
C*************************************************************C
C     CALLED BY RUFCODE TO COMPUTE THE INPUT                  C
C     IMPEDANCE ACROSS A SMALL GAP. SIMPSON'S                 C
C     THREE POINT INTEGRATION IS USED IN THE                  C
C     NUMERICAL EVALUATION OVER THE CELLS                     C
C                                                             C
C     OBSERVATION POINT INDEXED BY:     I                     C
C     SOURCE POINT INDEXED BY:          J                     C
C*************************************************************C
      REAL X(MDIM,5,5),Y(MDIM,5,5),XN(MDIM,5,5),YN(MDIM,5,5)
      REAL DSQ(MDIM,5),TAUS(MDIM,5),NDR,NPDR,NDNP,NDSP
      REAL XAI,YAI,XANI,YANI,XNJ,YNJ,X1,Y1,X2,Y2,X3,Y3,X4,Y4
      REAL DIST,XKK,AX(50),AY(50),ZMAG(50)
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      REAL*4 SIMP33(3,3),SIMP15(5)
      COMPLEX AA(5,5),BA(5,5),CA(5,5),DA(5,5)
      COMPLEX HSUM(50),ESUM(50),ZSUM(50),PHI(MDIM)
      COMPLEX HINC(50),EINC(50),AVG,ZT
      COMPLEX HZ,H1,CI,AONE,ATWO,ATHREE,AFOUR,PHIJ
      INTEGER NSEG(MDIM,3),ATOT
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XKK,XNI,YNI,XNJ,YNJ
      COMMON /OPTS/ XAI,YAI,XANI,YANI,PHIJ,POL
C
```

```
      DATA SIMP15/7,32,12,32,7/
      DATA SIMP33/1,4,1, 4,16,4, 1,4,1/
      DATA TOT15,TOT33/90,36/
C
      READ(5,200,END=600) IVOL,ATOT,AXA,AYA,AXB,AYB
      IF(IVOL .EQ. 0) GO TO 600
      AXA = AXA/WAVE
      AYA = AYA/WAVE
      AXB = AXB/WAVE
      AYB = AYB/WAVE
          TX = AXB - AXA
          TY = AYB - AYA
          D  = SQRT(TX*TX+TY*TY)
          DX = TX/(ATOT-1)
          DY = TY/(ATOT-1)
      WRITE(6,250) ANGINC
      WRITE(6,300)
      DO 205 INDX = 1,ATOT
         AX(INDX) = AXA + (INDX-1)*DX
         AY(INDX) = AYA + (INDX-1)*DY
205   CONTINUE
      XKK=XK
      DO 600 I=1,ATOT
         XAI = AX(I)
         YAI = AY(I)
         XANI = -TY/D
         YANI = TX/D
C
         HINC(I) = (0.,0.)
         EINC(I) = (0.,0.)
         HSUM(I) = (0.,0.)
         ESUM(I) = (0.,0.)
         ZSUM(I) = (0.,0.)
         ZMAG(I) =  0.
         AVG     =  0.
         ZT      =  0.
C
C********** COMPUTE INCIDENT FIELD
C
         TETA = RED*ANGINC
         CT1 = COS(TETA)
         ST1 = SIN(TETA)
         DOT1 = XANI*CT1+YANI*ST1
C
         IF(IPOL .EQ. 2) THEN
              POL = Z0
                ELSE
                  POL = 1./Z0
         ENDIF
         IF(IPOL .EQ. 2) THEN
           HINC(I) = (1./POL)*CEXP(-CI*XK*(XAI*CT1+YAI*ST1))
           EINC(I) = DOT1*CEXP(-CI*XK*(XAI*CT1+YAI*ST1))
         ENDIF
         IF(IPOL .EQ. 1) THEN
           EINC(I) =  (1./POL)*CEXP(-CI*XK*(XAI*CT1+YAI*ST1))
           HINC(I) = -DOT1*CEXP(-CI*XK*(XAI*CT1+YAI*ST1))
         ENDIF
C
         DO 500 J=1,MTOT
C
         PHIJ = PHI(J)
C
C********** CHECK NECESSITY OF ANALYTICAL EVALUATION
C
            IF(NSEG(J,3) .EQ. 1) THEN
             DMIN = DIST(XAI,YAI,X(J,3,3),Y(J,3,3))
             IF(NSEG(J,2) .EQ. 1) GO TO 28
             IF(DMIN .LT. 0.2 ) THEN
                  DO 26 KK=1,5,4
                   DO 27 LL=1,5
                    DD1 = DIST(XAI,YAI,X(J,KK,LL),Y(J,KK,LL))
                    DD2 = DIST(XAI,YAI,X(J,LL,KK),Y(J,LL,KK))
                    IF(DD1 .LT. DMIN) DMIN=DD1
                    IF(DD2 .LT. DMIN) DMIN=DD2
27                 CONTINUE
26                CONTINUE
               ENDIF
C
             ELSE IF(NSEG(J,3) .EQ. 2) THEN
               DMIN = DIST(XAI,YAI,X(J,3,1),Y(J,3,1))

             ENDIF
C
28           DKA = XK*DMIN
C
C*********  PERFORM ANALYTICAL EVALUATION IF DK IS SMALL
C
```

```
      IF(DKA .LT. 0.45) GO TO 650
C
C************************************************C
C                                                C
C       NUMERICAL EVALUATION OF NEAR FIELDS      C
C                                                C
C************************************************C
C
      IF(NSEG(J,3) .EQ. 2) GO TO 625
      IF(NSEG(J,2) .EQ. 1) THEN
C
C*********   RICHMOND CELL CURRENT EVALUATION
C
        RXA = XAI - X(J,3,3)
        RYA = YAI - Y(J,3,3)
        R2  = SQRT(RXA*RXA+RYA*RYA)
        NDR = (RXA*XANI+RYA*YANI)/R2
        RK = R2*XK
        CALL HANKZ1(RK,2,HZ,H1)
        HSUM(I) = HSUM(I)-PHIJ*DSQ(J,3)*TAUS(J,3)*XK*HZ/(POL*4.)
        ESUM(I) = ESUM(I)+PHIJ*DSQ(J,3)*TAUS(J,3)*XK*NDR*CI*H1/4.
      ENDIF
C
      IF(NSEG(J,2) .EQ. 2) THEN
C
C*********   VOLUME CELL CURRENT EVALUATION
C
        DO 610 L= 1,5,2
           DO 605 K= 1,5,2
             XJ = X(J,L,K)
             YJ = Y(J,L,K)
             XNJ = XN(J,L,K)
             YNJ = YN(J,L,K)
             RXA = XAI - XJ
             RYA = YAI - YJ
             R2  = SQRT(RXA*RXA+RYA*RYA)
             RK  = R2*XK
             NDR = (RXA*XANI+RYA*YANI)/R2
             CALL HANKZ1(RK,2,HZ,H1)
             AA(L,K) = -PHIJ*XK/(POL*4.)*HZ
             AA(L,K) = AA(L,K)*DSQ(J,L)*TAUS(J,K)
             BA(L,K) = -PHIJ*CI*NDR*XK/4.*H1
             BA(L,K) = BA(L,K)*DSQ(J,L)*TAUS(J,K)
605     CONTINUE
610   CONTINUE
C
C************ 2-D INTEGRATION, 3X3
C
       DO 630 K=1,5,2
         KK=(K+1)/2
         DO 640 L =1,5,2
           LL=(L+1)/2
           HSUM(I) = HSUM(I)+SIMP33(LL,KK)*AA(L,K)/TOT33
           ESUM(I) = ESUM(I)+SIMP33(LL,KK)*BA(L,K)/TOT33
640      CONTINUE
630    CONTINUE
      ENDIF
C
C************ CONTOUR CELL CURRENT EVALUATION
C
625   IF(NSEG(J,3) .EQ. 2) THEN
        KK = 1
        K = 1
         DO 615 L=1,5
            XJ = X(J,L,K)
            YJ = Y(J,L,K)
            XNJ = XN(J,L,K)
            YNJ = YN(J,L,K)
            RXA = XAI - XJ
            RYA = YAI - YJ
            R2  = SQRT(RXA*RXA+RYA*RYA)
            RK  = R2*XK
            NDR = (RXA*XANI+RYA*YANI)/R2
            CALL HANKZ1(RK,2,HZ,H1)
            CA(L,KK) = -PHIJ*CI/(POL*4.)*HZ
            CA(L,KK) = CA(L,KK)*DSQ(J,L)
            DA(L,KK) = -PHIJ*NDR*H1/4.
            DA(L,KK) = DA(L,KK)*DSQ(J,L)
615       CONTINUE
C
C************ 1-D INTEGRATION
C
          DO 620 L=1,5
             HSUM(I) = HSUM(I)+CA(L,KK)*SIMP15(L)/TOT15
             ESUM(I) = ESUM(I)+DA(L,KK)*SIMP15(L)/TOT15
620       CONTINUE
         ENDIF
```

```
       GO TO 500
C
C*****************************************C
C                                         C
C     SINGULAR EVALUATION OF NEAR FIELD   C
C                                         C
C*****************************************C
C
650        ITYPE = 2
           JTYPE = NSEG(J,3)
C
      X1 = X(J,1,1)
      Y1 = Y(J,1,1)
      X2 = X(J,5,1)
      Y2 = Y(J,5,1)
      IF(JTYPE .EQ. 2) THEN
            X3 = 11.
            Y3 = 11.
            X4 = 11.
            Y4 = 11.
            XNJ = XN(J,3,1)
            YNJ = YN(J,3,1)
               ELSE
                 X3 = X(J,5,5)
                 Y3 = Y(J,5,5)
                 X4 = X(J,1,5)
                 Y4 = Y(J,1,5)
                 XNJ = XN(J,3,3)
                 YNJ = YN(J,3,3)
      ENDIF
      II=1
      JJ=2
      XAO = XAI
      YAO = YAI
      CALL COORDS(XAO,YAO,X1,Y1,X2,Y2,X3,Y3,X4,Y4,JTYPE,II,JJ)
         SMALL = 1.E-5
         N1 = 1
         N2 = 1
         N3 = 1
         N4 = 1
         IF(SQRT(X1*X1+Y1*Y1) .LT. SMALL) THEN
            N1 = 0
         ELSE IF(SQRT(X2*X2+Y2*Y2) .LT. SMALL) THEN
            N2 = 0
         ELSE IF(SQRT(X3*X3+Y3*Y3) .LT. SMALL) THEN
            N3 = 0
         ELSE IF(SQRT(X4*X4+Y4*Y4) .LT. SMALL) THEN
            N4 = 0
         ENDIF
         IF(JTYPE .EQ. 1) THEN
         HSUM(I) = HSUM(I)+AONE(X3,Y3,N3)-AONE(X4,Y4,N4)
     &                    -AONE(X2,Y2,N2)+AONE(X1,Y1,N1)
C
         ESUM(I) = ESUM(I)+ATWO(X3,Y3,N3)-ATWO(X4,Y4,N4)
     &                    -ATWO(X2,Y2,N2)+ATWO(X1,Y1,N1)
         ELSE IF(JTYPE .EQ. 2) THEN
         HSUM(I) = HSUM(I)+ ATHREE(X2,Y2,N2)-ATHREE(X1,Y1,N1)
         ESUM(I) = ESUM(I)+ AFOUR(X2,Y2,N2)- AFOUR(X1,Y1,N1)
         ENDIF
500      CONTINUE
          IF(IPOL .EQ. 2) THEN
          ZSUM(I) = (ESUM(I) - EINC(I))/(HINC(I) - HSUM(I))
          ENDIF
          IF(IPOL .EQ. 1) THEN
          ZSUM(I) = (EINC(I) + HSUM(I))/(HINC(I) + ESUM(I))
          ENDIF
          ZMAG(I) = CABS(ZSUM(I))
          WRITE(6,350) AX(I),AY(I),ZSUM(I),ZMAG(I)
600   CONTINUE
C
C********** COMPUTE AVERAGE IMPEDANCE VALUE
      DO 510 I=1,ATOT
         ZT = ZT+ZSUM(I)
510   CONTINUE
      AVG = ZT/ATOT
      AVGM = CABS(AVG)
      WRITE(6,400) AVG,AVGM
C
C********** FORMATS
200   FORMAT(I2,I5,4F10.5)
250   FORMAT(//10X,'GAP IMPEDANCE FOR ',F5.2,' DEGREES INCIDENCE')
300   FORMAT(//8X,'LOCATION',25X,'  IMPEDANCE',//7X,'X',9X,
     &        'Y',17X,'REAL',5X,'IMAGIN',5X, 'MAGN')
350   FORMAT(2F10.5,10X,3F10.3)
400   FORMAT(/'   AVERAGE IMPEDANCE =',8X,3F10.3)
      RETURN
      END
```

```
C*************************************************************
      COMPLEX FUNCTION AONE(X,Y,N)
      COMPLEX CI,SING,PHIJ
      REAL XK,NDNP,NDSP,XANI,YANI,XNJ,YNJ,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      COMMON /OPTS/ XAI,YAI,XANI,YANI,PHIJ,POL
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      IF(N .EQ. 0) THEN
         AONE = (0.,0.)
         GO TO 100
      ENDIF
C
      NDNP = XANI*XNJ+YANI*YNJ
      NDSP = -(YANI*XNJ-XANI*YNJ)
C
      AONE = -PHIJ*XK/(POL*4.)*SING(0,X,Y,XK)
100   RETURN
      END
C
C*************************************************************
      COMPLEX FUNCTION ATWO(X,Y,N)
      COMPLEX CI,SING,PHIJ
      REAL XK,NDNP,NDSP,XANI,YANI,XNJ,YNJ,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      COMMON /OPTS/ XAI,YAI,XANI,YANI,PHIJ,POL
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      IF(N .EQ. 0) THEN
         ATWO = (0.,0.)
         GO TO 100
      ENDIF
C
      NDNP = XANI*XNJ+YANI*YNJ
      NDSP = -(YANI*XNJ-XANI*YNJ)
C
      ATWO= -PHIJ*NDSP*CI/4.*SING(1,Y,X,XK)
     &      -PHIJ*NDNP*CI/4.*SING(1,X,Y,XK)
100   RETURN
      END
C
C*************************************************************
      COMPLEX FUNCTION ATHREE(X,Y,N)
      COMPLEX CI,SING,PHIJ
      REAL XK,NDNP,NDSP,XANI,YANI,XNJ,YNJ,X,Y
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      COMMON /OPTS/ XAI,YAI,XANI,YANI,PHIJ,POL
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      IF(N .EQ. 0) THEN
         ATHREE = (0.,0.)
         GO TO 100
      ENDIF
C
      NDNP = XANI*XNJ+YANI*YNJ
      NDSP = -(YANI*XNJ-XANI*YNJ)
C
      ATHREE = -PHIJ*CI/(POL*4.)*SING(1,X,Y,XK)
100   RETURN
      END
C
C*************************************************************
      COMPLEX FUNCTION AFOUR(X,Y,N)
      COMPLEX CI,SING,HZ,PHIJ
      REAL XK,NDNP,NDSP,XANI,YANI,XNJ,YNJ,X,Y,RHO
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XK,XNI,YNI,XNJ,YNJ
      COMMON /OPTS/ XAI,YAI,XANI,YANI,PHIJ,POL
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      IF(N .EQ. 0) THEN
         AFOUR = (0.,0.)
         GO TO 100
      ENDIF
C
      NDNP = XANI*XNJ+YANI*YNJ
      NDSP = -(YANI*XNJ-XANI*YNJ)
      RHO = SQRT(X*X+Y*Y)
      ARG = XK*RHO
      CALL HANKZ1(ARG,0,HZ,H1)
C
      AFOUR = -PHIJ*NDSP*HZ/(XK*4.)
     &        -PHIJ*NDNP*SING(2,X,Y,XK)/(XK*4.)
100   RETURN
```

```
      END
C*********************************************************************C
C*********************************************************************C
C                                                                     C
      SUBROUTINE MTXRED(MDIM,MDIM2,MTOT,NVOL,IPOL,XK,X,Y,XN,YN,DSQ,
     &                  NSEG,YE,YM,F,TAUS,IPRINT,nrmcell)
C                                                                     C
C                                                VERSION Mar 1990     C
C*********************************************************************C
C     CALLED BY RUFCODE TO GENERATE MATRIX ELEMENTS TO SOLVE          C
C     INTEGRAL EQUATIONS BY METHOD OF MOMENTS.  THE MATRIX            C
C     ELEMENTS ARE COMPUTED USING SIMPSON'S THREE POINT AND FIVE      C
C     POINT INTEGRATION OVER THE CELLS.                               C
C                                                                     C
C     OBSERVATION POINT INDEXED BY:     I                             C
C     SOURCE POINT INDEXED BY:          J                             C
C*********************************************************************C
      REAL X(MDIM,5,5),Y(MDIM,5,5),XN(MDIM,5,5),YN(MDIM,5,5)
      REAL DSQ(MDIM,5),NDR,NPDR,NDNP,NDSP,SDSP,SDNP
      REAL TAUS(MDIM,5),TAUJ,DELJ
      REAL XI,YI,XNI,YNI,XNJ,YNJ,X1,Y1,X2,Y2,X3,Y3,X4,Y4
      REAL DIST,XKK
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      REAL*4 SIMP55(5,5),SIMP15(5),SIMP33(3,3),SIMP13(3)
      COMPLEX A(5,5),B(5,5),C(5,5),D(5,5)
      COMPLEX HZ,H1,CI,F(MDIM2,MDIM2)
      COMPLEX YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YE(MDIM,7),YM(MDIM,7)
      COMPLEX FONE,FTWO,FTHREE,FFOUR,f1,f3,ftemp
      INTEGER NSEG(MDIM,3)
      real bx(8),by(8),cc(4,4),bxi(8),byi(8)
      real Jac,Rho,bxj(8),byj(8)
      real ag(96),wg(96),ags(96),wgs(96)
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
      COMMON /ELMNTS/ YIMP1,YIMP2,YIMP3,YIMP4,YIMP5,YIMP7,
     &                XKK,XNI,YNI,XNJ,YNJ
C
      DATA SIMP13/1,4,1/
      DATA SIMP15/7,32,12,32,7/
      DATA SIMP33/1,4,1, 4,16,4, 1,4,1/
      DATA SIMP55/49,224,84,224,49, 224,1024,384,1024,224,
     &  84,384,144,384,84, 224,1024,384,1024,224, 49,224,84,224,49/
      DATA TOT13,TOT15,TOT33,TOT55/6.,90.,36.,8100./
C
c
        Iorder = 6
        Iorders= 16
c
        icount = 0
        do 2 i = Iorder,2,-2
           icount = icount + 1
           ag(icount)= XGAUS(i,Iorder)
           wg(icount)= WGAUS(i,Iorder)
   2    continue
        do 3 i = 1,Iorder,2
           icount = icount + 1
           ag(icount)= XGAUS(i,Iorder)
           wg(icount)= WGAUS(i,Iorder)
   3    continue
           icount = 0
        do 4 i = Iorders,2,-2
           icount = icount + 1
           ags(icount)= XGAUS(i,Iorders)
           wgs(icount)= WGAUS(i,Iorders)
   4    continue
        do 6 i = 1,Iorders,2
           icount = icount + 1
           ags(icount)= XGAUS(i,Iorders)
           wgs(icount)= WGAUS(i,Iorders)
   6    continue
      XKK=XK
      DO 300 I=1,MTOT
C
C******************** TAKE INTO ACCOUNT POLARIZATION *************
C                                                                   *
C    IMPEDANCE          E-POLARIZATION          H-POLARIZATION     *
C      YIMP1                1/Ur                    1/Er          *
C      YIMP2              Er - 1/Ur               Ur - 1/Er       *
C      YIMP3              d/ds(1/Ur)              d/ds(1/Er)      *
C      YIMP4              d/dn(1/Ur)              d/dn(1/Er)      *
C      YIMP5              (Ur-1)/Ur               (Er-1)/Er       *
C                                                                   *
C*********************************************************************
C
        IF(IPOL .EQ. 2) GO TO 5
        YIMP1=YM(I,1)
        YIMP2=YE(I,2)
        YIMP3=YM(I,3)
```

```
         YIMP4=YM(I,4)
         YIMP5=YM(I,5)
         YIMP7=YM(I,7)
         GO TO 7
5        YIMP1=YE(I,1)
         YIMP2=YM(I,2)
         YIMP3=YE(I,3)
         YIMP4=YE(I,4)
         YIMP5=YE(I,5)
         YIMP7=YE(I,7)
C
7        CLOSE=2.5*DSQ(I,3)*TWOPI
         IF(NSEG(I,3) .EQ. 1) THEN
           XI=X(I,3,3)
           YI=Y(I,3,3)
           XNI=XN(I,3,3)
           YNI=YN(I,3,3)
         ELSE
           XI=X(I,3,1)
           YI=Y(I,3,1)
           XNI=XN(I,3,1)
           YNI=YN(I,3,1)
         ENDIF
25       DO 280 J=1,MTOT
            TAUJ=TAUS(J,3)
            DELJ=DSQ(J,3)
C
C******  CHECK NECESSITY OF ANALYTICAL EVALUATION
C
         IF(NSEG(J,3) .EQ. 1 ) THEN
           IF(NSEG(J,2) .EQ. 1) GO TO 95
           DMIN=DIST(XI,YI,X(J,3,3),Y(J,3,3))
           IF(DMIN .LT. 0.2  .AND. NSEG(I,3) .EQ. 2) THEN
             DO 26 KK=1,5,4
                DO 27 LL =1,5
                      DD1 = DIST(XI,YI,X(J,KK,LL),Y(J,KK,LL))
                      DD2 = DIST(XI,YI,X(J,LL,KK),Y(J,LL,KK))
                      IF(DD1 .LT. DMIN) DMIN=DD1
                      IF(DD2 .LT. DMIN) DMIN=DD2
27              CONTINUE
26           CONTINUE
           ENDIF
         ELSE IF(NSEG(J,3) .EQ. 2) THEN
             DMIN=DIST(XI,YI,X(J,3,1),Y(J,3,1))
         ENDIF
C
28       DK=XK*DMIN
C**** Perform analytical evaluation if DK is
C**** small enough, or if I = J
C
            IF(NSEG(J,2) .NE. 1) THEN
            IF(DK .LT. .45 .OR. I.EQ.J) GO TO 270
            ENDIF
            IF(NSEG(J,2) .EQ. 1 .AND. I .EQ. J) GO TO 270
C*************************************************
C                                                *
C      NUMERICAL EVALUATION OF ELEMENTS          *
C                                                *
C*************************************************
C
C*** Check for integration over body contour or interior volume
C
30          IF(NSEG(J,3).EQ.2) GO TO 200
C
C******  INTEGRATION OVER VOLUME CELL
C
35          RX=XI-X(J,3,3)
            RY=YI-Y(J,3,3)
            R2=SQRT(RX*RX+RY*RY)
            RK=R2*XK
C
C******* Parametric CELL EVALUATION *******
C
 95          IF(NSEG(J,2) .EQ. 1) THEN
              f(i,j) = (0.0,0.0)

              bx(1) = X(j,1,5)
              bx(2) = X(j,5,5)
              bx(3) = X(j,5,1)
              bx(4) = X(j,1,1)
              bx(5) = X(j,3,5)
              bx(6) = X(j,5,3)
              bx(7) = X(j,3,1)
              bx(8) = X(j,1,3)

              by(1) = Y(j,1,5)
              by(2) = Y(j,5,5)
```

```
              by(3) = Y(j,5,1)
              by(4) = Y(j,1,1)
              by(5) = Y(j,3,5)
              by(6) = Y(j,5,3)
              by(7) = Y(j,3,1)
              by(8) = Y(j,1,3)

c
c Now complete the process
c

              do 101 lk = 1,8
                bxj(lk) = bx(lk)
                byj(lk) = by(lk)
                bxi(lk) = xi - bx(lk)
                byi(lk) = yi - by(lk)
 101          continue

c
c Map the Difference Vectors into u,v parameters
c
              call TwoMap(bxi,bx)
              call TwoMap(byi,by)
c
c Compute the Jacobian Parameters for this Bad Boy
c
              Call Jacobian(bxj,byj,Cc)
c
c Compute the Area of the cell
c
              if(nseg(j,1).gt.nrmcell.and.i.eq.1) then
               area = 0.0
               do 322 lk = 1,Iorder
                do 310 kl = 1,Iorder
                  Call TwoStuff(Cc,bx,by,ag(lk),ag(kl),
     &                          Xin,Yin,Rho,Jac)
                  area = area + (wg(lk)*wg(kl)*Jac)
 310             continue
 322            continue
                taus(j,3) = sqrt(area)
                dsq(j,3) = taus(j,3)
              endif
c
c Perform Iorder Gaussian Quadrature Integration
c
              itest = i + j
              il = 2 * j - 1
              ih = 2 * j + 1
              if(itest.lt.il.OR.itest.gt.ih) then
               do 122 lk = 1,Iorder
                do 110 kl = 1,Iorder
                  Call TwoStuff(Cc,bx,by,ag(lk),ag(kl),
     &                          Xin,Yin,Rho,Jac)
                  If(nseg(i,3).eq.1) then
c Observation pt. is on Volume Cell (F1)
                   Ftemp = F1(yimp2,xk,yimp3,yimp4,Xin,Yin,Rho,Jac)
                  else
c Observation pt. is on Contour Cell (F3)
                    Ftemp = F3(xk,xni,yni,yimp5,Xin,Yin,Rho,Jac)
                  endif
                  F(i,j) = F(i,j)+cmplx(wg(lk)*wg(kl),0.0)*Ftemp
 110            continue
 122           continue
c Self-Cell
              else
               do 121 lk = 1,Iorders
                do 111 kl = 1,Iorders
                  Call TwoStuff(Cc,bx,by,ags(lk),ags(kl),
     &                          Xin,Yin,Rho,Jac)
                  If(nseg(i,3).eq.1) then
c Observation pt. is on Volume Cell (F1)
                   Ftemp = F1(yimp2,xk,yimp3,yimp4,Xin,Yin,Rho,Jac)
                  else
c Observation pt. is on Contour Cell (F3)
                    Ftemp = F3(xk,xni,yni,yimp5,Xin,Yin,Rho,Jac)
                  endif
                  F(i,j) = F(i,j)+cmplx(wgs(lk)*wgs(kl),0.0)*Ftemp
 111            continue
 121           continue
              endif

c
c Correction term when self-cell
c
              if(i.eq.j) F(i,j) = F(i,j) + yimp1

              GO TO 280
```

```
            ENDIF
C
C******* RECTANGULAR CELL EVALUATION *******
C
            IF(RK.GT.CLOSE) THEN
              IACUR=2
                ELSE
                  IACUR=1
            ENDIF
            DO 105 L=1,5,IACUR
               DO 100 K=1,5,IACUR
                  XJ=X(J,L,K)
                  YJ=Y(J,L,K)
                  XNJ=XN(J,L,K)
                  YNJ=YN(J,L,K)
                  RX=XI-XJ
                  RY=YI-YJ
                  R2=SQRT(RX*RX+RY*RY)
                  RK=R2*XK
C
C******  COMPUTE DOT PRODUCTS
C
                  SDR=(RX*YNI-RY*XNI)/R2
                  NDR=(RX*XNI+RY*YNI)/R2
                  SPDR=(YNJ*RX-XNJ*RY)/R2
                  NPDR=(XNJ*RX+YNJ*RY)/R2
                  SDSP=XNI*XNJ+YNI*YNJ
                  SDNP=YNI*XNJ-XNI*YNJ
                  NDSP=-(YNI*XNJ-XNI*YNJ)
                  NDNP=XNI*XNJ+YNI*YNJ
                  CALL HANKZ1(RK,2,HZ,H1)
                  IF(NSEG(I,3).EQ.1) GO TO 70
C
C*** OBSER.PNT. IS ON BODY CONTOUR (F3)
C
                  C(L,K)=-XK*XK*YIMP5/4.*NDR*H1
                  C(L,K)=C(L,K)*DSQ(J,L)*TAUS(J,K)
                  GO TO 100
C
C*** OBSER.PNT. IS VOLUME CELL (F1)
C
70                A(L,K)= -CI*XK*XK/4.*YIMP2*HZ
     &                    +CI*XK/4.*YIMP3*SDR*H1
     &                    +CI*XK/4.*YIMP4*NDR*H1
C
                  A(L,K)=A(L,K)*DSQ(J,L)*TAUS(J,K)
C
100            CONTINUE
105         CONTINUE
            F(I,J)=(0.,0.)
            IF(IACUR.EQ.1) THEN
              IF(NSEG(I,3).EQ.1) THEN
                GO TO 112
              ELSE
                GO TO 125
              ENDIF
            ELSE IF(IACUR.EQ.2) THEN
              IF(NSEG(I,3).EQ.1) THEN
                GO TO 140
              ELSE
                GO TO 165
              ENDIF
            ENDIF
C
C******  2-D INTEGRATION,5X5
C
112         DO 120 K=1,5
               DO 115 L=1,5
                  F(I,J)= F(I,J) +SIMP55(L,K)*A(L,K)/TOT55
115            CONTINUE
120         CONTINUE
            GO TO 280
C
125         DO 135 K=1,5
               DO 130 L=1,5
                  F(I,J)= F(I,J) +SIMP55(L,K)*C(L,K)/TOT55
130            CONTINUE
135         CONTINUE
            GO TO 280
C
C******  2-D INTEGRATION,3X3
C
140         DO 150 K=1,5,2
               KK=(K+1)/2
               DO 145 L=1,5,2
                  LL=(L+1)/2
```

```
                  F(I,J)= F(I,J) +SIMP33(LL,KK)*A(L,K)/TOT33
145               CONTINUE
150            CONTINUE
               GO TO 280
C
165            DO 175 K=1,5,2
                  KK=(K+1)/2
                  DO 170 L=1,5,2
                     LL=(L+1)/2
                     F(I,J)= F(I,J) +SIMP33(LL,KK)*C(L,K)/TOT33
170               CONTINUE
175            CONTINUE
               GO TO 280
C
C******  POINT OF INTEGRATION IS BODY CONTOUR
C
200            IF(NSEG(J,3).EQ.2) THEN
                 K=1
                 KK=1
               ENDIF
               F(I,J)=(0.,0.)
               DO 215 L=1,5
                  XJ=X(J,L,K)
                  YJ=Y(J,L,K)
                  XNJ=XN(J,L,K)
                  YNJ=YN(J,L,K)
                  RX=XI-XJ
                  RY=YI-YJ
                  R2=SQRT(RX*RX+RY*RY)
                  RK=R2*XK
C
C******  COMPUTE DOT PRODUCTS
C
                  SDR=(RX*YNI-RY*XNI)/R2
                  NDR=(RX*XNI+RY*YNI)/R2
                  SPDR=(YNJ*RX-XNJ*RY)/R2
                  NPDR=(XNJ*RX+YNJ*RY)/R2
                  SDSP=XNI*XNJ+YNI*YNJ
                  SDNP=YNI*XNJ-XNI*YNJ
                  NDSP=-(YNI*XNJ-XNI*YNJ)
                  NDNP=XNI*XNJ+YNI*YNJ
                  CALL HANKZ1(RK,2,HZ,H1)
                  IF(NSEG(I,3).NE.1) GO TO 205
C
C*** OBSER.POINT IS VOLUME CELL (F2)
C
                  B(L,KK)=-XK*YIMP2/4.*HZ
     &                      +YIMP3/4.*SDR*H1
     &                      +YIMP4/4.*NDR*H1
                  B(L,KK)=B(L,KK)*DSQ(J,L)
                  GO TO 215
C
C*** OBSER.POINT IS ON BODY CONTOUR (F4)
C
205               D(L,KK)= .25*CI*XK*YIMP5*NDR*H1
                  D(L,KK)=D(L,KK)*DSQ(J,L)
C                 PRINT *, 'YIMP,NDR,H1,DSQ:',YIMP5,NDR,H1,DSQ(J,L)
215            CONTINUE
               IF(NSEG(I,3).NE.1) GO TO 230
C
C*****  COMPUTE 1D-INTEGRATIONS
C
               DO 220 L=1,5
                  F(I,J)= F(I,J)+B(L,KK)*SIMP15(L)/90.
220            CONTINUE
               GO TO 280
C
230            DO 235 L=1,5
                  F(I,J)=F(I,J)+D(L,KK)*SIMP15(L)/90.
C                 PRINT *, F(I,J),I,J
235            CONTINUE
               GO TO 280
C***************************************************
C                                                  *
C      SINGULAR EVALUATION OF ELEMENTS             *
C                                                  *
C***************************************************
270            ITYPE=NSEG(I,3)
               JTYPE=NSEG(J,3)
               IF(I.EQ.J) THEN
                  KRON=1
                    ELSE
                      KRON=0
               ENDIF
C
C********** RECTANGULAR CELL SINGULAR EVALUATION
C
```

```
            X1=X(J,1,1)
            Y1=Y(J,1,1)
            X2=X(J,5,1)
            Y2=Y(J,5,1)
            IF(JTYPE .EQ. 2) THEN
              X3=11.
              Y3=11.
              X4=11.
              Y4=11.
              XNJ=XN(J,3,1)
              YNJ=YN(J,3,1)
            ELSE
              X3=X(J,5,5)
              Y3=Y(J,5,5)
              X4=X(J,1,5)
              Y4=Y(J,1,5)
              XNJ=XN(J,3,3)
              YNJ=YN(J,3,3)
            ENDIF
            XO = XI
            YO = YI
            CALL COORDS(XO,YO,X1,Y1,X2,Y2,X3,Y3,X4,Y4,JTYPE,I,J)
              SMALL = 1.E-6
              N1 = 1
              N2 = 1
              N3 = 1
              N4 = 1
              IF(SQRT(X1*X1+Y1*Y1) .LT. SMALL) THEN
                    N1 = 0
              ELSE IF(SQRT(X2*X2+Y2*Y2) .LT. SMALL) THEN
                    N2 = 0
              ELSE IF(SQRT(X3*X3+Y3*Y3) .LT. SMALL) THEN
                    N3 = 0
              ELSE IF(SQRT(X4*X4+Y4*Y4) .LT. SMALL) THEN
                    N4 = 0
              ENDIF
              IF(ITYPE.EQ.1.AND.JTYPE.EQ.1) THEN
              F(I,J)= FONE(X3,Y3)- FONE(X4,Y4)
     &                -FONE(X2,Y2)+ FONE(X1,Y1)+FLOAT(KRON)*YIMP1
            ELSE IF(ITYPE.EQ.1.AND.JTYPE.EQ.2) THEN
              F(I,J)= FTWO(X2,Y2)-FTWO(X1,Y1)
            ELSE IF(ITYPE.EQ.2.AND.JTYPE.EQ.1) THEN
              F(I,J)= FTHREE(X3,Y3,N3)-FTHREE(X4,Y4,N4)
     &                -FTHREE(X2,Y2,N2)+FTHREE(X1,Y1,N1)
            ELSE IF(ITYPE.EQ.2.AND.JTYPE.EQ.2) THEN
                    IF(I .EQ. J) THEN
                      IS = 1
                        ELSE
                          IS = 0
                    ENDIF
              F(I,J)= FFOUR(X2,Y2,N2,IS)-FFOUR(X1,Y1,N1,IS)
     &                +FLOAT(KRON)
C           PRINT *, I,J,X2,Y2,X1,Y1,N2,N1,IS
            ENDIF
280       CONTINUE
300    CONTINUE
       CALL CHMTX(MDIM2,MTOT,NVOL,F,0,IPRINT)
C      CALL CHMTX(MDIM2,MTOT,NVOL,F,1,IPRINT)
C      CALL CHMTX(MDIM2,MTOT,NVOL,F,2,IPRINT)
C      CALL CHMTX(MDIM2,MTOT,NVOL,F,3,IPRINT)
C      CALL CHMTX(MDIM2,MTOT,NVOL,F,4,IPRINT)
320    RETURN
       END
C*********************************************************************
       REAL FUNCTION XGAUS(I,NN)
       DIMENSION X6(6),X8(8),X12(12),X16(16),X24(24),X48(48),X96(96)
C
       DATA X6/  .2386192,-.2386192,.6612094,
     &          -.6612094,.9324695,-.9324695/
       DATA X8/  .1834346,-.1834346,.5255324,-.5255324,
     &           .7966664,-.7966664,.9602898,-.9602898/
       DATA X12/ .1252334,-.1252334,.3678315,-.3678315,
     &           .5873179,-.5873179,.7699027,-.7699027,
     &           .9041172,-.9041172,.9815606,-.9815606/
       DATA X16/
     &.0950125,-.0950125,.2816036,-.2816036,.4580168,-.4580168,
     &.6178762,-.6178762,.7554044,-.7554044,.8656312,-.8656312,
     &.944575 ,-.944575 ,.989409 ,-.989409/
       DATA X24/
     &.0640570,-.0640570,.1911187,-.1911187,.3150427,-.3150427,
     &.4337935,-.4337935,.5454215,-.5454215,.6480936,-.6480936,
     &.7401242,-.7401242,.8200020,-.8200020,.8864155,-.8864155,
     &.9382745,-.9382745,.9747285,-.9747285,.9951872,-.9951872/
       DATA X48/
     &.0323802,-.0323802,.0970047,-.0970047,.1612224,-.1612224,
     &.2247638,-.2247638,.2873625,-.2873625,.3487559,-.3487559,
     &.4086865,-.4086865,.4669029,-.4669029,.5231610,-.5231610,
```

```
     &.5772247,-.5772247,.6288674,-.6288674,.6778724,-.6778724,
     &.7240341,-.7240341,.7671590,-.7671590,.8070662,-.8070662,
     &.8435882,-.8435882,.8765720,-.8765720,.9058791,-.9058791,
     &.9313867,-.9313867,.9529877,-.9529877,.9705916,-.9705916,
     &.9841246,-.9841246,.9935302,-.9935302,.9987710,-.9987710/
      DATA X96/
     &.0162767,-.0162767,.0488130,-.0488130,.0812975,-.0812975,
     &.1136959,-.1136959,.1459737,-.1459737,.1780969,-.1780969,
     &.2100313,-.2100313,.2417432,-.2417432,.2731988,-.2731988,
     &.3043649,-.3043649,.3352085,-.3352085,.3656969,-.3656969,
     &.3957976,-.3957976,.4254790,-.4254790,.4547094,-.4547094,
     &.4834580,-.4834580,.5116942,-.5116942,.5393881,-.5393881,
     &.5665104,-.5665104,.5930324,-.5930324,.6189258,-.6189258,
     &.6441634,-.6441634,.6687183,-.6687183,.6925645,-.6925645,
     &.7156768,-.7156768,.7380306,-.7380306,.7596023,-.7596023,
     &.7803690,-.7803690,.8003087,-.8003087,.8194003,-.8194003,
     &.8376235,-.8376235,.8549590,-.8549590,.8713885,-.8713885,
     &.8868945,-.8868945,.9014606,-.9014606,.9150714,-.9150714,
     &.9277125,-.9277125,.9393703,-.9393703,.9500327,-.9500327,
     &.9596883,-.9596883,.9683268,-.9683268,.9759392,-.9759392,
     &.9825173,-.9825173,.9880541,-.9880541,.9925439,-.9925439,
     &.9959818,-.9959818,.9983644,-.9983644,.9996895,-.9996895/
C
      IF(NN.EQ.6 ) XGAUS= X6(I)
      IF(NN.EQ.8 ) XGAUS= X8(I)
      IF(NN.EQ.12) XGAUS=X12(I)
      IF(NN.EQ.16) XGAUS=X16(I)
      IF(NN.EQ.24) XGAUS=X24(I)
      IF(NN.EQ.48) XGAUS=X48(I)
      IF(NN.EQ.96) XGAUS=X96(I)
      RETURN
      END
C*************
      REAL FUNCTION WGAUS(I,NN)
      DIMENSION W6(6),W8(8),W12(12),W16(16),W24(24),W48(48),W96(96)
C
      DATA W6/ .4679139,.4679139,.3607616,
     &         .3607616,.1713245,.1713245/
      DATA W8/ .3626838,.3626838,.3137066,.3137066,
     &         .2223810,.2223810,.1012285,.1012285/
      DATA W12/.2491470,.2491470,.2334925,.2334925,.2031674,.2031674,
     &         .1600783,.1600783,.1069393,.1069393,.0471753,.0471753/
      DATA W16/.1894506,.1894506,.1826034,.1826034,.1691565,.1691565,
     &         .1495960,.1495960,.1246290,.1246290,.0951585,.0951585,
     &         .0622535,.0622535,.0271525,.0271525/
      DATA W24/
     &.1279382,.1279382,.1258374,.1258374,.1216705,.1216705,
     &.1155057,.1155057,.1074443,.1074443,.0976186,.0976186,
     &.0861902,.0861902,.0733465,.0733465,.0592986,.0592986,
     &.0442774,.0442774,.0285314,.0285314,.0123412,.0123412/
      DATA W48/
     &.0647377,.0647377,.0644662,.0644662,.0639242,.0639242,
     &.0631142,.0631142,.0620394,.0620394,.0607044,.0607044,
     &.0591148,.0591148,.0572773,.0572773,.0551995,.0551995,
     &.0528902,.0528902,.0503590,.0503590,.0476167,.0476167,
     &.0446746,.0446746,.0415451,.0415451,.0382414,.0382414,
     &.0347772,.0347772,.0311672,.0311672,.0274265,.0274265,
     &.0235708,.0235708,.0196162,.0196162,.0155793,.0155793,
     &.0114772,.0114772,.0073276,.0073276,.0031533,.0031533/
      DATA W96/
     &.0325506,.0325506,.0325161,.0325161,.0324472,.0324472,
     &.0323438,.0323438,.0322062,.0322062,.0320345,.0320345,
     &.0318288,.0318288,.0315893,.0315893,.0313164,.0313164,
     &.0310103,.0310103,.0306714,.0306714,.0302999,.0302999,
     &.0298963,.0298963,.0294611,.0294611,.0289946,.0289946,
     &.0284974,.0284974,.0279700,.0279700,.0274130,.0274130,
     &.0268269,.0268269,.0262123,.0262123,.0255700,.0255700,
     &.0249006,.0249006,.0242048,.0242048,.0234834,.0234834,
     &.0227371,.0227371,.0219666,.0219666,.0211729,.0211729,
     &.0203568,.0203568,.0195191,.0195191,.0186607,.0186607,
     &.0177825,.0177825,.0168855,.0168855,.0159706,.0159706,
     &.0150387,.0150387,.0140909,.0140909,.0131282,.0131282,
     &.0121516,.0121516,.0111621,.0111621,.0101607,.0101607,
     &.0091487,.0091487,.0081269,.0081269,.0070965,.0070965,
     &.0060585,.0060585,.0050142,.0050142,.0039646,.0039646,
     &.0029107,.0029107,.0018540,.0018540,.0007968,.0007968/
      IF(NN.EQ.6 ) WGAUS= W6(I)
      IF(NN.EQ.8 ) WGAUS= W8(I)
      IF(NN.EQ.12) WGAUS=W12(I)
      IF(NN.EQ.16) WGAUS=W16(I)
      IF(NN.EQ.24) WGAUS=W24(I)
      IF(NN.EQ.48) WGAUS=W48(I)
      IF(NN.EQ.96) WGAUS=W96(I)
      RETURN
      END
C*****************************************************************C
```

```
C*****************************************************************C
C                                                                 C
C        SUBROUTINE   REDSUBS                                     C
C                                                                 C
C                                                                 C
C                                                      May 1990   C
C*****************************************************************C
C*****************************************************************C
C                                                                 C
C                                                                 C
C-----------------------------------------------------------------*
C                                                                 *
C  ** SUBROUTINE GEOVOL **                                        *
C                                                                 *
C  Called by RUFCODE to read input data specifying the geometrical *
C  and electrical characteristics of the scatterer.  This routine  *
C  computes the values of x and y and the the surface normal       *
C  vector of a 5x5 grid of sampling points for each volume cell.   *
C                                                                 *
C-----------------------------------------------------------------*
      SUBROUTINE GEOVOL(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,
     1               EMUL,HMUL,WAVE,TAUS,XAA,YAA,XBB,YBB,id,ident,
     &               nrmcell)
      COMPLEX YE(MDIM,7),YM(MDIM,7),CI
      COMPLEX PERM1E,PERM2E,PERM3E,PERM4E,PERM5E,ZCOMP
      COMPLEX PERM1U,PERM2U,PERM3U,PERM4U,PERM5U
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      REAL D,DX,DIS,DY,XA,YA,XB,YB,XNORM,YNORM,TX,TY
      REAL X(MDIM,5,5),Y(MDIM,5,5)
      REAL XN(MDIM,5,5),YN(MDIM,5,5),TAUS(MDIM,5)
      REAL EMUL(MDIM),HMUL(MDIM),S(MDIM,5),DSQ(MDIM,5)
      REAL XAA(MDIM),YAA(MDIM),XBB(MDIM),YBB(MDIM)
      REAL XTS(3),YTS(3),XTN(3),YTN(3)
      integer ident(mdim)
      INTEGER ZESPEC,ZMSPEC,ZETAP,ZMTAP,IETAP,IMTAP,NSEG(MDIM,3)
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      XK=TWOPI
      IVCELL=0
      IVOL=0
      M=0
      id = 0
C
C**** READ IN IMPEDANCE AND TAPERING SPECIFICATIONS
C
      EMUL(1)=REAL(ZCOMP(0,0,0,0))
C
C***** Read input parameters and  generate sampling points
C
10    READ (5,203,END=499) NL,N,XA,YA,XB,YB,THIC,angle,INFZ,INFZT,INFI
      IF(NL.EQ.0) GO TO 499
      ZESPEC= INFZ/10
      ZMSPEC= INFZ-(INFZ/10)*10
      ZETAP = INFZT/10
      ZMTAP = INFZT-(INFZT/10)*10
      IETAP = INFI/10
      IMTAP = INFI-(INFI/10)*10
      RXN = XB
      RYN = YB
120   XA=XA/WAVE
      YA=YA/WAVE
      XB=XB/WAVE
      YB=YB/WAVE
      AREA=THIC/(WAVE*WAVE)
      THIC=THIC/WAVE
      TAU = THIC/NL
      tau2 = tau/2.0
      IVOL = IVOL + 1
      TX = XB - XA
      TY = YB - YA
      D = SQRT(TX*TX+TY*TY)
      XNORM = -TY/D
      YNORM = TX/D
C
C ***** Curved Type Cells ******
C
      IF(nint(angle).ne.0) THEN
C***** CURVED Volume SEGMENT *******
C
         T = 0.5*RED*angle
         COST = COS(T)
         SINT = SIN(T)
         RADr = 0.5*D/SIN(T)
         ARC = 2.0*RADr*T
         ALF = 2.0*T/N
         XO = (XA+XB)*0.5-RADr*COST*XNORM
         YO = (YA+YB)*0.5-RADr*COST*YNORM
         DS = (4.0*RADr*ALF)
```

```
c
          do 122 k = 1,NL
           id = id + 1
           BETA = ATAN2((YA-YO),(XA-XO)) - alf/2.0
           rad = radr - thic/2.0 + tau2 +float(k-1)*tau
           Ru = Rad + tau2
           Rl = Rad - tau2
           Rcl = Rad - tau/4.0
           Rcu = Rad + tau/4.0
c          print*,'Radii:',Rl,Rcl,Rad,Rcu,Ru
           WRITE(6,360) IVOL,k,N,XA,YA,xb,yb,thic,angle

          DO 110 INDX=1,N
           IVCELL = IVCELL + 1
           IF(IVCELL .EQ. MDIM) WRITE(6,400) MDIM
           NSEG(IVCELL,1)=IVOL
           NSEG(IVCELL,2)= 1
           NSEG(IVCELL,3)= 1
           ident(ivcell) = id
           dsq(ivcell,3) = alf*Rad
           S(IVCELL,3) = 0.
           TAUS(IVCELL,3) = tau
C
             COSBTA = COS(BETA)
             SINBTA = SIN(BETA)
             thetas = beta + alf/2.0
             thetaf = beta - alf/2.0
             thetal = beta + alf/4.0
             thetar = beta - alf/4.0
c             print*,'Angles: ',ivcell,thetas,thetal,beta,thetar,thetaf
c
c Set up the points now
c
c Point 1,1
             x(ivcell,1,1) = XO+Rl * cos(thetas)
             y(ivcell,1,1) = YO+Rl * sin(thetas)
c Point 1,2
             x(ivcell,1,2) = XO+Rl * cos(thetal)
             y(ivcell,1,2) = YO+Rl * sin(thetal)
c Point 1,3
             x(ivcell,1,3) = XO+Rl * cos(beta)
             y(ivcell,1,3) = YO+Rl * sin(beta)
c Point 1,4
             x(ivcell,1,4) = XO+Rl * cos(thetar)
             y(ivcell,1,4) = YO+Rl * sin(thetar)
c Point 1,5
             x(ivcell,1,5) = XO+Rl * cos(thetaf)
             y(ivcell,1,5) = YO+Rl * sin(thetaf)
c Point 2,1
             x(ivcell,2,1) = XO+Rcl * cos(thetas)
             y(ivcell,2,1) = YO+Rcl * sin(thetas)
c Point 2,2
             x(ivcell,2,2) = XO+Rcl * cos(thetal)
             y(ivcell,2,2) = YO+Rcl * sin(thetal)
c Point 2,3
             x(ivcell,2,3) = XO+Rcl * cos(beta)
             y(ivcell,2,3) = YO+Rcl * sin(beta)
c Point 2,4
             x(ivcell,2,4) = XO+Rcl * cos(thetar)
             y(ivcell,2,4) = YO+Rcl * sin(thetar)
c Point 2,5
             x(ivcell,2,5) = XO+Rcl * cos(thetaf)
             y(ivcell,2,5) = YO+Rcl * sin(thetaf)
c Point 3,1
             x(ivcell,3,1) = XO+Rad * cos(thetas)
             y(ivcell,3,1) = YO+Rad * sin(thetas)
c Point 3,2
             x(ivcell,3,2) = XO+Rad * cos(thetal)
             y(ivcell,3,2) = YO+Rad * sin(thetal)
c Point 3,3
             x(ivcell,3,3) = XO+Rad * cos(beta)
             y(ivcell,3,3) = YO+Rad * sin(beta)
c Point 3,4
             x(ivcell,3,4) = XO+Rad * cos(thetar)
             y(ivcell,3,4) = YO+Rad * sin(thetar)
c Point 3,5
             x(ivcell,3,5) = XO+Rad * cos(thetaf)
             y(ivcell,3,5) = YO+Rad * sin(thetaf)
c Point 4,1
             x(ivcell,4,1) = XO+Rcu * cos(thetas)
             y(ivcell,4,1) = YO+Rcu * sin(thetas)
c Point 4,2
             x(ivcell,4,2) = XO+Rcu * cos(thetal)
             y(ivcell,4,2) = YO+Rcu * sin(thetal)
c Point 4,3
             x(ivcell,4,3) = XO+Rcu * cos(beta)
             y(ivcell,4,3) = YO+Rcu * sin(beta)
```

```
c Point 4,4
              x(ivcell,4,4) = XO+Rcu * cos(thetar)
              y(ivcell,4,4) = YO+Rcu * sin(thetar)
c Point 4,5
              x(ivcell,4,5) = XO+Ru * cos(thetaf)
              y(ivcell,4,5) = YO+Ru * sin(thetaf)
c Point 5,1
              x(ivcell,5,1) = XO+Ru * cos(thetas)
              y(ivcell,5,1) = YO+Ru * sin(thetas)
c Point 5,2
              x(ivcell,5,2) = XO+Ru * cos(thetal)
              y(ivcell,5,2) = YO+Ru * sin(thetal)
c Point 5,3
              x(ivcell,5,3) = XO+Ru * cos(beta)
              y(ivcell,5,3) = YO+Ru * sin(beta)
c Point 5,4
              x(ivcell,5,4) = XO+Ru * cos(thetar)
              y(ivcell,5,4) = YO+Ru * sin(thetar)
c Point 5,5
              x(ivcell,5,5) = XO+Ru * cos(thetaf)
              y(ivcell,5,5) = YO+Ru * sin(thetaf)
c
              do 109 lk = 1,5
                do 108 kl = 1,5
                 XN(ivCELL,kl,lk) = COSBTA
                 YN(ivCELL,kl,lk) = SINBTA
 108            continue
 109           continue
               BETA = BETA - ALF
               I=ivCELL
 110      CONTINUE
 122      continue
           GO TO 250
       ENDIF
C
C ******* Rectangular Cells *******
C
       TX=XB-XA
       TY=YB-YA
       D=SQRT(TX*TX+TY*TY)
       DX=TX/N
       DY=TY/N
       XNORM=-TY/D
       YNORM=TX/D
       DS = (D/N)
C
       DO 210 K=1,NL
          id = id + 1
          DIS = THIC/2. - TAU/2*(2*K-1)
          XAA(K)=(XA-DIS*TY/D)
          YAA(K)=(YA+DIS*TX/D)
          XBB(K)=(XB-DIS*TY/D)
          YBB(K)=(YB+DIS*TX/D)
          WRITE(6,350) IVOL,K,N,XAA(K),YAA(K),XBB(K),YBB(K)
C
          DO 155 INDX=1,N
             IVCELL=IVCELL+1
             IF (IVCELL .EQ. MDIM) WRITE(6,400) MDIM
             NSEG(IVCELL,1)=IVOL
             NSEG(IVCELL,2)=2
             NSEG(IVCELL,3)=1
             ident(ivcell) = id
             DO 150 KNDX=1,5
                DSQ(IVCELL,KNDX)=(D/N)
                ST=DS*(FLOAT(INDX)-.5)
                S(IVCELL,KNDX)=ST
                TAUS(IVCELL,KNDX)=TAU
                XSA=XAA(K)-Dble(KNDX-3)*(TY/D)*TAU/4.
                YSA=YAA(K)+Dble(KNDX-3)*(TX/D)*TAU/4.
                DO 140 JNDX=1,5
C                    XSB=XBB(K)-Dble(KNDX-3)*TY/D*TAU/4.
C                    YSB=YBB(K)+Dble(KNDX-3)*TX/D*TAU/4.
                     X(IVCELL,KNDX,JNDX)=(XSA+DX*.25D0*
     &                          Dble(4*INDX+JNDX-5))
                     Y(IVCELL,KNDX,JNDX)=(YSA+DY*.25D0*
     &                           Dble(4*INDX+JNDX-5))
                     XN(IVCELL,KNDX,JNDX)=(XNORM)
                     YN(IVCELL,KNDX,JNDX)=(YNORM)
 140             CONTINUE
 150          CONTINUE
 155       CONTINUE
 210    CONTINUE
 250    NST=M+1
C
C
C***** PERFORM TAPERING AND IMPEDANCE COMPUTATIONS
C
```

```
C
      DELL=.001
      DO 170 I=NST,IVCELL
         XTS(1)=X(I,3,3)+DELL*(X(I,3,2)-X(I,3,3))
         XTN(1)=X(I,3,3)+DELL*(X(I,2,3)-X(I,3,3))
         YTS(1)=Y(I,3,3)+DELL*(Y(I,3,2)-Y(I,3,3))
         YTN(1)=Y(I,3,3)+DELL*(Y(I,2,3)-Y(I,3,3))
         XTS(2)=X(I,3,3)
         XTN(2)=X(I,3,3)
         YTS(2)=Y(I,3,3)
         YTN(2)=Y(I,3,3)
         XTS(3)=X(I,3,3)+DELL*(X(I,3,4)-X(I,3,3))
         XTN(3)=X(I,3,3)+DELL*(X(I,4,3)-X(I,3,3))
         YTS(3)=Y(I,3,3)+DELL*(Y(I,3,4)-Y(I,3,3))
         YTN(3)=Y(I,3,3)+DELL*(Y(I,4,3)-Y(I,3,3))
         PERM1E=ZCOMP(XTS(1),YTS(1),ZESPEC,ZETAP)
         PERM2E=ZCOMP(XTS(2),YTS(2),ZESPEC,ZETAP)
         PERM3E=ZCOMP(XTS(3),YTS(3),ZESPEC,ZETAP)
         PERM4E=ZCOMP(XTN(1),YTN(1),ZESPEC,ZETAP)
         PERM5E=ZCOMP(XTN(3),YTN(3),ZESPEC,ZETAP)
C
         PERM1U=ZCOMP(XTS(1),YTS(1),ZMSPEC,ZMTAP)
         PERM2U=ZCOMP(XTS(2),YTS(2),ZMSPEC,ZMTAP)
         PERM3U=ZCOMP(XTS(3),YTS(3),ZMSPEC,ZMTAP)
         PERM4U=ZCOMP(XTN(1),YTN(1),ZMSPEC,ZMTAP)
         PERM5U=ZCOMP(XTN(3),YTN(3),ZMSPEC,ZMTAP)
         YE(I,1)=  1./PERM2E
         YE(I,2)=     PERM2E-1./PERM2U
         YE(I,3)=-1./(PERM2E*PERM2E)*(PERM3E-PERM1E)/(2.*DELL)
         YE(I,4)=-1./(PERM2E*PERM2E)*(PERM5E-PERM4E)/(2.*DELL)
         YE(I,5)= 0.
         YE(I,6)= 0.
         YE(I,7)= 0.
         YM(I,1)=  1./PERM2U
         YM(I,2)=     PERM2U-1./PERM2E
         YM(I,3)=-1./(PERM2U*PERM2U)*(PERM3U-PERM1U)/(2.*DELL)
         YM(I,4)=-1./(PERM2U*PERM2U)*(PERM5U-PERM4U)/(2.*DELL)
         YM(I,5)= 0.
         YM(I,6)= 0.
         YM(I,7)= 0.
         EMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IETAP))
         HMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IMTAP))
170   CONTINUE
      M = IVCELL
C
C******BEGIN LOOP AGAIN
C
      GOTO 10
C     GOTO 499
C
C*****FORMATS
C
203   FORMAT(I3,I5,6F10.5,3I3)
C
350   FORMAT(' VOLUME SEGMENT: ',I4,I4,I5,3X,4F10.5)
360   FORMAT(' VOLUME SEGMENT: ',I4,I4,I5,3X,6F10.5,3X,
     &       'PARAMETRIC  CELL')
400   FORMAT('0 WARNING....',I5,' POINTS HAVE BEEN GENERATED')
C
499   nrmcell = ivol
      RETURN
      END
C*****************************************************************C
C                                                                 *
C  ** SUBROUTINE GEOCON **                                        *
C                                                                 *
C  Called by RUFCODE to read input data specifying the geometrical *
C  and electrical characteristics of the scatterer.  This routine  *
C  computes the values of x and y and the the surface normal       *
C  vector of a grid of 5 sampling points for each contour cell.    *
C                                                                 *
C-----------------------------------------------------------------*
      SUBROUTINE GEOCON(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,L,
     1                  EMUL,HMUL,WAVE,TAUS,CXAA,CYAA,id,ident)
      COMPLEX YE(MDIM,7),YM(MDIM,7),CI,ZCOMP
      COMPLEX PERM1E,PERM2E,PERM3E,PERM4E,PERM5E,PERM6E,PERM7E,PERM8E
      COMPLEX PERM1U,PERM2U,PERM3U,PERM4U,PERM5U,PERM6U,PERM7U,PERM8U
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      REAL D,DX,DY,CXA,CYA,CXB,CYB,XNORM,YNORM,TX,TY
      REAL X(MDIM,5,5),Y(MDIM,5,5),XN(MDIM,5,5),YN(MDIM,5,5)
      REAL DSQ(MDIM,5),TAUS(MDIM,5)
      REAL S(MDIM,5),EMUL(MDIM),HMUL(MDIM)
      REAL CXAA(MDIM),CYAA(MDIM),CANGLE
      REAL XTS(3),YTS(3),XTN(3),YTN(3)
      INTEGER NSEG(MDIM,3),CCELL,ident(mdim)
      INTEGER ZESPEC(2),ZMSPEC(2),ZETAP(2),ZMTAP(2),IETAP,IMTAP
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
```

```
      XK=TWOPI
      CCELL= M
      IVOL=0
      L=M
C
C***** Read input parameters and  generate sampling points
C
10    READ(5,203,END=195) CANGLE,N,CXA,CYA,CXB,CYB,INFZ,INFZT,INFI
      IF(CANGLE .GT. 360) GO TO 195
      ZESPEC(1)= INFZ/1000
      ZESPEC(2)= INFZ/100-(INFZ/1000)*10
      ZMSPEC(1)= INFZ/10 -(INFZ/100)*10
      ZMSPEC(2)= INFZ    -(INFZ/10)*10
      ZETAP(1) = INFZT/1000
      ZETAP(2) = INFZT/100-(INFZT/1000)*10
      ZMTAP(1) = INFZT/10 -(INFZT/100)*10
      ZMTAP(2) = INFZT    -(INFZT/10)*10
      IETAP    = INFI/10
      IMTAP    = INFI-(INFI/10)*10
      CXA = CXA/WAVE
      CYA = CYA/WAVE
      CXB = CXB/WAVE
      CYB = CYB/WAVE
      TX = CXB - CXA
      TY = CYB - CYA
      D = SQRT(TX*TX+TY*TY)
      DX = TX/N
      DY = TY/N
      XNORM = -TY/D
      YNORM = TX/D
      DS = (D/N)
      WRITE(6,351)CANGLE,N,CXA,CYA,CXB,CYB
C
      IF(INT(CANGLE) .EQ. 0) GO TO 30
C
C***** CURVED CONTOUR SEGMENT *******
C
      T = 0.5*RED*CANGLE
      COST = COS(T)
      SINT = SIN(T)
      RAD = 0.5*D/SIN(T)
      ARC = 2.0*RAD*T
      ALF = T/N/2.0
      XO = (CXA+CXB)*0.5-RAD*COST*XNORM
      YO = (CYA+CYB)*0.5-RAD*COST*YNORM
      BETA = ATAN2((CYA-YO),(CXA-XO))
      DS = (4.0*RAD*ALF)
      id = id + 1
C
        DO 50 INDX=1,N
          CCELL = CCELL + 1
          IF(CCELL .EQ. MDIM) WRITE(6,400) MDIM
          NSEG(CCELL,1) = INT(CANGLE)
          NSEG(CCELL,2) = 0
          NSEG(CCELL,3) = 2
          ident(ccell) = id
C
           DO 40 JNDX=1,5
             DSQ(CCELL,JNDX) = DS
             TAUS(CCELL,JNDX) = 0.0
             ST = DS*(FLOAT(INDX)-0.5)
             S(CCELL,JNDX) = ST
             COSBTA = COS(BETA)
             SINBTA = SIN(BETA)
             X(CCELL,JNDX,1) = XO+RAD*COSBTA
             Y(CCELL,JNDX,1) = YO+RAD*SINBTA
             XN(CCELL,JNDX,1) = COSBTA
             YN(CCELL,JNDX,1) = SINBTA
             IF(JNDX .NE. 5) BETA = BETA - ALF
 40        CONTINUE
 50     CONTINUE
        GO TO 70
C
C***** STRAIGHT CONTOUR SEGMENT *****
 30   id = id + 1
      DO 220 INDX=1,N
         CCELL = CCELL + 1
         IF(CCELL .EQ. MDIM) WRITE(6,400) MDIM
         NSEG(CCELL,1) = INT(CANGLE)
         NSEG(CCELL,2) = 0
         NSEG(CCELL,3) = 2
         ident(ccell) = id
         CXAA(INDX) = CXA + (INDX-1)*DX
         CYAA(INDX) = CYA + (INDX-1)*DY
         DO 215 KCNDX=1,5
              DSQ(CCELL,KCNDX)=(D/N)
              ST=DS*(FLOAT(INDX)-0.5)
```

```
              S(CCELL,KCNDX) = ST
              TAUS(CCELL,KCNDX)=0.
              X(CCELL,KCNDX,1)=CXAA(INDX)+FLOAT(KCNDX-1)*DX/4.
              Y(CCELL,KCNDX,1)=CYAA(INDX)+FLOAT(KCNDX-1)*DY/4.
              XN(CCELL,KCNDX,1)=(XNORM)
              YN(CCELL,KCNDX,1)=(YNORM)
215       CONTINUE
220    CONTINUE
C
C******** PERFORM TAPERING AND IMPEDANCE COMPUTATIONS
C
70     DELL= 0.001
       DO 180 I=L+1,CCELL
          IF(NSEG(I,3).EQ.2) THEN
            XTS(1)=X(I,3,1)+DELL*(X(I,2,1)-X(I,3,1))
            XTN(1)=X(I,3,1)+DELL*XN(I,3,1)
            YTS(1)=Y(I,3,1)+DELL*(Y(I,2,1)-Y(I,3,1))
            YTN(1)=Y(I,3,1)+DELL*YN(I,3,1)
            XTS(2)=X(I,3,1)
            XTN(2)=X(I,3,1)
            YTS(2)=Y(I,3,1)
            YTS(2)=Y(I,3,1)
            XTS(3)=X(I,3,1)+DELL*(X(I,4,1)-X(I,3,1))
            XTN(3)=X(I,3,1)-DELL*XN(I,3,1)
            YTS(3)=Y(I,3,1)+DELL*(Y(I,4,1)-Y(I,3,1))
            YTN(3)=Y(I,3,1)-DELL+YN(I,3,1)
          ENDIF
          PERM1E=ZCOMP(XTS(1),YTS(1),ZESPEC(1),ZETAP(1))
          PERM2E=ZCOMP(XTS(2),YTS(2),ZESPEC(1),ZETAP(1))
          PERM3E=ZCOMP(XTS(3),YTS(3),ZESPEC(1),ZETAP(1))
          PERM4E=ZCOMP(XTN(1),YTN(1),ZESPEC(1),ZETAP(1))
          PERM5E=ZCOMP(XTS(1),YTS(1),ZESPEC(2),ZETAP(2))
          PERM6E=ZCOMP(XTS(2),YTS(2),ZESPEC(2),ZETAP(2))
          PERM7E=ZCOMP(XTS(3),YTS(3),ZESPEC(2),ZETAP(2))
          PERM8E=ZCOMP(XTN(3),YTN(3),ZESPEC(2),ZETAP(2))
C
          PERM1U=ZCOMP(XTS(1),YTS(1),ZMSPEC(1),ZMTAP(1))
          PERM2U=ZCOMP(XTS(2),YTS(2),ZMSPEC(1),ZMTAP(1))
          PERM3U=ZCOMP(XTS(3),YTS(3),ZMSPEC(1),ZMTAP(1))
          PERM4U=ZCOMP(XTN(1),YTN(1),ZMSPEC(1),ZMTAP(1))
          PERM5U=ZCOMP(XTS(1),YTS(1),ZMSPEC(2),ZMTAP(2))
          PERM6U=ZCOMP(XTS(2),YTS(2),ZMSPEC(2),ZMTAP(2))
          PERM7U=ZCOMP(XTS(3),YTS(3),ZMSPEC(2),ZMTAP(2))
          PERM8U=ZCOMP(XTN(3),YTN(3),ZMSPEC(2),ZMTAP(2))
C
          YE(I,1)=   1./PERM2E
          YE(I,2)=   0.
          YE(I,3)=   0.
          YE(I,4)=   0.
          YE(I,5)=  (PERM2E-1.)/PERM2E-(PERM6E-1.)/PERM6E
          YE(I,6)=   1./PERM6E
          YE(I,7)=  (PERM2E-1.)/PERM2E+(PERM6E-1.)/PERM6E
          YM(I,1)=   1./PERM2U
          YM(I,2)=   0.
          YM(I,3)=   0.
          YM(I,4)=   0.
          YM(I,5)=  (PERM2U-1.)/PERM2U-(PERM6U-1.)/PERM6U
          YM(I,6)=   1./PERM6U
          YM(I,7)=  (PERM2U-1.)/PERM2U+(PERM6U-1.)/PERM6U
          EMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IETAP))
          HMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IMTAP))
180    CONTINUE
       L= CCELL
C
C***** BEGIN LOOP AGAIN
C
       GOTO 10
195    CONTINUE
       WRITE(6,360) L
       GO TO 499
C
C***** Formats
C
203    FORMAT(F5.1,I3,4F10.5,2I5,I3)
351    FORMAT(' CONTOUR SEGMENT:',F7.1,3X,I3,3X,4F10.5)
360    FORMAT(' TOTAL NUMBER OF SAMPLING POINTS = ',I5)
400    FORMAT('0 WARNING... ',I5,' POINTS HAVE BEEN GENERATED')
C
499    RETURN
       END
C-----------------------------------------------------------------------*
C                                                                       *
C  ** FUNCTION ZCOMP **                                                 *
C                                                                       *
C  This subroutine computes the permitivity/permeability for a         *
C  given taper spec. (ZSPEC.NE.0) or just the taper coef.(ZSPEC=0).*
C                                                                       *
```

```
C-----------------------------------------------------------------*
C
      COMPLEX FUNCTION ZCOMP(X,Y,ZSPEC,TSPEC)
      PARAMETER(JDIM=50)
      REAL C(JDIM,4),P(JDIM),CC(4)
      COMPLEX ZA(JDIM),ZB(JDIM),ZAA,ZBB
      INTEGER SPECZ(JDIM),ZSPEC,SPECT(JDIM),TSPEC
      INTEGER TINFO(JDIM),TT
      DATA IREAD /0/
C
C**** READ MATERIAL SPECIFICATIONS
C
      IF(IREAD.NE.0) GO TO 30
      I=0
10    I=I+1
      READ (5,13,END=12) SPECZ(I),ZA(I),ZB(I)
      IF(SPECZ(I).EQ.0) THEN
12      NIMP=I-1
        IREAD=1
        GO TO 20
      ENDIF
      GO TO 10
13    FORMAT(I3,4F13.3)
C
C**** READ TAPER SPECIFICATIONS
C
20    I=0
22    I=I+1
      READ (5,26,END=24) SPECT(I),TINFO(I),(C(I,J),J=1,4),P(I)
      IF(SPECT(I).EQ.0) THEN
24      NTAP=I-1
        IREAD=1
        GO TO 1000
      ENDIF
      GO TO 22
26    FORMAT(I3,I3,5F10.5)
C
C**** SEARCH FOR PROPER MATERIAL SPECIFICATIONS
C
30    IF(ZSPEC.EQ.0) GO TO 35
      DO 32 I=1,NIMP
         IF(ZSPEC.EQ.SPECZ(I)) THEN
           ZAA=ZA(I)
           ZBB=ZB(I)
           GO TO 35
         ENDIF
32    CONTINUE
      PRINT *,'THIS MATERIAL SPEC. DOES NOT EXIST'
      GO TO 1000
C
C**** SEARCH FOR PROPER TAPER SPECIFICATIONS
C
35    DO 45 I=1,NTAP
         IF(TSPEC.EQ.SPECT(I)) THEN
           TT=TINFO(I)
           PP=P(I)
           DO 40 J=1,4
                CC(J)=C(I,J)
40         CONTINUE
           GO TO 50
         ENDIF
45    CONTINUE
      PRINT *,'TAPER SPEC. NOT FOUND'
      GO TO 1000
C
50    IF(ZSPEC.EQ.0) THEN
        ZCOMP=TAPER(X,Y,TT,CC,PP)
          ELSE
        ZCOMP=ZBB+(ZAA-ZBB)*TAPER(X,Y,TT,CC,PP)
      ENDIF
1000  RETURN
      END
C-----------------------------------------------------------------*
C                                                                 *
C  ** FUNCTION TAPER **                                          *
C                                                                 *
C  This subroutine computes resistive or current tapering         *
C  coefficients.                                                  *
C                                                                 *
C-----------------------------------------------------------------*
C                                                                 *
C  *tapering inputs*                                              *
C                                                                 *
C    JJTAP  :     ones digit = direction of taper                 *
C                   0 = NO TAPER'/                                *
C                   1 = TAPER(X),1-SIDED'/                        *
C                   2 = TAPER(X),2-SIDED'/                        *
```

```
C                   3 = TAPER(Y),1-SIDED'/                          *
C                   4 = TAPER(Y),2-SIDED'/                          *
C                   5 = TAPER(XP,YP); (TAPER W/ RESPECT TO POINT)'  *
C                 tens digit = tapering characteristic              *
C                   1 = LINEAR'/                                    *
C                   2 = GAUSSIAN'/                                  *
C                   3 = COS**N (HANNING)'/                          *
C                   4 = BLACKMAN-HARRIS')                           *
C    GEO    :     if dir. of taper = 1 then                         *
C                    GEO(1) = x coord where taper originates        *
C                    GEO(2) = x coord where taper ends              *
C                 if dir. of taper = 2 then                         *
C                    GEO(1) = first x coord where taper originates  *
C                    GEO(2) = first x coord where taper ends        *
C                    GEO(3) = second x coord where taper originates*
C                    GEO(4) = second x coord where taper ends       *
C                 if dir. of taper = 3 then                         *
C                    GEO(1) = y coord where taper originates        *
C                    GEO(2) = y coord where taper ends              *
C                 if dir. of taper = 4 then                         *
C                    GEO(1) = first y coord where taper originates  *
C                    GEO(2) = first y coord where taper ends        *
C                    GEO(3) = second y coord where taper originates*
C                    GEO(4) = second y coord where taper ends       *
C                 if dir. of taper = 5 then                         *
C                    GEO(1) = x coord where radial taper originates*
C                    GEO(2) = y coord where radial taper ends       *
C                    GEO(3) = dist. from (x,y) where tapering ends  *
C    PAR    :     if taper char. = 2 then                           *
C                    PAR = alpha coeff. for gaussian tapering       *
C                 if taper char. = 3 then                           *
C                                                                   *
C  *geometry inputs*                                                *
C    X      :     x-coordinate                                      *
C    Y      :     y-coordinate                                      *
C                                                                   *
C  *outputs*                                                        *
C    TAPER  :     tapering coefficient(resis. or curr.)             *
C                                                                   *
C-------------------------------------------------------------------*
      FUNCTION TAPER(X,Y,JJTAP,GEO,PAR)
      REAL*4 R,GEO(4),X,Y
      REAL PI
C
      DATA PI/3.141592654/
      DATA A0,A1,A2,A3/.35875,.48829,.14128,.01168/
C
      JDIR=MOD(JJTAP,10)
      IF(JDIR .EQ. 0) GO TO 500
100   JTAP=INT(FLOAT(JJTAP)/10.)
      DO 300 I=NST,NEND
         IF(JDIR .EQ. 1 .OR. JDIR .EQ. 2) THEN
           R=(X-GEO(1))/(GEO(2)-GEO(1))
         ELSE IF(JDIR .EQ. 3 .OR. JDIR .EQ. 4) THEN
           R=(Y-GEO(1))/(GEO(2)-GEO(1))
         ELSE IF(JDIR .EQ. 5) THEN
           R=SQRT((GEO(1)-X)**2+(GEO(2)-Y)**2)/GEO(3)
         ENDIF
         IF(R .LT. 0.) GO TO 220
         IF(R .GT. 1.) GO TO 280
         GO TO 240
C
220      IF(JDIR .EQ. 1 .OR. JDIR .EQ. 3) GO TO 260
         IF(JDIR .EQ. 2) R=(X-GEO(3))/(GEO(4)-GEO(3))
         IF(JDIR .EQ. 4) R=(Y-GEO(3))/(GEO(4)-GEO(3))
         IF(R .LT. 0.) GO TO 260
         IF(R .GT. 1.) GO TO 280
C
240      IF(JTAP .EQ. 1) TAPER=1.-R
         IF(JTAP .EQ. 2) TAPER=EXP(-.5*PAR*PAR*R*R)
         IF(JTAP .EQ. 3) TAPER=COS(PI/2.*R)**IFIX(PAR)
         IF(JTAP .EQ. 4) TAPER=
     1  (A0+A1*COS(PI*R)+A2*COS(2.*PI*R)+A3*COS(3.*PI*R))
         GO TO 290
C
C***** R<O.,  MULTIPLIER = 1
C
260      TAPER=1.
         GO TO 290
C
C***** R>1., MULTIPLIER = 0
C
280      TAPER=0.
         IF(JTAP .EQ. 2) TAPER=EXP(-.5*PAR*PAR)
290      CONTINUE
300   CONTINUE
C
```

```
C***** END OF LOOP, TERMINATE
C
      GO TO 1000
C
C***** JDIR=0 (NO TAPERING AT ALL; SET ALL MULT. = 1)
C
500   TAPER=1.
1000  RETURN
      END
C*********************************************************************
      SUBROUTINE CDBLE(AIN,AOUT)
      COMPLEX*8 AIN
      COMPLEX*16 AOUT,CI
      REAL HIM,HRE
      DATA CI/(0.,1.)/
C
      AREAL=REAL(AIN)
      CALL RDBLE(AREAL,HRE)
      AMAG=AIMAG(AIN)
      CALL RDBLE(AMAG,HIM)
C     PRINT *,'AREAL,HRE:',AREAL,HRE
C     PRINT *,'AMAG,HIM:',AMAG,HIM
      AOUT=HRE+CI*HIM
      RETURN
      END
C*********************************************************************
      SUBROUTINE RDBLE(ASING,ADBLE)
      REAL*4 ASING
      REAL ADBLE
      INTEGER*4 AINT
C
      IF(ASING.LT.0.) MUL=-1
      IF(ASING.GT.0.) MUL=1
      IF(ASING.EQ.0.) THEN
        ADBLE=0.
        GO TO 100
      ENDIF
C     PRINT *,'   ASING:',ASING
      ASING=ABS(ASING)
C     PRINT *,'   ASING:',ASING
      NPOW=INT(DLOG10(DBLE(ASING)/.9999999499999D0))
      IF(NPOW.LT.0) NPOW=NPOW-1
C     PRINT *,'   NPOW:',NPOW
      AINT=NINT(ASING*10.D0**(6-NPOW))
C     PRINT *,'10**(6-N):',10.D0**(6-NPOW)
C     PRINT *,'A, AINT:',ASING*10.D0**(6-NPOW),AINT
      ADBLE=Dble(AINT)*10.D0**(NPOW-6)*Dble(MUL)
C     PRINT *,'   ADBLE:',ADBLE
100   RETURN
      END
C*********************************************************************C
      FUNCTION PHASE(X)
      COMPLEX X,CI
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
10    XR=REAL(X)
      XI=AIMAG(X)
      PHASE=0.
      IF(XI .NE. 0. .OR. XR .NE. 0.) THEN
        PHASE=RED*ATAN2(XI,XR)
          ELSE
            PHASE=0.
      ENDIF
      RETURN
      END
C*********************************************************************C
      FUNCTION ATAN3(Y,X)
C*********************************************************************C
C                                                                     C
C     Arctangent function called by SMAIN.                            C
C                                                                     C
      ATAN3=0.0
      IF (Y.NE.0.0 .OR. X.NE.0.0) ATAN3=ATAN2(Y,X)
      RETURN
      END
C*********************************************************************C
C                                                                     C
      SUBROUTINE HANKZ1(R,N,HZERO,HONE)
C                                                                     C
C*********************************************************************C
C                                                                     C
C         Called by subroutines MTXE and MTXH to compute Hankel       C
C     functions of the first kind for orders one and zero.  The       C
C     argument is variable R and must be positive.                    C
C                                                                     C
C.....HANKEL FUNCTIONS ARE OF FIRST KIND--J+IY
C.....   N=0 RETURNS HZERO (H-zero)
```

```
C.....   N=1 RETURNS HONE  (H-one)
C.....   N=2 RETURNS HZERO AND HONE
C.....SUBROUTINE REQUIRES R>0
C.....SUBROUTINE ADAM MUST BE SUPPLIED BY USER
      DIMENSION A(7),B(7),C(7),D(7),E(7),F(7),G(7),H(7)
      COMPLEX HZERO,HONE
      DATA A,B,C,D,E,F,G,H/1.0,-2.2499997,1.2656208,-0.3163866,
     &0.0444479,-0.0039444,0.00021,0.36746691,0.60559366,-0.74350384,
     &0.25300117,-0.04261214,0.00427916,-0.00024846,0.5,-0.56249985,
     &0.21093573,-0.03954289,0.00443319,-0.00031761,0.00001109,
     &-0.6366198,0.2212091,2.1682709,-1.3164827,0.3123951,-0.0400976,
     &0.0027873,0.79788456,-0.00000077,-0.0055274,-0.00009512,
     &0.00137237,-0.00072805,0.00014476,-0.78539816,-0.04166397,
     &-0.00003954,0.00262573,-0.00054125,-0.00029333,0.00013558,
     &0.79788456,0.00000156,0.01659667,0.00017105,-0.00249511,
     &0.00113653,-0.00020033,-2.35619449,0.12499612,0.0000565,
     &-0.00637879,0.00074348,0.00079824,-0.00029166/
      IF (R.LE.0.0) GO TO 50
      IF (N.LT.0.OR.N.GT.2) GO TO 50
      IF (R.GT.3.0) GO TO 20
      X=R*R/9.0
      IF (N.EQ.1) GO TO 10
      CALL ADAM(A,X,BJ)
      CALL ADAM(B,X,Y)
      BY=0.6366198*ALOG(0.5*R)*BJ+Y
      HZERO=CMPLX(BJ,BY)
      IF (N.EQ.0) RETURN
 10   CALL ADAM(C,X,Y)
      BJ=R*Y
      CALL ADAM(D,X,Y)
      BY=0.6366198*ALOG(0.5*R)*BJ+Y/R
      HONE=CMPLX(BJ,BY)
      RETURN
 20   X=3.0/R
      IF (N.EQ.1) GO TO 30
      CALL ADAM(E,X,Y)
      FOOL=Y/SQRT(R)
      CALL ADAM(F,X,Y)
      T=R+Y
      BJ=FOOL*COS(T)
      BY=FOOL*SIN(T)
      HZERO=CMPLX(BJ,BY)
      IF (N.EQ.0) RETURN
 30   CALL ADAM(G,X,Y)
      FOOL=Y/SQRT(R)
      CALL ADAM(H,X,Y)
      T=R+Y
      BJ=FOOL*COS(T)
      BY=FOOL*SIN(T)
      HONE=CMPLX(BJ,BY)
      RETURN
 50   WRITE(6,90) N,R
 90   FORMAT(32H0SICK DATA IN HANKZ1  *QUIT*  N=,I2,2X,2HR=,E11.3)
      CALL SYSTEM
      END
C                                                                      C
C**********************************************************************C
C                                                                      C
      SUBROUTINE ADAM(C,X,Y)
C                                                                      C
C**********************************************************************C
C                                                                      C
C     Called by subroutine HANKZ1 to compute the value of a 7th        C
C     order polynomial whose argument is X and coefficients are        C
C     contained in vector C.                                           C
C                                                                      C
      DIMENSION C(7)
C
      Y=X*C(7)
C
         DO 10 I=1,5
         Y=X*(C(7-I)+Y)
 10      CONTINUE
C
      Y=Y+C(1)
      RETURN
      END
C
C**********************************************************************C
      BLOCK DATA
C**********************************************************************C
C                                                                      C
C     Contains the constants used in common block PIES                 C
C
      COMPLEX CI
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
```

```
      DATA PI,TWOPI,GAMA,RED,DIG,Z0,CI/3.141592654,6.283185308,
     &.57722156649,0.017453293,57.29577951,376.7343,(0.,1.)/
      END
C*****************************************************************C
C                                                                 C
      SUBROUTINE PRMTX(M,MDIM,A,IOPT)
C                                                VER APR 1986     C
C*****************************************************************C
C     This subroutine is used to print the matrix ]A( only.       C
C     Only 24 columns of the matrix are printed. The program is   C
C     intended for debugging purposes only and is controlled by   C
C     activating call statement in RESTE and/or RESTH             C
C*****************************************************************C
      COMPLEX A(MDIM,MDIM)
      DIMENSION B(24)
      WRITE (6,30)
      DO 35 I=1,5
      DO 45 J=1,M
      IF(J.GT.24) GO TO 45
      B(J)=CABS(A(I,J))
 45   CONTINUE
      IF(M .LE. 24) THEN
        MEND=M
          ELSE
            MEND=24
      ENDIF
      IF(IOPT.EQ.1) WRITE (6,40) (B(J),J=1,MEND)
      IF(IOPT.EQ.2) WRITE (6,41) (A(I,J),J=1,MEND)
 35   CONTINUE
 30   FORMAT('0',1X,'A(1,1) A(1,2) A(1,3) A(1,4) A(1,5) A(1,6) A(1,7)'
     &,' A(1,8)',6X,'A(1,10)',7X,'A(1,12)',7X,'A(1,14)',7X,'A(1,16)',
     & 7X,'A(1,18)')
 40   FORMAT(24F7.4)
 41   FORMAT(14( '(' , F7.4 , ',' , F7.4 , ') '),/,'  ',
     &       10( '(' , F7.4 , ',' , F7.4 , ') '))
      RETURN
      END
C*****************************************************************C
C                                                                 C
C                                                                 C
C-----------------------------------------------------------------*
C                                                                 *
C  ** SUBROUTINE Abnormal **                                      *
C                                                                 *
C  Called by RUFCODE to read input data specifying the geometrical *
C  and electrical characteristics of the scatterer.  This routine  *
C  computes the values of x and y and the the surface normal       *
C  vector of a 5x5 grid of sampling points for each abnormal cell. *
C                                                                 *
c By:  Leo C. Kempel                                              *
c Date:  17 May 1990                                              *
C-----------------------------------------------------------------*
      SUBROUTINE Abnormal(MDIM,NSEG,X,Y,XN,YN,S,DSQ,YE,YM,M,
     1               EMUL,HMUL,WAVE,TAUS,XAA,YAA,XBB,YBB,id,ident)
      COMPLEX YE(MDIM,7),YM(MDIM,7),CI
      COMPLEX PERM1E,PERM2E,PERM3E,PERM4E,PERM5E,ZCOMP
      COMPLEX PERM1U,PERM2U,PERM3U,PERM4U,PERM5U
      REAL PI,TWOPI,GAMA,RED,DIG,Z0
      REAL D,DX,DIS,DY,XA,YA,XB,YB,XNORM,YNORM,TX,TY
      REAL X(MDIM,5,5),Y(MDIM,5,5)
      REAL XN(MDIM,5,5),YN(MDIM,5,5),TAUS(MDIM,5)
      REAL EMUL(MDIM),HMUL(MDIM),S(MDIM,5),DSQ(MDIM,5)
      REAL XAA(MDIM),YAA(MDIM),XBB(MDIM),YBB(MDIM)
      REAL XTS(3),YTS(3),XTN(3),YTN(3)
      INTEGER ZESPEC,ZMSPEC,ZETAP,ZMTAP,IETAP,IMTAP,NSEG(MDIM,3)
      integer ident(mdim)
      COMMON /PIES/ PI,TWOPI,GAMA,RED,DIG,Z0,CI
C
      XK=TWOPI
      IVCELL=M
      IVOL=M
C
C***** Read input parameters and  generate sampling points
C

10    READ (5,203,END=499) NL,N,X1,Y1,X2,Y2,X3,Y3,X4,Y4
      IF(NL.EQ.0) GO TO 499
      read (5,204,end=499) X5,Y5,X6,Y6,X7,Y7,X8,Y8,XC,YC
      read (5,205,end=499) xnorm,ynorm,INFZ,INFZT,INFI
      ZESPEC= INFZ/10
      ZMSPEC= INFZ-(INFZ/10)*10
      ZETAP = INFZT/10
      ZMTAP = INFZT-(INFZT/10)*10
      IETAP = INFI/10
      IMTAP = INFI-(INFI/10)*10
c
c Normalize to the wavelength
```

```
c
      x1 = x1 / wave
      x2 = x2 / wave
      x3 = x3 / wave
      x4 = x4 / wave
      x5 = x5 / wave
      x6 = x6 / wave
      x7 = x7 / wave
      x8 = x8 / wave
      xc = xc / wave
c
      y1 = y1 / wave
      y2 = y2 / wave
      y3 = y3 / wave
      y4 = y4 / wave
      y5 = y5 / wave
      y6 = y6 / wave
      y7 = y7 / wave
      y8 = y8 / wave
      yc = yc / wave
c
      tau = tau / wave
      ds = ds / wave
c
      IVOL = IVOL + 1
c
C***** Abnormal Volume Segnment *******
C
      WRITE(6,360) IVOL,XC,YC,xnorm,ynorm

      IVCELL = IVCELL + 1
      IF(IVCELL .EQ. MDIM) WRITE(6,400) MDIM
      NSEG(IVCELL,1)=IVOL
      NSEG(IVCELL,2)= 1
      NSEG(IVCELL,3)= 1
      ident(ivcell) = 999
      dsq(ivcell,3) = 0.
      S(IVCELL,3) = 0.
      TAUS(IVCELL,3) = 0.
C
c
c Set up the points now
c
c Point 1,1
            x(ivcell,1,1) = X4
            y(ivcell,1,1) = Y4
c Point 1,2
            x(ivcell,1,2) = (X8 + X4) / 2.0
            y(ivcell,1,2) = (Y8 + Y4) / 2.0
c Point 1,3
            x(ivcell,1,3) = X8
            y(ivcell,1,3) = Y8
c Point 1,4
            x(ivcell,1,4) = (X1 + X8) / 2.0
            y(ivcell,1,4) = (Y1 + Y8) / 2.0
c Point 1,5
            x(ivcell,1,5) = X1
            y(ivcell,1,5) = Y1
c Point 2,1
            x(ivcell,2,1) = (X7 + X4) / 2.0
            y(ivcell,2,1) = (Y7 + Y4) / 2.0
c Point 2,2
            x(ivcell,2,2) = (X7 + X4 + X8 + X4) / 4.0
            y(ivcell,2,2) = (Y7 + Y4 + Y8 + Y4) / 4.0
c Point 2,3
            x(ivcell,2,3) = (X8 + XC) / 2.0
            y(ivcell,2,3) = (Y8 + YC) / 2.0
c Point 2,4
            x(ivcell,2,4) = (X1 + X5 + X8 + XC) / 4.0
            y(ivcell,2,4) = (Y1 + Y5 + Y8 + YC) / 4.0
c Point 2,5
            x(ivcell,2,5) = (X1 + X5) / 2.0
            y(ivcell,2,5) = (Y1 + Y5) / 2.0
c Point 3,1
            x(ivcell,3,1) = X7
            y(ivcell,3,1) = Y7
c Point 3,2
            x(ivcell,3,2) = (X7 + XC) / 2.0
            y(ivcell,3,2) = (Y7 + YC) / 2.0
c Point 3,3
            x(ivcell,3,3) = XC
            y(ivcell,3,3) = YC
c Point 3,4
            x(ivcell,3,4) = (X5 + XC) / 2.0
            y(ivcell,3,4) = (Y5 + YC) / 2.0
c Point 3,5
            x(ivcell,3,5) = X5
```

```
              y(ivcell,3,5) = Y5
c Point 4,1
              x(ivcell,4,1) = (X3 + X7) / 2.0
              y(ivcell,4,1) = (Y3 + Y7) / 2.0
c Point 4,2
              x(ivcell,4,2) = (X3 + X7 + XC + X6) / 4.0
              y(ivcell,4,2) = (Y3 + Y7 + YC + Y6) / 4.0
c Point 4,3
              x(ivcell,4,3) = (X6 + XC) / 2.0
              y(ivcell,4,3) = (Y6 + YC) / 2.0
c Point 4,4
              x(ivcell,4,4) = (X2 + X5 + X6 + XC) / 4.0
              y(ivcell,4,4) = (Y2 + Y5 + Y6 + YC) / 4.0
c Point 4,5
              x(ivcell,4,5) = (X2 + X5) / 2.0
              y(ivcell,4,5) = (Y2 + Y5) / 2.0
c Point 5,1
              x(ivcell,5,1) = X3
              y(ivcell,5,1) = Y3
c Point 5,2
              x(ivcell,5,2) = (X3 + X6) / 2.0
              y(ivcell,5,2) = (Y3 + Y6) / 2.0
c Point 5,3
              x(ivcell,5,3) = X6
              y(ivcell,5,3) = Y6
c Point 5,4
              x(ivcell,5,4) = (X2 + X6) / 2.0
              y(ivcell,5,4) = (Y2 + Y6) / 2.0
c Point 5,5
              x(ivcell,5,5) = X2
              y(ivcell,5,5) = Y2
c
              do 109 lk = 1,5
                do 108 kl = 1,5
                 XN(ivCELL,kl,lk) = xnorm
                 YN(ivCELL,kl,lk) = ynorm
 108            continue
 109          continue
C
C
C***** PERFORM TAPERING AND IMPEDANCE COMPUTATIONS
C
C
      NST=M+1
      DELL=.001
      DO 170 I=NST,IVCELL
         XTS(1)=X(I,3,3)+DELL*(X(I,3,2)-X(I,3,3))
         XTN(1)=X(I,3,3)+DELL*(X(I,2,3)-X(I,3,3))
         YTS(1)=Y(I,3,3)+DELL*(Y(I,3,2)-Y(I,3,3))
         YTN(1)=Y(I,3,3)+DELL*(Y(I,2,3)-Y(I,3,3))
         XTS(2)=X(I,3,3)
         XTN(2)=X(I,3,3)
         YTS(2)=Y(I,3,3)
         YTN(2)=Y(I,3,3)
         XTS(3)=X(I,3,3)+DELL*(X(I,3,4)-X(I,3,3))
         XTN(3)=X(I,3,3)+DELL*(X(I,4,3)-X(I,3,3))
         YTS(3)=Y(I,3,3)+DELL*(Y(I,3,4)-Y(I,3,3))
         YTN(3)=Y(I,3,3)+DELL*(Y(I,4,3)-Y(I,3,3))
         PERM1E=ZCOMP(XTS(1),YTS(1),ZESPEC,ZETAP)
         PERM2E=ZCOMP(XTS(2),YTS(2),ZESPEC,ZETAP)
         PERM3E=ZCOMP(XTS(3),YTS(3),ZESPEC,ZETAP)
         PERM4E=ZCOMP(XTN(1),YTN(1),ZESPEC,ZETAP)
         PERM5E=ZCOMP(XTN(3),YTN(3),ZESPEC,ZETAP)
C
         PERM1U=ZCOMP(XTS(1),YTS(1),ZMSPEC,ZMTAP)
         PERM2U=ZCOMP(XTS(2),YTS(2),ZMSPEC,ZMTAP)
         PERM3U=ZCOMP(XTS(3),YTS(3),ZMSPEC,ZMTAP)
         PERM4U=ZCOMP(XTN(1),YTN(1),ZMSPEC,ZMTAP)
         PERM5U=ZCOMP(XTN(3),YTN(3),ZMSPEC,ZMTAP)
         YE(I,1)=  1./PERM2E
         YE(I,2)=     PERM2E-1./PERM2U
         YE(I,3)=-1./(PERM2E*PERM2E)*(PERM3E-PERM1E)/(2.*DELL)
         YE(I,4)=-1./(PERM2E*PERM2E)*(PERM5E-PERM4E)/(2.*DELL)
         YE(I,5)= 0.
         YE(I,6)= 0.
         YE(I,7)= 0.
         YM(I,1)=  1./PERM2U
         YM(I,2)=     PERM2U-1./PERM2E
         YM(I,3)=-1./(PERM2U*PERM2U)*(PERM3U-PERM1U)/(2.*DELL)
         YM(I,4)=-1./(PERM2U*PERM2U)*(PERM5U-PERM4U)/(2.*DELL)
         YM(I,5)= 0.
         YM(I,6)= 0.
         YM(I,7)= 0.
         EMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IETAP))
         HMUL(I)=REAL(ZCOMP(XTS(2),YTS(2),0,IMTAP))
170   CONTINUE
      M = IVCELL
```

```
C
C******BEGIN LOOP AGAIN
C
      GOTO 10
C     GOTO 499
C
C*****FORMATS
C
203   FORMAT(I3,I5,8F10.5)
204   FORMAT(10F10.5)
205   FORMAT(2F10.5,3I3)
C
360   FORMAT(' ABNORMAL SEGMENT: ',I4,5X,2F10.5,13X,2F10.5,
     &         '  ABNORMAL CELL')
400   FORMAT('0 WARNING....',I5,' POINTS HAVE BEEN GENERATED')
C
499   RETURN
      END

C*====================================================================*C
C                                                                      C
      SUBROUTINE CGEFA(A,LDA,N,IPVT,INFO)
C                                                                      C
C*====================================================================*C
C                                                                      C
C NAASA  2.1.043 CGEFA    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
      INTEGER LDA,N,IPVT(1),INFO
      COMPLEX A(LDA,1)
C
C     CGEFA FACTORS A COMPLEX MATRIX BY GAUSSIAN ELIMINATION.
C
C     CGEFA IS USUALLY CALLED BY CGECO, BUT IT CAN BE CALLED
C     DIRECTLY WITH A SAVING IN TIME IF  RCOND  IS NOT NEEDED.
C     (TIME FOR CGECO) = (1 + 9/N)*(TIME FOR CGEFA) .
C
C     ON ENTRY
C
C        A       COMPLEX(LDA, N)
C                THE MATRIX TO BE FACTORED.
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C     ON RETURN
C
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS
C                WHICH WERE USED TO OBTAIN IT.
C                THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE
C                L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER
C                TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.
C
C        IPVT    INTEGER(N)
C                AN INTEGER VECTOR OF PIVOT INDICES.
C
C        INFO    INTEGER
C                = 0  NORMAL VALUE.
C                = K  IF  U(K,K) .EQ. 0.0 .  THIS IS NOT AN ERROR
C                     CONDITION FOR THIS SUBROUTINE, BUT IT DOES
C                     INDICATE THAT CGESL OR CGEDI WILL DIVIDE BY ZERO
C                     IF CALLED.  USE  RCOND  IN CGECO FOR A RELIABLE
C                     INDICATION OF SINGULARITY.
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
C     SUBROUTINES AND FUNCTIONS
C
C     BLAS CAXPY,CSCAL,ICAMAX
C     FORTRAN ABS,AIMAG,CMPLX,REAL
C
C     INTERNAL VARIABLES
C
      COMPLEX T
      INTEGER ICAMAX,J,K,KP1,L,NM1
C
      COMPLEX ZDUM
      REAL CABS1
      CABS1(ZDUM) = ABS(REAL(ZDUM)) + ABS(AIMAG(ZDUM))
C
CCC   Gaussian elimination with partial pivoting
C
      INFO = 0
      NM1 = N - 1
```

```
      IF (NM1 .LT. 1) GO TO 70
      DO 60 K = 1, NM1
         KP1 = K + 1
C
C        FIND L = PIVOT INDEX
C
         L = ICAMAX(N-K+1,A(K,K),1) + K - 1
         IPVT(K) = L
C
CCC      Zero pivot implies this column already triangularized
C
         IF (CABS1(A(L,K)) .EQ. 0.0E0) GO TO 40
C
CCC         Interchange if necessary
C
            IF (L .EQ. K) GO TO 10
               T = A(L,K)
               A(L,K) = A(K,K)
               A(K,K) = T
   10       CONTINUE
C
CCC         Compute multipliers
C
            T = -CMPLX(1.0E0,0.0E0)/A(K,K)
            CALL CSCAL(N-K,T,A(K+1,K),1)
C
CCC         Row elimination with column indexing
C
            DO 30 J = KP1, N
               T = A(L,J)
               IF (L .EQ. K) GO TO 20
                  A(L,J) = A(K,J)
                  A(K,J) = T
   20          CONTINUE
               CALL CAXPY(N-K,T,A(K+1,K),1,A(K+1,J),1)
   30       CONTINUE
         GO TO 50
   40    CONTINUE
            INFO = K
   50    CONTINUE
   60 CONTINUE
   70 CONTINUE
      IPVT(N) = N
      IF (CABS1(A(N,N)) .EQ. 0.0E0) INFO = N
      RETURN
      END
C                                                                      C
C**********************************************************************C
C                                                                      C
      SUBROUTINE CGESL(A,LDA,N,IPVT,B,JOB)
C                                                                      C
C**********************************************************************C
C                                                                      C
C NAASA  2.1.044 CGESL    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
      INTEGER LDA,N,IPVT(1),JOB
      COMPLEX A(LDA,1),B(1)
C
C     CGESL SOLVES THE COMPLEX SYSTEM
C     A * X = B  OR  CTRANS(A) * X = B
C     USING THE FACTORS COMPUTED BY CGECO OR CGEFA.
C
C     ON ENTRY
C
C        A       COMPLEX(LDA, N)
C                THE OUTPUT FROM CGECO OR CGEFA.
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C        IPVT    INTEGER(N)
C                THE PIVOT VECTOR FROM CGECO OR CGEFA.
C
C        B       COMPLEX(N)
C                THE RIGHT HAND SIDE VECTOR.
C
C        JOB     INTEGER
C                = 0         TO SOLVE  A*X = B ,
C                = NONZERO   TO SOLVE  CTRANS(A)*X = B  WHERE
C                            CTRANS(A)  IS THE CONJUGATE TRANSPOSE.
C
C     ON RETURN
C
C        B       THE SOLUTION VECTOR  X .
```

```
C
C     ERROR CONDITION
C
C        A DIVISION BY ZERO WILL OCCUR IF THE INPUT FACTOR CONTAINS A
C        ZERO ON THE DIAGONAL.  TECHNICALLY THIS INDICATES SINGULARITY
C        BUT IT IS OFTEN CAUSED BY IMPROPER ARGUMENTS OR IMPROPER
C        SETTING OF LDA .  IT WILL NOT OCCUR IF THE SUBROUTINES ARE
C        CALLED CORRECTLY AND IF CGECO HAS SET RCOND .GT. 0.0
C        OR CGEFA HAS SET INFO .EQ. 0 .
C
C     TO COMPUTE  INVERSE(A) * C  WHERE  C  IS A MATRIX
C     WITH  P  COLUMNS
C           CALL CGECO(A,LDA,N,IPVT,RCOND,Z)
C           IF (RCOND IS TOO SMALL) GO TO ...
C           DO 10 J = 1, P
C              CALL CGESL(A,LDA,N,IPVT,C(1,J),0)
C        10 CONTINUE
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
C     SUBROUTINES AND FUNCTIONS
C
C     BLAS CAXPY,CDOTC
C     FORTRAN CONJG
C
C     INTERNAL VARIABLES
C
      COMPLEX CDOTC,T
      INTEGER K,KB,L,NM1
C
      NM1 = N - 1
      IF (JOB .NE. 0) GO TO 50
C
C        JOB = 0 , SOLVE  A * X = B
C        FIRST SOLVE  L*Y = B
C
         IF (NM1 .LT. 1) GO TO 30
         DO 20 K = 1, NM1
            L = IPVT(K)
            T = B(L)
            IF (L .EQ. K) GO TO 10
               B(L) = B(K)
               B(K) = T
   10       CONTINUE
            CALL CAXPY(N-K,T,A(K+1,K),1,B(K+1),1)
   20    CONTINUE
   30    CONTINUE
C
C        NOW SOLVE  U*X = Y
C
         DO 40 KB = 1, N
            K = N + 1 - KB
            B(K) = B(K)/A(K,K)
            T = -B(K)
            CALL CAXPY(K-1,T,A(1,K),1,B(1),1)
   40    CONTINUE
      GO TO 100
   50 CONTINUE
C
C        JOB = NONZERO, SOLVE  CTRANS(A) * X = B
C        FIRST SOLVE  CTRANS(U)*Y = B
C
         DO 60 K = 1, N
            T = CDOTC(K-1,A(1,K),1,B(1),1)
            B(K) = (B(K) - T)/CONJG(A(K,K))
   60    CONTINUE
C
C        NOW SOLVE CTRANS(L)*X = Y
C
         IF (NM1 .LT. 1) GO TO 90
         DO 80 KB = 1, NM1
            K = N - KB
            B(K) = B(K) + CDOTC(N-K,A(K+1,K),1,B(K+1),1)
            L = IPVT(K)
            IF (L .EQ. K) GO TO 70
               T = B(L)
               B(L) = B(K)
               B(K) = T
   70       CONTINUE
   80    CONTINUE
   90    CONTINUE
  100 CONTINUE
      RETURN
      END
C                                                                      C
C**********************************************************************C
```

```
C                                                                       C
      SUBROUTINE CAXPY(N,CA,CX,INCX,CY,INCY)
C                                                                       C
C***********************************************************************C
C                                                                       C
C NAASA  1.1.014 CAXPY    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     CONSTANT TIMES A VECTOR PLUS A VECTOR.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX CX(1),CY(1),CA
      INTEGER I,INCX,INCY,IX,IY,N
C
      IF(N.LE.0)RETURN
      IF (ABS(REAL(CA)) + ABS(AIMAG(CA)) .EQ. 0.0 ) RETURN
      IF(INCX.EQ.1.AND.INCY.EQ.1)GOTO 20
C
CCC      Code for unequal increments or equal increments
CCC      Not equal to 1
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        CY(IY) = CY(IY) + CA*CX(IX)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
CCC      Code for both increments equal to 1
C
   20 DO 30 I = 1,N
        CY(I) = CY(I) + CA*CX(I)
   30 CONTINUE
      RETURN
      END
C                                                                       C
C***********************************************************************C
      COMPLEX FUNCTION CDOTC(N,CX,INCX,CY,INCY)
C***********************************************************************C
C                                                                       C
C NAASA  1.1.012 CDOTC    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     FORMS THE DOT PRODUCT OF TWO VECTORS, CONJUGATING THE FIRST
C     VECTOR.
C     JACK DONGARRA, LINPACK,  6/17/77.
C
      COMPLEX CX(1),CY(1),CTEMP
      INTEGER I,INCX,INCY,IX,IY,N
C
      CTEMP = (0.0,0.0)
      CDOTC = (0.0,0.0)
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1.AND.INCY.EQ.1)GOTO 20
C
CCC      Code for unequal increments or equal increments
CCC      Not equal to 1
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        CTEMP = CTEMP + CONJG(CX(IX))*CY(IY)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      CDOTC = CTEMP
      RETURN
C
CCC      Code for both increments equal to 1
C
   20 DO 30 I = 1,N
        CTEMP = CTEMP + CONJG(CX(I))*CY(I)
   30 CONTINUE
      CDOTC = CTEMP
      RETURN
      END
C                                                                       C
C***********************************************************************C
C                                                                       C
      SUBROUTINE  CSCAL(N,CA,CX,INCX)
C                                                                       C
C***********************************************************************C
C                                                                       C
```

```
C NAASA  1.1.019 CSCAL    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     SCALES A VECTOR BY A CONSTANT.
C     JACK DONGARRA, LINPACK,  6/17/77.
C
      COMPLEX CA,CX(1)
      INTEGER I,INCX,N,NINCX
C
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
CCC      Code for increment not equal to 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        CX(I) = CA*CX(I)
   10 CONTINUE
      RETURN
C
CCC      Code for increment equal to 1
C
   20 DO 30 I = 1,N
        CX(I) = CA*CX(I)
   30 CONTINUE
      RETURN
      END
C                                                                      C
C**********************************************************************C
C                                                                      C
      SUBROUTINE  CSSCAL(N,SA,CX,INCX)
C                                                                      C
C**********************************************************************C
C                                                                      C
C NAASA  1.1.018 CSSCAL   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     SCALES A COMPLEX VECTOR BY A REAL CONSTANT.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX CX(1)
      REAL SA
      INTEGER I,INCX,N,NINCX
C
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
CCC      Code for increment not equal to 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        CX(I) = CMPLX(SA*REAL(CX(I)),SA*AIMAG(CX(I)))
   10 CONTINUE
      RETURN
C
CCC      Code for increment equal to 1
C
   20 DO 30 I = 1,N
        CX(I) = CMPLX(SA*REAL(CX(I)),SA*AIMAG(CX(I)))
   30 CONTINUE
      RETURN
      END
C                                                                      C
C**********************************************************************C
      INTEGER FUNCTION ICAMAX(N,CX,INCX)
C**********************************************************************C
C                                                                      C
C NAASA  1.1.021 ICAMAX   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     FINDS THE INDEX OF ELEMENT HAVING MAX. ABSOLUTE VALUE.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX CX(1)
      REAL SMAX
      INTEGER I,INCX,IX,N
      COMPLEX ZDUM
      REAL CABS1
      CABS1(ZDUM) = ABS(REAL(ZDUM)) + ABS(AIMAG(ZDUM))
C
      ICAMAX = 1
      IF(N.LE.1)RETURN
      IF(INCX.EQ.1)GOTO 20
C
CCC      Code for increment not equal to 1
C
      IX = 1
      SMAX = CABS1(CX(1))
      IX = IX + INCX
      DO 10 I = 2,N
```

```
         IF(CABS1(CX(IX)).LE.SMAX) GO TO 5
         ICAMAX = I
         SMAX = CABS1(CX(IX))
    5    IX = IX + INCX
   10 CONTINUE
      RETURN
C
CCC      Code for increment equal to 1
C
   20 SMAX = CABS1(CX(1))
      DO 30 I = 2,N
         IF(CABS1(CX(I)).LE.SMAX) GO TO 30
         ICAMAX = I
         SMAX = CABS1(CX(I))
   30 CONTINUE
      RETURN
      END
C                                                                      C
C**********************************************************************C
      REAL FUNCTION SCASUM(N,CX,INCX)
C**********************************************************************C
C                                                                      C
C NAASA  1.1.010 SCASUM   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
C     TAKES THE SUM OF THE ABSOLUTE VALUES OF A COMPLEX VECTOR AND
C     RETURNS A SINGLE PRECISION RESULT.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX CX(1)
      REAL STEMP
      INTEGER I,INCX,N,NINCX
C
      SCASUM = 0.0E0
      STEMP = 0.0E0
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
CCC      Code for increment not equal to 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        STEMP = STEMP + ABS(REAL(CX(I))) + ABS(AIMAG(CX(I)))
   10 CONTINUE
      SCASUM = STEMP
      RETURN
C
CCC      Code for increment equal to 1
C
   20 DO 30 I = 1,N
        STEMP = STEMP + ABS(REAL(CX(I))) + ABS(AIMAG(CX(I)))
   30 CONTINUE
      SCASUM = STEMP
      RETURN
      END
C**********************************************************************C
C                                                                      C
C     The following subroutines are standard LINPACK routines          C
C     to perform L-U decomposition and back substitution on a          C
C     single precision complex matrix.  See CC-Memo 407 sec 2.1        C
C     for documentation on these routines.                             C
C                                                                      C
C**********************************************************************C
C                                                                      C
      SUBROUTINE CGECO(A,LDA,N,IPVT,RCOND,Z)
C                                                                      C
C**********************************************************************C
C                                                                      C
C NAASA  2.1.042 CGECO    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
C
      INTEGER LDA,N,IPVT(1)
      COMPLEX A(LDA,1),Z(1)
      REAL RCOND
C
C     CGECO FACTORS A COMPLEX MATRIX BY GAUSSIAN ELIMINATION
C     AND ESTIMATES THE CONDITION OF THE MATRIX.
C
C     IF  RCOND  IS NOT NEEDED, CGEFA IS SLIGHTLY FASTER.
C     TO SOLVE  A*X = B , FOLLOW CGECO BY CGESL.
C     TO COMPUTE  INVERSE(A)*C , FOLLOW CGECO BY CGESL.
C     TO COMPUTE  DETERMINANT(A) , FOLLOW CGECO BY CGEDI.
C     TO COMPUTE  INVERSE(A) , FOLLOW CGECO BY CGEDI.
C
C     ON ENTRY
C
C        A       COMPLEX(LDA, N)
C                THE MATRIX TO BE FACTORED.
C
```

```
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C     ON RETURN
C
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS
C                WHICH WERE USED TO OBTAIN IT.
C                THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE
C                L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER
C                TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.
C
C        IPVT    INTEGER(N)
C                AN INTEGER VECTOR OF PIVOT INDICES.
C
C        RCOND   REAL
C                AN ESTIMATE OF THE RECIPROCAL CONDITION OF  A .
C                FOR THE SYSTEM  A*X = B , RELATIVE PERTURBATIONS
C                IN  A  AND  B  OF SIZE  EPSILON  MAY CAUSE
C                RELATIVE PERTURBATIONS IN  X  OF SIZE  EPSILON/RCOND .
C                IF  RCOND  IS SO SMALL THAT THE LOGICAL EXPRESSION
C                          1.0 + RCOND .EQ. 1.0
C                IS TRUE, THEN  A  MAY BE SINGULAR TO WORKING
C                PRECISION.  IN PARTICULAR,  RCOND  IS ZERO  IF
C                EXACT SINGULARITY IS DETECTED OR THE ESTIMATE
C                UNDERFLOWS.
C
C        Z       COMPLEX(N)
C                A WORK VECTOR WHOSE CONTENTS ARE USUALLY UNIMPORTANT.
C                IF  A  IS CLOSE TO A SINGULAR MATRIX, THEN  Z  IS
C                AN APPROXIMATE NULL VECTOR IN THE SENSE THAT
C                NORM(A*Z) = RCOND*NORM(A)*NORM(Z) .
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
C     SUBROUTINES AND FUNCTIONS
C
C     LINPACK CGEFA
C     BLAS CAXPY,CDOTC,CSSCAL,SCASUM
C     FORTRAN ABS,AIMAG,AMAX1,CMPLX,CONJG,REAL
C
C     INTERNAL VARIABLES
C
      COMPLEX CDOTC,EK,T,WK,WKM
      REAL ANORM,S,SCASUM,SM,YNORM
      INTEGER INFO,J,K,KB,KP1,L
C
      COMPLEX ZDUM,ZDUM1,ZDUM2,CSIGN1
      REAL CABS1
      CABS1(ZDUM) = ABS(REAL(ZDUM)) + ABS(AIMAG(ZDUM))
      CSIGN1(ZDUM1,ZDUM2) = CABS1(ZDUM1)*(ZDUM2/CABS1(ZDUM2))
C
CCC   Compute 1-NORM of A
C
      ANORM = 0.0E0
      DO 10 J = 1, N
      ANORM = AMAX1(ANORM,SCASUM(N,A(1,J),1))
   10 CONTINUE
C
CCC   Factor
C
      CALL CGEFA(A,LDA,N,IPVT,INFO)
C
C     RCOND = 1/(NORM(A)*(ESTIMATE OF NORM(INVERSE(A)))) .
C     ESTIMATE = NORM(Z)/NORM(Y) WHERE  A*Z = Y  AND  CTRANS(A)*Y = E .
C     CTRANS(A)  IS THE CONJUGATE TRANSPOSE OF A .
C     THE COMPONENTS OF  E  ARE CHOSEN TO CAUSE MAXIMUM LOCAL
C     GROWTH IN THE ELEMENTS OF W  WHERE  CTRANS(U)*W = E .
C     THE VECTORS ARE FREQUENTLY RESCALED TO AVOID OVERFLOW.
C
C     SOLVE CTRANS(U)*W = E
C
      EK = CMPLX(1.0E0,0.0E0)
      DO 20 J = 1, N
         Z(J) = CMPLX(0.0E0,0.0E0)
   20 CONTINUE
      DO 100 K = 1, N
         IF (CABS1(Z(K)) .NE. 0.0E0) EK = CSIGN1(EK,-Z(K))
         IF (CABS1(EK-Z(K)) .LE. CABS1(A(K,K))) GO TO 30
            S = CABS1(A(K,K))/CABS1(EK-Z(K))
            CALL CSSCAL(N,S,Z,1)
            EK = CMPLX(S,0.0E0)*EK
   30    CONTINUE
         WK = EK - Z(K)
```

```
         WKM = -EK - Z(K)
         S = CABS1(WK)
         SM = CABS1(WKM)
         IF (CABS1(A(K,K)) .EQ. 0.0E0) GO TO 40
            WK = WK/CONJG(A(K,K))
            WKM = WKM/CONJG(A(K,K))
         GO TO 50
   40    CONTINUE
            WK = CMPLX(1.0E0,0.0E0)
            WKM = CMPLX(1.0E0,0.0E0)
   50    CONTINUE
         KP1 = K + 1
         IF (KP1 .GT. N) GO TO 90
            DO 60 J = KP1, N
               SM = SM + CABS1(Z(J)+WKM*CONJG(A(K,J)))
               Z(J) = Z(J) + WK*CONJG(A(K,J))
               S = S + CABS1(Z(J))
   60       CONTINUE
            IF (S .GE. SM) GO TO 80
               T = WKM - WK
               WK = WKM
               DO 70 J = KP1, N
                  Z(J) = Z(J) + T*CONJG(A(K,J))
   70          CONTINUE
   80       CONTINUE
   90    CONTINUE
         Z(K) = WK
  100 CONTINUE
      S = 1.0E0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
C
CCC   Solve CTRANS(L)*Y = V
C
      DO 120 KB = 1, N
         K = N + 1 - KB
         IF (K .LT. N) Z(K) = Z(K) + CDOTC(N-K,A(K+1,K),1,Z(K+1),1)
         IF (CABS1(Z(K)) .LE. 1.0E0) GO TO 110
            S = 1.0E0/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
  110    CONTINUE
         L = IPVT(K)
         T = Z(L)
         Z(L) = Z(K)
         Z(K) = T
  120 CONTINUE
      S = 1.0E0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
C
      YNORM = 1.0E0
C
CCC   Solve L*V = Y
C
      DO 140 K = 1, N
         L = IPVT(K)
         T = Z(L)
         Z(L) = Z(K)
         Z(K) = T
         IF (K .LT. N) CALL CAXPY(N-K,T,A(K+1,K),1,Z(K+1),1)
         IF (CABS1(Z(K)) .LE. 1.0E0) GO TO 130
            S = 1.0E0/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
            YNORM = S*YNORM
  130    CONTINUE
  140 CONTINUE
      S = 1.0E0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
      YNORM = S*YNORM
C
CCC   Solve U*Z = V
C
      DO 160 KB = 1, N
         K = N + 1 - KB
         IF (CABS1(Z(K)) .LE. CABS1(A(K,K))) GO TO 150
            S = CABS1(A(K,K))/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
            YNORM = S*YNORM
  150    CONTINUE
         IF (CABS1(A(K,K)) .NE. 0.0E0) Z(K) = Z(K)/A(K,K)
         IF (CABS1(A(K,K)) .EQ. 0.0E0) Z(K) = CMPLX(1.0E0,0.0E0)
         T = -Z(K)
         CALL CAXPY(K-1,T,A(1,K),1,Z(1),1)
  160 CONTINUE
C     MAKE ZNORM = 1.0
      S = 1.0E0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
      YNORM = S*YNORM
C
```

```
      IF (ANORM .NE. 0.0E0) RCOND = YNORM/ANORM
      IF (ANORM .EQ. 0.0E0) RCOND = 0.0E0
      RETURN
      END
c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c
c       Title:  Jacobian
c       Type:  Subroutine
c       Purpose:  Compute the Jacobian Coefficients for this Element
c       Input:  Element points x1,x2,x3,x4,x5,x6,x7,x8
c                              y1,y2,y3,y4,y5,y6,y7,y8
c       Output: Real Jacobian Coefficients C[4x4] ==> [u^(n-1),v^(n-1)]
c
c       By:  Leo C. Kempel
c            Radiation Laboritory  University of Michigan
c       Last Revision:  14 Mar. 1990
c
c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c
      Subroutine Jacobian(bx,by,C)
c
c Declare Variables
c
        real bx(8),by(8),C(4,4)
        real x1,x2,x3,x4,x5,x6,x7,x8
        real y1,y2,y3,y4,y5,y6,y7,y8
c
c Load the sample Points
c
        x1 = bx(1)
        x2 = bx(2)
        x3 = bx(3)
        x4 = bx(4)
        x5 = bx(5)
        x6 = bx(6)
        x7 = bx(7)
        x8 = bx(8)

        y1 = by(1)
        y2 = by(2)
        y3 = by(3)
        y4 = by(4)
        y5 = by(5)
        y6 = by(6)
        y7 = by(7)
        y8 = by(8)
c
c Compute The Coefficients
c
        C(1,1)=  - 0.25*X6*Y5 + 0.25*X8*Y5 + 0.25*X5*Y6 - 0.25*X7*Y6 +
     &          0.25*X6*Y7 -0.25*X8*Y7 - 0.25*X5*Y8 + 0.25*X7*Y8


        C(1,2) = 0.125*X5*Y1 + 0.25*X6*Y1 - 0.125*X7*Y1 - 0.25*X8*Y1 -
     &           0.125*X5*Y2 +
     &  0.25*X6*Y2 + 0.125*X7*Y2 -0.25*X8*Y2+0.125*X5*Y3+0.25*X6*Y3-
     &  0.125*X7*Y3 - 0.25*X8*Y3 -0.125*X5*Y4+0.25*X6*Y4+0.125*X7*Y4-
     &  0.25*X8*Y4-0.125*X1*Y5+0.125*X2*Y5-0.125*X3*Y5+0.125*X4*Y5-
     &  0.25*X1*Y6-0.25*X2*Y6-0.25*X3*Y6-0.25*X4*Y6+X8*Y6 +
     &  0.125*X1*Y7-0.125*X2*Y7+0.125*X3*Y7-0.125*X4*Y7+0.25*X1*Y8+
     &  0.25*X2*Y8+0.25*X3*Y8+0.25*X4*Y8-X6*Y8

        C(1,3) = -0.25*X2*Y1 - 0.25*X4*Y1 -0.125*X5*Y1+0.25*X6*Y1+
     &           0.125*X7*Y1 +
     &  0.25*X8*Y1+0.25*X1*Y2+0.25*X3*Y2+0.125*X5*Y2-0.25*X6*Y2-
     &  0.125*X7*Y2-0.25*X8*Y2-0.25*X2*Y3-0.25*X4*Y3+0.125*X5*Y3+
     &  0.25*X6*Y3-0.125*X7*Y3+0.25*X8*Y3+0.25*X1*Y4+0.25*X3*Y4-
     &  0.125*X5*Y4-0.25*X6*Y4+0.125*X7*Y4-0.25*X8*Y4+0.125*X1*Y5-
     &  0.125*X2*Y5-0.125*X3*Y5+0.125*X4*Y5+0.25*X6*Y5-0.25*X8*Y5-
     &  0.25*X1*Y6+0.25*X2*Y6-0.25*X3*Y6+0.25*X4*Y6-0.25*X5*Y6+
     &  0.25*X7*Y6-0.125*X1*Y7+0.125*X2*Y7+0.125*X3*Y7-0.125*X4*Y7-
     &  0.25*X6*Y7+0.25*X8*Y7-0.25*X1*Y8+0.25*X2*Y8-0.25*X3*Y8+
     &  0.25*X4*Y8 + 0.25*X5*Y8 - 0.25*X7*Y8

        C(1,4) = 0.25*X2*Y1 + 0.25*X3*Y1-0.5*X6*Y1-0.25*X1*Y2 -
     &           0.25*X4*Y2 + 0.5*X8*Y2 -
     & 0.25*X1*Y3-0.25*X4*Y3+0.5*X8*Y3+0.25*X2*Y4+0.25*X3*Y4 -
     & 0.5*X6*Y4 +
     & 0.5*X1*Y6+0.5*X4*Y6-X8*Y6-0.5*X2*Y8-0.5*X3*Y8 + X6*Y8

c****67**1*********2*********3 RufCode 4********5*********6*********7*****

       C(2,1) = 0.25*X5*Y1 + 0.125*X6*Y1 - 0.25*X7*Y1 - 0.125*X8*Y1 +
     &          0.25*X5*Y2 -
     &  0.125*X6*Y2-0.25*X7*Y2+0.125*X8*Y2+0.25*X5*Y3+0.125*X6*Y3-
     &  0.25*X7*Y3-0.125*X8*Y3+0.25*X5*Y4-0.125*X6*Y4-0.25*X7*Y4+
     &  0.125*X8*Y4-0.25*X1*Y5-0.25*X2*Y5-0.25*X3*Y5-0.25*X4*Y5 +
```

```
     &  X7*Y5-0.125*X1*Y6+0.125*X2*Y6-0.125*X3*Y6+0.125*X4*Y6+
     &  0.25*X1*Y7+0.25*X2*Y7+0.25*X3*Y7+0.25*X4*Y7- X5*Y7 +
     &  0.125*X1*Y8-0.125*X2*Y8+0.125*X3*Y8-0.125*X4*Y8

        C(2,2) = -0.75*X5*Y1 + 0.25*X6*Y1 - 0.25*X7*Y1 + 0.75*X8*Y1 -
     &           0.75*X5*Y2 +
     &  0.75*X6*Y2-0.25*X7*Y2+0.25*X8*Y2-0.25*X5*Y3+0.75*X6*Y3 -
     &  0.75*X7*Y3+0.25*X8*Y3-0.25*X5*Y4+0.25*X6*Y4-0.75*X7*Y4+
     &  0.75*X8*Y4+0.75*X1*Y5+0.75*X2*Y5+0.25*X3*Y5+0.25*X4*Y5-X6*Y5-
     &  X8*Y5-0.25*X1*Y6-0.75*X2*Y6-0.75*X3*Y6-0.25*X4*Y6+X5*Y6+
     &  X7*Y6+0.25*X1*Y7+0.25*X2*Y7+0.75*X3*Y7+0.75*X4*Y7-X6*Y7 -
     &  X8*Y7-0.75*X1*Y8-0.25*X2*Y8-0.25*X3*Y8-0.75*X4*Y8+X5*Y8+
     &  X7*Y8

        C(2,3) = 0.375*X3*Y1 + 0.625*X4*Y1 + 0.5*X5*Y1 - 0.375*X6*Y1 -
     &           0.5*X7*Y1 -
     &  0.625*X8*Y1+0.625*X3*Y2+0.375*X4*Y2+0.5*X5*Y2-0.625*X6*Y2 -
     &  0.5*X7*Y2-0.375*X8*Y2-0.375*X1*Y3-0.625*X2*Y3+0.5*X5*Y3+
     &  0.625*X6*Y3-0.5*X7*Y3+0.375*X8*Y3-0.625*X1*Y4-0.375*X2*Y4+
     &  0.5*X5*Y4+0.375*X6*Y4-0.5*X7*Y4+0.625*X8*Y4-0.5*X1*Y5-
     &  0.5*X2*Y5 -
     &  0.5*X3*Y5-0.5*X4*Y5+X6*Y5+X8*Y5+0.375*X1*Y6+0.625*X2*Y6 -
     &  0.625*X3*Y6-0.375*X4*Y6-X5*Y6+X7*Y6+0.5*X1*Y7+0.5*X2*Y7 +
     &  0.5*X3*Y7+0.5*X4*Y7-X6*Y7-X8*Y7+0.625*X1*Y8+0.375*X2*Y8 -
     &  0.375*X3*Y8-0.625*X4*Y8 - X5*Y8 + X7*Y8

        C(2,4) = 0.0

c****67**1*********2*********3 RufCode 4********5*********6*********7*****

        C(3,1) = 0.25*X2*Y1 + 0.25*X4*Y1 - 0.25*X5*Y1 - 0.125*X6*Y1 -
     &           0.25*X7*Y1 +
     &  0.125*X8*Y1-0.25*X1*Y2-0.25*X3*Y2+0.25*X5*Y2-0.125*X6*Y2 +
     &  0.25*X7*Y2+0.125*X8*Y2+0.25*X2*Y3+0.25*X4*Y3-0.25*X5*Y3 +
     &  0.125*X6*Y3-0.25*X7*Y3-0.125*X8*Y3-0.25*X1*Y4-0.25*X3*Y4 +
     &  0.25*X5*Y4+0.125*X6*Y4+0.25*X7*Y4-0.125*X8*Y4+0.25*X1*Y5 -
     &  0.25*X2*Y5+0.25*X3*Y5-0.25*X4*Y5+0.25*X6*Y5-0.25*X8*Y5 +
     &  0.125*X1*Y6+0.125*X2*Y6-0.125*X3*Y6-0.125*X4*Y6-0.25*X5*Y6 +
     &  0.25*X7*Y6+0.25*X1*Y7-0.25*X2*Y7+0.25*X3*Y7-0.25*X4*Y7 -
     &  0.25*X6*Y7+0.25*X8*Y7-0.125*X1*Y8-0.125*X2*Y8+0.125*X3*Y8 +
     &  0.125*X4*Y8 + 0.25*X5*Y8 - 0.25*X7*Y8

        C(3,2) = -0.625*X2*Y1 - 0.375*X3*Y1 + 0.625*X5*Y1 + 0.5*X6*Y1 +
     &           0.375*X7*Y1 -
     &  0.5*X8*Y1+0.625*X1*Y2+0.375*X4*Y2-0.625*X5*Y2+0.5*X6*Y2 -
     &  0.375*X7*Y2-0.5*X8*Y2+0.375*X1*Y3+0.625*X4*Y3-0.375*X5*Y3 +
     &  0.5*X6*Y3-0.625*X7*Y3-0.5*X8*Y3-0.375*X2*Y4-0.625*X3*Y4 +
     &  0.375*X5*Y4+0.5*X6*Y4+0.625*X7*Y4-0.5*X8*Y4-0.625*X1*Y5 +
     &  0.625*X2*Y5+0.375*X3*Y5-0.375*X4*Y5-X6*Y5+X8*Y5-0.5*X1*Y6 -
     &  0.5*X2*Y6-0.5*X3*Y6-0.5*X4*Y6+X5*Y6+X7*Y6-0.375*X1*Y7 +
     &  0.375*X2*Y7+0.625*X3*Y7-0.625*X4*Y7-X6*Y7+X8*Y7+0.5*X1*Y8+
     &  0.5*X2*Y8 + 0.5*X3*Y8 + 0.5*X4*Y8 - X5*Y8 - X7*Y8

        C(3,3) = 0.375*X2*Y1 - 0.375*X4*Y1 - 0.375*X5*Y1 - 0.375*X6*Y1 +
     &           0.375*X7*Y1 +
     &  0.375*X8*Y1-0.375*X1*Y2+0.375*X3*Y2+0.375*X5*Y2-0.375*X6*Y2-
     &  0.375*X7*Y2+0.375*X8*Y2-0.375*X2*Y3+0.375*X4*Y3+0.375*X5*Y3 +
     &  0.375*X6*Y3-0.375*X7*Y3-0.375*X8*Y3+0.375*X1*Y4-0.375*X3*Y4 -
     &  0.375*X5*Y4+0.375*X6*Y4+0.375*X7*Y4-0.375*X8*Y4+0.375*X1*Y5 -
     &  0.375*X2*Y5-0.375*X3*Y5+0.375*X4*Y5+0.75*X6*Y5-0.75*X8*Y5 +
     &  0.375*X1*Y6+0.375*X2*Y6-0.375*X3*Y6-0.375*X4*Y6-0.75*X5*Y6 +
     &  0.75*X7*Y6-0.375*X1*Y7+0.375*X2*Y7+0.375*X3*Y7-0.375*X4*Y7 -
     &  0.75*X6*Y7+0.75*X8*Y7-0.375*X1*Y8-0.375*X2*Y8+0.375*X3*Y8 +
     &  0.375*X4*Y8 + 0.75*X5*Y8 - 0.75*X7*Y8

        C(3,4) = 0.0

c****67**1*********2*********3 RufCode 4********5*********6*********7*****

        C(4,1) = -0.25*X3*Y1 - 0.25*X4*Y1 + 0.5*X7*Y1 - 0.25*X3*Y2 -
     &           0.25*X4*Y2 + 0.5*X7*Y2 +
     &  0.25*X1*Y3 + 0.25*X2*Y3 - 0.5*X5*Y3 + 0.25*X1*Y4 + 0.25*X2*Y4 -
     &  0.5*X5*Y4 +
     &  0.5*X3*Y5 + 0.5*X4*Y5 - X7*Y5 - 0.5*X1*Y7 - 0.5*X2*Y7 + X5*Y7

        C(4,2) = 0.0
        C(4,3) = 0.0
        C(4,4) = 0.0

c        print*,'The Jacobian Matrix:'
c        print*,c(1,1),c(1,2),c(1,3),c(1,4)
c        print*,c(2,1),c(2,2),c(2,3),c(2,4)
c        print*,c(3,1),c(3,2),c(3,3),c(3,4)
c        print*,c(4,1),c(4,2),c(4,3),c(4,4)
c        print*,' '
c
c All Done !!!
```

```
c
      End
c****67**1*********2*********3 RufCode 4********5*********6*********7*****


c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c
c       Title:  TwoFunc
c       Type:  Complex Function
c       Purpose:  Compute the functions F1,F3
c                 Auxillary Functions Jac,Xin,Yin,Rho
c       Input:  Element points(bx,by),muep,emloe,ko,DelEx,DelEy,nx,ny,u,v
c       Output: Function Value at a Point in (u,v) space
c
c       By:  Leo C. Kempel
c            Radiation Laboritory  University of Michigan
c       Last Revision:  14 Mar. 1990
c
c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c

      Subroutine TwoStuff(C,bx,by,u,v,Xin,Yin,Rho,Jac)
c
c Declare Variables
c
        integer l,k
        real C(4,4),bx(8),by(8),u,v

        real Rho,Xin,Yin,Jac
c
c Compute The mapped coordinates,Rho,koRho
c
        Xin = bx(1) + bx(2)*v + bx(3)*v**2 + bx(4)*u + bx(5)*u*v +
     &        bx(6)*u*v**2 + bx(7)*u**2 + bx(8)*v*u**2
        Yin = by(1) + by(2)*v + by(3)*v**2 + by(4)*u + by(5)*u*v +
     &        by(6)*u*v**2 + by(7)*u**2 + by(8)*v*u**2

        Rho = Sqrt(Xin**2 + Yin**2)
c
c Compute The Jacobian
c
        Jac = 0.0
        do 20  k = 1,4
           do 10 l = 1,4
              Jac = Jac +  C(l,k) * u**(l-1) * v**(k-1)
   10      continue
   20   continue
        Jac = Abs(Jac)
c
c Right ...
c
      End

c****67**1*********2*********3 RufCode 4********5*********6*********7*****

      Complex Function F1(muep,ko,DelEx,DelEy,Xin,Yin,Rho,Jac)
c
c Declare Variables
c
        real ko
        complex muep,DelEx,DelEy
        complex Ho,H1,cgunk,cgunkx,cgunky

        real Rho,koRho,Xin,Yin,Jac
c
c Compute The Function
c
        koRho = ko * Rho
        F1 = (0.0,0.0)

        CALL HANKZ1(koRho,2,Ho,H1)
        cgunk = cmplx((ko**2)/4.0*Jac,0.0)
        cgunkx = cmplx(ko/4.0*Xin/Rho*Jac)
        cgunky = cmplx(ko/4.0*Yin/Rho*Jac)
        F1 = F1 + (0.0,-1.0)*cgunk*muep*Ho
        F1 = F1 + (0.0,1.0)*cgunkx*DelEx*H1
        F1 = F1 + (0.0,1.0)*cgunky*DelEy*H1

c
c Right ...
c
        Return
      End
c****67**1*********2*********3 RufCode 4********5*********6*********7*****

      Complex Function F3(ko,nx,ny,emloe,Xin,Yin,Rho,Jac)
c
```

```
c Declare Variables
c
        real ko,nx,ny
        complex H1,Ho,cgunkx,cgunky,em1oe

        real Rho,koRho,Xin,Yin,Jac
c
c Compute The Function
c
        koRho = ko * Rho
        F3 = (0.0,0.0)

        CALL HANKZ1(koRho,1,Ho,H1)

        cgunkx = cmplx((ko**2)/4.0*nx*Xin/Rho*Jac,0.0)
        cgunky = cmplx((ko**2)/4.0*ny*Yin/Rho*Jac,0.0)

        F3 = -cgunkx*H1*em1oe
        F3 = F3 - cgunky*H1*em1oe
c
c Right ...
c
        Return
      End
c****67**1*********2*********3 RufwCode 4********5*********6*********7*****

c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c
c       Title:  TwoMap
c       Type:  Subroutine
c       Purpose:  Compute the Volume Map terms for either X(u,v) or Y(u,v)
c       Input:  Element points a(8)
c       Output: Map terms b(8)
c
c       By:  Leo C. Kempel
c            Radiation Laboritory  University of Michigan
c       Last Revision:  14 Mar. 1990
c
c****67**1*********2*********3 RufCode 4********5*********6*********7*****
c
      Subroutine TwoMap(a,b)
c
c Declare Variables
c
        real a(8)
        real b(8)
c
c Compute the Map parameters
c
        b(1) = 0.25*(2.0*(a(5)+a(6)+a(7)+a(8)) - (a(1)+a(2)+a(3)+a(4)))
        b(2) = 0.5 * (a(7) - a(5))
        b(3) = 0.25 * (a(1)+a(2)+a(3)+a(4) - 2.0 *(a(6) + a(8)))
        b(4) = 0.5 * (a(6) - a(8))
        b(5) = 0.25 * (a(1) - a(2) + a(3) - a(4))
        b(6) = 0.25 * (a(2) - a(1) + a(3) - a(4) - 2.0 * (a(6) - a(8)))
        b(7) = 0.25 * (a(1) + a(2) + a(3) + a(4) - 2.0 * (a(5) + a(7)))
        b(8) = 0.25 * (a(3) - a(1) - a(2) + a(4) + 2.0 * (a(5) - a(7)))
c
c All Done
c
      End
c****67**1*********2*********3 RufCode 4********5*********6*********7*****
```