ELECTROMAGNETIC SCATTERING FROM LARGE PLANAR PLATES OF ARBITRARY SHAPE

by

Kasra Barkeshli, Arindam Chatterjee and John L. Volakis

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

November 1989

389604-8-T = RL-2568

ABSTRACT

ELECTROMAGNETIC SCATTERING FROM LARGE PLANAR PLATES OF ARBITRARY SHAPE

by
Kasra Barkeshli , Arindam Chatterjee and John L. Volakis

The problem of electromagnetic scattering from large planar plates of arbitrary shape and resistivity is formulated using an improved conjugate gradient solution method. The method employs the discrete convolution theorem and Fast Fourier transform to achieve higher efficiency and accuracy over previously reported results. The accuracy of the technique is confirmed by comparison with measured data and patterns available based on alternative formulations for a variety of targets.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

LIST OF APPENDICES

Appendix

CHAPTER I

Introduction

The problem of electromagnetic scattering from large conducting and resistive plates is of considerable interest in electromagnetics as they constitute simple but nevertheless important components in man-made structures. Although they have been studied at a wide frequency range, experience with various numerical and asymptotic methods of solution as well as comparison with measured data reveals that there is a serious difficulty in predicting the scattering behavior of plates at grazing incidences where the edge currents and corner diffraction mechanisms are significant. Also, in the case of numerical methods, computer memory requirements play a major role in choosing the efficient method of solution.

This report presents a technique to solve the problem of scattering from large planar plates iteratively using the conjugate gradient (CG) method. The conjugate gradient method is an iterative solution technique and has recently been found useful in electromagnetic applications ([1],[2],[3], [4],[5] and [6]). It is shown that the CG method in conjunction with the Fast Fourier Transform (FFT) is an efficient and in many instances the viable alternative to direct methods where the memory requirement for large targets becomes prohibitive.

The CGFFT method was initially implemented by employing the analytical ex-

pression for the Fourier transform of the free space Green's function to carry out the convolution integrals appearing in the pertinent integral equations [4],[7]. The elements of the dyadic Green's function were subsequently formed by carrying out the differential operations in the transform domain. This approach assumes an infinite domain for the Green's function as the Fourier transform is defined over the whole space $(-\infty, +\infty)$. Thus—as far as the Green's function is concerned—the finiteness of the target's physical extent is not accounted for. As a result, unless a large FFT pad with extended zero elements is used, the method suffers from aliasing errors. This problem is more serious for the TM cases where the surface current density exhibits high spectral content due to edge singularity effects. It is the experience of the authors that a pad of at least three times the size of the target is often needed to get acceptable results for TM cases at oblique and close to grazing incidences. This makes the method inefficient in terms of both computer memory and speed particularly for monostatic computations since the solution must be repeated for each incidence. To alleviate this difficulty an alternative approach in handling the Green's function is taken resulting in improved accuracy and a substantial increase in speed. Here the problem is explicitly formulated in terms of discrete convolution of the current density with a discrete integral function of the Green's dyadic. In this case, the Fourier transform properly accounts for the finite domain of the Green's function, thus, avoiding aliasing.

The piecewise constant expansion functions were used with point-matching although a provision for incorporating piecewise linear or sinusoidal (roof-top) basis functions has also been kept.

An interactive Fortran code was developed based on this study which can handle a wide variety of targets with irregular symmetries. This is done by approximating

the target as a polygon with the coordinates of its corners specified by the user.

This report is divided into two parts. The first part describes the formulation of the problem in terms of both continuous and discrete Green's functions. This is followed by an evaluation of the performance of the two formulations. The second part is a user's manual for the computer program including sample test cases for reference purposes.

CHAPTER II

Formulation

2.1 Integral Equations

Consider a thin inhomogeneous non-magnetic plate of resistivity η illuminated by a plane wave $\mathbf{E}^i$ as shown in Figure 2.1. The integral equation for the unknown surface current density $\mathbf{J}$ is given by

$$\mathbf{E}^i(\mathbf{r}) = \eta(\mathbf{r})\mathbf{J}(\mathbf{r}) + jkZ \int_{s'} \mathbf{J}(\mathbf{r}') \cdot \bar{\Gamma}(\mathbf{r};\mathbf{r}')ds' \tag{2.1}$$

where Z denotes the free space impedance, $k = (2\pi/\lambda)$ is the free space wave number and $\bar{\Gamma}$ denotes the electric dyadic Green's function in unbounded space given by[4]

$$\bar{\Gamma}(\mathbf{r};\mathbf{r}') = \{\bar{\mathbf{I}} + \frac{1}{k^2}\nabla\nabla\}G(\mathbf{r};\mathbf{r}') \tag{2.2}$$

with

$$G(\mathbf{r};\mathbf{r}') = \frac{e^{-jk|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|} \tag{2.3}$$

in which $\mathbf{r}$ and $\mathbf{r}'$ denote the observation and integration points, both on the plate's surface. The explicit form of $\bar{\Gamma}$ is

$$\bar{\Gamma} = \begin{pmatrix} (1 + \frac{1}{k^2}\frac{\partial^2}{\partial x^2}) & \frac{1}{k^2}\frac{\partial^2}{\partial x \partial y} \\ \frac{1}{k^2}\frac{\partial^2}{\partial y \partial x} & (1 + \frac{1}{k^2}\frac{\partial^2}{\partial y^2}) \end{pmatrix} G(\mathbf{r};\mathbf{r}') \tag{2.4}$$

Thus equation (2.1) represents a set of coupled integral equations:

$$E_x^i(x,y) = \eta(x,y)J_x(x,y) - [(1+\frac{1}{k^2}\frac{\partial^2}{\partial x^2})\Pi_x + \frac{1}{k^2}\frac{\partial^2}{\partial x \partial y}\Pi_y] \tag{2.5}$$

$$E_y^i(x,y) = \eta(x,y)J_y(x,y) - [\frac{1}{k^2}\frac{\partial^2}{\partial y \partial x}\Pi_x + (1+\frac{1}{k^2}\frac{\partial^2}{\partial y^2})\Pi_y] \tag{2.6}$$

where $\mathbf{\Pi}$ is the electric vector Hertz potential given by

$$\mathbf{\Pi}(\mathbf{r}) = -jkZ\int_{s'}\mathbf{J}(\mathbf{r}')G(\mathbf{r};\mathbf{r}')ds'. \tag{2.7}$$

The integral in (2.1) is a convolution and can therefore be evaluated by invoking the convolution theorem. Introducing the spectral frequencies f_x and f_y and Fourier transform pair

$$g = \mathcal{F}^{-1}\{\tilde{g}\} \iff \tilde{g}, \tag{2.8}$$

with

$$\tilde{g}(f_x,f_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)e^{-j2\pi(f_x x + f_y y)}dxdy, \tag{2.9}$$

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \tilde{g}(f_x,f_y)e^{j2\pi(f_x x + f_y y)}df_x df_y. \tag{2.10}$$

and employing the convolution theorem, the above system of equations can be formally written as

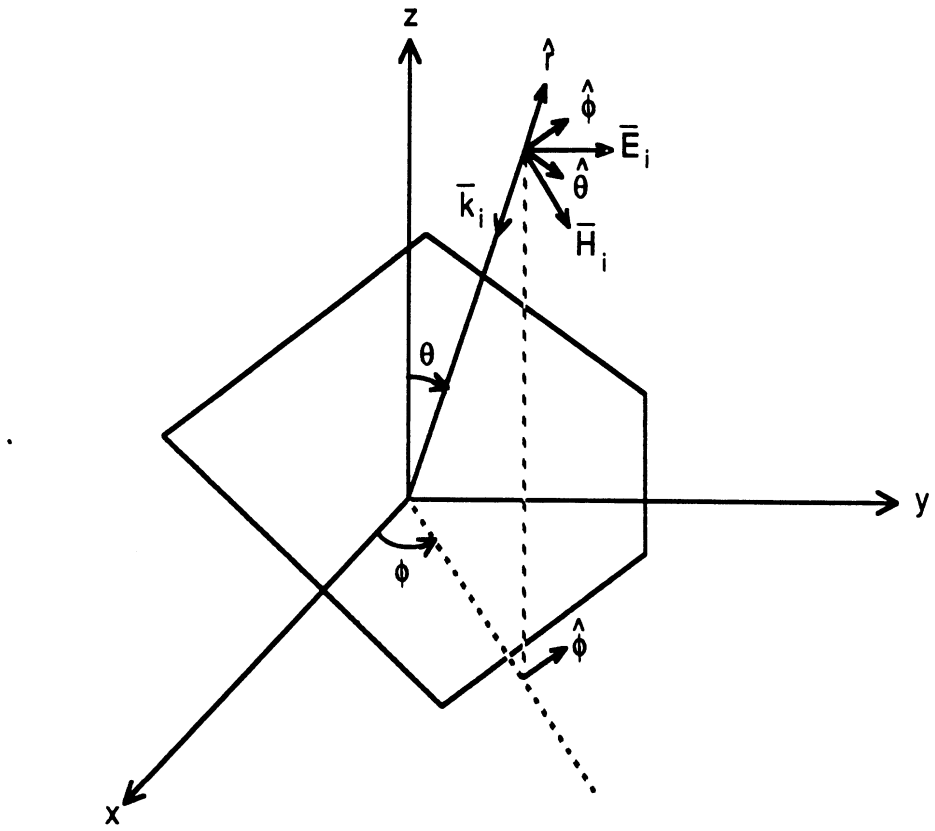
$$\mathbf{E}^i = \eta\mathbf{J} + jkZ\mathcal{F}^{-1}\{\tilde{\bar{\Gamma}}\cdot\tilde{\mathbf{J}}\}. \tag{2.11}$$

The above equation can now be efficiently solved via the conjugate gradient method. The description of the CG method and the algorithm used in this study are available in [1],[7].

2.2 Continuous Convolution

In a traditional implementation of the CGFFT method, the current density is expanded in terms of a subsectional surface basis function $\bar{\Psi}$ as

$$\mathbf{J}(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\mathbf{J}_{mn}\cdot\bar{\Psi}_{mn}(x,y) \tag{2.12}$$



$\bar{E}_i \parallel \hat{\theta}$	$\alpha = 0$	H-pol.
$\bar{E}_i \parallel \hat{\phi}$	$\alpha = \frac{\pi}{2}$	E-pol.

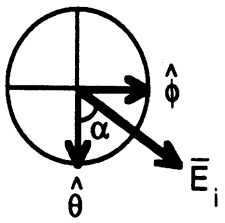


Fig. 2.1. Geometry of a polygonal plate illuminated by a plane wave.

where

$$\bar{\Psi}_{mn}(x,y) = \bar{\Psi}(x - x_m, y - y_n), \tag{2.13}$$

$$\bar{\Psi}(x,y) = \begin{pmatrix} \psi_x(x,y) & 0 \\ 0 & \psi_y(x,y) \end{pmatrix}, \tag{2.14}$$

and ψ_x and ψ_y are the expansion functions in the x and y directions, respectively. The Fourier transform of the current density vector is thus given by (Appendix I)

$$\tilde{\mathbf{J}} = \hat{\mathbf{J}} \cdot \tilde{\bar{\Psi}} \tag{2.15}$$

where $\hat{\mathbf{J}}$ is the discrete Fourier transform of the sampled current. Similarly, the Fourier transform of the dyadic Green's function is expressed as

$$\tilde{\bar{\Gamma}} = \begin{pmatrix} (1 - f_x^2) & (-f_x f_y) \\ (-f_x f_y) & (1 - f_y^2) \end{pmatrix} \tilde{G}(f_x, f_y), \tag{2.16}$$

with

$$G(x,y) = \frac{e^{-jk_0\sqrt{x^2+y^2}}}{4\pi\sqrt{x^2+y^2}} \iff \tilde{G}(f_x,f_y) = \frac{1}{2k_0\sqrt{f_x^2+f_y^2-1}}, \tag{2.17}$$

in which f_x and f_y are the spectral frequencies and use has been made of the following relationship for the partial derivative

$$\frac{\partial g(x)}{\partial x} \iff j2\pi f_x \tilde{g}(f_x) \cdot \tag{2.18}$$

Equations (2.16) and (2.17) constitute analytical expressions for the free space dyadic Green's function. Substituting these into (2.11) and testing the resulting equation at discrete points (point-matching), the following expression is obtained

$$\mathbf{E}_{ij} = \eta_{ij}\mathbf{J}_{ij} + jkZ\mathcal{F}^{-1}\{(\tilde{\bar{\Gamma}} \cdot \tilde{\bar{\Psi}}) \cdot \hat{\mathbf{J}}\} \cdot \tag{2.19}$$

where the subscript ij denotes the value of the quantity at the point (x_i,y_j) on the plate. It should be noted that to perform the Fourier transformation implied

by (2.11), an FFT pad twice the size of the plate must be employed. In general, however, a much larger pad is required when the analytical transform of the Green's function is used. To comply with a standard size FFT pad, we must choose $N' = 2^{\gamma}$, where γ is any integer and $N' > 2N-1 > N_{Nyquist}$ with N reperesenting the number of elements/unknowns. Thus,

$$\gamma \geq \log_2(2N-1) + \rho \tag{2.20}$$

in which ρ is referred to as the order of the FFT pad.

2.3 Discrete Convolution

The use of analytical expressions for the Fourier transform of the Green's function developed in the previous section assumes the entire space as the domain of the Green's function. Since the finiteness of the structure is not accounted for in this approach, aliasing errors are inevitable. For TM scattering, this necessitates the use of prohibitively large FFT pads in order to achieve reasonable accuracy. This approach also suffers from difficulties associated with the numerical handling of the known ring singularity of the transform of G[1]. By employing the discrete Fourier transform of the Green's function defined over an area twice the domain of the scatterer, aliasing is avoided (provided the required sampling criterion is satisfied). This leads to substantially more accurate results and faster convergence. More importantly, the results are insensitive to the FFT pad provided it is at least twice the plate size.

In the new method, the integral equation is put into a discrete convolutional form for direct application of the *discrete* convolution theorem. Substituting for the

[1]The singularity corresponds to the circular ring $f_x^2 + f_y^2 = 1$ in equation (2.17).

current expansion in the integral on the right hand side of (2.1) yields

$$\int_{s'}[\sum_{mn}\mathbf{J}_{mn}\cdot\bar{\mathbf{\Psi}}_{mn}(x,y)]\cdot\bar{\mathbf{\Gamma}}(x,y;x',y')ds' \tag{2.21}$$

which upon interchanging the order of summation and integration, may be written as

$$\sum_{mn}\mathbf{J}_{mn}\cdot\int_{s'}\bar{\mathbf{\Psi}}_{mn}(x',y')\cdot\bar{\mathbf{\Gamma}}(x,y;x',y')ds'\cdot \tag{2.22}$$

Introducing (2.22) into (2.1) and applying point-matching yields the system of equations

$$\mathbf{E}^{\mathbf{i}}{}_{ij}=\eta_{ij}\mathbf{J}_{ij}+jk_0Z_0\sum_{mn}\bar{\mathbf{\Xi}}_{ij}\cdot\mathbf{J}_{mn}\cdot \tag{2.23}$$

The dyadic function $\bar{\mathbf{\Xi}}$

$$\bar{\mathbf{\Xi}}=\begin{pmatrix}\xi^{xx} & \xi^{xy}\\ \xi^{yx} & \xi^{yy}\end{pmatrix} \tag{2.24}$$

is the new discrete kernel whose elements are given by

$$\begin{aligned}
\xi^{xx}_{ij} &= (1+\frac{1}{k^2}\frac{\partial^2}{\partial y^2})\int_{\sigma_{mn}}G(x,y;x',y')\psi_x(x'-x_m,y'-y_n)ds'\Big|_{(x,y)=(x_i,y_j)}\\
\xi^{xy}_{ij} &= \frac{1}{k^2}\frac{\partial^2}{\partial x\partial y}\int_{\sigma_{mn}}G(x,y;x',y')\psi_y(x'-x_m,y'-y_n)ds'\Big|_{(x,y)=(x_i,y_j)}\\
\xi^{yx}_{ij} &= \frac{1}{k^2}\frac{\partial^2}{\partial x\partial y}\int_{\sigma_{mn}}G(x,y;x',y')\psi_x(x'-x_m,y'-y_n)ds'\Big|_{(x,y)=(x_i,y_j)}\\
\xi^{yy}_{ij} &= (1+\frac{1}{k^2}\frac{\partial^2}{\partial x^2})\int_{\sigma_{mn}}G(x,y;x',y')\psi_y(x'-x_m,y'-y_n)ds'\Big|_{(x,y)=(x_i,y_j)}
\end{aligned} \tag{2.25}$$

where σ_{mn} is the incremental surface element corresponding to the mnth cell on the plate $(x_m-\frac{\Delta x}{2}<x<x_m+\frac{\Delta x}{2}\ ,\ y_n-\frac{\Delta y}{2}<y<y_n+\frac{\Delta y}{2})$. Obviously, the convolutional nature of the operation is preserved once the above functions are evaluated at the appropriate field points. Applying the discrete convolution theorem in (2.23) now yields

$$\mathbf{E}_{ij}=\eta_{ij}\mathbf{J}_{ij}+jk_0Z_0\mathcal{DFT}^{-1}\{\hat{\bar{\mathbf{\Xi}}}\cdot\hat{\mathbf{J}}\}\cdot \tag{2.26}$$

In the above, $\hat{\bar{\Xi}}$ denotes the discrete Fourier transform of $\bar{\Xi}$ and its accurate evaluation is a key to the success of the formulation. To calculate these elements, it is noted that in the discrete sense, the partial derivatives may be carried out by finite difference formulae. In particular, using a 3-point difference formula, we obtain the relation (see Appendix I)

$$\frac{\partial}{\partial x}g(x) \iff j2\pi f_x Sinc(\pi f_x \Delta x)\tilde{g}(f_x) \tag{2.27}$$

which may be considered as the discrete counterpart to (2.18). Using (2.27) $\hat{\bar{\Xi}}$ may be expressed as

$$\hat{\bar{\Xi}} = \begin{pmatrix} (1-[2\pi f_x Sinc(\pi f_x \Delta x)]^2) & -(2\pi)^2 f_x f_y Sinc(\pi f_x \Delta x) Sinc(\pi f_y \Delta y) \\ -(2\pi)^2 f_y f_x Sinc(\pi f_y \Delta y) Sinc(\pi f_x \Delta x) & (1-[2\pi f_y Sinc(\pi f_y \Delta y)]^2) \end{pmatrix} \hat{\xi}, \tag{2.28}$$

where $\hat{\xi}$ is the discrete Fourier transform of the sequence

$$\begin{aligned} \xi_{mn} &= \int_{\sigma_{mn}} \frac{e^{-jk\sqrt{x^2+y^2}}}{4\pi\sqrt{x^2+y^2}} ds \\ &= \int_{x_m-\frac{\Delta x}{2}}^{x_m+\frac{\Delta x}{2}} \int_{y_n-\frac{\Delta y}{2}}^{y_n+\frac{\Delta y}{2}} \frac{e^{-jk\sqrt{x^2+y^2}}}{4\pi\sqrt{x^2+y^2}} ds. \end{aligned} \tag{2.29}$$

Note that in (2.27) and (2.28),

$$Sinc(x) = sin(x)/x \tag{2.30}$$

It is noted that ξ has an integrable singularity when $y_m = y_n = 0$, which corresponds to the self-cell interaction. This term can be evaluated analytically using various approximations as stated below.

Approximate integration: An approximate integration of the self-cell term yields[8]

$$\begin{aligned} \xi_{00} \simeq \frac{1}{\pi} & \{\Delta x \ln|\tan(\frac{\pi}{4}+\frac{1}{2}\tan^{-1}\frac{\Delta y}{\Delta x})| \\ & +\Delta y \ln|\tan(\frac{\pi}{4}+\frac{1}{2}\tan^{-1}\frac{\Delta x}{\Delta y})| - jk\frac{\Delta x \Delta y}{2}\}. \end{aligned} \tag{2.31}$$

Taylor series expansion: expanding the integrand in a four-term Taylor series expansion

$$\frac{e^{-jkR}}{R} \simeq \frac{1}{R}\left(1 - jkR + \frac{(jkr)^2}{2} - \frac{(jkR)^3}{6}\right). \tag{2.32}$$

The integral can then be expressed as

$$\xi_{00} \simeq \frac{1}{4\pi}(I_1 - jkI_2 - \frac{k^2}{2}I_3 + j\frac{k^3}{6}I_4)\Big|_{x=-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}}\Big|_{y=-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \tag{2.33}$$

where

$$\begin{aligned} I_1 &= \int \frac{1}{R}ds = x\ln(y+R) + y\ln(x+R) \\ I_2 &= \int ds = \Delta x \Delta y \\ I_3 &= \int Rds = \frac{xyR}{3} + \frac{x^3}{6}\ln(y+R) + \frac{y^3}{6}\ln(x+R) \\ I_4 &= \int R^2 ds = \frac{xyR^2}{3}. \end{aligned}$$

Circular disk approximation: Using a circular disk approximation

$$\begin{aligned} \xi_{00} &\simeq \int_0^{2\pi}\int_0^{r_0} \frac{e^{-jkr}}{4\pi r} r dr d\phi \\ &= \frac{r_0}{2}e^{-jkr_0/2}\mathrm{Sinc}(\frac{kr_0}{2}) \\ r_0 &= \sqrt{\Delta x \Delta y/\pi}. \end{aligned} \tag{2.34}$$

Figure 2.2 shows a comparison of these for the square cells ($\Delta x = \Delta y = \Delta$) of different sizes. As seen, they all give values that are essentially indistinguishable for $\Delta < 0.1\lambda$.

The remaining terms ξ_{mn} are evaluated numerically using Guassian quadrature integration. The solution of (2.26) via the CG method will be referred to in the report as the CGDFT method.

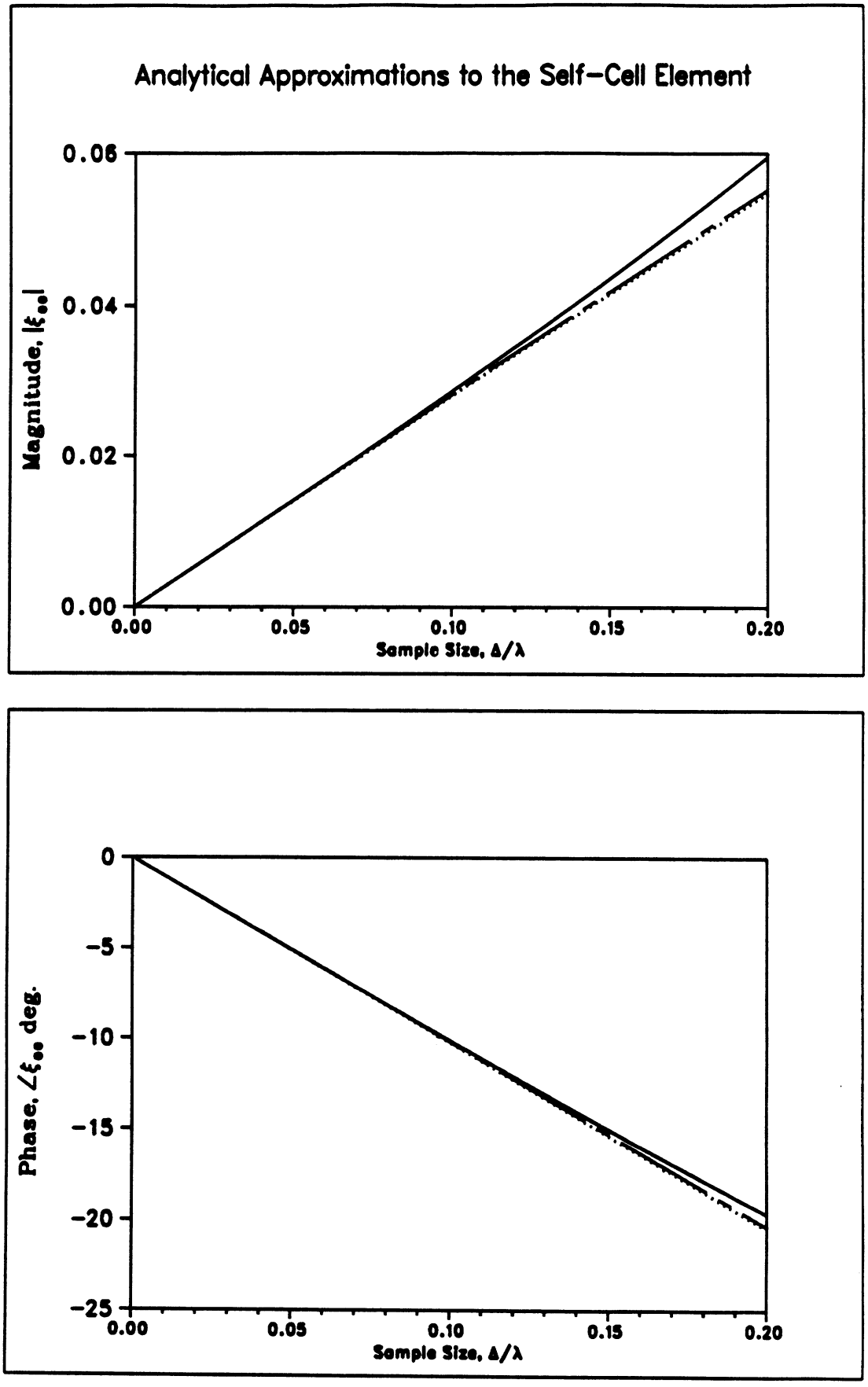


Fig.2.2 **Evaluation of the self-cell element using approximate integration(—), four-term Taylor series expansion(•••), and circular disk approximation(– –).**

CHAPTER III

Results

The above formulations were employed to compute the scattering by a number of plates. In this chapter, a sample of numerical results some of which are compared with available measured data or results based on alternative formulations are given.

3.1 Comparison with measured data

Figure 3.1 shows the radar cross section for a rectangular conducting plate illuminated by E and H polarized plane waves. The edge-on behavior of a plate of constant width ($b = 2\lambda$) and varying length ($2\lambda < a < 2.5\lambda$) is given in figure 3.2 with the electric field aligned with the shorter side. The results obtained are compared with measurement data reported by Hey and Senior[9]. It should be noted that the CGFFT method employing the continuous transform of the Green's function was found inadequate for an accurate prediction of the edge-on scattering behavior[4]. This is clearly demonstrated in Figure 3.3 where we compare the RCS of a square plate (of side 2λ) as computed by the CGDFT method with $\rho = 1$. It is found to be in good agreement with the corresponding measured data.

Since plates of arbitrary shape may be modeled as polygons, circular and triangular plates can also be treated just as easily. Figure 3.4 gives the radar cross

section of a circular plate of radius a with $ka = 5$. These results agree well with the Sommerfeld-Macdonald approximation for close-to-normal incidence angles. However at oblique incidences, this approximation breaks down. Table 3.1 provides a comparison with measured data[10] of the edge-on backscatter as computed with the CGDFT as a further testament of the method's accuracy at close-to-normal incidences. Also, in figure 3.5 backscatter RCS data for an equilateral triangular plate of side 2λ are compared with the corresponding measured data. The agreement is clearly good considering a possible small inaccuracy in the measurements due to alignment difficulties.

The scattering characteristics of geometrically complex targets may also be simulated by approximating the target by a polygon of n sides. This is illustrated in Figure 3.6 where the plate has been modeled as a polygon of 8 corners. The near-grazing (conical) azimuthal scattering pattern of this target is shown in Figure 3.7.

Resistive plates are considered next. In practice conducting surfaces are replaced with resistive cards for cross section reduction purposes. By defining η as a function of position, different resistivity tapers may be treated. As an example, the resistivity can be expressed by a nonlinear function

$$\eta(x,y) = \eta_c + (\eta_e - \eta_c)\left[\left(\frac{|x-x_0|}{x_l/2}\right)^{\tau_x} + \left(\frac{|y-y_0|}{y_l/2}\right)^{\tau_y}\right] \tag{3.1}$$

where η_c and η_e may be considered as the resistivities at the center and the edges of a rectangular plate, respectively and τ_x and τ_y denote the tapers in the corresponding directions. Figure 3.8 shows the effect of uniform and non-uniform (parabolic) resistive tapers on the monostatic cross section of a $2\lambda \times 2\lambda$ plate. The parabolic resistivity profile is shown in figure 3.9.

3.2 Efficiency

It was mentioned in the introduction that applying the traditional approach (using the analytical expression for the transform of the Green's function) requires a large FFT pad to reduce aliasing effects and characterize the weakly singular behavior of $\tilde{G}$ in the vicinity of the ring $f_x^2 + f_y^2 = 1$. Figure 3.3 illustrates the convergence of the far zone scattered field for a square plate as the size of the FFT pad is progressively increased. Also shown is the improved result using the method presented here. It is noted that a pad of at least 3 times the number of unknowns is needed to obtain acceptable results with the traditional method and although the general behavior of the backscattering cross section approaches that of the correct result, the convergence to the measured value is not clear near grazing incidence even with higher order pads. On the other hand, the new method gives a reasonable prediction using a FFT pad of order 1.

In an earlier work [11], the vectorizable nature of the CGFFT algorithm was explored by identifying the major processes involved in a given iteration. Since in the CG algorithm a considerable amount of computation time is spent in operations which exhibit no data recurrence at a particular point in the iteration, these operations may be vectorized to increase the speed of calculations. This is particularly the case for the FFT which is a highly vectorizable algorithm and plays a major role in the speed and efficiency of the optimized code. In this study both scalar and optimized FFT routines (available on the IBM 3090's ESSL library) were used. Sample timing information are given in Tables 3.2 and 3.3. When the data from both of these tables are combined, one observes an overall speedup of more than 60 times over the scalar implementation using the continuous transform of the Green's

function. It should be noted that a vectorization of 97% is observed when employing the vector FFT routine.

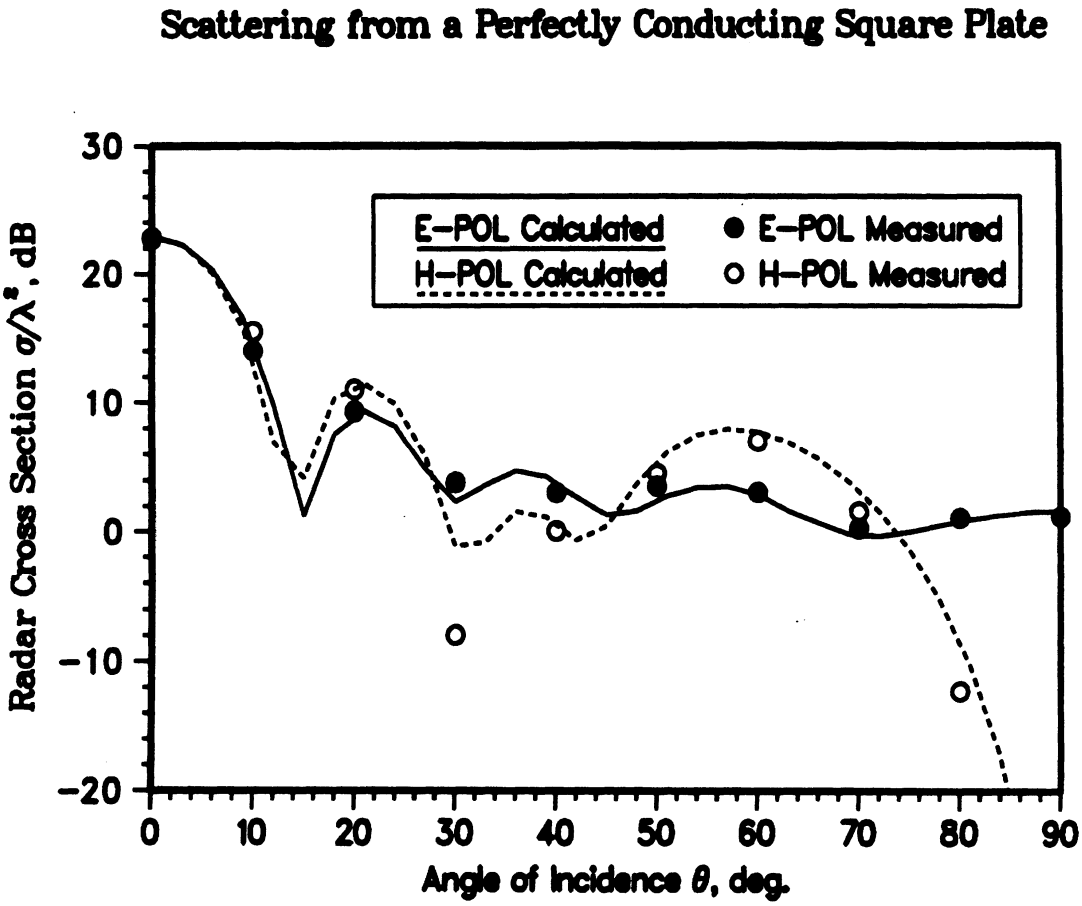


Figure 3.1: Backscattering pattern of a $2\lambda \times 2\lambda$ square plate

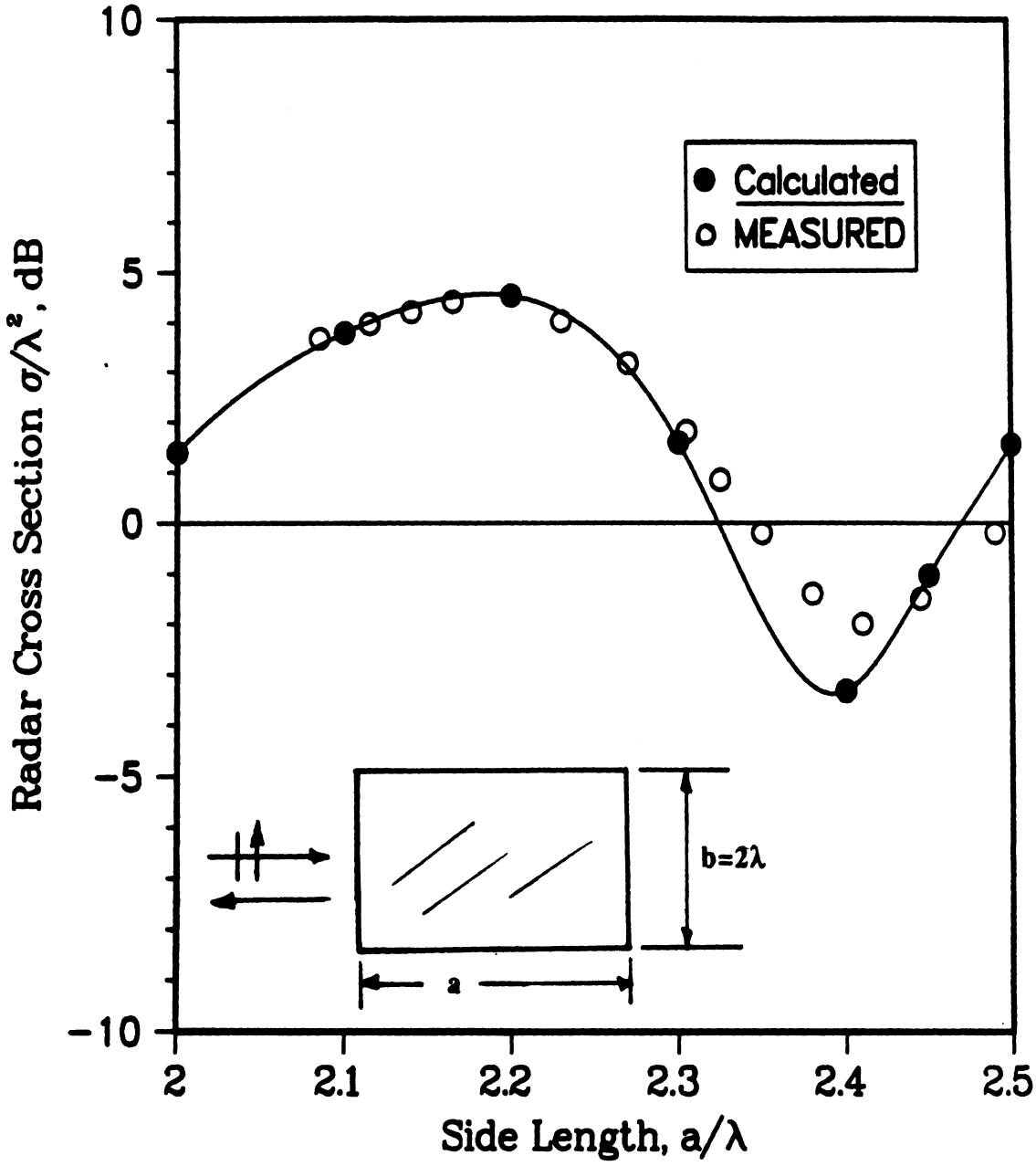


Figure 3.2: E-polarization edge-on RCS for a rectangular plate

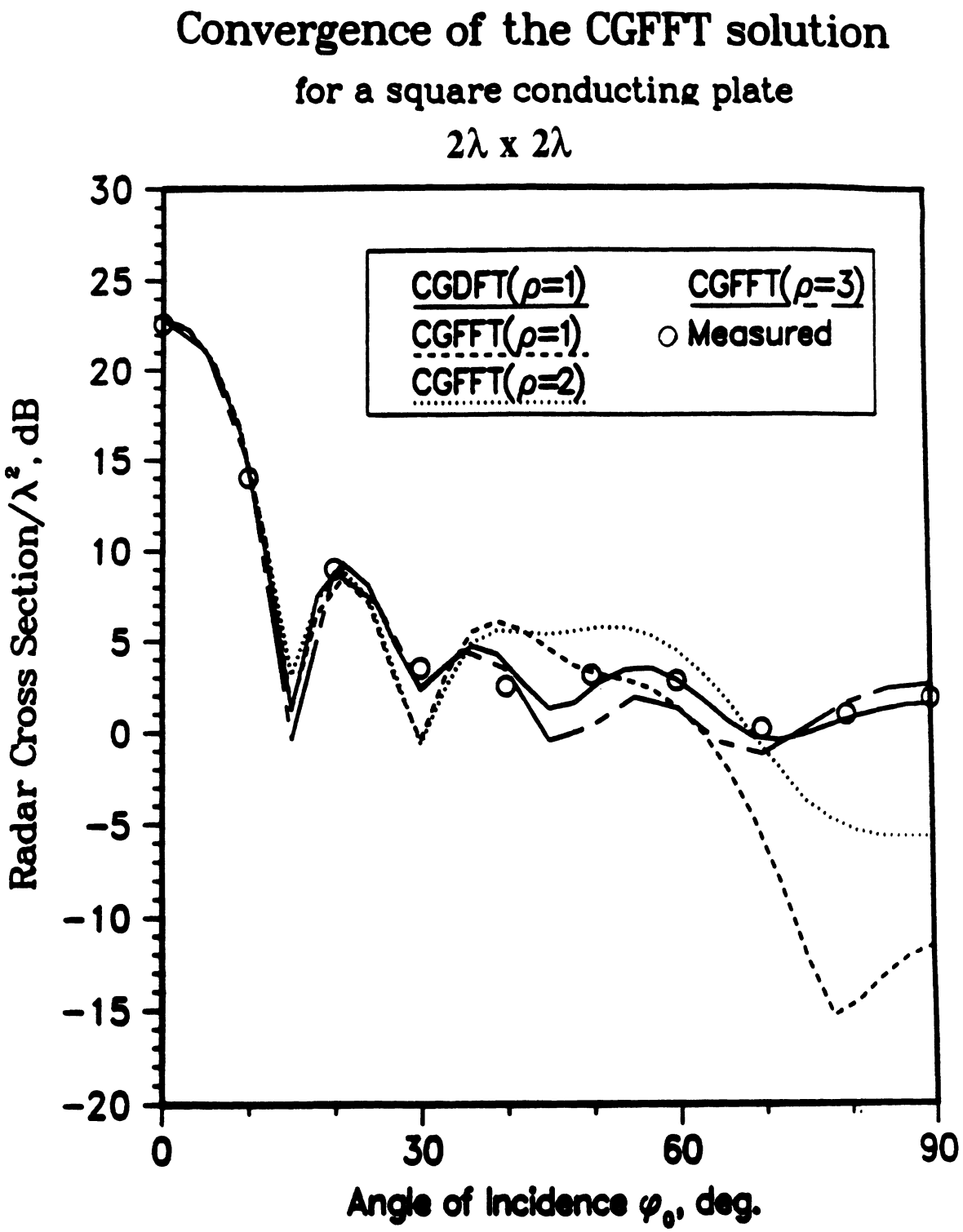
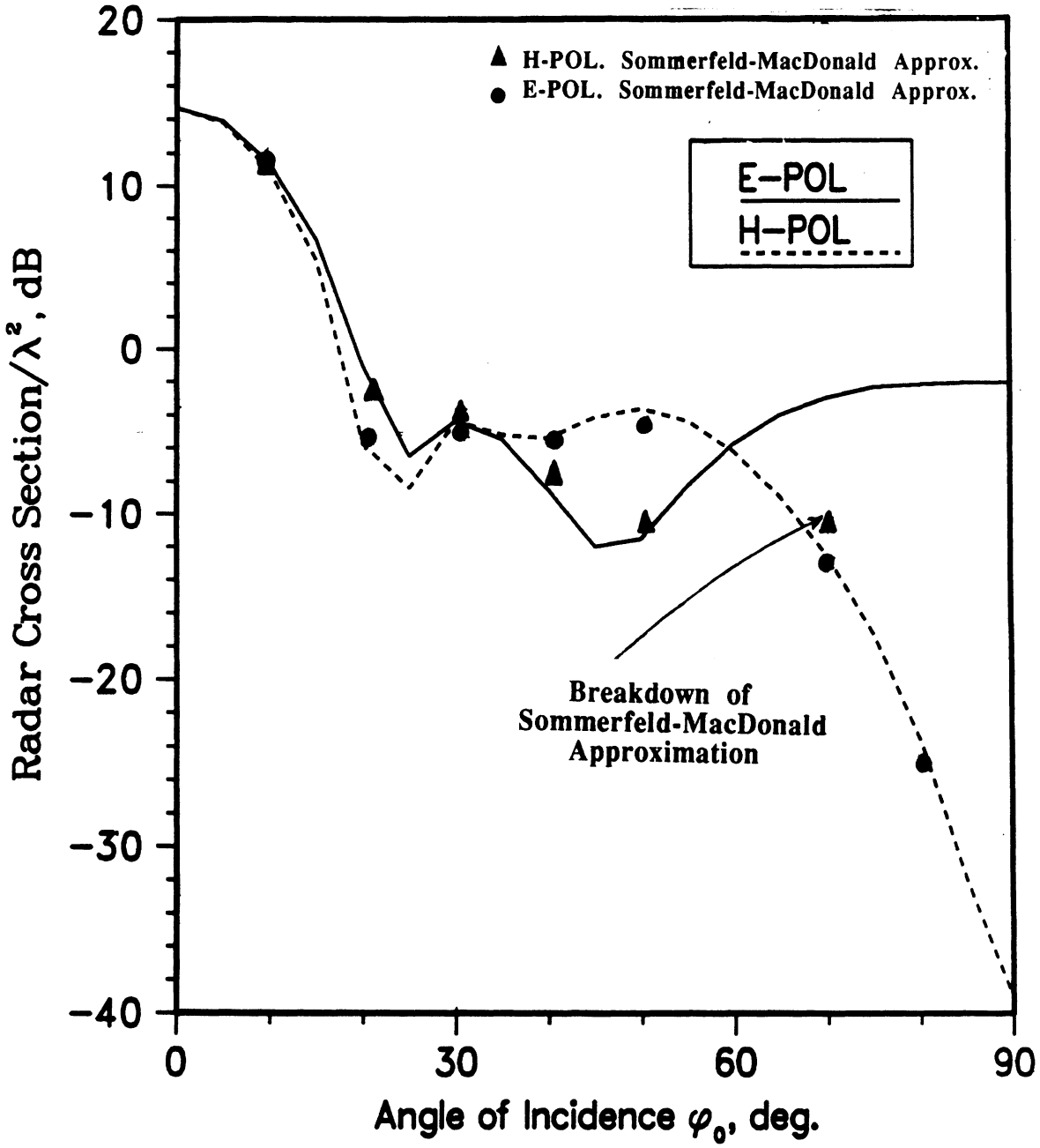


Figure 3.3: Convergence of the CGFFT solution for a square conducting plate

Backscattering pattern of a circular plate (ka=5)



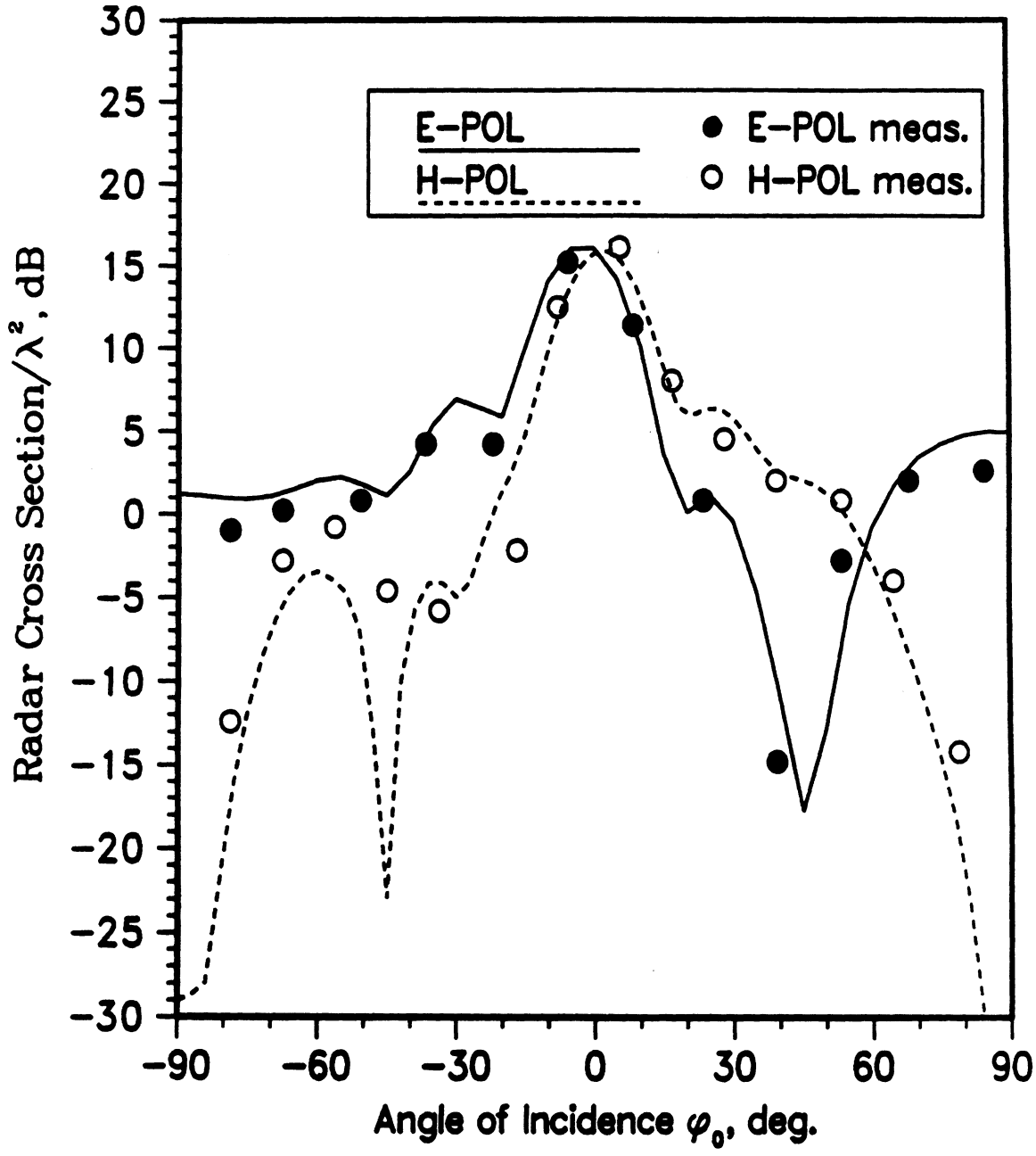


Figure 3.5: Backscatter pattern for an equilateral triangular plate of side 2λ

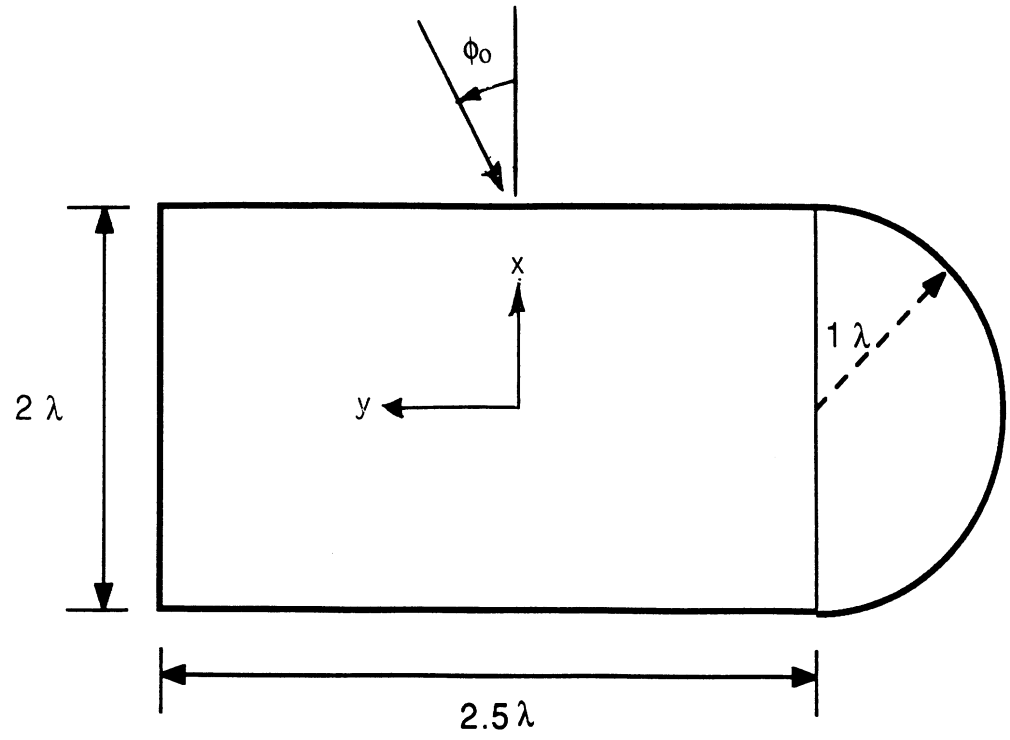


Figure 3.6: Target geometry of the test plate

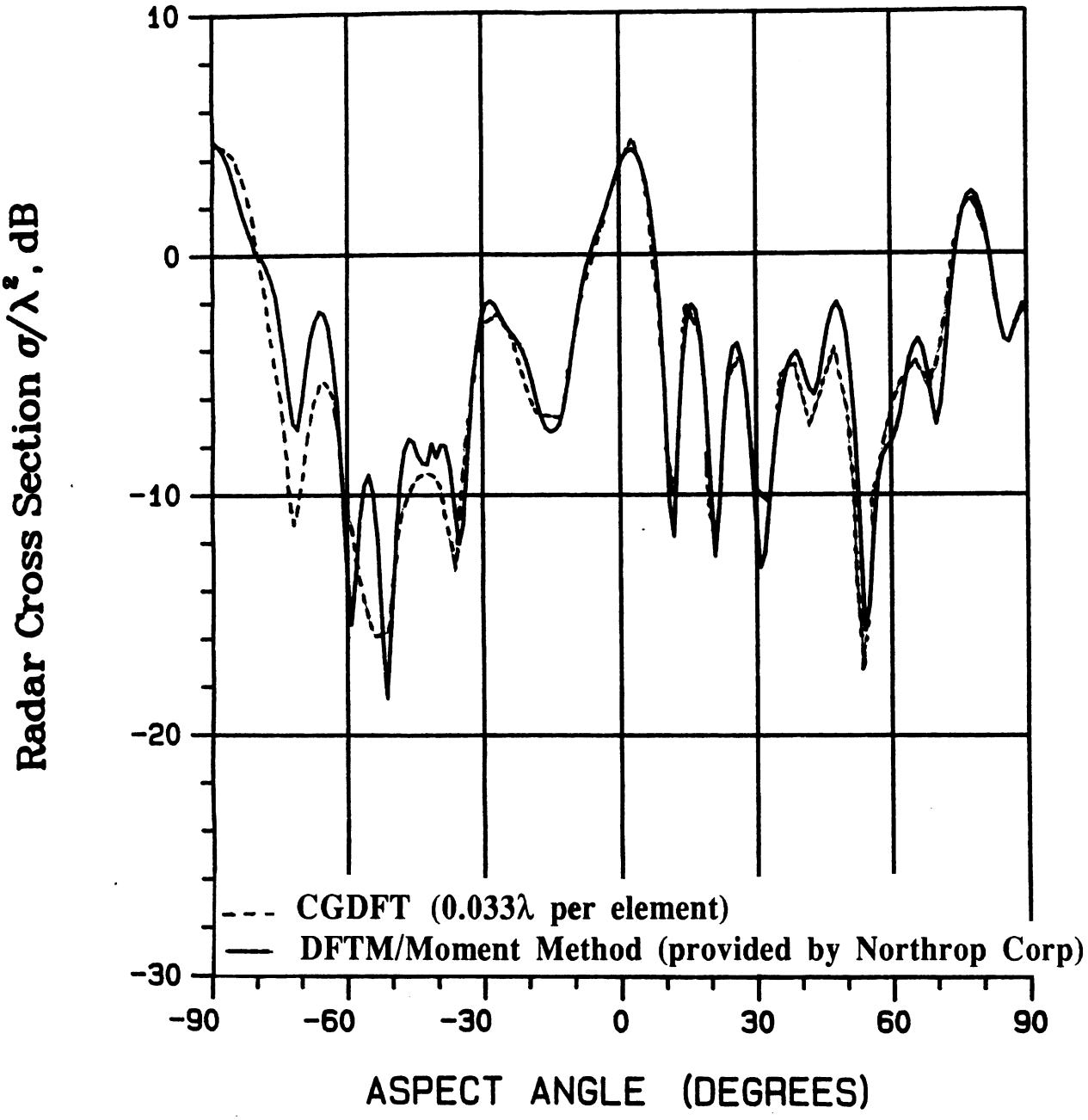


Figure 3.7: E-Polarization Conical ($\theta_0 = 80^o$) backscattering pattern of the test plate

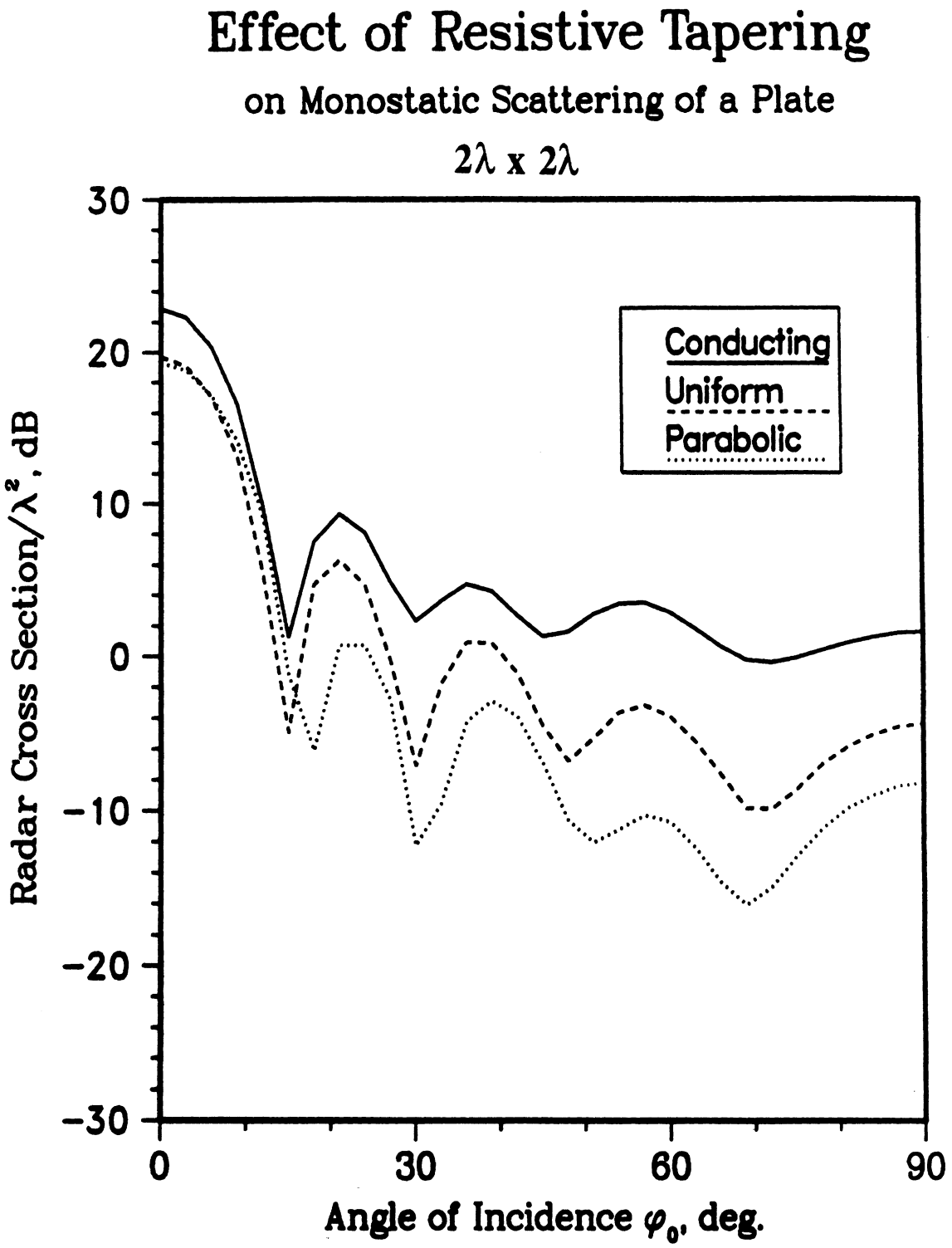


Figure 3.8: Effect of resistive tapering on the monostatic scattering of a 2x2 λ^2 square plate

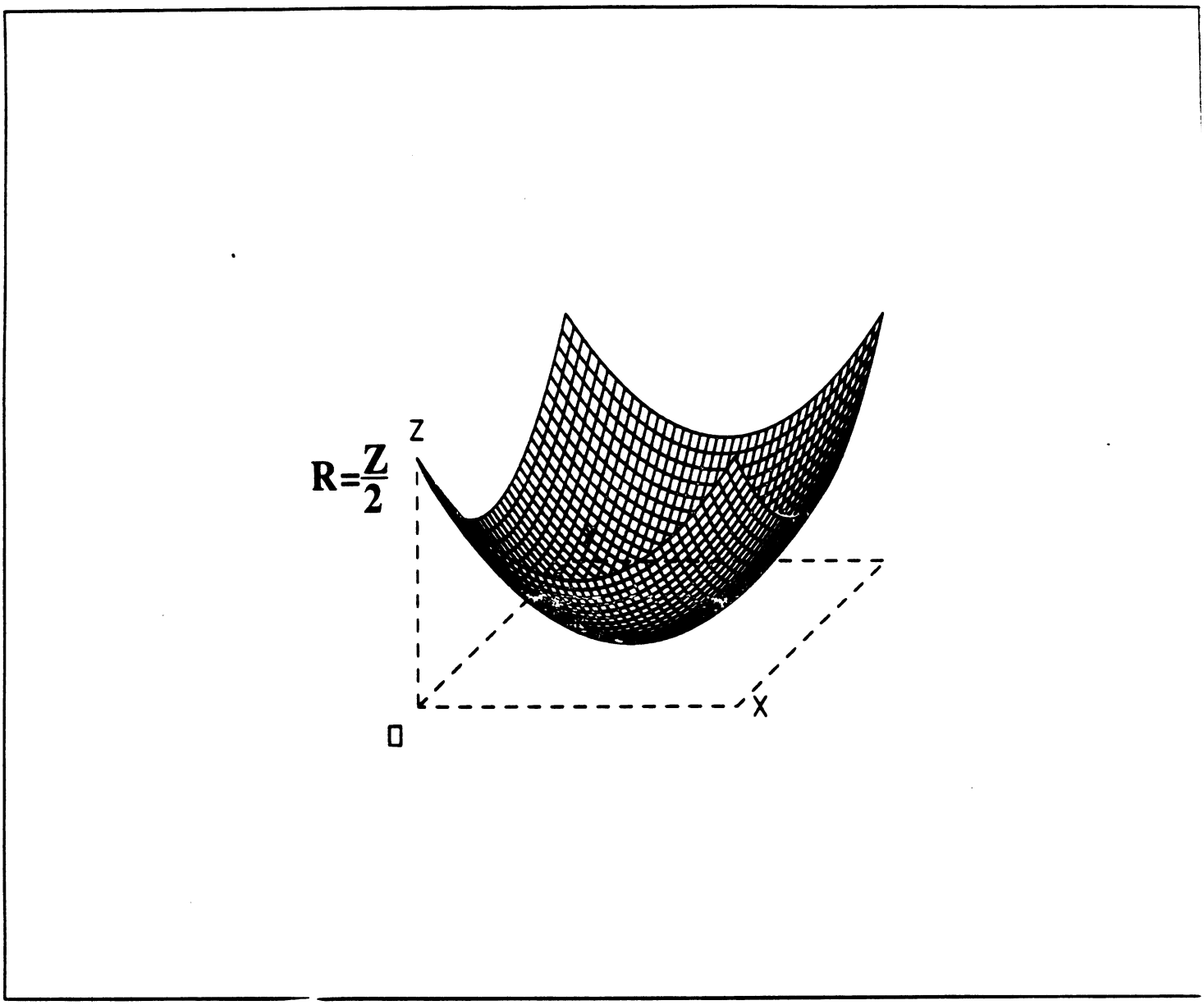


Figure 3.9: 3-Dimensional profile of a parabolic taper (Max $R=\frac{Z}{2}$)

ka	radius(a)	Measured values	Calculated values
1.0	0.1592	-7.696	-8.653
1.5	0.2387	-6.383	-6.274
2.7	0.4297	-5.229	-3.099

Table 3.1: Comparison between measured and calculated values at edge-on incidence (E-pol.) for a circular disk.

Green's function	FFT Pad order	Time (in ms) per iteration	Relative speedup
Discrete	1	65.49	20
Analytical	1	65.62	19.96
	2	291.21	4.5
	3	1310.62	1

Table 3.2: Relative speedup between algorithms employing the Analytical and the Discrete transform of the Green's function.

Green's Function	FFT Pad order	Time(in ms) per iteration		Vector Speedup
		Vectorized	Non-vectorized	
Discrete	1	65.49	216.26	3.30
Analytical	1	65.62	215.75	3.28
	2	291.21	840.34	2.89
	3	1310.62	3597.12	2.75

Table 3.3: Speedup between vectorized and non-vectorized code.

CHAPTER IV

Conclusions and Future Work

The formulation discussed here may be easily extended to cylindrical structures with a minor modification of the dyadic Green's function and the incident field expressions. It should be noted that the problem is simplified for TM case where the longitudinal current component is the only unknown and only a single integral equation is to be solved.

CHAPTER V

Users' Manual

This chapter describes the significance of the various input variables in the CGFFT code and explains the procedure for running the program. It is the intention of the authors that the reader be fully acquainted with the CGFFT code after reading this manual. The program can be run in either interactive or non-interactive mode. The interactive mode is designed to help make the task of the inexperienced user easier. The interactive mode of running the program will be described first.

5.1 Input variables

5.1.1 Target geometry generation

The CGFFT code is capable of performing scattering computations for targets of irregular symmetries (this will be shown later). The user can generate a target of his/her own choice by specifying the co-ordinates of the corners of the target. The program prompts for the target choice on the screen by printing the following message:

```
**SCATTERING FROM THIN CONDUCTING PLATES**


SPECIFY THE TARGET
```

```
1: RECTANGULAR PLATE

2: CIRCULAR DISC

3: TRIANGULAR PLATE

4: POLYGONAL PLATE

5: TEST CASE

ENTER THE TARGET NUMBER:
```

Option 5 includes a few special targets which were generated for testing purposes. It should be emphasized here that these targets could just as well be generated using the 'TARGET # 4: POLYGONAL PLATE' choice. On entering the target number, the program assigns the inputted value to the variable 'NTARG'. Inputting a number less than 1 or greater than 5 evokes the response 'WRONG TARGET! TRY AGAIN...'. If the target number specified was between 1 and 4, the program asks the user about the

```
TARGET CENTER AND ENCLOSURE LIMITS: XO,YO,XL,YL
```

The target center should be the point of symmetry for symmetrical targets and a convenient point (from which it is easier to calculate the corner co-ordinates) near the center point for asymmetrical ones. 'XO,YO' are the input variables for the target center. The enclosure limits are the length and breadth of the target specified by the variables 'XL,YL'. For non-rectangular targets, the enclosure limits are specified by the dimensions of the largest rectangle that would just fit the geometry. The next message printed by the program is

```
SAMPLE DENSITIES:UPLX,UPLY
```

The number of samples per wavelength desired by the user can thus be specified. If the sampling is low, aliasing errors occur whereas oversampling results in high

memory requirement. The optimum number of samples per wavelength can thus only be decided by trial and error but the authors have found that 20-25 samples per wavelength (for large targets of sides 2 wavelengths long) give reliable results. The variables 'UPLX,UPLY' specify the sample densities. The target geometry specification is now complete except for the order of the FFT pad.

`FFT PAD OF ORDER:`

The discrete Fourier transforms are carried out by a radix 2 FFT routine. The sampling interval and the size of the FFT pads are chosen so that the Nyquist criterion in both the spatial and frequency domains as well as the requirements of linearity in the convolutions are satisfied. Thus, the period N' of the array to be transformed is chosen so that

$$N' = 2^{\gamma} : N' > N_{Nyquist}, N' > 2 \times N - 1 \tag{5.1}$$

where N is the number of unknowns and γ is an integer. In practice γ is chosen according to the rule

$$\gamma \geq \log_2(2N - 1) + \rho \tag{5.2}$$

where ρ is an integer setting the *order* of the FFT pad dimension to ensure adequate frequency sampling in the spectral domain when performing the inverse transform operation. The array elements beyond the scatterer's physical range are set to zero.

For computations using the discrete Green's Function, the pad should be of order 1 whereas for calculations with the analytical Green's Function, the pad order is limited only by the size of the memory and time considerations (the authors have found that a pad order exceeding 3 is too time-consuming without yielding a significant improvement in the results).

For target numbers 1, 3 and 4, the program asks for the coordinates of the corner-points:

```
ENTER THE COORDINATES OF EACH CORNER SEQUENTIALLY:
```

The only thing to remember here is that the co-ordinates of the points should be entered such that the points are arranged in either clockwise or counter-clockwise fashion. For target number 2 (a circle), the program asks the user for the radius of the circular target.

For option number 5 (test cases), the geometry is automatically generated since the parameters are specified within the code.

```
GEOMETRY DISPLAY? 1)YES 2)NO
```

In order to make sure that the geometry generated is correct, it is recommended that the user check the geometry before plunging into the CGFFT calculations. Entering '1' in response to the above message, the geometry appears in the form of asterisks in a grid of dots. The entire target is actually modelled as a matrix where the asterisks indicate the target points('1') and the dots signify zeroes.

The user may continue with the calculations or modify the geometry or exit from the program at this point depending on the response to the message:

```
1) CONTINUE   2) MODIFY GEOMETRY   3) EXIT
```

5.1.2 Resistivity taper specification

The plate's resistivity may be specified externally or internally by setting the variable 'TPR' to a real value (TPR≥ -2.0). If a value of -1.0 is selected, the plate is assumed to be perfectly conducting and calculations involving resistivity are

omitted. A non-negative value for TPR denotes the taper in the x and y directions as indicated in equation (3.1).

```
RESISTIVITY TAPER(REAL):
 -2.) RESISTIVITY VECTOR READ EXTERNALLY
 -1.) PERFECTLY CONDUCTING PLATE
  0.) UNIFORM
  1.) LINEAR
  2.) PARABOLIC
```

The variables 'ETC' and 'ETE' read in the resistivity values at the center and the edge of the plate, respectively. For non-symmetrical targets, it is advisable to read in the resistivity values externally.

5.1.3 Scattering pattern computations

This section enables the user to specify the range of the θ and ϕ angles, the polarization angle ψ, the type of scattering calculations(monostatic or bistatic), the basis function, the Green's function (Analytical or Discrete) and the tolerance desired.

```
TETA & PHI
```

The program asks the user the initial θ and ϕ angles for which the scattering computations are to be done. The code accepts both positive and negative values for θ and ϕ.

```
POLARIZATION ANGLE : 0.)H [HZ=0] , 90.)E [EZ=0]
```

If H-polarization scattering calculations are desired, 0. should be entered. This sets ψ to 0. For E-polarization computations, ψ should be set to 90.

1)BISTATIC 2)MONOSTATIC(BACKSCATTERING) COMPUTATIONS

For bistatic calculations, 1 is to be inputted. Entering 2 causes the program to carry out monostatic (backscattering) computations.

1)ELEVATION 2)AZIMUTH CUT

In order to find out the elevation pattern, the user should input 1. For azimuthal pattern calculation, 2 should be entered.

INITIAL & FINAL ANGLES AND INCREMENT

The user can substantially decrease the computation time by specifying the initial and final angles so as to exploit the symmetry inherent in the chosen target. A note of caution for the inexperienced user:even though the initial and final angles are real, the input for the angle increment expects an integer.

BASIS: 1)CONVENTIONAL DFT 2)SURFACE PULSE 3)ROOF-TOP

The user can employ either of the three basis functions for scattering computations. However, in this code the surface pulse basis has been only incorporated.

1)DISCRETE OR 2)ANALYTICAL TRANSFORM OF THE DYADIC G

The user can employ either the Discrete or Analytical transform of the Dyadic Green's function. As discussed in Chapter 2, the analytical transform was employed in the original version of the code. It is present as an option but its use is not recommended due to its associated inefficiency.

MAX & MIN NO. OF ITERATIONS:ITMAX,ITMIN & TOLERANCE

The user can specify the maximum and minimum number of iterations. However, it is recommended by the authors that the maximum number of iterations be kept to a reasonable value (say, 100) and the minimum number kept to 2. The tolerance value specified by the authors during running the program was .035 (3.5%). It is left to the user to tailor the values of 'ITMAX,ITMIN & TOLERANCE' according to his needs.

5.2 Output

The information about the number of iterations and the tolerance achieved for each angle is displayed on the screen. In this way, the angles for which convergence was not achieved can be identified. The values for the scattering computations are stored in a data file in such a way that the data can be easily plotted.

5.3 Sample problems

A sample problem is presented in this section to allow the user a better insight into the running of the CGFFT code. In the problem solved here, the backscattering pattern of a polygonal plate is computed using the discrete transform of the dyadic Green's function.

```
**SCATTERING FROM THIN CONDUCTING PLATES**


SPECIFY THE TARGET

          1: RECTANGULAR PLATE

          2: CIRCULAR DISC

          3: TRIANGULAR PLATE

          4: POLYGONAL PLATE
```

5: TEST CASE

```
ENTER THE TARGET NUMBER:
4

TARGET CENTER AND ENCLOSURE LIMITS: XO,YO,XL,YL
0. 0. 2. 2.

SAMPLE DENSITIES:UPLX,UPLY
25.5 25.5

FFT PAD OF ORDER:
1

ENTER THE COORDINATES OF EACH CORNER SEQUENTIALLY:
CORNER #1:
-2.232    0.0
CORNER #2:
-0.5     -1.0
CORNER #3:
 0.5     -1.0
CORNER #4:
 1.088   -0.809
CORNER #5:
 1.451   -0.309
```

```
CORNER #6:
 1.451     0.309
CORNER #7:
 1.088     0.809
CORNER #8:
 0.5       1.0
CORNER #9:
-0.5       1.0

GEOMETRY DISPLAY? 1)YES 2)NO
2

1) CONTINUE   2) MODIFY GEOMETRY   3) EXIT
1

TETA & PHI
90. 90.

POLARIZATION ANGLE : 0.)H [HZ=0] , 90.)E [EZ=0]
90.

1)BISTATIC 2)MONOSTATIC(BACKSCATTERING) COMPUTATIONS
2

1)ELEVATION  2)AZIMUTH CUT
```

```
1

INITIAL & FINAL ANGLES AND INCREMENT
90. 270. 3

BASIS: 1)CONVENTIONAL DFT 2)SURFACE PULSE 3)ROOF-TOP
2

1)DISCRETE OR 2)ANALYTICAL TRANSFORM OF THE DYADIC G
1

MAX & MIN NO. OF ITERATIONS:ITMAX,ITMIN & TOLERANCE
150 2 .035
```

In the non-interactive mode of running the above program, the input values are stored in a file and the program reads from it. This mode is faster than the interactive mode.

APPENDICES

APPENDIX A

Spectral Domain Considerations

A.1 Fourier Transform of the Current Density

The Fourier transform of the current density vector is defined as

$$\tilde{\mathbf{J}}(f_x, f_y) = \int_{-\infty}^{\infty} \mathbf{J}(x,y) e^{-j2\pi(f_x x + f_y y)} dxdy \,. \tag{A.1}$$

Substituting for the current expansion (2.12)in the above,

$$\begin{aligned} \tilde{\mathbf{J}} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \sum_{mn} \mathbf{J}_{mn} \cdot \bar{\mathbf{\Psi}}_{mn}(x,y) e^{-j2\pi(f_x x + f_y y)} dxdy \\ &= \sum_{mn} \mathbf{J}_{mn} \cdot \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \bar{\mathbf{\Psi}}_{mn}(x,y) e^{-j2\pi(f_x x + f_y y)} dxdy \end{aligned} \tag{A.2}$$

and noting that

$$\begin{aligned} \tilde{\psi}_{mn} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \psi(x - x_m, y - y_n) e^{-j2\pi(f_x x + f_y y)} dxdy \\ &= \tilde{\psi} e^{-j2\pi(f_x x_m + f_y y_n)}, \end{aligned} \tag{A.3}$$

equation (A.1) reduces to

$$\tilde{\mathbf{J}} = \tilde{\bar{\mathbf{\Psi}}}(f_x, f_y) \cdot \sum_{mn} \mathbf{J}_{mn} e^{-j2\pi(f_x x_m + f_y y_n)} \tag{A.4}$$

with the summation on the right hand side defined as the discrete Fourier transform of $\mathbf{J}$:

$$\hat{\mathbf{J}} = \sum_{mn} \mathbf{J}_{mn} e^{-j2\pi(f_x x_m + f_y y_n)}. \tag{A.5}$$

Thus, the desired relationship between the continuous and discrete Fourier transforms of the current density vector is

$$\tilde{\mathbf{J}} = \tilde{\bar{\Psi}} \cdot \hat{\mathbf{J}}. \tag{A.6}$$

A.2 Discrete Differentiation and its Transform

Numerical differentiation of a discrete function can be carried out by calculating finite differences at a number of points. A 3-point finite difference formula is given by

$$\frac{\partial g}{\partial x}(x_i) = \frac{g(x_i + \frac{\Delta x}{2}) - g(x_i - \frac{\Delta x}{2})}{\Delta x} \tag{A.7}$$

with its transform

$$\frac{\partial g}{\partial x} \Longleftrightarrow \tilde{g}(f_x)\frac{e^{-j2\pi f_x \frac{\Delta x}{2}} - e^{+j2\pi f_x \frac{\Delta x}{2}}}{\Delta x} \tag{A.8}$$

which can be further reduced to

$$\frac{\partial g}{\partial x} \Longleftrightarrow -j2\pi f_x Sinc(\pi f_x \Delta x)\tilde{g}. \tag{A.9}$$

More accurate expressions may be derived by using higher order difference formulae. For the 5-point case,

$$\frac{\partial g}{\partial x}(x_i) = \frac{8g(x_i + \frac{\Delta x}{2}) - g(x_i + \frac{3\Delta x}{2}) + g(x_i - \frac{3\Delta x}{2}) - 8g(x_i - \frac{\Delta x}{2})}{12\Delta x} \tag{A.10}$$

and the corresponding result is

$$\frac{\partial g}{\partial x} \Longleftrightarrow -j2\pi f_x [\frac{2}{3} Sinc(\pi f_x \Delta x) - \frac{1}{4} Sinc(3\pi f_x \Delta x)]\tilde{g}. \tag{A.11}$$

APPENDIX B

Listing

This appendix provides a listing of the computer program PLTCGF developed and used in this study.

```
C    ======================================================================
C                          DESCRIPTION OF THE PROGRAM
C    ======================================================================
C     THIS PROGRAM COMPUTES THE CURRENT DENSITY AND THE RADAR SCATTERING
C     CROSS SECTION OF PLANAR PLATES OF ARBITRARY SHAPES MODELED BY
C     POLYGONS. ALSO, SEVERAL SPECIFIC GEOMETRIES ARE TREATED SEPARATELY
C     INCLUDING RECTANGLES, TRIANGLES AND CIRCLES.
C    ======================================================================
C     KASRA BARKESHLI
C     RADIATION LABORATORY
C     EECS DEPARTMENT
C     UNIVERSITY OF MICHIGAN
C     JULY 1989
C    ======================================================================
C
C     COMPILATION PROCEDURE:
C
C....MTS (WITH VECTORIZATION)
C
C     ***********************************************************************
C     NOTICE: FOR VECTORIZATION ON THE IBM-3090, THE FOLLOWING STATEMENT
C     AND SIMILAR ONES IN THE SUBSEQUENT SUBROUTINES MUST BE UNCOMENTED
C     WITH THE '@' SYMBOL POSITIONED IN COLUMN 1.
C     ***********************************************************************
C@PROCESS DIRECTIVE ('*VDIR')
C
                                PROGRAM PLTCGF
C
C       $ RUN NEW:FORTRANVS SCARDS=PLTCGF PRINT=-LIST SPUNCH=-LOAD
C                           PAR=OPT(3) VECTOR(LEVEL(2),REPORT)
C       $ RUN -LOAD+NAAS:NEW.ESSL+*FORTRANLIB
C
C....APOLLO WORKSTATIONS
C
C       $ ftn pltcgf
C       $ bind pltcgf.bin -b pltcgf.obj
C       $ pltcgf.obj
C
C    ======================================================================
C                          PARAMETER DESCRIPTION
C    ======================================================================
C
C...GEOMETRY
C
C        NTARG.....................TARGET NUMBER
C        X0,Y0.....................TARGET CENTER COORDINATES
C        XL,YL.....................TARGET ENCLOSURE
C        UPLX,UPLY.................NUMBER OF UNKNOWNS PER WAVELENGTH IN
C                                   X AND Y DIRECTIONS
C        MX,MY.....................NUMBER OF UNKNOWNS IN X AND Y DIRECTIONS
C        NX,NY.....................FFT PAD SIZE IN X AND Y DIRECTIONS
C        IPAD......................ORDER OF THE FFT PAD
C        ITAG......................INTEGER ARRAY CONTAINING THE ADDRESSES
C                                   OF THE ACTUAL POINTS ON THE TARGET
C...ELECTRICAL PROPERTY
C
C        ETA.......................RESISTIVITY
C        ITAP......................TAPERING PARAMETER:
C                                   0) UNIFORM RESISTIVITY
C                                   1) LINEAR TAPER
C                                   2) PARABOLIC TAPER
C                                   3) RESISTIVITY VECTOR READ EXTERNALLY
C
C...PATTERN COMPUTATION
C
C        TETA,PHI..................SPHERICAL ANGLES OF INCIDENCE
```

```
C       PSI.....................POLARIZATION ANGLE:
C                               0 )H POLARIZATION HZ=0
C                               90)E POLARIZATION EZ=0
C       ISCAT...................PATTERN COMPUTATION FLAG:
C                               1)BISTATIC
C                               2)MONOSTATIC (BACKSCATTERING)
C       ICUT....................NUMBER OF BISTATIC PATTERN CUTS
C       ANGI,ANGF,INC...........INITIAL AND FINAL ANGLES AND INCREMENT
C
C...GREEN'S FUNCTION
C
C       IG......................METHOD OF GREENS FUNCTION EVALUATION:
C                               1)DISCRETE TRANSFORM
C                               2)ANALYTICAL TRANSFORM
C...EXPANSION FUNCTIONS
C
C       IBASE...................TYPE OF BASIS FUNCTION FOR CURRENT EXPANSION:
C                               1)CONVENTIONAL DFT
C                               2)SURFACE PULSE
C                               3)ROOF-TOP (NOT AVAILABLE)
C
C...ITERATION
C
C       ITMAX,ITMIN,TOL.........MAXIMIM AND MINIMUM NUMBER OF ITERATIONS
C                               AND ITERATION TOLERANCET
C
C...CGFFT VARIABLES
C
C       E.......................EXCITATION (INCIDENCE) FIELD
C       JS......................UNKNOWN CURRENT DENSITY
C       R.......................RESIDUAL VECTOR
C       P.......................SEARCH VECTOR
C       A,AT....................THE INTEGRO-DIFFERENTIAL OPERATOR AND
C                               ITS FOURIER TRANSFORM
C       ZT,W....................SPECTRAL DOMAIN WORKING AREA
C
C...TIMING
C
C       BGTIME,INTIME,ENTIME....BEGIN, INTERMEDIATE AND END TIMES
C
C   ================================================================
        PARAMETER(MXT=128,MYT=128,NXF=256,NYF=256)
        PARAMETER(MT=MXT*MYT,MTI=2*MXT*MYT,NTF=NXF*NYF,NTFI=2*NXF*NYF)
        INTEGER ITAG(MT)
        INTEGER*4 IT(3)
        REAL X(MT),Y(MT),RES(MT)
        COMPLEX GXX(NXF,NYF),GXY(NXF,NYF),GYX(NXF,NYF),GYY(NXF,NYF)
        COMPLEX JS(MTI),E(MTI),A(MTI),R(MTI),P(MTI)
        COMPLEX ZT(NTFI),AT(NTFI),W(NTF)
        COMPLEX XJ
        CHARACTER*2 FILE(2)
        COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
        DATA TPI/6.28318530717959/,Z0/376.991118/,XJ/(0.,1.)/
        DATA FILE/'jx','jy'/
C
C   ----------------------------------------------------------------
C
C...TARGET GEOMETRY SPECIFICATION
C
        WRITE(6,*)
        WRITE(6,*)'**SCATTERING FROM THIN CONDUCTING PLATES**'
1       WRITE(6,*)
        WRITE(6,*)'SPECIFY THE TARGET:'
        WRITE(6,*)
        WRITE(6,*)'1: RECTANGULAR PLATE'
        WRITE(6,*)'2: CIRCULAR DISK'
```

```
        WRITE(6,*)'3: TRIANGULAR PLATE'
        WRITE(6,*)'4: POLYGONAL PLATE'
        WRITE(6,*)'5: TEST TARGETS'
        WRITE(6,*)
        READ(5,*,ERR=1)NTARG
        IF((NTARG.GE.1).AND.(NTARG.LE.4))THEN
          WRITE(6,*)'TARGET CENTER AND ENCLOSURE LIMITS: X0,Y0,XL,YL'
          READ(5,*,ERR=1)X0,Y0,XL,YL
        ELSEIF(NTARG.EQ.5)THEN
          WRITE(6,*)'TEST CASES'
          WRITE(6,*)'6: TARGET # 1'
          WRITE(6,*)'7: TARGET # 2'
          WRITE(6,*)'8: TARGET # 3'
          WRITE(6,*)'9: TARGET # 4'
          READ(5,*,ERR=1)NTARG
        ENDIF
        IF((NTARG.LT.0).OR.(NTARG.GT.9))THEN
          WRITE(6,*)'WRONG TARGET! TRY AGAIN...'
          GO TO 1
        ENDIF
        WRITE(6,*)'SAMPLE DENSITIES:UPLX,UPLY'
        READ(5,*)UPLX,UPLY
        WRITE(6,*)'FFT PAD OF ORDER:'
        READ(5,*)IPAD
        CALL GEOMTR(NTARG,X0,Y0,XL,YL,UPLX,UPLY,IPAD,ITAG,X,Y)
        WRITE(6,*)'1) CONTINUE   2) MODIFY GEOMETRY   3) EXIT'
        READ(5,*)IRES
        IF(IRES.EQ.2)THEN
          GO TO 1
        ELSEIF(IRES.EQ.3)THEN
          GO TO 600
        ENDIF
        IF((MX.GT.MXT).OR.(MY.GT.MYT).OR.(NX.GT.NXF).OR.(NY.GT.NYF))THEN
          WRITE(6,2)
          WRITE(6,3)MX,MY,NX,NY
          WRITE(6,4)MXT,MYT,NXF,NYF
          GO TO 600
        ENDIF
2       FORMAT(/,'**REQUIRED ARRAY DIMENSIONS EXCEED THOSE SPECIFIED.')
3       FORMAT('  REQUIRED  : MXT=',I3,' MYT=',I3,' NXF=',I3,' NYF=',I3)
4       FORMAT('  SPECIFIED :',4(5X,I3),/,'MODIFY DIMENSIONS & RERUN.')
        NT=NX*NY
C
C   ---------------------------------------------------------------------
C
C...RESISTIVITY PARAMETERS
C
        WRITE(6,*)'RESISTIVITY TAPER(REAL):'
        WRITE(6,*)' -2.) RESISTIVITY VECTOR READ EXTERNALLY'
        WRITE(6,*)' -1.) PERFECTLY CONDUCTING PLATE'
        WRITE(6,*)'  0.) UNIFORM'
        WRITE(6,*)'  1.) LINEAR'
        WRITE(6,*)'  2.) PARABOLIC'
        READ(5,*)TPR
        INDX=0
        IF(TPR.EQ.-2.)THEN
C
C...READ THE RESISTIVITY VALUES FROM AN EXTERNAL FILE
C
          OPEN(11,FILE='rest.in')
          DO 2000 J=1,MY
             DO 1000 I=1,MX
                INDX=INDX+1
                READ(11,*)RES(INDX)
1000         CONTINUE
2000       CONTINUE
```

```
          CLOSE(11)
        ELSEIF(TPR.EQ.0.)THEN
C
C...GET THE RESISTIVITY VALUE FOR THE UNIFORM CASE
C
          WRITE(6,*)'ETA='
          READ(5,*)ETA
          DO 4000 J=1,MY
             DO 3000 I=1,MX
                INDX=INDX+1
                RES(INDX)=ETA
3000         CONTINUE
4000       CONTINUE
        ELSEIF(TPR.GT.0.)THEN
C
C...GET THE RESISTIVITY TAPER AND SAVE
C
          OPEN(10,FILE='rest.out')
          WRITE(10,*)'RESISTIVITY TAPER'
          WRITE(10,*)MX,MY
          WRITE(10,*)(I,I=1,MX)
          WRITE(10,*)(J,J=1,MY)
          WRITE(6,*)'ETA(CENTER), ETA(EDGE)'
          READ(5,*)ETC,ETE
          WIDTHX=XL/2.
          WIDTHY=YL/2.
          DO 6000 J=1,MY
             DO 5000 I=1,MX
                INDX=INDX+1
                AC1=ABS(X(INDX)-X0)
                AC2=ABS(Y(INDX)-Y0)
                RES(INDX)=
     +          ETC+(ETE-ETC)*((AC1/WIDTHX)**TPR+(AC2/WIDTHY)**TPR)
                WRITE(10,*)RES(INDX)
5000         CONTINUE
6000       CONTINUE
          CLOSE(10)
        ENDIF
C
C   ---------------------------------------------------------------
C
C...PARAMETERS FOR SCATTERING PATTERN COMPUTATIONS
C
        WRITE(6,*)'TETA & PHI'
        READ(5,*)TETA,PHI
        WRITE(6,*)'POLARIZATION ANGLE : 0.)H [HZ=0] , 90.)E [EZ=0]'
        READ(5,*)PSI
        WRITE(6,*)'1)BISTATIC 2)MONOSTATIC(BACKSCATTERING) COMPUTATIONS'
        READ(5,*)ISCAT
        WRITE(6,*)'1)ELEVATION  2)AZIMUTH CUT'
        READ(5,*)ICUT
        WRITE(6,*)'INITIAL & FINAL ANGLES AND INCREMENT'
        READ(5,*)ANGI,ANGF,INC
C
C   ---------------------------------------------------------------
C
C...THE SURFACE BASIS FUNCTIONS TO BE INCORPORATED
C
        WRITE(6,*)'BASIS: 1)CONVENTIONAL DFT 2)SURFACE PULSE 3)ROOF-TOP'
        READ(5,*)IBASE
C
C   ---------------------------------------------------------------
C
C...THE METHOD OF GREENS FUNCTION EVALUATION
C
        WRITE(6,*)'1)DISCRETE OR 2)ANALYTICAL TRANSFORM OF THE DYADIC G'
```

```
        READ(5,*)IG
        CALL GREENS(GXX,GXY,GYX,GYY,IG,IBASE)
C
C   ---------------------------------------------------------------
C
C...ITERATION PARAMETERS
C
        WRITE(6,*)'MAX & MIN NO. OF ITERATIONS:ITMAX,ITMIN & TOLERANCE'
        READ(5,*)ITMAX,ITMIN,TOL
C
C   ---------------------------------------------------------------
C
C...INITIAL GUESS (OF ZERO)
C
        DO 6 I = 1,MGI
           JS(I) = (0.,0.)
6       CONTINUE
        OPEN(UNIT=3,FILE='sigma0')
        WRITE(3,7)
7       FORMAT('INPUT DATA.',/,'"SIGMA0"')
10      CONTINUE
        WRITE(6,15)TETA,PHI
15      FORMAT('ANGLES OF INCIDENCE: TETA=',F5.1,' , PHI=',F5.1)
        ITER = 0
C
C   ---------------------------------------------------------------
C
C...INCIDENT FIELD AND ITS NORM
C
        CALL INCIDE(E,TETA,PHI,PSI,X,Y)
        ENORM = VNORMX(E)
C
C   ---------------------------------------------------------------
C
C...CGFFT ZEROTH ORDER RESIDUAL CALCULATION
C
        CALL OPERAT(A,-1,JS,TPR,RES,ZT,GXX,GXY,GYX,GYY,AT,W,ITAG)
        DO 20 I = 1,MGI
           R(I) = A(I)-E(I)
           P(I) = (0.,0.)
20      CONTINUE
30      CONTINUE
C
C   ---------------------------------------------------------------
C
C    RESET TIME
C
        CALL TIME(0,PR,RES,RC4,RC8)
C
C...CGFFT MAIN ITERATION LOOP
C
        CALL OPERAT(A,1,R,TPR,RES,ZT,GXX,GXY,GYX,GYY,AT,W,ITAG)
        B = 1./VNORMX(A)
        DO 40 I = 1,MGI
           P(I) = P(I)-B*A(I)
40      CONTINUE
        CALL OPERAT(A,-1,P,TPR,RES,ZT,GXX,GXY,GYX,GYY,AT,W,ITAG)
        T = 1./VNORMX(A)
        DO 50 I = 1,MGI
           R(I) = R(I)+T*A(I)
           JS(I) = JS(I)+T*P(I)
50      CONTINUE
C
C   ---------------------------------------------------------------
C
C...TOLERANCE CHECK
```

```
C
        ITER = ITER+1
C   COMPUTE RELATIVE RESIDUAL ERROR
        RSS = SQRT(VNORMX(R)/ENORM)
        IF((RSS.LE.TOL).AND.(ITER.GT.ITMIN)) THEN
          WRITE(6,400)
        ELSE IF(ITER.GE.ITMAX) THEN
          WRITE(6,500)
        ELSE
          GO TO 30
        END IF
60      FORMAT(5(1X,G12.5))
C
C    CPU TIME FOR THIS ANGLE OF INCIDENCE
C
        CALL TIME(26,PR,RES,RC4,RC8)
C
C   ----------------------------------------------------------------------
C
C...SCATTERING CALCULATIONS
C
        IF(ISCAT.EQ.2)THEN
          CALL SCATER(SIG,JS,TETA,PHI,X,Y)
          IF(ICUT.EQ.1)THEN
            WRITE(6,180)TETA,ITER,RSS,SIG
            WRITE(3,200)TETA,SIG
            TETA=TETA+INC
            IF(TETA.LE.ANGF) THEN
              GO TO 10
            ENDIF
          ELSEIF(ICUT.EQ.2)THEN
            WRITE(6,180)PHI,ITER,RSS,SIG
            WRITE(3,200)PHI,SIG
            PHI=PHI+INC
            IF(PHI.LE.ANGF) THEN
              GO TO 10
            ENDIF
          ENDIF
        ELSEIF(ISCAT.EQ.1)THEN
          WRITE(6,*)ITER,RSS
C
C...FOR BISTATIC CASE, SAVE THE CURRENT DENSITY
C
          DO 110 I=1,2
             L1=(I-1)*MG
             OPEN(I,FILE=FILE(I))
             WRITE(I,*)'TITLE'
             WRITE(I,*)MX,MY
             WRITE(I,*)(J,J=1,MX)
             WRITE(I,*)(K,K=1,MY)
             DO 70 L=1,NT
                W(L)=(0.,0.)
70           CONTINUE
             DO 80 L = 1,MG
                W(ITAG(L)) = JS(L+L1)
80           CONTINUE
             DO 100 K=1,MY
                L2=(K-1)*NX
                DO 90 J=1,MX
                   WRITE(I,*)CABS(W(L2+J))
90              CONTINUE
100          CONTINUE
             CLOSE(I)
110       CONTINUE
          IANGI=ANGI
          IANGF=ANGF
```

```
          IF(ICUT.EQ.1)THEN
            DO 150 I=IANGI,IANGF,INC
               TETA=I
               CALL SCATER(SIG,JS,TETA,PHI,X,Y)
               WRITE(6,180)TETA,SIG
               WRITE(3,200)TETA,SIG
150         CONTINUE
          ELSEIF(ICUT.EQ.2)THEN
            DO 160 I=IANGI,IANGF,INC
               PHI=I
               CALL SCATER(SIG,JS,TETA,PHI,X,Y)
               WRITE(6,180)PHI,SIG
               WRITE(3,200)PHI,SIG
160         CONTINUE
          ENDIF
        ENDIF
        WRITE(3,*)'END OF DATA.'
        CLOSE(3)
        CLOSE(9)
        WRITE(6,*)'SCATTERING COMPUTATIONS COMPLETED'
        WRITE(6,*)'ANOTHER TARGET? 1) YES  2)NO'
        READ(5,*)IRES
        IF(IRES.EQ.1)GO TO 1
        WRITE(6,*)'NORMAL TERMINATION.'
 180    FORMAT('ANGLE=',1X,F5.1,2X,'SIGMA0=',F9.3)
 200    FORMAT(F5.1,2X,F9.3)
 300    FORMAT(I4,1X,G13.4)
 400    FORMAT('CONVERGENCE  ACHIEVED.')
 500    FORMAT('ITMAX EXCEEDED; CONVERGENCE CRITERION NOT MET.')
 600    STOP
        END
C   ================================================================
C   *                    TARGET GEOMETRY SPECIFICATIONS              *
C   ================================================================
        SUBROUTINE GEOMTR(NTARG,X0,Y0,XL,YL,UPLX,UPLY,IPAD,ITAG,XT,YT)
        INTEGER ITAG(*)
        REAL XT(*),YT(*),XC(10),YC(10)
        CHARACTER*1 ITARG(256,256)
        LOGICAL TAG,TRITAG
        COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
        TAG=.FALSE.
        IF(NTARG.EQ.1)THEN
C
C...RECTANGULAR PLATE
C
        ELSEIF(NTARG.EQ.2)THEN
C
C...CIRCULAR DISK
C
          WRITE(6,*)'ENTER THE RADIUS:'
          READ(5,*)RAD
        ELSEIF((NTARG.EQ.3).OR.(NTARG.EQ.4))THEN
          IF(NTARG.EQ.3)THEN
C
C...TRIANGULAR PLATE
C
            NC=3
          ELSEIF(NTARG.EQ.4)THEN
C
C...POLYGONAL PLATE
C
            WRITE(6,*)'NUMBER OF CORNERS:'
            READ *,NC
          ENDIF
          NT=NC-2
          WRITE(6,*)'ENTER THE COORDINATES OF EACH CORNER SEQUENTIALLY:'
```

```
          DO 30 I=1,NC
             WRITE(6,*)'CORNER #',I
             READ(5,*)XC(I),YC(I)
30        CONTINUE
        ELSEIF(NTARG.EQ.6)THEN
          XL=3.
          YL=2.
          X0=1.5
          Y0=0.
          C1=SQRT(3.)
        ELSEIF(NTARG.EQ.7)THEN
          XL=4.
          YL=2.
          X0=2.
          Y0=0.
          C1=SQRT(3.)
          C2=1.+C1
        ELSEIF(NTARG.EQ.8)THEN
          XL=3.5
          YL=2.
          X0=1.75
          Y0=0.
          C1=2.5
        ELSEIF(NTARG.EQ.9)THEN
          XL=2.5
          YL=2.
          X0=.25
          Y0=0.
          C1=SQRT(3.)
          C2=C1/2.
        ENDIF
        XL2=XL/2.
        YL2=YL/2.
        XSTRT=X0-XL2
        YSTRT=Y0-YL2
        MX=XL*UPLX
        MY=YL*UPLY
        NX=2**(INT(ALOG(2.*MX)/ALOG(2.))+IPAD)
        NY=2**(INT(ALOG(2.*MY)/ALOG(2.))+IPAD)
        DX=1./UPLX
        DY=1./UPLY
        DO 20 J=1,NY
           DO 10 I=1,NX
              ITARG(I,J)='.'
10         CONTINUE
20      CONTINUE
        INDX=0
        DO 60 J=1,MY
           Y=YSTRT+(J-0.5)*DY
           L1=(J-1)*NX
           DO 50 I=1,MX
              X=XSTRT+(I-0.5)*DX
              L=L1+I
C
C   ------------------------------------------------------------------
C
C...TARGET # 1 RECTANGLE
C
              IF(NTARG.EQ.1)THEN
                TAG=.TRUE.
C   ------------------------------------------------------------------
C
C...TARGET # 2  CIRCULAR DISK
C
              ELSEIF(NTARG.EQ.2)THEN
                IF( (SQRT((X-X0)**2+(Y-Y0)**2)).LT.RAD )TAG=.TRUE.
```

```
C
C   ------------------------------------------------------------------
C
C...TARGETS # 3 AND 4  TRIANGLE AND POLYGON
C
               ELSEIF((NTARG.EQ.3).OR.(NTARG.EQ.4))THEN
                 DO 40 K=1,NT
                    IF(
     +             TRITAG(X,Y,
     +                    XC(1),YC(1),XC(K+1),YC(K+1),XC(K+2),YC(K+2))
     +              )TAG=.TRUE.
40               CONTINUE
C
C   ------------------------------------------------------------------
C
C...TARGET # 6
C
               ELSEIF(NTARG.EQ.6)THEN
                 IF(
     +             (X.LE.C1).AND.(ABS(Y).LE.X/C1)
     +                      .OR.
     +             (X.GE.C1).AND.(((X-C1)**2+Y**2).LE.1.)
     +              )TAG=.TRUE.
C
C   ------------------------------------------------------------------
C
C...TARGET # 7
C
               ELSEIF(NTARG.EQ.7)THEN
                 IF(   (X.LE.C1).AND.(ABS(Y).LE.X/C1)
     +                      .OR.
     +               (X.GE.C1).AND.(X.LE.C2).AND.(ABS(Y).LE.1.)
     +                      .OR.
     +               (X.GE.C2).AND.(((X-C2)**2+Y**2).LE.1.)   )TAG=.TRUE.
C
C   ------------------------------------------------------------------
C
C...TARGET # 8
C
               ELSEIF(NTARG.EQ.8)THEN
                 IF(   (X.LE.C1).AND.(ABS(Y).LE.1.)
     +                      .OR.
     +               (X.GE.C1).AND.(((X-C1)**2+Y**2).LE.1.)   )TAG=.TRUE.
C
C   ------------------------------------------------------------------
C
C...TARGET # 9
C
               ELSEIF(NTARG.EQ.9)THEN
                 IF(   ((ABS(Y).LE.C2).AND.(Y.GE.C1*(ABS(X)-.5)))
     +                      .OR.
     +               ((X.GE.0.).AND.(Y.LE.C2)
     +                      .AND.(((X-.5)**2+Y**2).LE.1.))   )THEN
                    TAG=.TRUE.
                    IF((X.LE.0.).AND.(ABS(Y).LE.DY/2.))THEN
                      TAG=.FALSE.
                    ENDIF
                 ENDIF
               ENDIF
C
C...TAG ASSIGNMENT
C
               IF(TAG)THEN
                 ITARG(I,J)='*'
                 INDX=INDX+1
                 ITAG(INDX)=L
```

```
                XT(INDX)=X
                YT(INDX)=Y
                TAG=.FALSE.
              ENDIF
50          CONTINUE
60        CONTINUE
C
C   ---------------------------------------------------------------
C
C...TOTAL NUMBER OF 'NONZERO' ELEMENTS
C
          WRITE(6,*)
          WRITE(6,*)'XI= ',XSTRT+0.5*DX,'  XF= ',X,'  MX= ',MX,'  NX= ',NX
          WRITE(6,*)'YI= ',YSTRT+0.5*DY,'  YF= ',Y,'  MY= ',MY,'  NY= ',NY
          WRITE(6,*)
          MG=INDX
          MGI=2*MG
          WRITE(6,*)'GEOMETRY DISPLAY? 1)YES 2)NO'
          READ(5,*)IRES
          IF(IRES.EQ.1)THEN
            DO 70 J=NY,1,-1
               WRITE(6,80)(ITARG(I,J),I=1,NX)
70          CONTINUE
80          FORMAT(128(1X,A1))
          ENDIF
          WRITE(6,*)
          WRITE(6,*)'TARGET GEOMETRY SPECIFICATION COMPLETED.'
          RETURN
          END
C   ================================================================
C   *  IDENTIFYING A POINT IN THE INTERIOR OF A TRIANGLE SPECIFIED BY  *
C   *  ITS CORNERS                                                    *
C   ================================================================
          LOGICAL FUNCTION TRITAG(X,Y,X1,Y1,X2,Y2,X3,Y3)
          TRITAG=.FALSE.
          Z1 = ( X -  X1) * ( Y2-Y1 ) - ( X2-X1 ) * ( Y  -Y1 )
          Z2 = ( X3 - X1) * ( Y2-Y1 ) - ( X2-X1 ) * ( Y3 -Y1 )
          Z3 = ( X  - X1) * ( Y3-Y1 ) - ( X3-X1 ) * ( Y  -Y1 )
          Z4 = ( X2 - X1) * ( Y3-Y1 ) - ( X3-X1 ) * ( Y2 -Y1 )
          Z5 = ( X  - X2) * ( Y3-Y2 ) - ( X3-X2 ) * ( Y  -Y2 )
          Z6 = ( X1 - X2) * ( Y3-Y 2) - ( X3-X2 ) * ( Y1 -Y2 )
          IF(
     +       (Z1*Z2 .GE. 0.0).AND.(Z3*Z4 .GE. 0.0).AND.(Z5*Z6 .GE. 0.0)
     +       )TRITAG=.TRUE.
          RETURN
          END
C   ================================================================
C   * EVALUATION OF DYADIC GREEN'S FUNCTION AND ITS FOURIER TRANSFORM  *
C   ================================================================
          SUBROUTINE GREENS(GXX,GXY,GYX,GYY,IG,IBASE)
          REAL*8 AUX1(20000),AUX2(20000)
          REAL*8 DX2D,DY2D,DFX,DFY,FX,FY,DERX,DERY,CXX,CXY,CYY,DBSINC
          REAL*8 TPID,TPI2D,S
          COMPLEX XJ,XJK,MYCSQR,FCT
          COMPLEX GI,G0,GX,GY,G0SELF,G0NSLF
          COMPLEX GXX(NX,NY),GXY(NX,NY),GYX(NX,NY),GYY(NX,NY)
          INTEGER N(2)
          COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
          DATA PI/3.141592653589793/,TPI/6.28318530717959/
          DATA TPI2/39.47841760435747/,TPI2D/39.47841760435747D0/
          DATA FPI/12.56637061435917/,Z0/376.991118/
          DATA TPID/6.28318530717959D0/
          DATA XJ/(0.,1.)/,XJK/(0.,6.28318530717959)/
          FCT=XJ*Z0/TPI/NT
          DS=DX*DY
          DX2=DX/2.
```

```
        DY2=DY/2.
        DX2D=DBLE(DX2)
        DY2D=DBLE(DY2)
        DFX=1.D0/DBLE(DX*NX)
        DFY=1.D0/DBLE(DY*NY)
        NXH = NX/2+1
        NYH = NY/2+1
C   ------------------------------------------------------------------
        DO 2 J=1,NY
           DO 1 I=1,NX
              GXX(I,J)=(0.,0.)
              GXY(I,J)=(0.,0.)
              GYX(I,J)=(0.,0.)
              GYY(I,J)=(0.,0.)
1          CONTINUE
2       CONTINUE
C
C   ------------------------------------------------------------------
C
C...DISCRETE GREEN'S FUNCTION
C
        IF(IG.EQ.1)THEN
          N(1)=NX
          N(2)=NY
          RH0=SQRT(DS/PI)
          RBOUND=RH0
          DO 20 J=-MY+1,MY-1
             IF(J.LT.0)THEN
               JQ=NY+J
             ELSE
               JQ=J
             ENDIF
             ETA=J*DY
             ETAP=ETA+DY2
             ETAM=ETA-DY2
             DO 10 I=-MX+1,MX-1
                IF(I.LT.0)THEN
                  IP=NX+I
                ELSE
                  IP=I
                ENDIF
                PSI=I*DX
                PSIP=PSI+DX2
                PSIM=PSI-DX2
                R=SQRT(PSI**2+ETA**2)
                IF(R.GT.RBOUND)THEN
                  G0 = G0NSLF(R)
                ELSE
                  G0 =   G0SELF(PSIP,ETAP) + G0SELF(PSIM,ETAM)
     +               - ( G0SELF(PSIP,ETAM) + G0SELF(PSIM,ETAP) )
                ENDIF
                GXX(IP+1,JQ+1) = G0 / FPI
                GXX(IP+1,JQ+1)=FCT*GXX(IP+1,JQ+1)
10           CONTINUE
20        CONTINUE
C       CALL SCFT2(1,GXX,1,NX,GXX,1,NX,NX,NY,1,1.0,AUX1,20000,AUX2,20000)
C       CALL SCFT2(0,GXX,1,NX,GXX,1,NX,NX,NY,1,1.0,AUX1,20000,AUX2,20000)
          CALL FFTNDM(GXX,N,2,-1)
          DO 40 J=1,NY
             IF(J.LE.NYH)FY=(J-1)*DFY
             IF(J.GT.NYH)FY=(J-NY-1)*DFY
             DERY=FY*DBSINC(TPID*FY*DY2D)
             CYY=1.D0-DERY*DERY
             DO 30 I=1,NX
                IF(I.LE.NXH)FX=(I-1)*DFX
                IF(I.GT.NXH)FX=(I-NX-1)*DFX
```

```
                DERX=FX*DBSINC(TPID*FX*DX2D)
                CXX=1.D0-DERX*DERX
                CXY=-DERX*DERY
                GYY(I,J)=GXX(I,J)*SNGL(CYY)
                GXY(I,J)=GXX(I,J)*SNGL(CXY)
                GYX(I,J)=GXY(I,J)
                GXX(I,J)=GXX(I,J)*SNGL(CXX)
30            CONTINUE
40          CONTINUE
C
C   ------------------------------------------------------------------
C
C...ANALYTICAL TRANSFORM OF THE GREEN'S FUNCTION
C
          ELSEIF(IG.EQ.2)THEN
            DO 60 J=1,NY
               IF(J.LE.NYH)FY=(J-1)*DFY
               IF(J.GT.NYH)FY=(J-NY-1)*DFY
               CYY=TPI2D*(1.D0-FY*FY)
               DO 50 I=1,NX
                  IF(I.LE.NXH)FX=(I-1)*DFX
                  IF(I.GT.NXH)FX=(I-NX-1)*DFX
                  CXX=TPI2D*(1.D0-FX*FX)
                  CXY=-TPI2D*FX*FY
                  S=FX*FX+FY*FY-1.D0
                  GI=2.*TPI*MYCSQR(S)
                  IF(CABS(GI).LT.1.E-20)GI=1.E-10
                  GXX(I,J)=FCT*SNGL(CXX)/GI
                  GXY(I,J)=FCT*SNGL(CXY)/GI
                  GYX(I,J)=GXY(I,J)
                  GYY(I,J)=FCT*SNGL(CYY)/GI
50             CONTINUE
60           CONTINUE
          ENDIF
C
C   ------------------------------------------------------------------
C
C...BASIS FUNCTIONS
C
          DO 80 J=1,NY
             IF(J.LE.NYH)FY=(J-1)*DFY
             IF(J.GT.NYH)FY=(J-NY-1)*DFY
             DO 70 I=1,NX
                IF(I.LE.NXH)FX=(I-1)*DFX
                IF(I.GT.NXH)FX=(I-NX-1)*DFX
                IF(IBASE.EQ.1)THEN
                ELSEIF(IBASE.EQ.2)THEN
                  GXX(I,J)=GXX(I,J)*PWCNST(DX,FX)*PWCNST(DY,FY)
                  GXY(I,J)=GXY(I,J)*PWCNST(DX,FX)*PWCNST(DY,FY)
                  GYX(I,J)=GYX(I,J)*PWCNST(DX,FX)*PWCNST(DY,FY)
                  GYY(I,J)=GYY(I,J)*PWCNST(DX,FX)*PWCNST(DY,FY)
                ELSEIF(IBASE.EQ.3)THEN
                  GXX(I,J)=GXX(I,J)*PWSINE(DX,FX)*PWCNST(DY,FY)
                  GXY(I,J)=GXY(I,J)*PWCNST(DX,FX)*PWSINE(DY,FY)
                  GYX(I,J)=GYX(I,J)*PWSINE(DX,FX)*PWCNST(DY,FY)
                  GYY(I,J)=GYY(I,J)*PWCNST(DX,FX)*PWSINE(DY,FY)
                ENDIF
70           CONTINUE
80        CONTINUE
          RETURN
          END
C   ==================================================================
C   *                              G0NSLF                              *
C   ==================================================================
          COMPLEX FUNCTION G0NSLF(R)
          COMPLEX GFREES,XJK
```

```
      COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
      DATA TPI2/39.47841760435747/,XJK/(0.,6.28318530717959)/
      GFREES(R)=CEXP(-XJK*R)/R
      G0NSLF=TPI2*DS*GFREES(R)
      RETURN
      END
C     ================================================================
C     *                              G0SELF                            *
C     ================================================================
      COMPLEX FUNCTION G0SELF(X,Y)
      COMPLEX XJK
      COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
      DATA TPI2/39.47841760435747/,XJK/(0.,6.28318530717959)/
      X2=X*X
      Y2=Y*Y
      X3=X*X2
      Y3=Y*Y2
      XY=X*Y
      R2=X2+Y2
      R=SQRT(R2)
      ALX=ALOG(X+R)
      ALY=ALOG(Y+R)
      G1=X*ALY+Y*ALX
      G2=DS
      G3=XY*R/3.+(X3*ALY+Y3*ALX)/6.
      G4=XY*R2/3.
      G0SELF=TPI2*(G1-XJK*G2-TPI2/2.*G3+XJK*TPI2/6.*G4)
      RETURN
      END
C     ================================================================
C     *   TRANSFORM OF THE SURFACE PIECEWISE CONSTANT EXPANSION FUNCTION *
C     ================================================================
      REAL FUNCTION PWCNST(D,F)
      DATA TPI/6.28318530717959/
      C=TPI*F
      PWCNST=SNSINC(C*D/2.)
      RETURN
      END
C     ================================================================
C     *    TRANSFORM OF THE OVERLAPPING SINUSOIDAL EXPANSION FUNCTION    *
C     ================================================================
      REAL FUNCTION PWSINE(D,F)
      DATA TPI/6.28318530717959/
      C=TPI*F
      C1=TPI*TPI*(1.-F*F)
      IF(C.EQ.0.)THEN
        PWSINE=1.
      ELSE
        PWSINE=2.*TPI*(COS(C*D)-COS(TPI*D))/(D*C1*SIN(TPI*D))
      ENDIF
      RETURN
      END
C     ================================================================
C     *                            INCIDENT FIELD                      *
C     ================================================================
C@PROCESS DIRECTIVE ('*VDIR')
      SUBROUTINE INCIDE(E,TETA,PHI,PSI,X,Y)
      REAL X(*),Y(*)
      COMPLEX E(*),XJ,PHASE
      COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
      DATA PI/3.141592653589793/,TPI/6.28318530717959/,Z0/376.991118/
      DATA XJ/(0.,1.)/
      STE = SIN(TETA*PI/180.)
      CTE = COS(TETA*PI/180.)
      SPH = SIN(PHI*PI/180.)
      CPH = COS(PHI*PI/180.)
```

```
        SPS = SIN(PSI*PI/180.)
        CPS = COS(PSI*PI/180.)
        RX = TPI*STE*CPH
        RY = TPI*STE*SPH
        E0=Z0
        EX = E0 * ( CPS*CTE*CPH - SPS*SPH )
        EY = E0 * ( CPS*CTE*SPH + SPS*CPH )
C*VDIR IGNORE RECRDEPS
        DO 10 I=1,MG
              PHASE   = CEXP ( XJ * ( RX*X(I) + RY*Y(I)) )
              E(I)    = EX * PHASE
              E(I+MG) = EY * PHASE
10      CONTINUE
        RETURN
        END
C   ================================================================
C   *                        EUCLEDEAN NORM CALCULATION              *
C   ================================================================
        REAL FUNCTION VNORMX(W)
        REAL*8 SUM,AMAG
        COMPLEX W(*)
        COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
        SUM = 0.D0
        DO 10 I = 1,MGI
           AMAG = DBLE ( CABS ( W(I) ) )
           SUM = SUM+AMAG*AMAG
10      CONTINUE
        VNORMX = SNGL(SUM)
        RETURN
        END
C   ================================================================
C   * INTEGRO-DIFFERENTIAL OPERATOR INVOLVING CONVOLUTIONS CARRIED OUT *
C   * IN THE TRANSFORM DOMAIN                                        *
C   ================================================================
C@PROCESS DIRECTIVE ('*VDIR')
        SUBROUTINE
     +    OPERAT(A,ISIGN,Z,TPR,RES,ZT,GXX,GXY,GYX,GYY,AT,W,ITAG)
        INTEGER ITAG(*)
       REAL RES(*)
        COMPLEX XJ,A(*),Z(*),ZT(*),AT(*),W(*)
        COMPLEX GXX(NX,NY),GXY(NX,NY),GYX(NX,NY),GYY(NX,NY)
        COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS,MT
        DATA PI/3.141592653589793/,TPI/6.2831853071796D0/,Z0/376.991118/
        DATA XJ/(0.,1.)/
C
C   ----------------------------------------------------------------
C
C...FORWARD FOURIER TRANSFORM
C
        CALL SPECTR(Z,ZT,-1,W,ITAG)
C                   I  O
C   ----------------------------------------------------------------
C
C...REGULAR OPERATOR
C
        IF(ISIGN.EQ.-1)THEN
C*VDIR IGNORE RECRDEPS
          DO 20 J=1,NY
             L=(J-1)*NX
             DO 10 I=1,NX
                K=L+I
                AT(K)=GXX(I,J)*ZT(K)+GXY(I,J)*ZT(K+NT)
10           CONTINUE
20        CONTINUE
C*VDIR IGNORE RECRDEPS
          DO 40 J=1,NY
```

```
            L=NT+(J-1)*NX
            DO 30 I=1,NX
               K=L+I
               AT(K)=GYX(I,J)*ZT(K-NT)+GYY(I,J)*ZT(K)
30          CONTINUE
40        CONTINUE
C
C   ---------------------------------------------------------------------
C
C...ADJOINT OPERATOR
C
        ELSEIF(ISIGN.EQ.1)THEN
C*VDIR IGNORE RECRDEPS
          DO 60 J=1,NY
             L=(J-1)*NX
             DO 50 I=1,NX
                K=L+I
                AT(K)=CONJG(GXX(I,J))*ZT(K)+CONJG(GXY(I,J))*ZT(K+NT)
50           CONTINUE
60        CONTINUE
C*VDIR IGNORE RECRDEPS
          DO 80 J=1,NY
             L=NT+(J-1)*NX
             DO 70 I=1,NX
                K=L+I
                AT(K)=CONJG(GYX(I,J))*ZT(K-NT)+CONJG(GYY(I,J))*ZT(K)
70           CONTINUE
80        CONTINUE
        ENDIF
C
C   ---------------------------------------------------------------------
C
C...INVERSE FOURIER TRANSFORM
C
        CALL SPECTR(A,AT,1,W,ITAG)
C                   O  I
C   ---------------------------------------------------------------------
C
C...CALCULATIONS FOR RESISTIVE PLATES
C
          IF(TPR.GT.-1.)THEN
          IF(ISIGN.EQ.-1) THEN
C*VDIR IGNORE RECRDEPS
            DO 100 J=1,MY
              L=(J-1)*MX
              DO 90 I=1,MX
                K=L+I
                  A(K)=RES(K)*Z(K)+A(K)
90            CONTINUE
100         CONTINUE
C*VDIR IGNORE RECRDEPS
            DO 120 J=1,MY
               L=MT+(J-1)*MX
               DO 110 I=1,MX
                 K=L+I
                   A(K)=RES(K-MT)*Z(K-MT)+A(K)
110            CONTINUE
120       CONTINUE
          ELSEIF(ISIGN.EQ.1) THEN
C*VDIR IGNORE RECRDEPS
            DO 140 J=1,MY
               L=(J-1)*MX
               DO 130 I=1,MX
                  K=L+I
                  A(K)=RES(K)*Z(K)+A(K)
130            CONTINUE
```

```
140          CONTINUE
             DO 160 J=1,MY
               L=MT+(J-1)*MX
                 DO 150 I=1,MX
                   K=L+I
                   A(K)=RES(K-MT)*Z(K-MT)+A(K)
150              CONTINUE
160       CONTINUE
         ENDIF
         ENDIF
         RETURN
         END
C    ================================================================
C    *                      FOURIER TRANSFORM VIA FFT                 *
C    ================================================================
         SUBROUTINE SPECTR(Z,ZT,ISIGN,W,ITAG)
         INTEGER N(2),ITAG(*)
         REAL*8 AUX1(20000),AUX2(20000)
         COMPLEX Z(*),ZT(*),W(*)
         COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
         N(1)=NX
         N(2)=NY
C
C    ----------------------------------------------------------------
C
C...FORWARD FOURIER TRANSFORM
C
         IF(ISIGN.EQ.-1)THEN
           DO 10 I = 1,2
C    PRECONDITIONING IN SPATIAL DOMAIN
              DO 15 II=1,NT
                 W(II)=(0.,0.)
15            CONTINUE
              L1 = (I-1)*MG
              DO 20 L = 1,MG
                 W(ITAG(L)) = Z(L1+L)
20            CONTINUE
C          CALL SCFT2(1,W,1,NX,W,1,NX,NX,NY,1,1.0,AUX1,20000,AUX2,20000)
C          CALL SCFT2(0,W,1,NX,W,1,NX,NX,NY,1,1.0,AUX1,20000,AUX2,20000)
              CALL FFTNDM(W,N,2,-1)
              L1 = (I-1)*NT
              DO 40 L = 1,NT
                 ZT(L1+L) = W(L)
40            CONTINUE
10         CONTINUE
C
C    ----------------------------------------------------------------
C
C...INVERSE FOURIER TRANSFORM
C
         ELSE IF(ISIGN.EQ.1)THEN
           DO 50 I = 1,2
              L1 = (I-1)*NT
              DO 60 L = 1,NT
                 W(L) = ZT(L1+L)
60            CONTINUE
C         CALL SCFT2(1,W,1,NX,W,1,NX,NX,NY,-1,1.0,AUX1,20000,AUX2,20000)
C         CALL SCFT2(0,W,1,NX,W,1,NX,NX,NY,-1,1.0,AUX1,20000,AUX2,20000)
              CALL FFTNDM(W,N,2,1)
              L1 = (I-1)*MG
              DO 70 L = 1,MG
                 Z(L1+L) = W(ITAG(L))
70            CONTINUE
50         CONTINUE
         END IF
         RETURN
```

```
      END
C     ================================================================
C     *    ROUTINE FOR COMPUTATION OF FAR ZONE SCATTERING COEFFICIENT     *
C     ================================================================
      SUBROUTINE SCATER(SIG,JS,TETA,PHI,X,Y)
      REAL X(*),Y(*)
      COMPLEX JS(*),XJ,PHASE,SX,SY,NTETA,NPHI
      COMMON /DIM/MX,MY,NX,NY,NT,MG,MGI,DX,DY,DS
      DATA PI/3.141592653589793/,TPI/6.2831853071796D0/,XJ/(0.,1.)/
C     ----------------------------------------------------------------
      STE = SIN(TETA*PI/180.)
      CTE = COS(TETA*PI/180.)
      SPH = SIN(PHI*PI/180.)
      CPH = COS(PHI*PI/180.)
      RX = TPI*STE*CPH
      RY = TPI*STE*SPH
      SX = (0.,0.)
      SY = (0.,0.)
      DO 10 I=1,MG
            PHASE = CEXP(XJ*(RX*X(I)+RY*Y(I)))
            SX = SX + JS(I)     * PHASE
            SY = SY + JS(I+MG) * PHASE
10    CONTINUE
      SX = DS*SNSINC(DX*RX/2.)*SNSINC(DY*RY/2.)*SX
      SY = DS*SNSINC(DX*RX/2.)*SNSINC(DY*RY/2.)*SY
      NTETA = CTE*(CPH*SX+SPH*SY)
      NPHI =-SPH*SX+CPH*SY
      SIG = PI*(CABS(NTETA)**2+CABS(NPHI)**2)
      IF(SIG.LE.1.E-10)SIG=1.E-10
      SIG = 10.*ALOG10(SIG)
      RETURN
      END
C     ================================================================
C     *                              SNSINC                            *
C     ================================================================
      REAL FUNCTION SNSINC(A)
      IF(A.EQ.0)THEN
        SNSINC = 1.
      ELSE
        SNSINC = SIN(A) / A
      ENDIF
      RETURN
      END
C     ================================================================
C     *                              DBSINC                            *
C     ================================================================
      DOUBLE PRECISION FUNCTION DBSINC(A)
      REAL*8 A
      IF(A.EQ.0.D0)THEN
        DBSINC = 1.D0
      ELSE
        DBSINC = DSIN(A) / A
      ENDIF
      RETURN
      END
C     ================================================================
C     *              SQUARE ROOT OF A SIGNED REAL NUMBER               *
C     ================================================================
      COMPLEX FUNCTION MYCSQR(A)
      REAL*8  A
      COMPLEX XJ
      DATA XJ/(0.,1.)/
      IF(A.GE.0D0)MYCSQR=SNGL(DSQRT(A))
      IF(A.LT.0D0)MYCSQR=-XJ*SNGL(DSQRT(-A))
      RETURN
      END
```

```
C    ================================================================
C    *  FFT ROUTINE FOR COMPUTATION OF N DIMENSIONAL FOURIER TRANSFORM  *
C    *  TASKS: 1)BIT REVERSAL 2)TRIGONOMETRIC RECURRENCE 3)TRANSFORM    *
C    *         ALL DONE IN THIS PROGRAM                                 *
C    ================================================================
C                          PARAMETER DESCRIPTION
C    ================================================================
C        DATA.................A REAL ARRAY IN WHICH DATA ARE STORED AS IN
C                              A MULTIDIMENSIONAL COMPLEX FORTRAN ARRAY
C        NDIM.................DIMENSION OF DATA AND THE FFT
C        NN...................INTEGER ARRAY OF LENGTH NDIM
C        ISIGN................DIRECTION OF THE TRANSFORM:
C                            -1    -FORWARD FFT
C                             1    -INVERSE  FFT TIMES THE PRODUCT OF
C                                   LENGTHS OF ALL DIMENSIONS
C    ================================================================
         SUBROUTINE  FFTNDM(DATA,NN,NDIM,ISIGN)
         REAL*8     . WR,WI,WPR,WPI,WTEMP,THETA
         DIMENSION   NN(NDIM),DATA(*)
         NTOT=1
         DO 10 IDIM=1,NDIM
            NTOT=NTOT*NN(IDIM)
10       CONTINUE
         NPREV=1
         DO 80 IDIM=1,NDIM
            N=NN(IDIM)
            NREM=NTOT/(N*NPREV)
            IP1=2*NPREV
            IP2=IP1*N
            IP3=IP2*NREM
            I2REV=1
C
C    ----------------------------------------------------------------
C
C...BIT REVERSAL SECTION
C
            DO 40 I2=1,IP2,IP1
               IF(I2.LT.I2REV)THEN
                 DO 30 I1=I2,I2+IP1-2,2
                    DO 20 I3=I1,IP3,IP2
                       I3REV=I2REV+I3-I2
                       TEMPR=DATA(I3)
                       TEMPI=DATA(I3+1)
                       DATA(I3)=DATA(I3REV)
                       DATA(I3+1)=DATA(I3REV+1)
                       DATA(I3REV)=TEMPR
                       DATA(I3REV+1)=TEMPI
20                  CONTINUE
30                CONTINUE
               END IF
               IBIT=IP2/2
 1             IF((IBIT.GE.IP1).AND.(I2REV.GT.IBIT))THEN
                 I2REV=I2REV-IBIT
                 IBIT=IBIT/2
               GO TO 1
               END IF
               I2REV=I2REV+IBIT
40          CONTINUE
C
C    ----------------------------------------------------------------
C
C...DANIELSON-LANCZOS FORMULA
C
            IFP1=IP1
 2          IF(IFP1.LT.IP2)THEN
              IFP2=2*IFP1
```

```
              THETA=ISIGN*6.28318530717959D0/(IFP2/IP1)
              WPR=-2.D0*DSIN(0.5D0*THETA)**2
              WPI=DSIN(THETA)
              WR=1.D0
              WI=0.D0
              DO 70 I3=1,IFP1,IP1
                 DO 60 I1=I3,I3+IP1-2,2
                    DO 50 I2=I1,IP3,IFP2
                       K1=I2
                       K2=K1+IFP1
                       TEMPR=SNGL(WR)*DATA(K2)-SNGL(WI)*DATA(K2+1)
                       TEMPI=SNGL(WR)*DATA(K2+1)+SNGL(WI)*DATA(K2)
                       DATA(K2)=DATA(K1)-TEMPR
                       DATA(K2+1)=DATA(K1+1)-TEMPI
                       DATA(K1)=DATA(K1)+TEMPR
                       DATA(K1+1)=DATA(K1+1)+TEMPI
50                  CONTINUE
60                CONTINUE
                  WTEMP=WR
C
C   ------------------------------------------------------------------
C
C...TRIGONOMETRIC RECURRENCE
C
                  WR=WR*WPR-WI*WPI+WR
                  WI=WI*WPR+WTEMP*WPI+WI
70             CONTINUE
               IFP1=IFP2
               GO TO 2
               END IF
               NPREV=N*NPREV
80          CONTINUE
            RETURN
            END
```

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] T. K. Sarkar, A. Ercument, and M. Rao, "Application of FFT and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies," *IEEE Trans. Antenna Propagat.*, AP-34(5):635–640, May 1985.

[2] N. N. Bojarski, k-space formulation of the electromagnetic scattering problem, Technical Report AFAL-TR-71-5, March 1971.

[3] P. M. Vanden Berg, "Iterative computational techniques in scattering based upon the integrated square error criterion," *IEEE Trans. Antenna Propagat.*, AP-32:1063–1071, October 1984.

[4] T. J. Peters and J. L. Volakis, "The application of a conjugate gradient fft method to scattering from thin planar material plates," *IEEE Trans. Antenna Propagat.*, AP-36(4):518–526, April 1988.

[5] M. F. Catedra, J. G. Cuevas, and L. Nuno, "A scheme to analyze conducting plates of resonant size using the conjugate gradient method and the fast fourier transform," *IEEE Trans. Antenna Propagat.*, AP-36:1744–1752, December 1988.

[6] C. Y. Shen, K. J. Glover, M. I. Sander, and A. D. Varvatsis, "The discrete fourier transform method of solving differential-integral equations in scattering theory," *IEEE Trans. Antenna Propagat.*, AP-37:1032–1041, August 1989.

[7] K. Barkeshli and J. L. Volakis, "Improving the convergence rate of the conjugate gradient fft method using subdomain basis functions," *IEEE Trans. Antenna Propagat.*, AP-37(4), April 1989.

[8] Y. Rahmat-Samii and R. Mittra, "Electromagnetic coupling through small apertures in a conducting screen," *IEEE Trans. Antenna Propagat.*, AP-25(2), March 1977.

[9] J. S. Hey and T. B. A. Senior, "Electromagnetic scattering by thin conducting plates at glancing incidence," *Proceedings of the Physical Society*, 72:981, 1958.

[10] J. S. Hey, G. S. Stewart, J. T. Pinson and P. E. V. Prince, "The scattering of electromagnetic waves by conducting spheres and disks," *Proceedings of the Physical Society*, 69:1038, 1956.

[11] K. Barkeshli and J. L. Volakis, "A vector concurrent application of a conjugate gradient FFT algorithm to electromagnetic radiation and scattering problems," *IEEE Transactions on Magnetics*, 25(4), July 1989.