Forward

This report contains two separate papers. One describes a new implementation for formulating the scattering by flat resistive plates and the other is an application of the same formulation for computing the scattering by finite frequency selective surfaces.

Table of Contents

A BICONJUGATE GRADIENT FFT SOLUTION FOR SCATTERING BY PLANAR PLATES

Jian-Ming Jin and John L. Volakis

Radiation Laboratory

Department of Electrical Engineering and Computer Science

The University of Michigan

Ann Arbor, Michigan 48109-2122

ABSTRACT

An efficient numerical solution of the scattering by planar perfectly conducting or resistive plates is presented. The electric field integral equation is discretized using roof-top subdomain functions as testing and expansion basis and the resulting system is solved via the biconjugate gradient (BiCG) method in conjunction with the fast Fourier transform (FFT). Unlike other formulations employed in conjunction with the conjugate gradient FFT (CG-FFT) method, in this formulation the derivatives associated with the dyadic Green's function are transferred to the testing and expansion basis, thus reducing the singularity of the kernel. This leads to substantial improvements in the convergence of the solution as demonstrated by the included results.

I. INTRODUCTION

The problem of electromagnetic scattering by a perfectly conducting or resistive plate amounts to that of solving the electric field integral equation

$$\hat{\mathbf{n}} \times \mathbf{E}^{inc}(\mathbf{r}) = \eta(\mathbf{r})\, \hat{\mathbf{n}} \times \mathbf{J}(\mathbf{r}) + jk_0 Z_0\, \hat{\mathbf{n}} \times \iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{J}(\mathbf{r}')dS' \tag{1}$$

where $\hat{\mathbf{n}}$ denotes the unit vector normal to the plate, $\mathbf{E}^{inc}$ denotes the incident electric field, η is the resistivity of the plate and $\mathbf{J}$ is the unknown electric current. Also, $k_0 = 2\pi/\lambda$ is the free space wavenumber, Z_0 is the free space intrinsic impedance, S denotes the area of the plate and $\bar{\bar{\mathrm{G}}}_0$ is the free-space dyadic Green's function given by

$$\bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') = \left(\bar{\bar{\mathrm{I}}} + \frac{1}{k_0^2}\nabla\nabla\right) G_0(\mathbf{r},\mathbf{r}') \tag{2}$$

with

$$\bar{\bar{\mathrm{I}}} = \hat{\mathbf{x}}\hat{\mathbf{x}} + \hat{\mathbf{y}}\hat{\mathbf{y}} + \hat{\mathbf{z}}\hat{\mathbf{z}} \quad \text{and} \quad G_0(\mathbf{r},\mathbf{r}') = \frac{e^{-jk_0|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|}.$$

The integral equation (1) can be solved using the traditional moment method procedure in conjunction with a direct solution of the system. Alternatively, to reduce the memory demand required for matrix inversion, various iterative methods have been employed in the past and the conjugate gradient FFT (CG-FFT) method is most popular among these. Several different implementation schemes of the CG-FFT method have been proposed [1]-[8] and these differ in the manner in which the two del operators appearing in (2) are treated. The simplest approach is to employ the analytical Fourier transform of G_0 and handle the spatial derivatives implied by the del operators in the spectral domain where they become simple algebraic multiplicative factors [1]-[3]. This approach usually requires a large FFT pad to reduce aliasing errors since the analytical

Fourier transform of G_0 extends over the entire space. To reduce the size of the FFT pad and to eliminate aliasing errors, another approach has been considered where the integral equation is first cast in a discrete form before invoking the convolution theorem to evaluate the integrals [4]-[6]. In connection with this approach, the derivatives implied by the del operators can be approximated via finite differences and calculated via the discrete Fourier transform as described in [5] and [6].

The above approaches share a common feature that the del operators act directly upon G_0. However, by using Gauss' or the divergence theorem these differential operators can be transferred to the testing and/or expansion functions provided they are differentiable. This concept has been adopted in [7] where the roof-top basis functions were employed for the expansion of **J** and one of the del operators was transferred from G_0 to **J**. Point matching was then used to cast the integral equation in a discrete form before invoking the convolution theorem. Most recently, another approach was proposed by Zwamborn and van den Berg [8] who employed the roof-top functions for testing the integral equation (1). By invoking Gauss' theorem, one of the del operators was transferred to the testing function and the other was left to operate on the vector potential which was then expanded in terms of the roof-top basis functions. Unfortunately, in [8] the analytical Fourier transform of G_0 was still employed and thus a large FFT pad was required to suppress aliasing errors, reducing the efficiency of the solution.

The approaches described in [7] and [8] demonstrated that a much more accurate and efficient solution can be achieved by transferring one of the del operators from the singular kernel to the expansion or testing functions. One would then expect that if both del operators are transferred from the free space Green's function the resulting

discrete system should yield a more accurate and efficient solution. In the following this proposition is examined by transferring one of the del operators to the testing function and the other to the expansion function. The roof-top expansion basis are employed to discretize the surface current density but in contrast to traditional formulations the resulting system of equations is obtained by setting to zero the variation of a certain functional stationary about the true solution of the current density. The resulting system is identical to that resulting from an application of Galerkin's procedure but the adopted variational approach leads to the system construction in less steps. The biconjugate gradient (BiCG) algorithm is employed for the system solution and it is shown that the convergence rate is substantially improved in comparison with previous CG-FFT implementations. In those implementations the CG algorithm was employed to ensure convergence and this increased the operation count per iteration in addition to squaring the system's condition. In contrast, the BiCG algorithm, although not monotonically decreasing the error, requires only one matrix-vector product per iteration (since the matrix is symmetric) and does not square the condition number which generally translates to improved convergence rates.

II. FORMULATION AND DISCRETIZATION

In this section, we formulate the system of equations for the solution of the integral equation (1). To this end we introduce the functional

$$\begin{aligned} F(\mathbf{J}') &= \iint_S \eta(\mathbf{r})\mathbf{J}'(\mathbf{r}) \bullet \mathbf{J}'(\mathbf{r}) dS + jk_0 Z_0 \iint_S \mathbf{J}'(\mathbf{r}) \bullet \left[\iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{J}'(\mathbf{r}') dS' \right] dS \\ &\quad -2 \iint_S \mathbf{J}'(\mathbf{r}) \bullet \mathbf{E}^{\mathrm{inc}}(\mathbf{r}) dS \end{aligned} \tag{3}$$

which we assert to be stationary about $\mathbf{J}' = \mathbf{J}$, where $\mathbf{J}$ is the true solution of (1). To

prove this assertion, let us set

$$\mathbf{J}'(\mathbf{r}) = \mathbf{J}(\mathbf{r}) + \mathbf{u}(\mathbf{r}) \tag{4}$$

where $\mathbf{u}$ is an arbitrary vector which lies in the plane of S. The first variation of $\mathbf{J}$ is then given by

$$\begin{aligned} \delta F &= 2\iint_S \eta(\mathbf{r})\mathbf{u}(\mathbf{r}) \bullet \mathbf{J}(\mathbf{r})dS + jk_0Z_0 \iint_S \mathbf{J}(\mathbf{r}) \bullet \left[\iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{u}(\mathbf{r}')dS'\right] dS \\ &\quad + jk_0Z_0 \iint_S \mathbf{u}(\mathbf{r}) \bullet \left[\iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{J}(\mathbf{r}')dS'\right] dS \\ &\quad - 2\iint_S \mathbf{u}(\mathbf{r}) \bullet \mathbf{E}^{inc}(\mathbf{r})dS \end{aligned} \tag{5}$$

and since $\bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}')$ is equal to its transpose, we can rewrite (5) as

$$\delta F = 2\iint_S \mathbf{u}(\mathbf{r}) \bullet \left[\eta(\mathbf{r})\mathbf{J}(\mathbf{r}) + jk_0Z_0 \iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{J}(\mathbf{r}')dS' - \mathbf{E}^{inc}(\mathbf{r})\right] dS \tag{6}$$

Upon setting $\mathbf{u} = \mathbf{v} \times \hat{\mathbf{n}}$, where $\mathbf{v}$ is arbitrary in both magnitude and direction, it follows that

$$\delta F = 2\iint_S \mathbf{v}(\mathbf{r}) \bullet \left\{\hat{\mathbf{n}} \times \left[\eta(\mathbf{r})\mathbf{J}(\mathbf{r}) + jk_0Z_0 \iint_S \bar{\bar{\mathrm{G}}}_0(\mathbf{r},\mathbf{r}') \bullet \mathbf{J}(\mathbf{r}')dS' - \mathbf{E}^{inc}(\mathbf{r})\right]\right\} dS \tag{7}$$

and upon invoking the stationarity requirement that $\delta F = 0$ we recover (1).

Let us now proceed with the discretization of F and for convenience we will henceforth drop the distinction between $\mathbf{J}$ and $\mathbf{J}'$. We will first attempt to reduce the singularity of the improper kernel in (3) by transferring the del operators to the current density in much the same way as done in [9] and [10]. By substituting (2) into (3) and invoking a common vector identity along with the divergence theorem, (3) can be written as

$$F = \iint_S \eta(\mathbf{r})\mathbf{J}(\mathbf{r}) \bullet \mathbf{J}(\mathbf{r})dS + jk_0Z_0 \iint_S \mathbf{J}(\mathbf{r}) \bullet \left\{\iint_S \mathbf{J}(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}')dS'\right.$$

$$+\frac{1}{k_0^2}\nabla \iint_S \left[\nabla' \bullet \mathbf{J}(\mathbf{r}')\right] G_0(\mathbf{r},\mathbf{r}')dS' - \frac{1}{k_0^2}\nabla \oint_C \hat{\mathbf{n}}_c' \bullet \mathbf{J}(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}')dl' \Big\} dS$$
$$-2\iint_S \mathbf{J}(\mathbf{r}) \bullet \mathbf{E}^{inc}(\mathbf{r})dS \tag{8}$$

where C denotes the perimeter of the plate and $\hat{\mathbf{n}}_c$ is the unit vector which lies in the plane of the plate and is normal to C. However, since $\hat{\mathbf{n}}_c' \bullet \mathbf{J}(\mathbf{r}') = 0$, the corresponding contour integral vanishes. Further application of the vector identity and the divergence theorem on the third right-hand side term of (8) gives

$$\begin{aligned} F \;=\; & \iint_S \eta(\mathbf{r})\mathbf{J}(\mathbf{r}) \bullet \mathbf{J}(\mathbf{r})dS + jk_0Z_0 \iint_S \mathbf{J}(\mathbf{r}) \bullet \left[\iint_S \mathbf{J}(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}')dS'\right] dS \\ & + \frac{Z_0}{jk_0}\iint_S \left[\nabla \bullet \mathbf{J}(\mathbf{r})\right] \left\{\iint_S \left[\nabla' \bullet \mathbf{J}(\mathbf{r}')\right] G_0(\mathbf{r},\mathbf{r}')dS'\right\} dS \\ & - \frac{Z_0}{jk_0}\oint_C \hat{\mathbf{n}}_c \bullet \mathbf{J}(\mathbf{r}) \left\{\iint_S \left[\nabla' \bullet \mathbf{J}(\mathbf{r}')\right] G_0(\mathbf{r},\mathbf{r}')dS'\right\} dS \\ & - 2\iint_S \mathbf{J}(\mathbf{r}) \bullet \mathbf{E}^{inc}(\mathbf{r})dS \end{aligned} \tag{9}$$

where we again note that $\hat{\mathbf{n}}_c \bullet \mathbf{J}(\mathbf{r}) = 0$ and thus the fourth right-hand side term vanishes. Clearly, the singularity of all kernels in (9) are integrable and we can proceed with their discretization in the standard manner provided the divergence of the chosen expansion basis can be analytically defined.

Without loss of generality, let us assume that the plate is in the x-y plane and (9) can then be written as

$$\begin{aligned} F \;=\; & \iint_S \eta\left(J_x^2 + J_y^2\right) dS + jk_0Z_0 \iint_S \left\{J_x\left[\iint_S J_xG_0dS'\right] + J_y\left[\iint_S J_yG_0dS'\right]\right\} dS \\ & + \frac{Z_0}{jk_0}\iint_S \left(\frac{\partial J_x}{\partial x} + \frac{\partial J_y}{\partial y}\right)\left[\iint_S \left(\frac{\partial J_x}{\partial x'} + \frac{\partial J_y}{\partial y'}\right) G_0dS'\right] dS \\ & - 2\iint_S (J_xE_x^{inc} + J_yE_y^{inc})dS \end{aligned} \tag{10}$$

With the intend of computing the surface integrals via the FFT we place the plate in a rectangular area which is then divided into $(M+1) \times (N+1)$ small rectangles whose

side lengths are Δx and Δy along the x and y directions, respectively. To discretize (10) we expand the x and y components of the current density as

$$J_x(x,y) = \sum_{m=1}^{M}\sum_{n=1}^{N+1} J_x^D(m,n)T_{mn}^x(x,y) \tag{11a}$$

$$J_y(x,y) = \sum_{m=1}^{M+1}\sum_{n=1}^{N} J_y^D(m,n)T_{mn}^y(x,y) \tag{11b}$$

where $J_{x,y}^D(m,n)$ denote the sample values of the current density at $(x = m\Delta x, (n-1)\Delta y \le y \le n\Delta y)$ for J_x and at $((m-1)\Delta x \le x \le m\Delta x, y = n\Delta y)$ for J_y, which are non-zero only if the associated cell is within S. The functions $T_{mn}^{x,y}$ represent the roof-top basis given by

$$T_{mn}^x(x,y) = \begin{cases} 1 - \frac{|x-m\Delta x|}{\Delta x}, & \text{for } |x-m\Delta x| \le \Delta x \text{ and } |y-(n-0.5)\Delta y| < \frac{\Delta y}{2} \\ 0, & \text{else} \end{cases} \tag{12a}$$

$$T_{mn}^y(x,y) = \begin{cases} 1 - \frac{|y-n\Delta y|}{\Delta y}, & \text{for } |y-n\Delta y| \le \Delta y \text{ and } |x-(m-0.5)\Delta x| < \frac{\Delta x}{2} \\ 0, & \text{else} \end{cases} \tag{12b}$$

from which it is readily found that

$$\frac{\partial J_x}{\partial x} = \sum_{m=1}^{M+1}\sum_{n=1}^{N+1} \frac{J_x^D(m,n) - J_x^D(m-1,n)}{\Delta x} P_{mn}(x,y) \tag{13a}$$

$$\frac{\partial J_y}{\partial y} = \sum_{m=1}^{M+1}\sum_{n=1}^{N+1} \frac{J_y^D(m,n) - J_y^D(m,n-1)}{\Delta y} P_{mn}(x,y) \tag{13b}$$

where $P_{mn}(x,y)$ is the pulse function defined by

$$P_{mn}(x,y) = \begin{cases} 1, & \text{for } |x-(m-0.5)\Delta x| < \frac{\Delta x}{2} \text{ and } |y-(n-0.5)\Delta y| < \frac{\Delta y}{2} \\ 0, & \text{else} \end{cases} \tag{14}$$

Substituting (11) and (13) into (10) yields the functional F in terms of the unknown current samples $J_{x,y}^D(m,n)$. To obtain a system of equations for $J_{x,y}^D(m,n)$ we must

enforce $\delta F = 0$ and this is equivalent to setting

$$\frac{\partial F}{\partial J_x^D(m,n)} = 0 \tag{15a}$$

$$\frac{\partial F}{\partial J_y^D(m,n)} = 0 \tag{15b}$$

Upon performing the differentiations we find that

$$\begin{aligned}\frac{1}{2}\frac{\partial F}{\partial J_x^D(m,n)} &= \sum_{m'=1}^{M}\sum_{n'=1}^{N+1}\left[R^x(m,n;m',n') + jk_0Z_0G_T^x(m,n;m',n')\right]J_x^D(m',n') \\ &\quad + \frac{Z_0}{jk_0\Delta x}\sum_{m'=1}^{M+1}\sum_{n'=1}^{N+1}\left[G_P(m,n;m',n') - G_P(m+1,n;m',n')\right] \\ &\quad \cdot\left\{\frac{1}{\Delta x}\left[J_x^D(m',n') - J_x^D(m'-1,n')\right]\right. \\ &\quad \left. + \frac{1}{\Delta y}\left[J_y^D(m',n') - J_y^D(m',n'-1)\right]\right\} - b_x(m,n)\end{aligned} \tag{16a}$$

$$\begin{aligned}\frac{1}{2}\frac{\partial F}{\partial J_y^D(m,n)} &= \sum_{m'=1}^{M+1}\sum_{n'=1}^{N}\left[R^y(m,n;m',n') + jk_0Z_0G_T^y(m,n;m',n')\right]J_y^D(m',n') \\ &\quad + \frac{Z_0}{jk_0\Delta y}\sum_{m'=1}^{M+1}\sum_{n'=1}^{N+1}\left[G_P(m,n;m',n') - G_P(m,n+1;m',n')\right] \\ &\quad \cdot\left\{\frac{1}{\Delta x}\left[J_x^D(m',n') - J_x^D(m'-1,n')\right]\right. \\ &\quad \left. + \frac{1}{\Delta y}\left[J_y^D(m',n') - J_y^D(m',n'-1)\right]\right\} - b_y(m,n)\end{aligned} \tag{16b}$$

in which

$$R^{x,y}(m,n;m',n') = \iint_S \eta\, T_{mn}^{x,y}\, T_{m'n'}^{x,y}\, dS \tag{17a}$$

$$G_T^{x,y}(m,n;m',n') = \iint_S T_{mn}^{x,y}\left[\iint_S T_{m'n'}^{x,y} G_0\, dS'\right] dS \tag{17b}$$

$$G_P(m,n;m'n') = \iint_S P_{mn}\left[\iint_S P_{m'n'} G_0\, dS'\right] dS \tag{17c}$$

$$b_{x,y}(m,n) = \iint_S T_{mn}^{x,y}\, E_{x,y}^{inc}\, dS \tag{17d}$$

The system implied by (15) is symmetric and can be solved via a direct method such as Gaussian elimination or LU decomposition. By resorting to an iterative solution, though, such as the CG and the BiCG method, substantial memory reduction can be achieved. Such a BiCG solution is discussed next, where the FFT is also employed for computing the required matrix-vector products, thus avoiding a need to explicitly generate the matrix.

III. BICONJUGATE GRADIENT FFT SOLUTION

The BiCG algorithm employed herein for the solution of $Ax = b$ in which A is symmetric, is as follows [11]:

Initialize the residual and search vectors with an initial guess x_0:

$$p_0 = r_0 = b - Ax_0$$

Iterate for $k = 0, 1, 2, \ldots$:

$$\begin{aligned} \alpha_k &= \frac{\langle r_k, r_k \rangle}{\langle p_k, Ap_k \rangle} \\ x_{k+1} &= x_k + \alpha_k p_k \\ r_{k+1} &= r_k - \alpha_k A p_k \\ \beta_k &= \frac{\langle r_{k+1}, r_{k+1} \rangle}{\langle r_k, r_k \rangle} \\ p_{k+1} &= r_{k+1} + \beta_k p_k \end{aligned}$$

Terminate when

$$\text{err} = \frac{||r_{k+1}||}{||b||} < \text{tolerance}$$

In the algorithm, $\langle x, y \rangle = x^T y$, where T denotes the transpose of the column.

It is observed that in the entire algorithm, A is only involved in the matrix-vector product Ap_k. For the problem considered here this product can be computed efficiently via the FFT without a need to generate the square matrix. To describe how we compute the Ap_k, let us first examine G_T^x defined in (17b). This can be explicitly written as

$$G_T^x(m,n;m',n') = \int_{(m-1)\Delta x}^{(m+1)\Delta x}\int_{(n-1)\Delta y}^{n\Delta y}\left(1-\frac{|x-m\Delta x|}{\Delta x}\right)\int_{(m'-1)\Delta x}^{(m'+1)\Delta x}\int_{(n'-1)\Delta y}^{n'\Delta y} \cdot\left(1-\frac{|x'-m'\Delta x|}{\Delta x}\right)G_0(x-x';y-y')dx'dy'dxdy \qquad (18)$$

and on letting $\tilde{x} = x - m\Delta x$ and $\tilde{y} = y - n\Delta y$, we have

$$G_T^x(m,n;m',n') = \int_{-\Delta x}^{\Delta x}\int_{-\Delta y}^{0}\left(1-\frac{|\tilde{x}|}{\Delta x}\right)\int_{-\Delta x}^{\Delta x}\int_{-\Delta y}^{0}\left(1-\frac{|\tilde{x}'|}{\Delta x}\right) \cdot G_0(\tilde{x}-\tilde{x}'+(m-m')\Delta x;\tilde{y}-\tilde{y}'+(n-n')\Delta y)d\tilde{x}'d\tilde{y}'d\tilde{x}d\tilde{y}\,. \qquad (19)$$

It is then clear that $G_T^x(m,n;m',n')$ is a function of the differences $(m-m')$ and $(n-n')$ and in a similar manner it can also be shown that $G_T^y(m,n;m',n')$ and $G_P(m,n;m',n')$ are functions of $(m-m')$ and $(n-n')$. Making use of this property, the system (15) can be more explicitly written as

$$\sum_{i=-1}^{+1} J_x^D(m+i,n)R^x(m,n;m+i,n) + jk_0Z_0J_x^D(m,n)\otimes G_T^x(m,n) + \frac{Z_0}{jk_0\Delta x}\left\{\frac{1}{\Delta x}\left[J_x^D(m,n)-J_x^D(m-1,n)\right]+\frac{1}{\Delta y}\left[J_y^D(m,n)-J_y^D(m,n-1)\right]\right\} \otimes\left[G_P(m,n)-G_P(m+1,n)\right] = b_x(m,n) \qquad (20a)$$

$$\sum_{i=-1}^{+1} J_y^D(m,n+i)R^y(m,n;m,n+i) + jk_0Z_0J_y^D(m,n)\otimes G_T^y(m,n) + \frac{Z_0}{jk_0\Delta y}\left\{\frac{1}{\Delta x}\left[J_x^D(m,n)-J_x^D(m-1,n)\right]+\frac{1}{\Delta y}\left[J_y^D(m,n)-J_y^D(m,n-1)\right]\right\} \otimes\left[G_P(m,n)-G_P(m,n+1)\right] = b_y(m,n) \qquad (20b)$$

where $G_T^{x,y}(m,n) = G_T^{x,y}(m,n;0,0)$, $G_P(m,n) = G_P(m,n;0,0)$ and the symbol $\otimes$ denotes convolution. The use of the FFT in evaluating the convolutions is now obvious. Specifically, let us define $\tilde{J}_x^{DP}(u,v)$ to be the discrete Fourier transform of the sample train $J_x^{DP}(m,n)$ defined as

$$J_x^{DP}(m,n) = \begin{cases} J_x^D(m,n) \text{ for } 1 \le m \le M, 1 \le n \le N+1 \\ 0 \text{ for } M+1 \le m \le 2M+2, N+2 \le n \le 2N+2 \end{cases} \tag{21}$$

and it is seen that $J_x^{DP}(m,n)$ is simply $J_x^D(m,n)$ padded with zeros so that its size is increased by a factor of two in each dimension. Similarly, we let $\tilde{G}_T^{xP}(u,v)$ to denote the discrete Fourier transform of $G_T^{xP}(m,n)$ where

$$G_T^{xP}(m,n) = \begin{cases} G_T^x(m,n) \text{ for } 1 \le m \le M+1, 1 \le n \le N+1 \\ G_T^x(2M+4-m,n) \text{ for } M+2 \le m \le 2M+2, 1 \le n \le N+1 \\ G_T^x(m,2N+4-n) \text{ for } 1 \le m \le M+1, N+2 \le n \le 2N+2 \\ G_T^x(2M+4-m,2N+4-n) \\ \quad \text{for } M+2 \le m \le 2M+2, N+2 \le n \le 2N+2 \end{cases} \tag{22}$$

and in an analogous manner we can define $\tilde{J}_y^{DP}(u,v)$, $\tilde{G}_T^{yP}(u,v)$ and $\tilde{G}_P^P(u,v)$ to be the discrete Fourier transforms of the corresponding quantities. With these definitions, (20) can be more compactly written as

$$\begin{aligned} &\sum_{i=-1}^{+1} J_x^D(m+i,n) R^x(m,n;m+i,n) + \mathcal{F}_D^{-1}\Big\{ jk_0 Z_0 \tilde{J}_x^{DP}(u,v) \tilde{G}_T^{xP}(u,v) \\ &+ \frac{Z_0}{jk_0 \Delta x}\left[\frac{1}{\Delta x}\left(1 - F_x^*(u)\right)\tilde{J}_x^{DP}(u,v) + \frac{1}{\Delta y}\left(1 - F_y^*(v)\right)\tilde{J}_y^{DP}(u,v)\right] \\ &\cdot \left(1 - F_x(u)\right)\tilde{G}_P^P(u,v)\Big\} = b_x(m,n) \end{aligned} \tag{23a}$$

$$\begin{aligned} &\sum_{i=-1}^{+1} J_y^D(m,n+i) R^y(m,n;m,n+i) + \mathcal{F}_D^{-1}\Big\{ jk_0 Z_0 \tilde{J}_y^{DP}(u,v) \tilde{G}_T^{yP}(u,v) \\ &+ \frac{Z_0}{jk_0 \Delta y}\left[\frac{1}{\Delta x}\left(1 - F_x^*(u)\right)\tilde{J}_x^{DP}(u,v) + \frac{1}{\Delta y}\left(1 - F_y^*(v)\right)\tilde{J}_y^{DP}(u,v)\right] \\ &\cdot \left(1 - F_y(v)\right)\tilde{G}_P^P(u,v)\Big\} = b_y(m,n) \end{aligned} \tag{23b}$$

in which the asterisk denotes complex conjugation,

$$F_x(u) = \exp\left(j\frac{2\pi u}{2M+2}\right) \tag{24a}$$

$$F_y(v) = \exp\left(j\frac{2\pi v}{2N+2}\right) \tag{24b}$$

and $\mathcal{F}_D^{-1}$ is the inverse discrete Fourier transform operator. Expressions (23) provide the most explicit definition of the matrix-vector product required in the BiCG algorithm. We remark that if the roof-top functions T^x_{mn} and T^y_{mn} are approximated by the pulse functions $P_{(m+1/2)n}$ and $P_{m(n+1/2)}$, respectively, and if mid-point integration is used in the evaluations of the first area integral in (17b) and (17c), it follows that (23) are identical to those derived in [7].

IV. NUMERICAL RESULTS

To demonstrate the efficiency of the proposed solution, we proceed to compare it with other CG-FFT solutions. In [8], a comprehensive comparison was presented for the plane wave scattering by a square $\lambda \times \lambda$ plate at normal incidence and this example was therefore chosen here for benchmarking purposes. This method's convergence characteristics are displayed in Figure 1 for two sampling rates and a comparison of the iteration count with four other methods [8] is given in Table 1. The characteristics of these four methods can be briefly described as follows:

- Method 1: Employs the analytical Fourier transform of the Green's function and pulse expansion functions. The spatial derivatives of the del operators are replaced by algebraic factors in the spectral domain. System is solved via the CG method (Generally, the BiCG method does not converge for this system)

- Method 2: Employs the analytical Fourier transform of the Green's function and piecewise sinusoidal expansion functions. The spatial derivatives of the del operators are replaced by algebraic factors in the spectral domain. System is solved via the CG method

- Method 3: Employs the discrete Fourier transform of the Green's function (integrated over the cell's area) and pulse expansion functions. The spatial derivatives of the del operators are approximated by finite differences. System is solved via the CG method

- Method 4: Employs the analytical Fourier transform of the Green's function and roof-top expansion functions. One of the del operators is transferred to the roof-top testing function. System is solved via the CG method

- This method: Employs the discrete Fourier transform of the Green's function (integrated over the cell's area). One del operator is transferred to the roof-top testing function and the other to the roof-top expansion function. System is solved via the BiCG method

From Table 1 it is clear that the proposed method attains convergence in at most half the number of iterations required by the best of the other four methods. Furthermore, considering that the BiCG solution requires only one matrix-vector product per iteration, whereas the CG solution requires two such products, the proposed method is even more efficient in terms of CPU time.

Table 1: Number of iterations required for convergence of the solution for a $\lambda \times \lambda$ plate illuminated with a plane wave at normal incidence

Discretization	err	Method 1	Method 2	Method 3	Method 4	This method
17×17 cells	10^{-2}	185	115	125	46	22
17×17 cells	10^{-3}	440	310	202	67	29
33×33 cells	10^{-2}	338	162	128	108	34
33×33 cells	10^{-3}	1480	800	1330	170	47

We next turn our attention to the accuracy of the proposed solution method. For such an assesment we considered the three plates depicted in Figure 2. Only one of these is rectangular whereas the perimeter of the other two includes circular and planar sections. For those non-rectangular plates the surface was still modeled as a collection of rectangular cells. The RCS of each plate was computed at 10 degrees off grazing for the horizontal polarization of incidence and the corresponding patterns are given in Figures 3-5. The criteria used for terminating the iteration were (a) the normalized root mean square error err (see algorithm) was less than 0.1, (b) the variation in RCS was less than 0.1 dB for 10 consecutive iterations, and (c) the number of iterations reached 200 (this occurred only in very few cases). The average number of iterations to reach convergence is given in Table 2. For the perfectly conducting case, the computed RCS patterns via this method are compared with a solution based on a traditional moment method implementation using triangular patches and linear basis to model the surface [12] and, as seen, the agreement is good.

Table 2: Average number of iterations required for convergence of the solution in Figures 3-5

Problem	No. of unknowns	Average no. of iterations
Fig. 3(a)	4366	79.4
Fig. 3(b)	4366	9.2
Fig. 4(a)	6056	117.1
Fig. 4(b)	6056	13.2
Fig. 5(a)	12034	145.6
Fig. 5(b)	12034	12.4

V. CONCLUSIONS

An efficient solution was presented for the scattering by planar conducting or resistive plates. A unique feature of the formulation is the transfer of the del operators contained in the dyadic Green's function to the expansion and testing functions. The resulting integral equation was then discretized by employing the roof-top functions for both expansion and testing. The resultant matrix was shown to be Toeplitz and was solved via the biconjugate gradient method in conjunction with the FFT. Numerical results were presented which demonstrated the efficiency, accuracy and capability of the solution over previous CG-FFT implementations.

REFERENCES

[1] T. K. Sarkar, E. Arvas, and S. 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. Antennas Propagat.*, vol. AP-34, pp. 635-640, May 1986.

[2] T. J. Peters and J. L. Volakis, "Application of a conjugate gradient FFT method to scattering from thin planar material plates," *IEEE Trans. Antennas Propagat.*, vol. AP-36, pp. 518-526, April 1988.

[3] K. Barkeshli and J. L. Volakis, "Improving the convergence rate of the conjugate gradient FFT method using subdomain basis functions," *IEEE Trans. Antennas Propagat.*, vol. AP-37, pp. 893-900, July 1989.

[4] N. N. Bojarski, "k-space formulation of the electromagnetic scattering problem," Air Force Avionics Lab. Technical Report AFAL-TR-71-75, March 1971.

[5] C. Y. Shen, K. J. Glover, M. I. Sancer, and A. D. Varvatsis, "The discrete Fourier transform method of solving differential-integral equations in scattering theory," *IEEE Trans. Antennas Propagat.*, vol. AP-37, pp. 1032-1041, Aug. 1989.

[6] K. Barkeshli and J. L. Volakis, "On the implementation of the conjugate gradient Fourier transform method for scattering by planar plates," *IEEE Antennas & Propagation Magazine*, vol. 32, pp. 20-26, April 1990.

[7] M. F. Cátedra, 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. Antennas Propagat.*, vol. AP-36, pp. 1744-1752, Dec. 1988.

[8] A. P. M. Zwamborn and P. M. van den Berg, "A weak form of the conjugate gradient FFT method for plate problems," *IEEE Trans. Antennas Propagat.*, vol. AP-39, pp. 224-228, Feb. 1991.

[9] J. R. Mautz and R. F. Harrington, "Electromagnetic transmission through a rectangular aperture in a perfectly conducting plane," Scientific Report No. 10, Contract F19628-73-C-0047 with Air Force Cambridge Research Laboratories, Hanscom A.F.B., Mass., Feb. 1976.

[10] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 409-419, May 1982.

[11] C. F. Smith, A. F. Peterson, and R. Mittra, "The biconjugate gradient method for electromagnetic scattering," *IEEE Trans. Antennas Propagat.*, vol. AP-38, pp. 938-940, June 1990.

[12] Courtesy of Ken Van-de-Houten and Vaughn Cable, Lockheed Advanced Development Co., Burbank, CA.

FIGURE CAPTIONS

Fig. 1 Convergence curves for a $\lambda \times \lambda$ plate with normal incidence. Solid line: 17×17 cells; Dashed line: 33×33 cells.

Fig. 2 Geometries of three plates for computations. (a) Target 1; (b) Target 2; (c) Target 3.

Fig. 3 Backscatter RCS for Target 1; $\mathbf{E}^{inc} = \hat{\phi}E$, $\theta^{inc} = 80^o$, 4366 unknowns. (a) $\eta = 0$; (b) $\eta = Z_0/4$. (9897 unknowns were used in generating the TRIMOM data.)

Fig. 4 Backscatter RCS for Target 2; $\mathbf{E}^{inc} = \hat{\phi}E$, $\theta^{inc} = 80^o$, 6056 unknowns. (a) $\eta = 0$; (b) $\eta = Z_0/4$. (10722 unknowns were used in generating the TRIMOM data.)

Fig. 5 Backscatter RCS for Target 3; $\mathbf{E}^{inc} = \hat{\phi}E$, $\theta^{inc} = 80^o$, 12034 unknowns. (a) $\eta = 0$; (b) $\eta = Z_0/4$. (6354 unknowns were used in generating the TRIMOM data.)

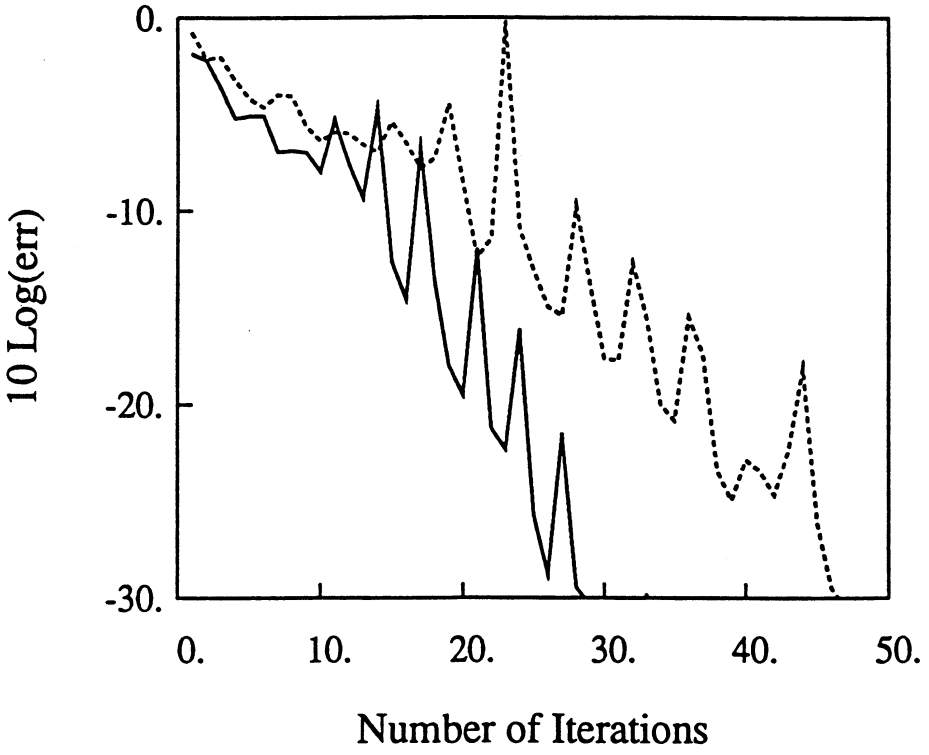
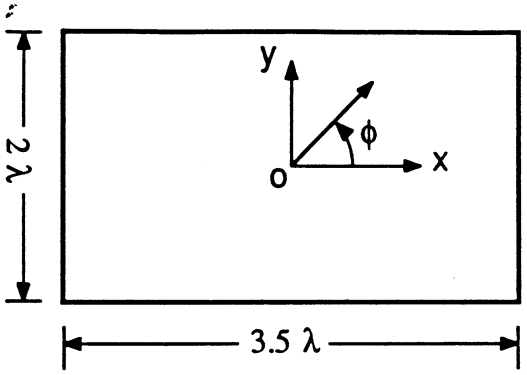
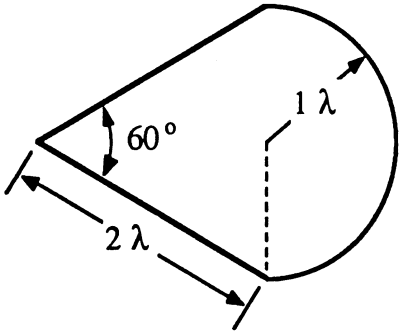


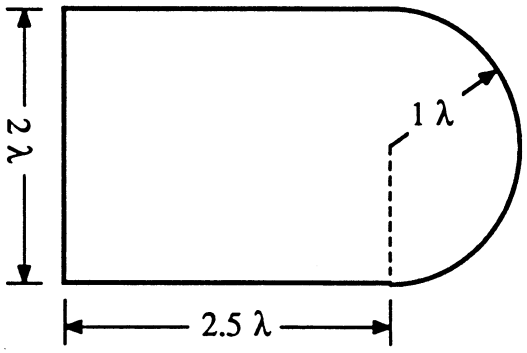
Fig. 1



(a)

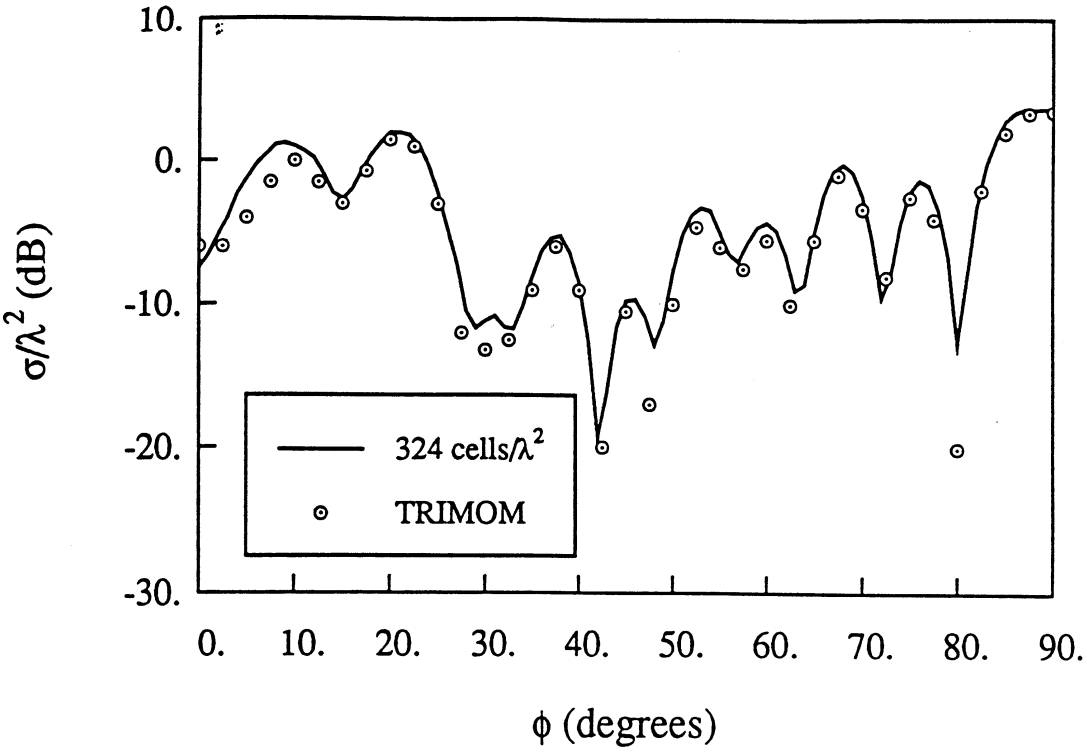


(b)

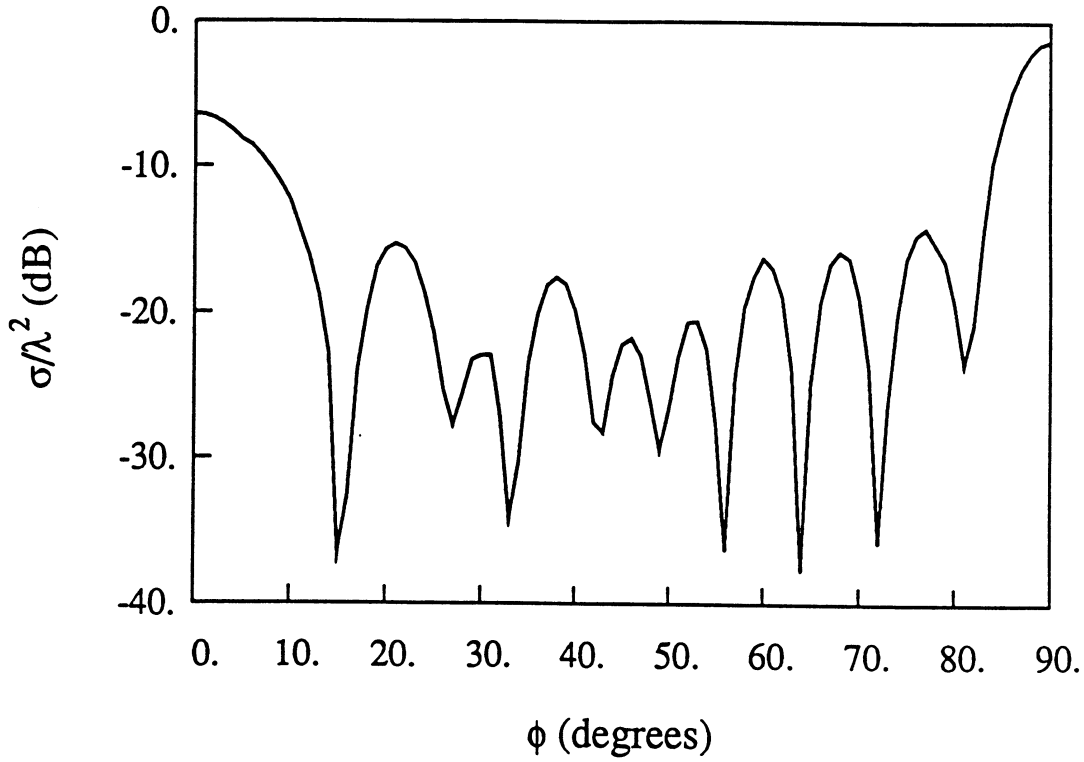


(c)

Fig. 2

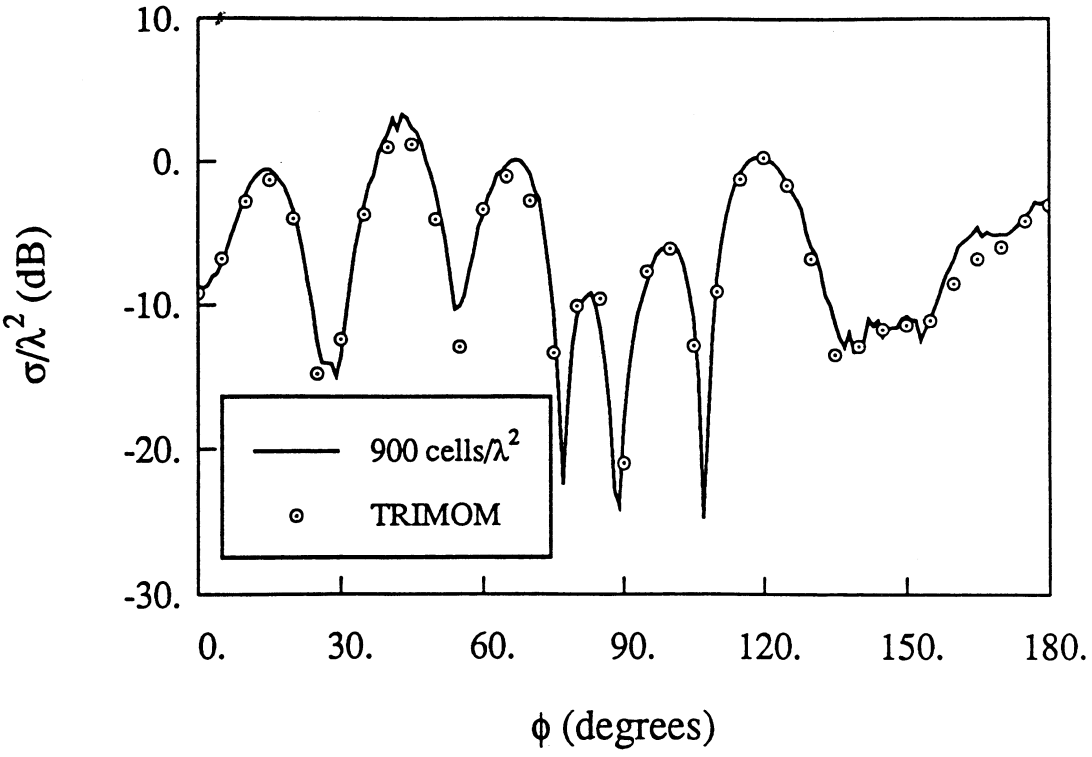

10.
0.
-10.
-20.
-30.
σ/λ² (dB)
324 cells/λ²
TRIMOM
0. 10. 20. 30. 40. 50. 60. 70. 80. 90.
ϕ (degrees)

(a)

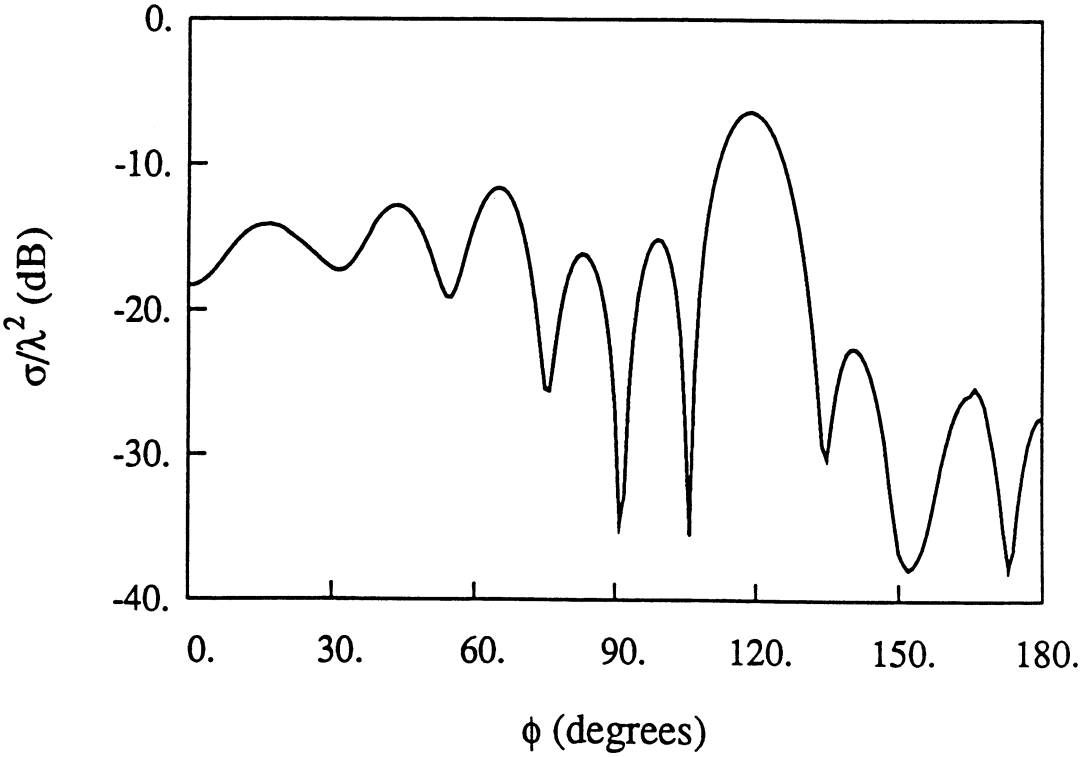

0.
-10.
-20.
-30.
-40.
σ/λ² (dB)
0. 10. 20. 30. 40. 50. 60. 70. 80. 90.
ϕ (degrees)

(b)

Fig. 3

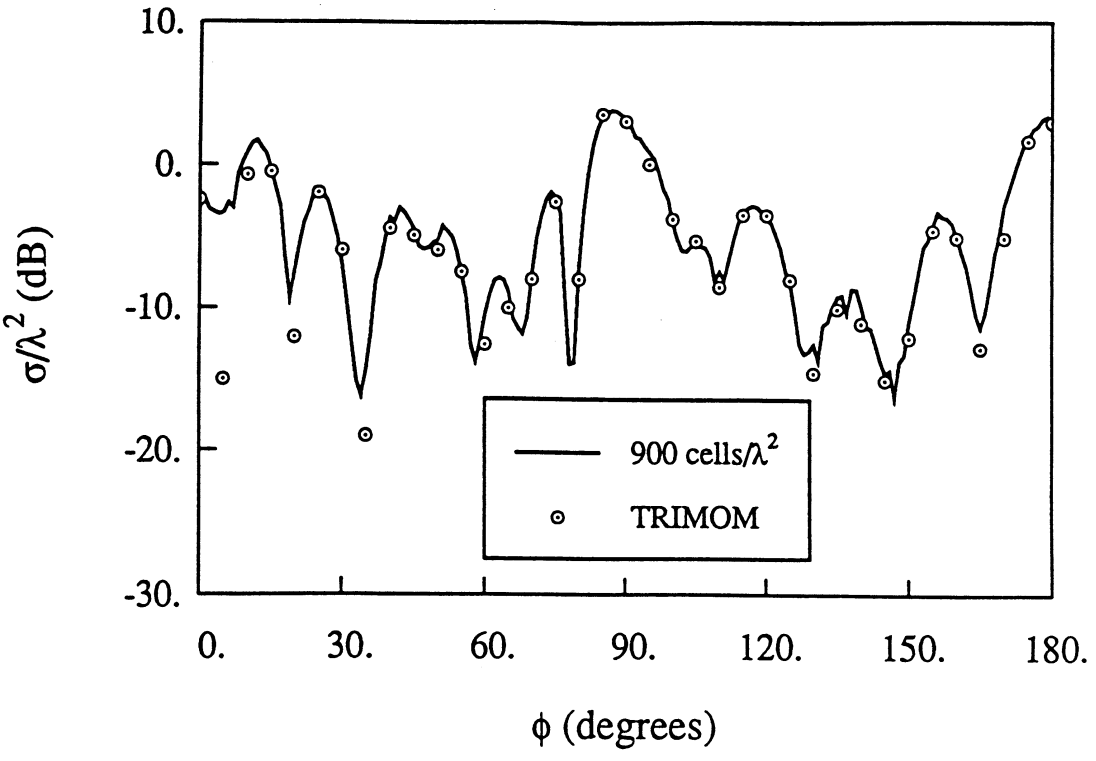

10.
0.
-10.
-20.
-30.
σ/λ² (dB)
900 cells/λ²
TRIMOM
0.
30.
60.
90.
120.
150.
180.
ϕ (degrees)

(a)

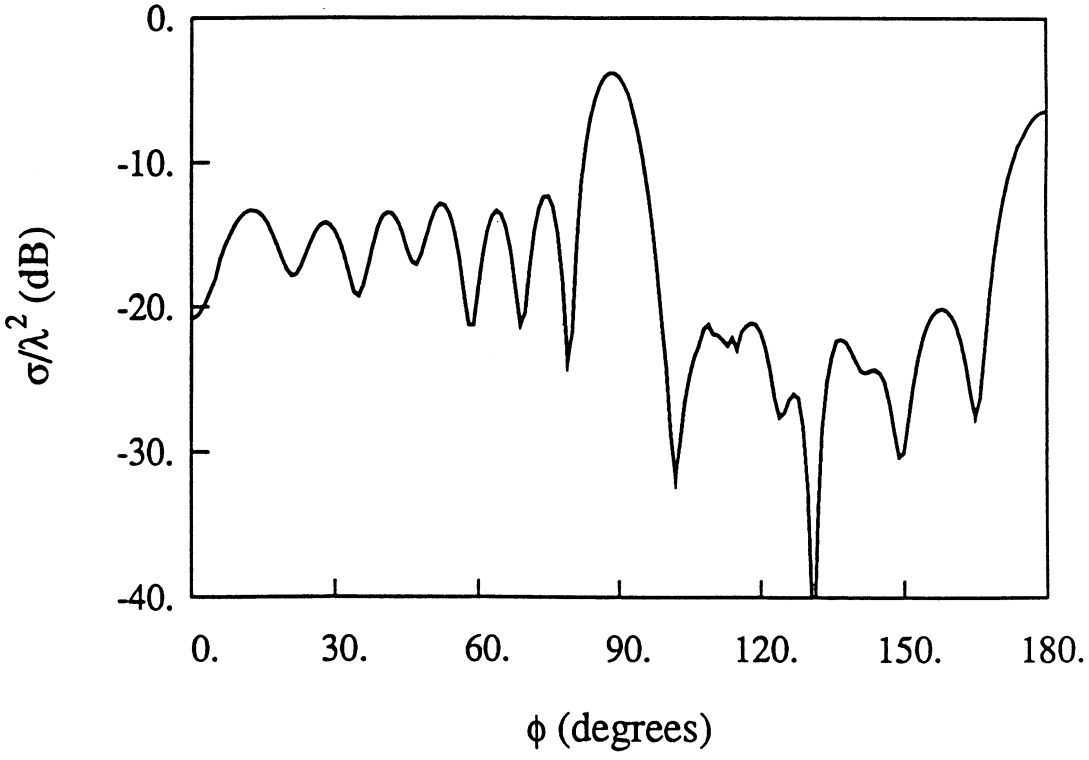

0.
-10.
-20.
-30.
-40.
σ/λ² (dB)
0.
30.
60.
90.
120.
150.
180.
ϕ (degrees)

(b)

Fig. 4


10.
0.
-10.
-20.
-30.
σ/λ² (dB)
900 cells/λ²
TRIMOM
0.
30.
60.
90.
120.
150.
180.
ϕ (degrees)

(a)


0.
-10.
-20.
-30.
-40.
σ/λ² (dB)
0.
30.
60.
90.
120.
150.
180.
ϕ (degrees)

(b)

Fig. 5

```
                      PROGRAM PLTBCG
C*********************************************************************C
C     THIS PROGRAM COMPUTES ELECTROMAGNETIC SCATTERING BY A CONDUCTING C
C     PLATE WITH ZERO THICKNESS OR AN APERTURE IN AN INFINITESIMALLY   C
C     THIN CONDUCTING PLANE. THE PROBLEM IS FORMULATED USING INTEGRAL  C
C     EQUATION. THE RESULTING SYSTEM OF EQUATIONS IS SOLVED USING      C
C     THE BICG-FFT TECHNIQUE. ONCE THE ELECTRIC CURRENTS ARE COMPUTED, C
C     THE PROGRAM COMPUTES THE FAR FIELD PATTERN IN RCS.               C
C                                                                      C
C     JIANMING JIN                                                     C
C     RADIATION LABORATORY                                             C
C     DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE        C
C     THE UNIVERSITY OF MICHIGAN                                       C
C     ANN ARBOR, MICHIGAN 48109                                        C
C*********************************************************************C
C                                                                      C
C     INPUT DATA DESCRIPTION:                                          C
C                                                                      C
C       XL - PLATE DIMENSION ALONG THE X AXIS (REAL)                   C
C       YL - PLATE DIMENSION ALONG THE Y AXIS (REAL)                   C
C       NX - NUMBER OF DISCRETE POINTS ALONG THE X AXIS (INTEGER)      C
C       NY - NUMBER OF DISCRETE POINTS ALONG THE Y AXIS (INTEGER)      C
C       WAVE - WAVELENGTH (REAL)                                       C
C       KODE - INTEGER                                                 C
C            KODE=0 - COMPUTE BISTATIC SCATTERING                      C
C            KODE=1 - COMPUTE BACKSCATTERING                           C
C       PHIINC - INCIDENCE ANGLE OF PHI (REAL)                         C
C       THEINC - INCIDENCE ANGLE OF THETA (REAL)                       C
C       PHIBEG - BEGINNING ANGLE OF PHI (REAL)                         C
C       PHIFIN - FINAL ANGLE OF PHI (REAL)                             C
C       PHISTP - INCREMENT ANGLE OF PHI (REAL)                         C
C       THEBEG - BEGINNING ANGLE OF THETA (REAL)                       C
C       THEFIN - FINAL ANGLE OF THETA (REAL)                           C
C       THESTP - INCREMENT ANGLE OF THETA (REAL)                       C
C*********************************************************************C
C
C.....INPUT DATA
      PRINT*,'INPUT DIMENSION OF THE PLATE: XL,YL (in cm)'
      print*,'If you are running test case #1: set XL=3, YL=2'
      print*,'                             #2: set XL=4, YL=2'
      print*,'                             #3: set XL=3.5, YL=2'
      print*,'                             #4: set XL=2.5, YL=2'
      READ*,XL,YL
      PRINT*,XL,YL
      PRINT*,'INPUT NUMBER OF POINTS ALONG EACH SIDES: NX,NY'
      READ*,NX,NY
      PRINT*,NX,NY
      NXFT=2**(INT(ALOG(2.*NX)/ALOG(2.))+1)
      NYFT=2**(INT(ALOG(2.*NY)/ALOG(2.))+1)
      PRINT*, 'FFT PAD SIZE:', NXFT, NYFT
      PRINT*,'INPUT WAVELENGTH: WAVE (in cm)'
      READ*,WAVE
      PRINT*,WAVE
```

```
      PRINT*,'INPUT KODE: 0 FOR BISTATIC, 1 FOR BACKSCATTERING'
      READ*,KODE
      PRINT*,KODE
      PRINT*,'INPUT BEGINNING, FINAL AND INCREMENT ANGLES: PHIBEG,PHIFIN
     &,PHISTP'
      READ*,PHIBEG,PHIFIN,PHISTP
      PRINT*,PHIBEG,PHIFIN,PHISTP
      PRINT*,'INPUT BEGINNING, FINAL AND INCREMENT ANGLES: THEBEG,THEFIN
     &,THESTP'
      READ*,THEBEG,THEFIN,THESTP
      PRINT*,THEBEG,THEFIN,THESTP
      IF(KODE.EQ.1) GOTO 1
      PRINT*,'INPUT INCIDENCE ANGLE IN DEGREE: PHIINC,THEINC'
      READ*,PHIINC,THEINC
      PRINT*,PHIINC,THEINC
  1   CONTINUE
      PRINT*,'INPUT PLOARIZATION ANGLE IN DEGREE: ALPHA'
      PRINT*,'0 FOR TE (Ez=0) CASE; 90 FOR TM (Hz=0) CASE'
      READ*,ALPHA
      PRINT*,ALPHA
      PRINT*,'INPUT TOLERANCE AND MAXIMUM ITERATIONS'
      READ*,TOL, ITMAX
      PRINT*,TOL, ITMAX
      PRINT*,'PRINT OUT CURRENT: 1 FOR YES, 0 FOR NO'
      READ*,IPRC
      PRINT*,IPRC
C
C.....CALL SUBROUTINE PLATE TO SOLVE THE PROBLEM
      CALL PLATE(XL,YL,NX,NY,NXFT,NYFT,WAVE,KODE,PHIINC,THEINC,ALPHA,
     &     PHIBEG,PHIFIN,PHISTP,THEBEG,THEFIN,THESTP,TOL,ITMAX,IPRC)
      STOP
      END



      SUBROUTINE PLATE(XL,YL,NX,NY,NXFT,NYFT,WAVE,KODE,PHIINC,THEINC,
     &   ALPHA,PHIBEG,PHIFIN,PHISTP,THEBEG,THEFIN,THESTP,TOL,ITMAX,IPRC)
C*********************************************************************C
C     THIS PROGRAM COMPUTES THE ELECTRIC CURRENTS ON A THIN CONDUCTING C
C     PLATE OR ELECTRIC FIELDS ON AN APERTURE IN A THIN CONDUCTING     C
C     PLANE. THE PROBLEM IS FORMULATED USING INTEGRAL EQUATION.        C
C     THE RESULTING SYSTEM OF EQUATIONS IS SOLVED USING THE BICG-FFT   C
C     TECHNIQUE. ONCE THE ELECTRIC CURRENTS OR FIELDS ARE COMPUTED,    C
C     THE PROGRAM COMPUTES THE FAR FIELD PATTERN.                      C
C*********************************************************************C
      PARAMETER (NMAX=25000,NSMAX=20000,NFT=70000)
C*********************************************************************C
C     NMAX - MAXIMUM NUMBER OF EDGES                                   C
C     NSMAX - MAXIMUM NUMBER OF CELLS                                  C
C     NFT - MAXIMUM FFT PAD                                            C
C*********************************************************************C
      COMPLEX AX(NMAX),AY(NMAX),EX(NMAX),EY(NMAX),
     &        PX(NMAX),PY(NMAX),RX(NMAX),RY(NMAX),
     &        BX(NMAX),BY(NMAX),EPS,MU,B,T,C1,CNORM,ZL
```

```
      COMPLEX AXX(2,2,NMAX),AXY(2,2,NMAX),AYX(2,2,NMAX),
     &        AYY(2,2,NMAX),GXX(NFT),GYX(NFT),GYY(NFT)
C*****************************************************************C
      INTEGER NEX(NMAX),NEY(NMAX),COUNT
      REAL KO
      PI=3.141592654
      KO=2.*PI/WAVE
      DX=XL/FLOAT(NX-1)
      DY=YL/FLOAT(NY-1)
      NXT=(NX-1)*NY
      NYT=NX*(NY-1)
      NT3=NXT+NYT
      PRINT*,'DX=',DX,'  DY=',DY
C
C
C.....CALL DIRIC TO IDENTIFY THE EDGES WHOSE NORMAL ELECTRIC CURRENT OR
C ... TANGENTIAL ELECTRIC FIELDS SHOULD BE SET TO ZERO
      CALL DIRIC(NX,NY,LEX,LEY,NEX,NEY)
      LET=LEX+LEY
      NUM=NT3-LET
      PRINT*,'NUMBER OF UNKNOWNS =', NUM
C
      NXP=NX-1
      NYP=NY-1
C
C.....CALL GREENF TO COMPUTE THE DISCRETIZED SURFACE INTEGRAL
      CALL GREENF(GXX,GYX,GYY,NX,NY,DX,DY,KO,AXX,AYX,AXY,AYY,
     &            NXFT,NYFT)

C
      IF(KODE.EQ.1) THEN
      PHI=PHIBEG
      THETA=THEBEG
      ELSE
      THETA=THEINC
      PHI=PHIINC
      END IF
C
 1111 CONTINUE

C.....INITIALIZE ELECTRIC FIELDS
      DO 201 I=1,NXT
 201     EX(I)=CMPLX(0.0,0.0)
      DO 202 I=1,NYT
 202     EY(I)=CMPLX(0.0,0.0)

C.....COMPUTE THE KNOWN VECTORS ON THE UPPER APERTURE
      CALL FDINC(BX,BY,NX,NY,DX,DY,KO,THETA,PHI,ALPHA)
C*****************************************************************C
C.....OUTPUT DATA DESCRIPTION:                                    C
C     BX - COMPLEX VECTOR   KNOWN EXCITATION                      C
C     BY - COMPLEX VECTOR   KNOWN EXCITATION                      C
C*****************************************************************C
```

```
C.....CALCULATES THE EUCLIDIAN NORM OF KNOWN VECTOR
        CALL BNDRY(BX,NEX,LEX)
        CALL BNDRY(BY,NEY,LEY)
        VNRM=VNORM(BX,NXT)+VNORM(BY,NYT)

        RSS=0.0
        ITER=0
        RCS=0.
        COUNT=0

C.....COMPUTE THE RESIDUAL FOR THE INTIAL FIELDS
        DO 204 I=1,NXT
 204        RX(I)=BX(I)
        DO 205 I=1,NYT
 205        RY(I)=BY(I)


C.....COMPUTE THE SEARCH VECTOR
        DO 207 I=1,NXT
 207        PX(I)=RX(I)
        DO 208 I=1,NYT
 208        PY(I)=RY(I)


C.....BEGIN ITERATION TO FIND SOLUTION
 30     CONTINUE


C.....COMPUTE COMPLEX VECTOR [A]*P
      DO 11 I=1,NXT
  11  AX(I)=CMPLX(0.0,0.0)
      DO 12 I=1,NYT
  12  AY(I)=CMPLX(0.0,0.0)

      CALL ADDAX(AX,AY,PX,PY,GXX,GYX,GYY,NX,NY,NXFT,NYFT,AXX,AYY)

C.....ENFORCE BOUNDARY CONDITION
        CALL BNDRY(AX,NEX,LEX)
        CALL BNDRY(AY,NEY,LEY)


        C1=CNORM(RX,RX,NXT)+CNORM(RY,RY,NYT)

        T = C1/(CNORM(PX,AX,NXT)+CNORM(PY,AY,NYT))

      DO 210 I=1,NXT
         EX(I)=EX(I)+T*PX(I)
 210     RX(I)=RX(I)-T*AX(I)
      DO 211 I=1,NYT
         EY(I)=EY(I)+T*PY(I)
 211     RY(I)=RY(I)-T*AY(I)

C.....CALCULATES THE EUCLIDIAN NORM OF RESIDUAL VECTOR
        B=(CNORM(RX,RX,NXT)+CNORM(RY,RY,NYT))/C1
```

```
C.....COMPUTE THE SEARCH VECTOR
        DO 213 I=1,NXT
 213       PX(I)=RX(I)+B*PX(I)
        DO 214 I=1,NYT
 214       PY(I)=RY(I)+B*PY(I)

        ITER = ITER+1

        VNRMR=VNORM(RX,NXT)+VNORM(RY,NYT)
        RSS = SQRT(VNRMR/VNRM)

C.....IF NOT TO MONITOR THE ITERATIVE PROCEDURE, COMMENT THE NEXT LINE
        PRINT 140,ITER,RSS,CABS(T)

        CALL SCATER(EX,EY,NMAX,NXP,NYP,DX,DY,KO,THETA,PHI,SIG)
C.....SET A CHECK ON RCS
       IF(ABS(SIG-RCS).LT.0.1) THEN
       COUNT=COUNT+1
       ELSE
       RCS=SIG
       COUNT=0
       END IF

C.....IF THE CRITERIA IS SATISFIED, THEN
        IF((RSS.LE.TOL).OR.(COUNT.EQ.10)) THEN
          PRINT 120

C.....CALL SCATER TO COMPUTE THE RADAR CROSS SECTION

        IF(KODE.EQ.1) THEN
      PRINT*,     '       THETA      PHI          SIG(DB)      SIGTHETA
     &SIGPHI'
        CALL SCATER(EX,EY,NMAX,NXP,NYP,DX,DY,KO,THETA,PHI,SIG)
C
        ELSE
      PRINT*,     '       THETA      PHI          SIG(DB)      SIGTHETA
     &SIGPHI'
      IIF=INT((PHIFIN-PHIBEG)/PHISTP)+1
      DO 1002 II=1,IIF
      PHI=PHIBEG+FLOAT(II-1)*PHISTP
      JJF=INT((THEFIN-THEBEG)/THESTP)+1
      DO 1002 JJ=1,JJF
      THETA=THEBEG+FLOAT(JJ-1)*THESTP
      CALL SCATER(EX,EY,NMAX,NXP,NYP,DX,DY,KO,THETA,PHI,SIG)
 1002 CONTINUE

C.....PRINT OUT THE FIELD AND CURRENT DATA
        IF(IPRC.GT.0) THEN
        PRINT 150
        DO 216 I=1,NXT
        IF(EX(I).EQ.(0.,0.)) PHX=0.
        IF(EX(I).NE.(0.,0.)) PHX=ATAN2(AIMAG(EX(I)),REAL(EX(I)))*180./PI
        PRINT 90,I,CABS(EX(I)),PHX
```

```
 216    CONTINUE
        PRINT 151
        DO 217 I=1,NYT
        IF(EY(I).EQ.(0.,0.)) PHY=0.
        IF(EY(I).NE.(0.,0.)) PHY=ATAN2(AIMAG(EY(I)),REAL(EY(I)))*180./PI
        PRINT 90,I,CABS(EY(I)),PHY
 217    CONTINUE
 90     FORMAT(I5,5X,2G13.4,2G13.4,2G13.4)
        ELSE
        END IF

       END IF
C
C.....IF THE CRITERIA IS NOT SATISFIED, THEN
        ELSE
          IF(ITER.EQ.ITMAX) THEN
            PRINT 130

          ELSE
            GO TO 30
          END IF
        END IF

      IF(KODE.EQ.1) THEN
        THETA=THETA+THESTP
        IF(THETA.LE.THEFIN) GOTO 1111
        PHI=PHI+PHISTP
        THETA=THEBEG
        IF(PHI.LE.PHIFIN) GOTO 1111
      ENDIF
C
 120    FORMAT('CONVERGENCE ACHIEVED, WOW!!'//)
 130    FORMAT('ITMAX EXCEEDED; NO CONVERGENCE.')
 140    FORMAT('ITER=',I4,'  RSS=',G14.4,'  |T|=',G14.4)
 150    FORMAT(/'   THE DISCRETE ELECTRIC CURRENTS'/
     &       '  SIDE      MAG(JY)     PHASE(JY)')
 151    FORMAT(/'  SIDE      MAG(JX)     PHASE(JX)')
        RETURN
        END



      SUBROUTINE DIRIC(NX,NY,LEX,LEY,NEX,NEY)
C*********************************************************************
C     THIS SUBROUTINE IDENTIFIES THE EDGES WHOSE TANGENTIAL ELECTRIC  *
C     FIELDS SHOULD SET TO ZERO                                       *
C     NX - NUMBER OF POINTS IN THE X-DIRECTION                        *
C     NY - NUMBER OF POINTS IN THE Y-DIRECTION                        *
C     LEX - NUMBER OF SIDES WITH EX=0                                 *
C     LEY - NUMBER OF SIDES WITH EY=0                                 *
C     NEX - VECTOR INDICATING THE SIDES WITH EX=0                     *
C     NEY - VECTOR INDICATING THE SIDES WITH EY=0                     *
C*********************************************************************
      INTEGER NEX(*),NEY(*)
```

```
      NXY=NX*NY
      LEX=2*(NX-1)
      LEY=2*(NY-1)
      LX=0
      LY=0
      NX1=NX-1
      NY1=NY-1
      DO 402 I=1,NX1
      LX=LX+1
 402  NEX(LX)=I
      DO 403 I=1,NX1
      LX=LX+1
 403  NEX(LX)=NX1*NY-NX1+I
      DO 404 I=1,NY1
      LY=LY+1
      NEY(LY)=NX*(I-1)+1
      LY=LY+1
 404  NEY(LY)=NX*I
      IF(LX.NE.LEX) GOTO 10
      IF(LY.NE.LEY) GOTO 10

C   ---------------------------------------------------------------------
C
C...TARGET GEOMETRY SPECIFICATION
C
        WRITE(*,*)
        WRITE(*,*)'**SCATTERING FROM THIN CONDUCTING PLATES**'
1       WRITE(*,*)
        WRITE(*,*)'SPECIFY THE TARGET:'
        WRITE(*,*)
        WRITE(*,*)'1: RECTANGULAR PLATE'
        WRITE(*,*)'2: CIRCULAR DISK'
        WRITE(*,*)'3: TRIANGULAR PLATE'
        WRITE(*,*)'4: POLYGONAL PLATE'
        WRITE(*,*)'5: TEST TARGETS'
        WRITE(*,*)
        READ*,NTARG
        IF(NTARG.EQ.1) RETURN
        IF((NTARG.GE.1).AND.(NTARG.LE.4))THEN
          WRITE(*,*)'ENCLOSURE CENTER AND LIMITS: X0,Y0,XL,YL'
          READ*,X0,Y0,XL,YL
        ELSEIF(NTARG.EQ.5)THEN
          WRITE(*,*)'TEST CASES'
          WRITE(*,*)'6: TARGET # 1'
          WRITE(*,*)'7: TARGET # 2'
          WRITE(*,*)'8: TARGET # 3'
          WRITE(*,*)'9: TARGET # 4'
          READ*,NTARG
        ENDIF
        IF((NTARG.LT.0).OR.(NTARG.GT.9))THEN
          WRITE(*,*)'WRONG TARGET! TRY AGAIN...'
          GO TO 1
        ENDIF
        CALL GEOMTR(NX1,NY1,NTARG,X0,Y0,XL,YL,LX,LY,LEX,LEY,NEX,NEY)
```

```
      RETURN
   10 WRITE(*,20)
   20 FORMAT('THERE IS AN ERROR IN SUBROUTINE DIRIC')
      END

C   ======================================================================
C   *                     TARGET GEOMETRY SPECIFICATIONS                 *
C   ======================================================================
      SUBROUTINE GEOMTR(MX,MY,NTARG,X0,Y0,XL,YL,LX,LY,LEX,LEY,NEX,NEY)
      INTEGER NEX(*),NEY(*)
      REAL XC(10),YC(10)
      CHARACTER*1 ITARG(256,256)
      LOGICAL TAG,TRITAG
      TAG=.FALSE.
      IF(NTARG.EQ.1)THEN
C
C...RECTANGULAR PLATE
C
      ELSEIF(NTARG.EQ.2)THEN
C
C...CIRCULAR DISK
C
        WRITE(*,*)'ENTER THE RADIUS:'
        READ*,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(*,*)'NUMBER OF CORNERS:'
          READ *,NC
        ENDIF
        NT=NC-2
        WRITE(*,*)'ENTER THE COORDINATES OF EACH CORNER SEQUENTIALLY:'
        DO 30 I=1,NC
           WRITE(*,*)'CORNER #',I
           READ*,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
        NX=MX
        NY=MY
        DX=XL/MX
        DY=YL/MY
        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
                 TAG=.FALSE.
               ELSE

       LEX=LEX+1
       LX=LX+1
       NEX(LX)=MX*(J-1)+I
       LEX=LEX+1
       LX=LX+1
       NEX(LX)=MX*J+I
       LEY=LEY+1
       LY=LY+1
       NEY(LY)=(MX+1)*(J-1)+I
       LEY=LEY+1
       LY=LY+1
       NEY(LY)=(MX+1)*(J-1)+I+1

               ENDIF
50          CONTINUE
60        CONTINUE
C
C    ------------------------------------------------------------------
C
C...TOTAL NUMBER OF 'NONZERO' ELEMENTS
C
         WRITE(*,*)
         WRITE(*,*)'XI= ',XSTRT+0.5*DX,'  XF= ',X,'  MX= ',MX,'  NX= ',NX
         WRITE(*,*)'YI= ',YSTRT+0.5*DY,'  YF= ',Y,'  MY= ',MY,'  NY= ',NY
         WRITE(*,*)
         MG=INDX
         MGI=2*MG
         WRITE(*,*)'TOTAL NUMBER OF UNKNOWNS: 2 *',MG
         WRITE(*,*)'GEOMETRY DISPLAY? 1)YES 2)NO'
         READ*,IRES
         IF(IRES.EQ.1)THEN
           DO 70 J=NY,1,-1
              WRITE(*,80)(ITARG(I,J),I=1,NX)
70         CONTINUE
80         FORMAT(255(1X,A1))
         ENDIF
         WRITE(*,*)
         WRITE(*,*)'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



      SUBROUTINE GREENF(GXX,GYX,GYY,NX,NY,DX,DY,KO,AXX,AYX,AXY,AYY,
     &                  NXFT,NYFT)
C*********************************************************************
C     THIS SUBROUTINE FIRST GENERATES THE DISCRETIZED GREEN'S        *
C     FUNCTION AND THEN TRANSFORMS IT USING FFT                      *
C     DX  -  INCREMENT IN THE X DIRECTION                            *
C     DY  -  INCREMENT IN THE Y DIRECTION                            *
C     G   -  AN ARRAY WITH OUTPUT AS TRANSFORMED GREEN'S FUNCTION    *
C*********************************************************************
      COMPLEX AXX(2,2,*),AXY(2,2,*),AYX(2,2,*),AYY(2,2,*),G1
      COMPLEX GXX(*),GYX(*),GYY(*),ETA
      REAL KO,KR,XI(2),X1(3),
     &     PNX(2),PNY(2),NEX(2),NEY(2),NEXP(2),NEYP(2),IL1(2)
      INTEGER NN(2)
      DATA X1/-0.333333333,0.,0.33333333333/
      DATA XI/-0.5773502692,0.5773502692/
      DATA IL1/2,1/
      PI=3.141592654
      TPI=2.*PI
      WAVE=TPI/KO
      SKO=KO*KO
      AREA=DX*DY
      RLIM=0.01*SQRT(DX*DX+DY*DY)
C
      NXP=NX-1
      NYP=NY-1
      NX1=NX+1
      NY1=NY+1
      NXY1=NX1*NY1
      DO 601 L=1,NXY1
      DO 601 I=1,2
      DO 601 J=1,2
      AXX(I,J,L)=0.
      AXY(I,J,L)=0.
      AYX(I,J,L)=0.
 601  AYY(I,J,L)=0.
C
C.....START HERE TO GENERATE THE MATRIX ELEMENTS
C
```

```
      YC=DY
      XC=DX
C
        PNX(1)=-1./DX
        PNX(2)=1./DX
        PNY(1)=-1./DY
        PNY(2)=1./DY
C
      DO 607 JP=1,NY1
      YPC=FLOAT(JP-1)*DY
      DO 607 IP=1,NX1
      XPC=FLOAT(IP-1)*DX
      KP=(JP-1)*NX1+IP
C
      IF((JP.LE.4).AND.(IP.LE.4)) THEN
C
        DO 605 JJ=1,3
        Y=YC+X1(JJ)*DY
        NEX(1)=0.5-X1(JJ)
        NEX(2)=0.5+X1(JJ)
        DO 605 II=1,3
        X=XC+X1(II)*DX
        NEY(1)=0.5-X1(II)
        NEY(2)=0.5+X1(II)
C
          DO 604 JJP=1,3
          YP=YPC+X1(JJP)*DY
          NEXP(1)=0.5-X1(JJP)
          NEXP(2)=0.5+X1(JJP)
          DO 604 IIP=1,3
          XP=XPC+X1(IIP)*DX
          NEYP(1)=0.5-X1(IIP)
          NEYP(2)=0.5+X1(IIP)
C
      XX=X-XP
      YY=Y-YP
      R=SQRT(XX*XX+YY*YY)
      IF(R.LE.RLIM) THEN
C.....USING ANALYTICAL INTEGRATION FOR 1/R
      XL=X-DX/6.
      XU=X+DX/6.
      YL=Y-DY/6.
      YU=Y+DY/6.
      XLP=XP-DX/6.
      XUP=XP+DX/6.
      YLP=YP-DY/6.
      YUP=YP+DY/6.
      A1=FINT(XUP,YUP,XU,YU)
      A2=FINT(XLP,YLP,XU,YU)
      A3=FINT(XLP,YUP,XU,YU)
      A4=FINT(XUP,YLP,XU,YU)
      A5=FINT(XUP,YUP,XL,YL)
      A6=FINT(XLP,YLP,XL,YL)
      A7=FINT(XLP,YUP,XL,YL)
```

```
      A8=FINT(XUP,YLP,XL,YL)
      A9=FINT(XUP,YUP,XL,YU)
      A10=FINT(XLP,YLP,XL,YU)
      A11=FINT(XLP,YUP,XL,YU)
      A12=FINT(XUP,YLP,XL,YU)
      A13=FINT(XUP,YUP,XU,YL)
      A14=FINT(XLP,YLP,XU,YL)
      A15=FINT(XLP,YUP,XU,YL)
      A16=FINT(XUP,YLP,XU,YL)
      G1=A1+A2-A3-A4+A5+A6-A7-A8-A9-A10+A11+A12-A13-A14+A15+A16
      G1=G1/(AREA/9.)
C.....USING GAUSSIAN FORMULA (2X2) FOR INTEGRATION
      DO 602 J4=1,2
      Y2=YY+XI(J4)*DY/6.
      DO 602 I4=1,2
      X2=XX+XI(I4)*DX/6.
      R=SQRT(X2*X2+Y2*Y2)
      KR=KO*R
      SI=SIN(KR)
      CO=COS(KR)
 602  G1=G1+(CMPLX(CO,-SI)-1.)/R*(AREA/36.)
      G1=G1/TPI
      ELSE
C.....USING ONE POINT FORMULA FOR INTEGRATION
      R=SQRT(XX*XX+YY*YY)
      KR=KO*R
      SI=SIN(KR)
      CO=COS(KR)
      G1=CMPLX(CO,-SI)/(TPI*R)*(AREA/9.)
      END IF
      G1=-G1
C
      DO 603 I1=1,2
      DO 603 J1=1,2
      AXX(I1,J1,KP)=AXX(I1,J1,KP)+(NEX(I1)*NEXP(J1)*SKO-
     &                PNY(I1)*PNY(J1))*G1*AREA/9.
      AXY(I1,J1,KP)=AXY(I1,J1,KP)+PNY(I1)*PNX(J1)*G1*AREA/9.
      AYX(I1,J1,KP)=AYX(I1,J1,KP)+PNX(I1)*PNY(J1)*G1*AREA/9.
      AYY(I1,J1,KP)=AYY(I1,J1,KP)+(NEY(I1)*NEYP(J1)*SKO-
     &                PNX(I1)*PNX(J1))*G1*AREA/9.
 603  CONTINUE
C
 604  CONTINUE
 605  CONTINUE
C
      ELSE
C
      XX=XC-XPC
      YY=XC-YPC
C.....USING ONE POINT FORMULA FOR INTEGRATION
      R=SQRT(XX*XX+YY*YY)
      KR=KO*R
      SI=SIN(KR)
      CO=COS(KR)
```

```
      G1=CMPLX(CO,-SI)/(TPI*R)*AREA
      G1=-G1
C
      DO 606 I1=1,2
      DO 606 J1=1,2
      AXX(I1,J1,KP)=AXX(I1,J1,KP)+(0.25*SKO-
     &                PNY(I1)*PNY(J1))*AREA*G1
      AXY(I1,J1,KP)=AXY(I1,J1,KP)+PNY(I1)*PNX(J1)*G1*AREA
      AYX(I1,J1,KP)=AYX(I1,J1,KP)+PNX(I1)*PNY(J1)*G1*AREA
      AYY(I1,J1,KP)=AYY(I1,J1,KP)+(0.25*SKO-
     &                PNX(I1)*PNX(J1))*AREA*G1
 606  CONTINUE
C
      END IF
C
 607  CONTINUE
C
C
C.....GENERATE GREEN'S FUNCTION IN FTT PAD
C
      NFT=NXFT*NYFT
      DO 915 J=1,NYP
      K1=J*NX1+1
      K2=(J-1)*NXFT
      DO 915 I=1,NXP
      IK1=I+K1
      I1=I+K2
      GXX(I1)=(AXX(1,1,IK1)+AXX(1,2,IK1-NX1)+AXX(2,2,IK1)
     &        +AXX(2,1,IK1+NX1))/NFT
      GYX(I1)=AYX(1,1,IK1)/NFT
      IF(I.EQ.1) GOTO 913
      I2=NXFT-I+2+K2
      GXX(I2)=GXX(I1)
      GYX(I2)=GYX(I1)
 913  CONTINUE
      IF(J.EQ.1) GOTO 915
      I3=NXFT*(NYFT-J+1)+I
      GXX(I3)=GXX(I1)
      GYX(I3)=GYX(I1)
      IF(I.EQ.1) GOTO 915
      I4=NXFT*(NYFT-J+2)-I+2
      GXX(I4)=GXX(I1)
      GYX(I4)=GYX(I1)
 915  CONTINUE
C
      DO 917 J=1,NYP
      K1=J*NX1+1
      K2=(J-1)*NXFT
      DO 917 I=1,NXP
      IK1=I+K1
      I1=I+K2
      GYY(I1)=(AYY(1,1,IK1)+AYY(1,2,IK1-1)+AYY(2,1,IK1+1)
     &        +AYY(2,2,IK1))/NFT
      IF(I.EQ.1) GOTO 916
```

```
      I2=NXFT-I+2+K2
      GYY(I2)=GYY(I1)
  916 CONTINUE
      IF(J.EQ.1) GOTO 917
      I3=NXFT*(NYFT-J+1)+I
      GYY(I3)=GYY(I1)
      IF(I.EQ.1) GOTO 917
      I4=NXFT*(NYFT-J+2)-I+2
      GYY(I4)=GYY(I1)
  917 CONTINUE
C
C.....MODIFY TO ACCOMMODATE RESISITIVE PLATE
      PRINT*,'INPUT THE COMPLEX RESISTIVITY ETA / Z_0:'
      READ*,ETA
      PRINT*,ETA
      GXX(1)=GXX(1)+(0.,4.)*ETA*KO*AREA/(3.*NFT)
      GYY(1)=GYY(1)+(0.,4.)*ETA*KO*AREA/(3.*NFT)
      GXX(2)=GXX(2)+(0.,2.)*ETA*KO*AREA/(3.*NFT)
      GXX(NXFT)=GXX(2)
      GYY(NXFT+1)=GYY(NXFT+1)+(0.,2.)*ETA*KO*AREA/(3.*NFT)
      GYY(NXFT*(NYFT-1)+1)=GYY(NXFT+1)
C.....MODIFICATION ENDS
C
      NN(1)=NXFT
      NN(2)=NYFT
      CALL  FFTN(GXX,NN,2,1)
      CALL  FFTN(GYX,NN,2,1)
      CALL  FFTN(GYY,NN,2,1)
C
C
      RETURN
      END


      REAL FUNCTION FINT(XP,YP,X,Y)
      XX=X-XP
      YY=Y-YP
      R=SQRT(XX*XX+YY*YY)
      IF((ABS(XX).LE.0.0).OR.(ABS(YY).LE.0.0)) THEN
      SUM=-R*R*R/6.
      ELSE
      SUM=0.5*XX*YY*(XX*ALOG(YY+R)+YY*ALOG(XX+R))
     &    -0.25*XX*YY*(XX+YY)-R*R*R/6.
      END IF
      FINT=SUM
      RETURN
      END


      SUBROUTINE FDINC(BX,BY,NX,NY,DX,DY,KO,THETA,PHI,ALPHA)
C*********************************************************************
C.....THIS SUBROUTINE COMPUTES THE EXCITATION TERM INVOLVING THE     *
C ... INCIDENT FIELD ON THE APERTURE                                 *
C     THETA - INCIDENT ANGLE                                         *
```

```
C      PHI   - INCIDENT ANGLE                                              *
C      ALPHA - THE ANGLE BETWEEN H VECTOR AND PHI DIRECTION               *
C              ALPHA = 0 DEGREE - HZ=0 H-POL. OR TM POL.                  *
C              ALPHA = 90 DEGREES - EZ=0 E-POL. OR TE POL.                *
C*********************************************************************
      COMPLEX BX(*),BY(*),BXX,BYY,CJ
      REAL KO,HOLD
      NXT=(NX-1)*NY
      NYT=NX*(NY-1)
      DO 701 I=1,NXT
  701    BX(I)=CMPLX(0.,0.)
      DO 702 I=1,NYT
  702    BY(I)=CMPLX(0.,0.)
      NX1=NX-1
      NY1=NY-1
      RED=0.01745329
      CJ=CMPLX(0.0,1.0)
      TH=RED*THETA
      PH=RED*PHI
      AL=RED*ALPHA
      AREA=DX*DY
      SINT=SIN(TH)
      COST=COS(TH)
      SINP=SIN(PH)
      COSP=COS(PH)
      SINA=SIN(AL)
      COSA=COS(AL)
      DO 703 J=1,NY1
      Y=(FLOAT(J)-0.5)*DY
      DO 703 I=1,NX1
      X=(FLOAT(I)-0.5)*DX
      K=(J-1)*NX1+I
      HOLD=KO*SINT*(X*COSP+Y*SINP)
      HY=SINA*COST*SINP-COSA*COSP
      BXX=-CJ*KO*HY*CMPLX(COS(HOLD),SIN(HOLD))*AREA
      HX=SINA*COST*COSP+COSA*SINP
      BYY=CJ*KO*HX*CMPLX(COS(HOLD),SIN(HOLD))*AREA
      DO 704 I1=1,2
      IF(I1.EQ.1) L=(J-1)*NX1+I
      IF(I1.EQ.2) L=J*NX1+I
      BX(L)=BX(L)+BXX
      IF(I1.EQ.1) L=(J-1)*NX+I
      IF(I1.EQ.2) L=(J-1)*NX+I+1
  704 BY(L)=BY(L)+BYY
  703 CONTINUE
      RETURN
      END


      SUBROUTINE BNDRY(EX,NEX,LEX)
C*********************************************************************
C.....THIS SUBROUTINE ENFORCES THE BOUNDARY CONDITION ON THE NODAL    *
C ... FIELDS                                                          *
C*********************************************************************
```

```
      COMPLEX EX(*)
      INTEGER NEX(*)
      DO 801 I=1,LEX
      I1=NEX(I)
 801     EX(I1)=CMPLX(0.0,0.0)
      RETURN
      END


C*********************************************************************C
C   THIS SUBROUTINE CALCULATES THE EUCLIDIAN NORM OF A VECTOR         C
C*********************************************************************C
      REAL    FUNCTION VNORM(A,N)
      COMPLEX A(N)
      SUM=0.
      DO 10 I=1,N
      SUM=SUM+CABS(A(I))**2
 10   CONTINUE
      VNORM=SUM
      RETURN
      END


C*********************************************************************C
C   THIS SUBROUTINE CALCULATES THE EUCLIDIAN NORM OF A VECTOR         C
C*********************************************************************C
      COMPLEX    FUNCTION CNORM(A1,A,NO)
      COMPLEX A(NO),A1(NO),SUM
      SUM=(0.,0.)
      DO 20 I=1,NO
      SUM=SUM+A1(I)*A(I)
 20   CONTINUE
      CNORM=SUM
      RETURN
      END


      SUBROUTINE ADDAX(AX,AY,EX,EY,GXX,GYX,GYY,NX,NY,NXFT,NYFT,X,Y)
C*********************************************************************
C.....THIS SUBROUTINE GENERATES VECTORS AX AND AY CONTRIBUTED BY     *
C ... BOUNDARY INTEGRALS                                            *
C*********************************************************************
      COMPLEX AX(*),AY(*),EX(*),EY(*),GXX(*),GYX(*),GYY(*),
     &        X(*),Y(*),XX,YY
      INTEGER NN(2)
      NXP=NX-1
      NYP=NY-1
      NFT=NXFT*NYFT

      DO 910 I=1,NFT
      X(I)=CMPLX(0.,0.)
 910  Y(I)=CMPLX(0.,0.)

      DO 915 J=1,NY
```

```
      K1=(J-1)*NXP
      K2=(J-1)*NXFT
      DO 915 I=1,NXP
      I1=I+K1
      I2=I+K2
      X(I2)=EX(I1)
  915 CONTINUE

      DO 917 J=1,NYP
      K1=(J-1)*NX
      K2=(J-1)*NXFT
      DO 917 I=1,NX
      I1=I+K1
      I2=I+K2
      Y(I2)=EY(I1)
  917 CONTINUE

      NN(1)=NXFT
      NN(2)=NYFT
      CALL  FFTN(X,NN,2,1)
      CALL  FFTN(Y,NN,2,1)

      DO 920 I=1,NFT
      X(I)=GXX(I)*X(I)
      Y(I)=GYY(I)*Y(I)
  920 CONTINUE

      CALL  FFTN(X,NN,2,-1)
      CALL  FFTN(Y,NN,2,-1)

      DO 925 J=1,NY
      K1=(J-1)*NXP
      K2=(J-1)*NXFT
      DO 925 I=1,NXP
      I1=I+K1
      I2=I+K2
      AX(I1)=AX(I1)+X(I2)
  925 CONTINUE

      DO 927 J=1,NYP
      K1=(J-1)*NX
      K2=(J-1)*NXFT
      DO 927 I=1,NX
      I1=I+K1
      I2=I+K2
      AY(I1)=AY(I1)+Y(I2)
  927 CONTINUE
C
      DO 930 I=1,NFT
      X(I)=CMPLX(0.,0.)
  930 Y(I)=CMPLX(0.,0.)

      DO 935 J=1,NYP
      K1=(J-1)*NXP
```

```
      K2=(J-1)*NXFT
      DO 935 I=1,NXP
      I1=I+K1
      I2=I+K2
      X(I2)=EX(I1)-EX(I1+NXP)
  935 CONTINUE

      DO 937 J=1,NYP
      K1=(J-1)*NX
      K2=(J-1)*NXFT
      DO 937 I=1,NXP
      I1=I+K1
      I2=I+K2
      Y(I2)=EY(I1)-EY(I1+1)
  937 CONTINUE

      CALL  FFTN(X,NN,2,1)
      CALL  FFTN(Y,NN,2,1)

      DO 940 I=1,NFT
      X(I)=GYX(I)*X(I)
      Y(I)=GYX(I)*Y(I)
  940 CONTINUE

      CALL  FFTN(X,NN,2,-1)
      CALL  FFTN(Y,NN,2,-1)

      DO 945 J=1,NYP
      K1=(J-1)*NXP
      K2=(J-1)*NXFT
      DO 945 I=1,NXP
      I1=I+K1
      I2=I+K2
      AX(I1)=AX(I1)+Y(I2)
      AX(I1+NXP)=AX(I1+NXP)-Y(I2)
  945 CONTINUE

      DO 947 J=1,NYP
      K1=(J-1)*NX
      K2=(J-1)*NXFT
      DO 947 I=1,NXP
      I1=I+K1
      I2=I+K2
      AY(I1)=AY(I1)+X(I2)
      AY(I1+1)=AY(I1+1)-X(I2)
  947 CONTINUE
C
      RETURN
      END



      SUBROUTINE SCATER(EX,EY,NMAX,NXP,NYP,DX,DY,KO,TETA,PHI,SIG)
        EXTERNAL SINC
```

```
      REAL KO
      COMPLEX EX(*),EY(*),XJ,SX,SY,MX,MY,NTETA,NPHI
      DATA    PI/3.141592653589793/,TPI/6.28318530717959/,XJ/(0.,1.0)/
C     ---------------------------------------------------------------------
      STE=SIN(TETA*PI/180.)
      CTE=COS(TETA*PI/180.)
      SPH=SIN(PHI*PI/180.)
      CPH=COS(PHI*PI/180.)
      RX=KO*STE*CPH
      RY=KO*STE*SPH
      SX=(0.,0.)
      SY=(0.,0.)
      DO 20 J=1,NYP
         L=(J-1)*NXP
         Y=(FLOAT(J)-0.5)*DY
         DO 10 I=1,NXP
            K=L+I
            X=(FLOAT(I)-0.5)*DX
            I1=(J-1)*NXP+I
            I2=J*NXP+I
            I3=(J-1)*(NXP+1)+I
            I4=(J-1)*(NXP+1)+I+1
            MY=-0.5*(EX(I1)+EX(I2))
            MX=0.5*(EY(I3)+EY(I4))
            SX=SX+MX*CEXP(XJ*(RX*X+RY*Y))
            SY=SY+MY*CEXP(XJ*(RX*X+RY*Y))
10       CONTINUE
20    CONTINUE
      WAVE=2.*PI/KO
      AREA=DX*DY/WAVE/WAVE
      SX=AREA*SINC(0.5*DX*RX)*SINC(0.5*DY*RY)*SX
      SY=AREA*SINC(0.5*DX*RX)*SINC(0.5*DY*RY)*SY
      NTETA= CTE*(CPH*SX+SPH*SY)
      NPHI=-SPH*SX+CPH*SY
      SIGT=PI*CABS(NTETA)**2
      SIGP=PI*CABS(NPHI)**2
      SIG = PI*(CABS(NTETA)**2+CABS(NPHI)**2)
C
C    THE FACTOR OF 4 FROM IMAGE THEORY
C

      SIGT=4.*SIGT
      SIGP=4.*SIGP
      SIG  = 4.*SIG
      IF(SIG.LE.1.E-8)SIG=1.E-8
      SIG = 10.*ALOG10(SIG) -6.
      IF(SIGT.LE.1.E-8)SIGT=1.E-8
      SIGT = 10.*ALOG10(SIGT) -6.
      IF(SIGP.LE.1.E-8)SIGP=1.E-8
      SIGP = 10.*ALOG10(SIGP) -6.
      WRITE(*,30) TETA,PHI,SIG,SIGT,SIGP
  30  FORMAT(5F12.2)
      RETURN
      END
```

```
C   ======================================================================
C                          SINGLE PRECISION SINC
C   ======================================================================
        REAL FUNCTION SINC(A)
        REAL A
C   ----------------------------------------------------------------------
        IF(A.EQ.0.)THEN
          SINC=1.
        ELSE
          SINC=SIN(A)/A
        ENDIF
        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     -FOREWARD FFT
C                           -1     -INVERSE  FFT TIMES THE PRODUCT OF
C                                   LENGTHS OF ALL DIMENSIONS
C   ======================================================================
        SUBROUTINE  FFTN(DATA,NN,NDIM,ISIGN)
C        REAL*8      WR,WI,WPR,WPI,WTEMP,THETA
        DOUBLE PRECISION  WR,WI,WPR,WPI,WTEMP,THETA
        DIMENSION   NN(NDIM),DATA(*)
        NTOT=1
        DO 10 IDIM=1,NDIM
 10          NTOT=NTOT*NN(IDIM)
C        ENDDO
        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   BIT REVERSAL SECTION
           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
C                   ENDDO
 30               CONTINUE
C                 ENDDO
                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             ENDDO
C    DANIELSON-LANCZOS FORMULA
              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
C                     ENDDO
 60                CONTINUE
C                   ENDDO
                  WTEMP=WR
C    TRIGONOMETRIC RECURRENCE
                  WR=WR*WPR-WI*WPI+WR
                  WI=WI*WPR+WTEMP*WPI+WI
 70             CONTINUE
C               ENDDO
              IFP1=IFP2
              GO TO 2
```

```
            END IF
            NPREV=N*NPREV
 80       CONTINUE
C         ENDDO
          RETURN
          END
```

SCATTERING BY A FINITE FREQUENCY SELECTIVE SURFACE

Jian-Ming Jin and John L. Volakis

Radiation Laboratory

Department of Electrical Engineering and Computer Science

The University of Michigan

Ann Arbor, Michigan 48109-2122

ABSTRACT

An exact solution is presented for finite frequency selective surfaces (FSS) which is used to assess the accuracy of an existing approximate solution. Exact scattering patterns are given for planar FSS structures modelled with as many as 123000 unknowns and it is concluded that the approximate solution is reasonably accurate away from grazing.

I. INTRODUCTION

Frequency selective surfaces (FSS), comprising an array of conducting/resistive patches or perforations in a conducting screen, find numerous applications in radome designs and dichroic reflector antenna systems. An FSS is typically analyzed on the assumption that it is infinite in extent. Floquet's theorem is then invoked to express the currents of the FSS in terms of a periodic function whose period is equal to the area over a single cell, thus reducing the computational region to that of the single patch [1]-[3]. In practice, though, the FSS is finite and for an exact analysis it is required that the currents over the FSS be determined without *a priori* assumptions on their functionality. The FSS structure must then be modelled in its entirety and this implies excessive memory and computational demands [4], [5]. For this reason, an approximate solution was proposed in [6] where the currents computed for the infinite FSS were integrated over the extent of the corresponding finite FSS to evaluate its scattering. Due to the lack of available exact data for three-dimensional finite FSS, this approximate solution was only validated for a unidirectionally truncated model (finite in one direction and infinite in the other) by comparing it with the exact data presented in [7], [8]. In this communication we present new exact data for large three-dimensional FSS structures which are used to assess the accuracy of the approximate solution based on the infinite FSS model. The exact data were generated from an iterative solution of a new discrete form of the integral equation. This is briefly discussed in the subsequent section and because of its low memory requirement coupled with its fast convergence it was possible to obtain results for large FSS of practical interest.

II. APPROXIMATE AND EXACT SOLUTIONS

To obtain the approximate solution for the scattering by a finite FSS we first consider the corresponding infinite periodic FSS. This is readily analyzed by invoking Floquet's theorem to reduce the domain of the unknown currents to that over a single FSS cell. Following the analysis given in [6], roof-top basis functions are employed for representing the current distribution on the reference patch and the resulting integral equation is discretized via Galerkin's method. The system is then solved via the conjugate gradient method in conjunction with the fast Fourier transform (FFT) [6]. Once the currents on the reference patch are obtained, the FSS scattering is approximately found by multiplying the radiated field from the reference patch with the array factor of the finite FSS.

The exact treatment of the finite FSS involves a direct solution of the integral equation

$$\hat{\mathbf{z}} \times \mathbf{E}^{inc}(\mathbf{r}) = jk_0 Z_0 \, \hat{\mathbf{z}} \times \iint_S \bar{\bar{\mathbf{G}}}_0(\mathbf{r}, \mathbf{r}') \bullet \mathbf{J}(\mathbf{r}') dS' \tag{1}$$

In this, Z_0 is the free space intrinsic impedance, $k_0 = 2\pi/\lambda$ is the free space wavenumber, $\mathbf{E}^{inc}$ denotes the incident field, $\bar{\bar{\mathbf{G}}}_0$ is the free-space dyadic Green's function and $\mathbf{J}$ is the unknown surface current density on the patches of the FSS occupying the surface S in the x-y plane. To discretize the integral equation, S is subdivided into small rectangular elements of width Δx along the x direction and Δy along the y direction. Assuming $M+1$ subdivisions along the x direction and $N+1$ subdivisions along the y direction, by using roof-top basis the current density may be expanded as

$$J_x(x,y) = \sum_{m=1}^{M} \sum_{n=1}^{N+1} J_x^D(m,n) T_{mn}^x(x,y) \tag{2a}$$

$$J_y(x,y) = \sum_{m=1}^{M+1}\sum_{n=1}^{N} J_y^D(m,n)T_{mn}^y(x,y) \tag{2b}$$

where $J_{x,y}^D(m,n)$ denote the discrete current density and $T_{mn}^{x,y}$ are the roof-top basis. They are defined as

$$T_{mn}^x(x,y) = \begin{cases} 1-\frac{|x-m\Delta x|}{\Delta x}, \text{ for } |x-m\Delta x| \le \Delta x \text{ and } |y-(n-0.5)\Delta y| < \frac{\Delta y}{2} \\ 0, \qquad \text{else} \end{cases} \tag{3a}$$

$$T_{mn}^y(x,y) = \begin{cases} 1-\frac{|y-n\Delta y|}{\Delta y}, \text{ for } |y-n\Delta y| \le \Delta y \text{ and } |x-(m-0.5)\Delta x| < \frac{\Delta x}{2} \\ 0, \qquad \text{else} \end{cases} \tag{3b}$$

When the expansions in (2) are substituted into (1), upon employing Galerkin's technique we obtain

$$jk_0Z_0J_x^D(m,n) \otimes G_T^x(m,n) + \frac{Z_0}{jk_0\Delta x}\left[\frac{1}{\Delta x}\left(J_x^D(m,n) - J_x^D(m-1,n)\right)\right.$$
$$\left.+\frac{1}{\Delta y}\left(J_y^D(m,n) - J_y^D(m,n-1)\right)\right] \otimes (G_P(m,n) - G_P(m+1,n)) = b_x(m,n) \tag{4a}$$

$$jk_0Z_0J_y^D(m,n) \otimes G_T^y(m,n) + \frac{Z_0}{jk_0\Delta y}\left[\frac{1}{\Delta x}\left(J_x^D(m,n) - J_x^D(m-1,n)\right)\right.$$
$$\left.+\frac{1}{\Delta y}\left(J_y^D(m,n) - J_y^D(m,n-1)\right)\right] \otimes (G_P(m,n) - G_P(m,n+1)) = b_y(m,n) \tag{4b}$$

where the symbol $\otimes$ denotes convolution and

$$G_T^x(m-m',n-n') = \int_{-\Delta x}^{\Delta x}\int_{-\Delta y}^{0}\left(1-\frac{|\tilde{x}|}{\Delta x}\right)\int_{-\Delta x}^{\Delta x}\int_{-\Delta y}^{0}\left(1-\frac{|\tilde{x}'|}{\Delta x}\right)$$
$$\cdot G_0(\tilde{x}-\tilde{x}'+(m-m')\Delta x, \tilde{y}-\tilde{y}'+(n-n')\Delta y)d\tilde{x}'d\tilde{y}'d\tilde{x}d\tilde{y} \tag{5a}$$

$$G_P(m-m',n-n') = \int_{-\Delta x}^{0}\int_{-\Delta y}^{0}\int_{-\Delta x}^{0}\int_{-\Delta y}^{0}$$
$$\cdot G_0(\tilde{x}-\tilde{x}'+(m-m')\Delta x, \tilde{y}-\tilde{y}'+(n-n')\Delta y)d\tilde{x}'d\tilde{y}'d\tilde{x}d\tilde{y} \tag{5b}$$

$$b_x(m,n) = \int_{-\Delta x}^{\Delta x}\int_{-\Delta y}^{0}\left(1-\frac{|\tilde{x}|}{\Delta x}\right)E_x^{inc}(\tilde{x}+m\Delta x, \tilde{y}+n\Delta y)d\tilde{x}d\tilde{y} \tag{5c}$$

in which G_0 is the free space scalar Green's function. G_T^y and b_y are obtained from G_T^x and b_x by interchanging $\tilde{x}$ and $\tilde{x}'$ with $\tilde{y}$ and $\tilde{y}'$. It is important to note that in deriving (4) from (1) we employed the divergence theorem twice to transfer the double gradient operations from the Green's function G_0 to the expansion and the weighting functions.

The most important aspect of (4) is its convolutional form which was explicitly brought out through the transformations $\tilde{x} = x - m\Delta x$ and $\tilde{y} = y - n\Delta y$. This form is necessary for the application of the discrete Fourier transform to evaluate the convolutions and solve the system via an iterative algorithm using only $O(N)$ storage. For example, the conjugate gradient (CG) or biconjugate gradient (BCG) algorithm could be used here. However, in most previous applications of the CG-FFT or BCG-FFT algorithm, point matching was employed and this compromised the convergence rate of the solution. As will be seen later, though, the application of Galerkin's technique in conjunction with the transfer of the two ∇ operators to the expansion and weighting functions leads to substantial improvements in the convergence of the CG-FFT and BCG-FFT. One can then handle large size problem with reasonable computational efforts and this is demonstrated in the next section.

III. NUMERICAL RESULTS

Consider the planar FSS geometry shown in Figure 1. It is a square array of square perfectly conducting patches 0.8 cm×0.8 cm in size and each patch is separated 0.2 cm from its adjacent. To assess the accuracy of the aforementioned approximate analysis technique we will examine the bistatic scattering of the FSS as obtained by the exact and approximate solutions. For this purpose we assume a plane wave excitation in the

direction ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$) with an operating frequency of 24 GHz. Radar cross section (RCS) patterns corresponding to this excitation are given in Figures 2-5 and each figure applies to a different FSS size. Figure 2 refers to the scattering by a single patch, Figure 3 displays the pattern of a 4×4 patch array whereas Figures 4 and 5 include the patterns for the 12×12 and 25×25 patch arrays, respectively. In the figures, the solid line is the numerically exact solution whereas the dashed line represents the approximate solution, and in all cases we observe a good agreement between the two solutions near the pattern peaks. In general, the agreement is better for the larger arrays and this is more so for the transverse magnetic (TM) to z incidence than the transverse electric (TE) to z incidence results. The largest discrepancy between the exact and approximate solutions occurs in the TE_z incidence near grazing angles and this persists even for the larger arrays. RCS comparisons for other incidences are given in Figures 6 and 7. In particular, Figures 6 and 7 show the comparison of the approximate and exact RCS solutions for the 12×12 patch array when the plane wave is incident at ($\phi^{inc} = 45^o, \theta^{inc} = 45^o$) and ($\phi^{inc} = 45^o, \theta^{inc} = 75^o$) , respectively. Again, a reasonable agreement is observed between the two solutions.

In generating the data in Figures 2-7 we employed 200 unknowns in implementing the approximate solution. For the exact solution the numbers of unknowns used for the 4×4, 12×12 and 25×25 arrays were, respectively, 2964, 28084 and 123504. The corresponding convergence curves (backscatter RCS versus the number of iterations) for the exact solution with ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$) are given in Figure 8. These demonstrate the impressive convergence of the BCG-FFT in connection with the described formulation. Even in the case of the 25×25 patch array, convergence was achieved within 70 iterations

whereas in previous implementations the convergence was substantially slower.

V. CONCLUSIONS

In this comminuication an exact solution for the scattering by a finite FSS was employed to validate a previously proposed approximate solution. It was found that away from grazing, the approximate solution is reasonably accurate and is therefore suited for engineering design purposes.

REFERENCES

[1] C. C. Chen, "Transmission through a conducting screen perforated periodically with apertures," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-18, pp. 627-632, Sept. 1970.

[2] S. W. Lee, "Scattering by dielectric-loaded screen," *IEEE Trans. Antennas Propagat.*, vol. AP-19, pp. 656-665, Sept. 1971.

[3] J. P. Montgomery, "Scattering by an infinite periodic array of thin conductors on a dielectric sheet," *IEEE Trans. Antennas Propagat.*, vol. AP-23 pp. 70-75, Jan 1975.

[14 R. Kastner and R. Mittra, "Iterative analysis of finite-sized planar frequency selective surfaces with rectangular patches or perforations," *IEEE Trans. Antennas Propagat.*, vol. AP-35, pp. 372-377, April 1987.

[5] K. Merewether and R. Mittra, "Spectral-domain analysis of a finite frequency selective surface with cross-shaped conducting patches," *1988 IEEE AP-S International Symposium Digest,* vol. 2, pp. 742-745, June 6-10, 1988.

[6] J. M. Jin and J. L. Volakis, "Electromagnetic scattering by a perfectly conducting patch array on a dielectric slab," *IEEE Trans. Antennas Propagat.*, vol. AP-38, pp. 556-563, April 1990.

[7] W. L. Ko and R. Mittra, "Scattering by a truncated periodic array," *IEEE Trans. Antennas Propagat.*, vol. AP-36, pp. 496-503, April 1988.

[8] P. W. Grounds and K. J. Webb, "Numerical analysis of finite frequency selective surfaces," in *1988 IEEE Antennas Prograt. Soc. Int. Symp. Dig.*, vol. II, pp. 746-749, June 1988.

FIGURE CAPTIONS

Fig. 1 Geometry of a finite frequency selective surface comprising an array of conducting patches.

Fig. 2 Bistatic scattering pattern for a single patch (1×1 array); ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 3 Bistatic scattering pattern for a 4×4 array; ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 4 Bistatic scattering pattern for a 12×12 array; ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 5 Bistatic scattering pattern for a 25×25 array; ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 6 Bistatic scattering pattern for a 12×12 array; ($\phi^{inc} = 45^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 7 Bistatic scattering pattern for a 12×12 array; ($\phi^{inc} = 45^o, \theta^{inc} = 75^o$), f = 24 GHz. (a) TM incidence. (b) TE incidence.

Fig. 8 Backscatter RCS versus number of iterations; ($\phi^{inc} = 0^o, \theta^{inc} = 45^o$), f = 24 GHz. (a) For a 4×4 array (2964 unknowns). (b) For a 12×12 array (28084 unknowns). (c) For a 25×25 array (123504 unknowns).

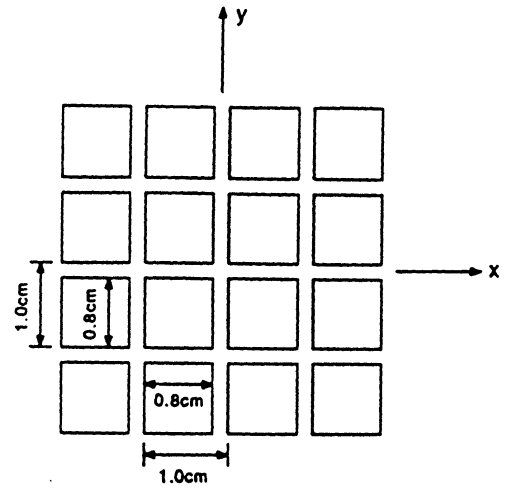
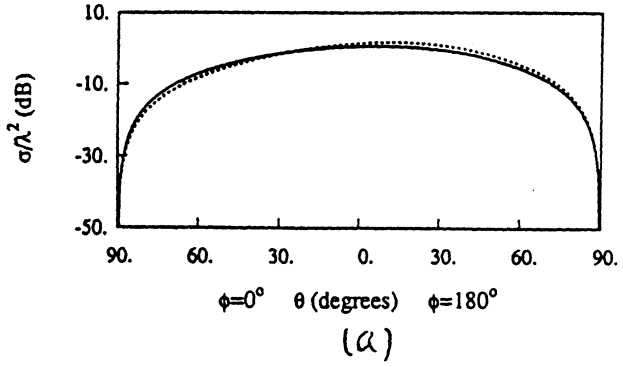
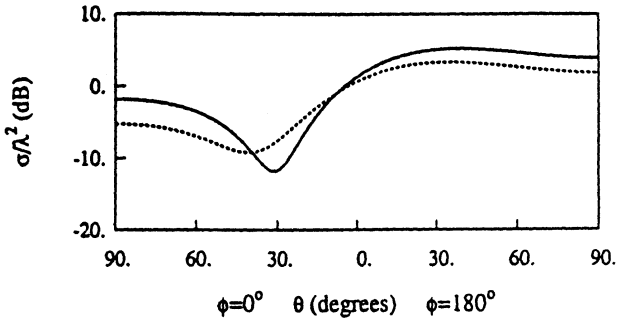


Fig. 1

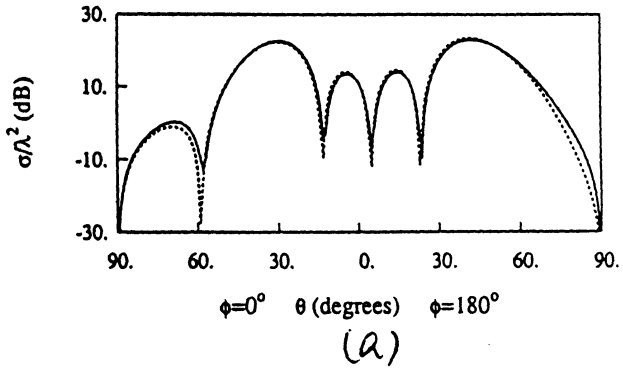


(a)

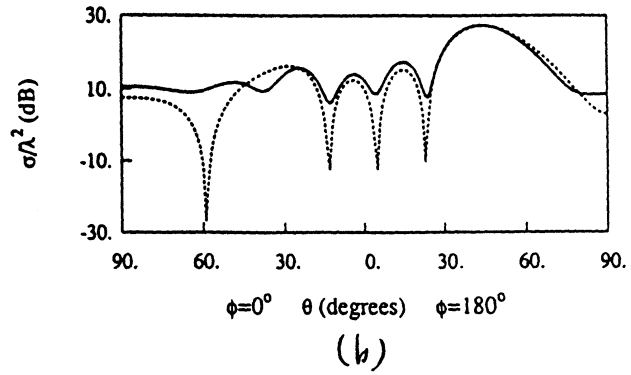


(b)

Fig. 2

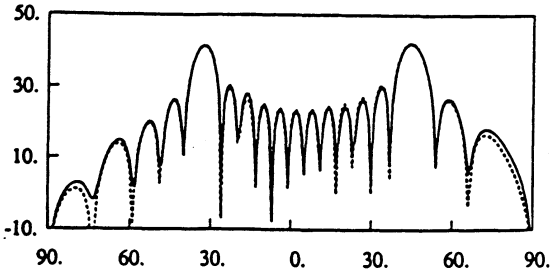


(a)

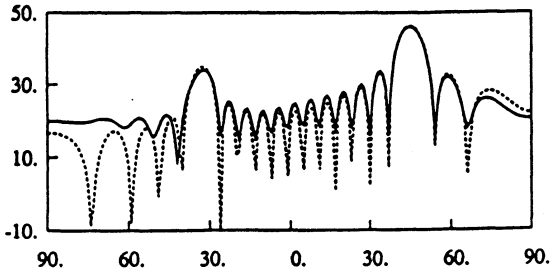


(b)

Fig. 3

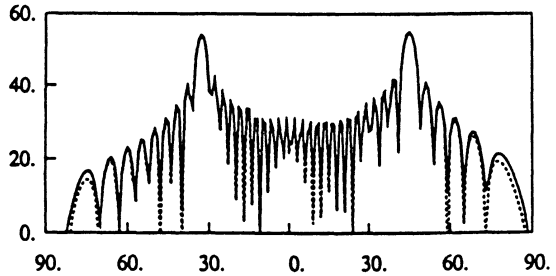


(a)

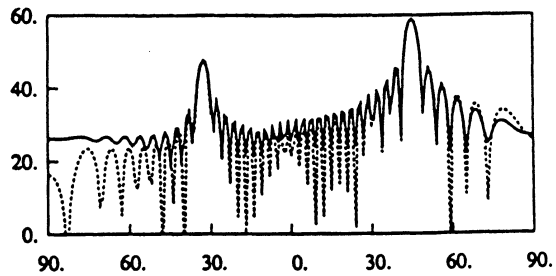


(b)

Fig. 4

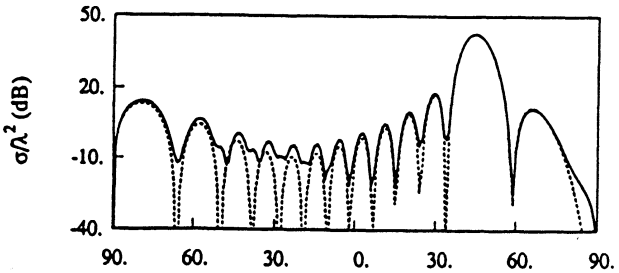


(a)

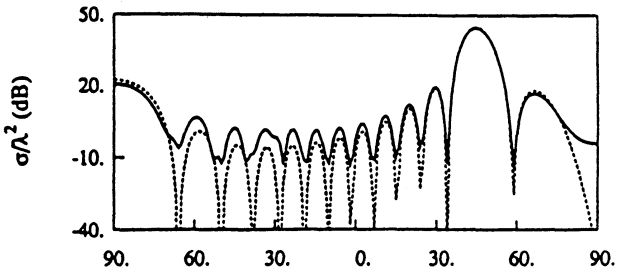


(b)

Fig. 5

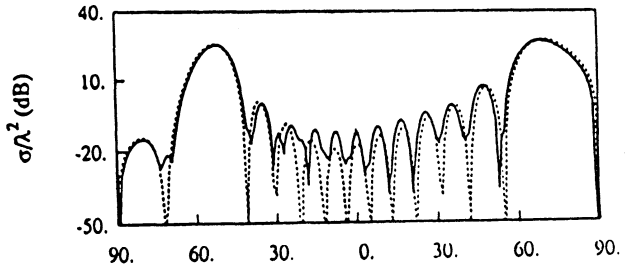


(a)

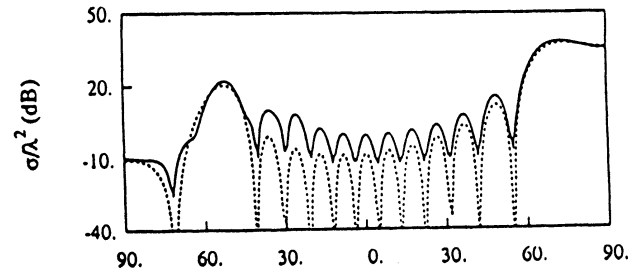


(b)

Fig. 6

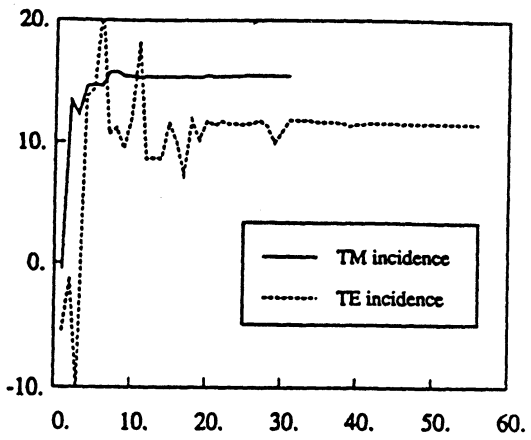


(a)

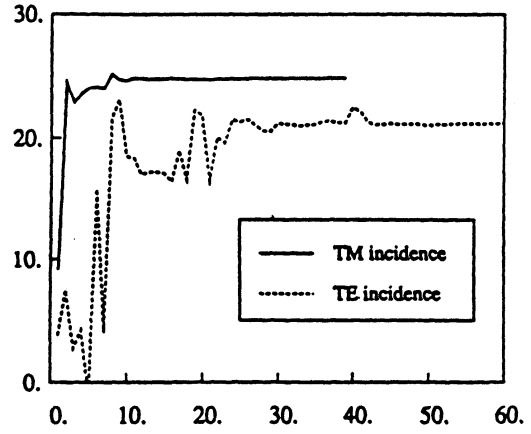


(b)

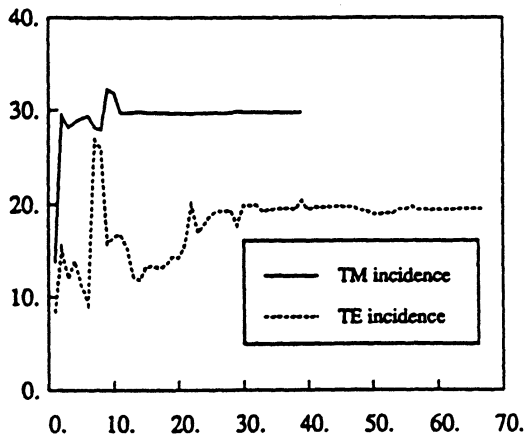
Fig. 7


20.
10.
0.
-10.
0.
10.
20.
30.
40.
50.
60.
TM incidence
TE incidence
Number of Iterations

(a)


30.
20.
10.
0.
0.
10.
20.
30.
40.
50.
60.
TM incidence
TE incidence
Number of Iterations

(b)


40.
30.
20.
10.
0.
0.
10.
20.
30.
40.
50.
60.
70.
TM incidence
TE incidence
Number of Iterations

(c)

Fig. 8