Radar Cross Section of a Rectangular Cavity - A Massively Parallel Calculation

L. D. Vann* L.T. Wille† J.S. Bagby* H.F. Helmken*

March 27, 1995

Abstract

A sequential code that calculates the radar cross section of a rectangular waveguide cavity was modified to execute on a MasPar MP-1 massively parallel computer with a SIMD architecture. The code uses the mode matching method of analysis to produce a set of matrix equations. Steps taken to accomplish the parallelization are discussed and some specific examples of program modification are presented. Timing results for wideband data are given that demonstrate the power of parallel computers for this type of application. Suggestions are made for further improvements as increased memory space becomes available.

1. Introduction

Radar Cross Section (RCS) analysis has important applications in many areas such as target detection, identification, and imaging. Objects of interest range from the very simple (perfectly conducting spheres or simple cylinders) to the very complex (aircraft, ships, buildings, etc.). In general, the RCS of an object depends on its shape and composition as well as the frequency, aspect angle, and polarization of the incident field. The object may consist of a combination of dielectric and magnetic materials or plasmas and may contain multiple layers of these materials with differing thicknesses. The ability to calculate the RCS of complex objects within a reasonable length of time is heavily dependent upon available computer power.

Many problems in computational electromagnetics are inherently parallel to some degree (e.g., calculations are made for multiple frequencies and/or aspect angles), and it is of interest to exploit this parallelism by developing algorithms for massively parallel computers. While better results might be obtained by carefully rethinking the usual approach to a problem and developing a parallel algorithm from the ground up, there is also some obvious practical benefit in porting sequential codes already in use to a massively parallel format. As an illustration of the latter point, in this paper we have modified an existing sequential RCS cavity code [1] to execute on a massively parallel computer.

The rectangular cavity code is part of the McPTD program developed by Lee *et al.* [1] and adopted by the U.S. Electromagnetic Code Consortium. The code uses the mode matching

*Department of Electrical Engineering, Florida Atlantic University, Boca Raton, FL 33431.

†Department of Physics, Florida Atlantic University, Boca Raton, FL 33431.

method of analysis in which the electric and magnetic fields are expanded in terms of modal functions with unknown coefficients, and the tangential fields are matched over the interfaces of the regions. Because this method of analysis involves the manipulation of large matrices for each frequency and aspect angle of interest, computational run time becomes impractical when using the sequential code over a wide range of frequencies and aspect angles as the number of modes is increased to achieve sufficient accuracy.

The remainder of this paper is divided as follows. Section 2 provides a general description of rectangular cavities considered by the McPTD code. A formulation for the cavity and free-space fields is also included in this section along with a summary of how the field matching produces a set of matrix equations. Section 3 contains a discussion of some parallel programming concepts and how they relate to the MasPar MP-1 computer used for this implementation. Section 4 details the steps taken to parallelize the sequential code and gives some specific examples of code modification. Section 5 presents timing results of the sequential versus parallel code for wideband data, and Section 6 summarizes the paper.

2. Geometrical Description of Cavity and Field Formulations

Figure 1 shows the general geometry of rectangular cavities considered by the McPTD code [1]. The cavity aperture is located on an infinite ground plane that may have a number of isotropic dielectric or plasma slabs on top. In addition, the interior of the cavity may contain a number of isotropic dielectric layers of differing thicknesses and constitutive parameters. The allowable number of interior and exterior layers is determined by array dimensions specified within the code. These array dimensions may be increased to accommodate a particular cavity of interest.

The unknown fields of the regions are expanded as a sum of TE^z and TM^z modes using suitable modal functions with unknown modal expansion coefficients. Once the electric and magnetic fields have been expressed in terms of these TE^z and TM^z modes, the tangential components of the total fields are matched over the interfaces resulting in an infinite number of simultaneous linear equations due to the infinite number of possible modes. This infinite matrix must then be truncated in order to determine the unknown modal expansion coefficients.

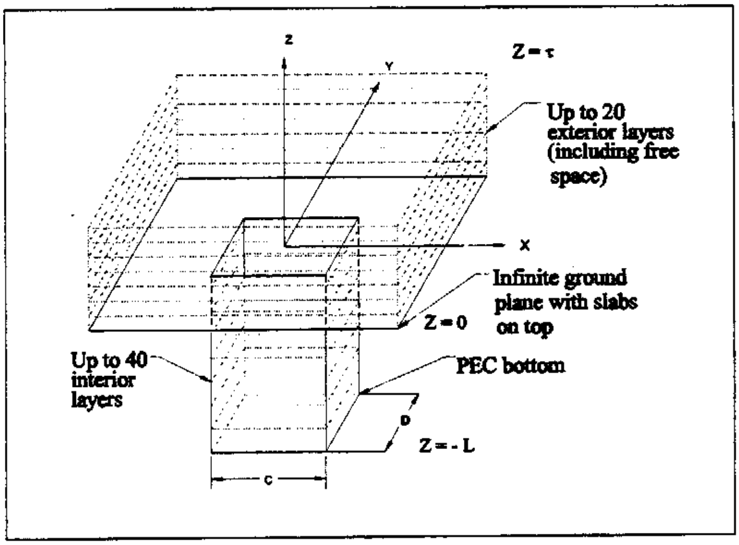
z
Y
Z=τ
Up to 20 exterior layers (including free space)
X
Infinite ground plane with slabs on top
Z=0
Up to 40 interior layers
PEC bottom
D
Z=-L
c

Fig. 1: Geometry of Rectangular Waveguide

2.1. Cavity Fields

Field components within the cavity were derived in terms of the magnetic and electric vector potentials **A** and **F**. These vector potentials can be chosen in such a way as to reduce the vector Helmholtz equations $\nabla^2\mathbf{A} + k^2\mathbf{A} = 0$ and $\nabla^2\mathbf{F} + k^2\mathbf{F} = 0$ to the scalar Helmholtz equation $\nabla^2\Psi + k^2\Psi = 0$ [2][3]. In order to satisfy the boundary conditions for the rectangular waveguide $(E_x = 0$ at $y = -\frac{d}{2}, y = \frac{d}{2}$, and $E_y = 0$ at $x = -\frac{c}{2}, x = \frac{c}{2})$, the appropriate solution for the wave equation in terms of the modal function $\Phi_{mn}(x,y)$for the TE^z case (with $e^{j\omega t}$ time dependence assumed) is [1]

$$\Psi_i = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\left[A_{mn}^{(i)}e^{-jk_{zmn}^{(i)}z} + B_{mn}^{(i)}e^{+jk_{zmn}^{(i)}z}\right]\cdot\Phi_{mn}(x,y) \tag{2.1}$$

where

$$\Phi_{mn} = \cos\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\cos\left[\frac{n\pi}{2}\left(y+\frac{d}{2}\right)\right]. \tag{2.2}$$

The TM^z case has the additional boundary condition that $E_z = 0$ at $x = -\frac{c}{2}, x = \frac{c}{2}, y = -\frac{d}{2}$, and $y = \frac{d}{2}$; the appropriate solution in terms of the modal function $\overline{\Phi}_{mn}(x,y)$ is [1]

$$\overline{\Psi}_i = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\left[\overline{A}_{mn}^{(i)}e^{-j\overline{k}_{zmn}^{(i)}z} + \overline{B}_{mn}^{(i)}e^{+j\overline{k}_{zmn}^{(i)}z}\right]\cdot\overline{\Phi}_{mn}(x,y) \tag{2.3}$$

where

$$\overline{\Phi}_{mn} = \sin\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\sin\left[\frac{n\pi}{2}\left(y+\frac{d}{2}\right)\right]. \tag{2.4}$$

In the above equations, (i) denotes the i^{th} layer of the waveguide and the wavenumber is

$$k_{z_{mn}}^{(i)} = \sqrt{k_o^2 \epsilon_i \mu_i - \left(\frac{m\pi}{c}\right)^2 - \left(\frac{n\pi}{d}\right)^2} \tag{2.5}$$

for the outgoing wave. Once the scalar potentials have been determined for the TE^z and TM^z cases, the associated fields can be found [2][3].

The modal coefficients in the top layer can be expressed in terms of those in the bottom layer by considering reflection and transmission at each interface and formulating a forward-propagation matrix [4]. Thus for a total of I interior layers, the modal coefficients of the first layer can be related to the modal coefficients of the I^{th} layer as follows:

$$\begin{gathered} \begin{bmatrix} A_1 \\ B_1 \end{bmatrix} = \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(I-1)I} \cdot \begin{bmatrix} A_I \\ B_I \end{bmatrix} \\ \begin{bmatrix} \overline{A}_1 \\ \overline{B}_1 \end{bmatrix} = \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(I-1)I} \cdot \begin{bmatrix} \overline{A}_I \\ \overline{B}_I \end{bmatrix} \end{gathered} \tag{2.6}$$

where $\overline{V}$ is the reflection/transmission coefficient matrix for each layer.

Because the cavity is terminated by a Perfect Electric Conductor (PEC) at $z = -L$ one has [4]:

$$\begin{gathered} B_I = (-1)e^{j2k_{z_I}L} A_I \\ \overline{B}_I = (+1)e^{j2k_{z_I}L} \overline{A}_I \end{gathered} \tag{2.7}$$

so that, finally, the expressions for $A_1, B_1, \overline{A}_1, \overline{B}_1$ can be written as

$$\begin{gathered} \begin{bmatrix} A_1 \\ B_1 \end{bmatrix} = \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(I-1)I} \cdot \begin{bmatrix} A_I \\ (+1)e^{j2k_{zI}L} A_I \end{bmatrix} \\ \begin{bmatrix} \overline{A}_1 \\ \overline{B}_1 \end{bmatrix} = \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(I-1)I} \cdot \begin{bmatrix} \overline{A}_I \\ (-1)e^{j2k_{zI}L} \overline{A}_I \end{bmatrix} . \end{gathered} \tag{2.8}$$

The above formulation gives an expression for the fields in the cavity in terms of two unknown coefficients $\{A_I, \overline{A}_I\}$ for each mode (m, n) under consideration.

2.2. Free-Space Fields

In the unbounded upper half space region ($z > 0$ in Figure 1), the eigenvalues become continuous and thus the solution of the scalar wave equation is a continuous superposition of spectral components which takes the form of a Fourier integral [5]:

TE^z Mode:

$$\Psi_g(x, y, z) = \frac{1}{(2\pi)^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \left[E^{(g)} e^{-jk_z^{(g)}z} + F^{(g)} e^{+jk_z^{(g)}z}\right] e^{-j(k_x x + k_y y)} dk_x dk_y \tag{2.9}$$

TM^z Mode:

$$\overline{\Psi}_g(x, y, z) = \frac{1}{(2\pi)^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \left[\overline{E}^{(g)} e^{-jk_z^{(g)}z} + \overline{F}^{(g)} e^{+jk_z^{(g)}z}\right] e^{-j(k_x x + k_y y)} dk_x dk_y \tag{2.10}$$

where (g) denotes the gth layer and (G) indicates the total number of layers above the ground plane. In order to match fields over the aperture of the cavity, the coefficients at the aperture $(E, F, \overline{E}, \overline{F})$ are related to the coefficients in the first layer above the ground plane by using the forward-propagation reflection and transmission matrix of equation (2.6) with appropriate substitutions as follows:

$$\begin{aligned} \begin{bmatrix} E_1 \\ F_1 \end{bmatrix} &= \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(G-1)G} \cdot \begin{bmatrix} E \\ F \end{bmatrix} \\ \begin{bmatrix} \overline{E}_1 \\ \overline{F}_1 \end{bmatrix} &= \overline{V}_{(1)(2)} \cdot \overline{V}_{(2)(3)} \cdots \overline{V}_{(G-1)G} \cdot \begin{bmatrix} \overline{E} \\ \overline{F} \end{bmatrix} . \end{aligned} \tag{2.11}$$

The coefficients F and $\overline{F}$ are the known amplitudes of the incident field in the upper half space. Therefore, (2.11) gives an expression for the fields in the first layer above the ground plane in terms of two unknown coefficients $(E, \overline{E})$. The preceding discussion is for the general case where the number of layers above the ground plane is greater than one. The cavity examined for this paper contained no layers above the ground plane, hence (2.11) was not used.

Prior to finding the fields, the integrals of (2.9) and (2.10) are discretized for numerical evaluation as follows:

$$\begin{bmatrix} \psi(x,y,z) \\ \overline{\psi}(x,y,z) \end{bmatrix} = \tfrac{ab}{(2\pi)^2} \textstyle\sum_{-\infty}^{+\infty} \sum_{-\infty}^{+\infty} \begin{bmatrix} E(k_x,k_y) + F \\ \overline{E}(k_x,k_y) + \overline{F} \end{bmatrix} \times \\ e^{-j(k_x x + k_y y + k_z z)} \Delta k_x \Delta k_y \tag{2.12}$$

where a and b represent the number of integration points in terms of wavelength. The integration interval and range are given by two dimensionless parameters [1]: P' (number of integration points per wavelength) and R (integration range parameter). The integration limits of (2.9) and (2.10) normalized with respect to the free space wavenumber, k_o, become $-R \leq \left(\frac{k_x}{k_o}, \frac{k_y}{k_o}\right) \leq R$. In addition, $\Delta k_x = \Delta k_y = \frac{k_o}{P'}$, $a = b = \frac{2\pi P'}{k_o} = P'\lambda_o$, and $P = RP'$ where the total number of integration points is equal to $(2P+1)^2$. Equations (2.9) and (2.10) can be approximated as

$$\begin{bmatrix} \psi(x,y,z) \\ \overline{\psi}(x,y,z) \end{bmatrix} = \textstyle\sum_{p=-P}^{P} \sum_{q=-P}^{P} \begin{bmatrix} E_{pq} + F_{pq} \\ \overline{E}_{pq} + \overline{F}_{pq} \end{bmatrix} \times \\ e^{-j\left(k_{x_p} x + k_{y_q} y + k_{z_{pq}} z\right)} \tag{2.13}$$

where

$$\begin{aligned} k_{x_p} &= \tfrac{2p\pi}{a} - k_o \sin\theta \cos\phi \\ k_{y_q} &= \tfrac{2q\pi}{b} - k_o \sin\theta \sin\phi \\ k_{z_{pq}} &= \sqrt{k_o^2 - u_p^2 - v_q^2} \end{aligned} \tag{2.14}$$

In the above relations, θ and ϕ are the incident angles of the fields. The fields for the upper half space can now be found in terms of the unknown modal expansion coefficients by substituting the potentials of (2.13) into the fields expressions given in [2][3].

In order to find solutions for the modal expansion coefficients contained in the field expressions, the tangential components of the electric and magnetic fields must be matched at the interface $z = 0$. Considering first the total electric fields, boundary conditions require that

$$\mathbf{E}_t(z = 0-) = \mathbf{E}_t(z = 0+),\ |x| < \frac{c}{2} \text{ and } |y| < \frac{d}{2} \tag{2.15}$$

Based on the previous discussion, the total tangential electric fields at $z = 0-$ and $z = 0+$ can be written as

$$\begin{aligned}\mathbf{E}_t\ |_{z=0-} = \quad & \textstyle\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}[A_{mn} + B_{mn}]\,\frac{n\pi}{d}\cos\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\sin\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_x \\ & +\tfrac{-m\pi}{c}\sin\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\cos\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_y \\ & +\tfrac{-k_{z_{mn}}}{k_o Y_o \epsilon}\left[\overline{A}_{mn} - \overline{B}_{mn}\right]\tfrac{m\pi}{c}\cos\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\sin\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_x \\ & +\tfrac{n\pi}{d}\sin\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\cos\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_y\}\}\end{aligned} \tag{2.16}$$

$$\begin{aligned}\mathbf{E}_t\ |_{z=0+}= \quad & \textstyle\sum_{p=-P}^{P}\sum_{q=-P}^{P}\{[E_{pq} + F_{pq}]\{j\frac{2q\pi}{b}\hat{a}_x + j\frac{2p\pi}{a}\hat{a}_y\} \\ & + \tfrac{k_{z_{pq}}}{jk_o Z_o \mu}\left[\overline{E}_{pq} + \overline{F}_{pq}\right]\{\tfrac{-2p\pi}{a}\hat{a}_x + \tfrac{-2q\pi}{b}\hat{a}_y\}\Big\}\, e^{-j\left(\frac{2p\pi}{a}x + \frac{2q\pi}{b}y\right)}\end{aligned} \tag{2.17}$$

Similarly, the total tangential magnetic fields must be matched at the interface. That is,

$$\mathbf{H}_t(z = 0-) = \mathbf{H}_t(z = 0+),\ |x| < \frac{c}{2} \text{ and } |y| < \frac{d}{2} \tag{2.18}$$

Now the total tangential magnetic fields at $z = 0-$ and $z = 0+$ can be written as

$$\begin{aligned}\mathbf{H}_t\ |_{z=0-}= \quad & \textstyle\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\Big\{[A_{mn} - B_{mn}]\,\frac{k_{z_{mn}}}{k_o Z_o \mu}\{\frac{m\pi}{c}\sin\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\times \\ & \cos\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_x + \tfrac{n\pi}{d}\cos\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\sin\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_y\} \\ & + \left[\overline{A}_{mn} + \overline{B}_{mn}\right]\{\tfrac{n\pi}{d}\sin\left[\frac{n\pi}{d}\left(x+\frac{c}{2}\right)\right]\cos\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_x \\ & +\tfrac{-m\pi}{c}\cos\left[\frac{m\pi}{c}\left(x+\frac{c}{2}\right)\right]\sin\left[\frac{n\pi}{d}\left(y+\frac{d}{2}\right)\right]\hat{a}_y\}\end{aligned} \tag{2.19}$$

$$\begin{aligned}\mathbf{H}_t\ |_{z=0+}= \quad & \textstyle\sum_{p=-P}^{P}\sum_{q=-P}^{P}\Big\{[E_{pq} + F_{pq}]\,\frac{k_{z_{pq}}}{jk_o Z_o \mu}\{\frac{-2p\pi}{a}\hat{a}_x + \frac{-2q\pi}{b}\hat{a}_y\} \\ & + \left[\overline{E}_{pq} + \overline{F}_{pq}\right]\{j\tfrac{-2q\pi}{b}\hat{a}_x + j\tfrac{2p\pi}{a}\hat{a}_y\}\}\, e^{-j\left(\frac{2p\pi}{a}x + \frac{2q\pi}{b}y\right)}\end{aligned} \tag{2.20}$$

The orthogonality of the modal functions is used to sift out a particular modal expansion coefficient in the cavity by performing the following operations:

$$\begin{aligned}&\int_{-\frac{d}{2}}^{\frac{d}{2}}\int_{-\frac{c}{2}}^{\frac{c}{2}}(A_{mn} + B_{mn})\cdot\left(E_x^{TE_{mn}} + E_y^{TE_{mn}}\right)_{cavity}\cdot\left(\mathbf{E}_t(z = 0-)\right)dxdy = \\ &\int_{-\frac{d}{2}}^{\frac{d}{2}}\int_{-\frac{c}{2}}^{\frac{c}{2}}(A_{mn} + B_{mn})\cdot\left(E_x^{TE_{mn}} + E_y^{TE_{mn}}\right)_{cavity}\cdot\left(\mathbf{E}_t(z = 0+)\right)dxdy\end{aligned} \tag{2.21}$$

$$\int_{-\frac{d}{2}}^{\frac{d}{2}} \int_{-\frac{c}{2}}^{\frac{c}{2}} \left(\overline{A}_{mn} - \overline{B}_{mn}\right) \cdot \left(E_x^{TM_{mn}} + E_y^{TM_{mn}}\right)_{cavity} \cdot \left(\mathbf{E}_t(z = 0-)\right) dxdy = \\ \int_{-\frac{d}{2}}^{\frac{d}{2}} \int_{-\frac{c}{2}}^{\frac{c}{2}} \left(\overline{A}_{mn} - \overline{B}_{mn}\right) \cdot \left(E_x^{TM_{mn}} + E_y^{TM_{mn}}\right)_{cavity} \cdot \left(\mathbf{E}_t(z = 0+)\right) dxdy \tag{2.22}$$

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (E_{pq} + F_{pq}) \cdot \left(E_x^{TE_{pq}} + E_y^{TE_{pq}}\right)_{free-space} \cdot \left(\mathbf{H}_t(z = 0-)\right) dxdy = \\ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (E_{pq} + F_{pq}) \cdot \left(E_x^{TE_{pq}} + E_y^{TE_{pq}}\right)_{free-space} \cdot \left(\mathbf{H}_t(z = 0+)\right) dxdy \tag{2.23}$$

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (\overline{E}_{pq} + \overline{F}_{pq}) \cdot \left(E_x^{TM_{pq}} + E_y^{TM_{pq}}\right)_{free-space} \cdot \left(\mathbf{H}_t(z = 0-)\right) dxdy = \\ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (\overline{E}_{pq} + \overline{F}_{pq}) \cdot \left(E_x^{TM_{pq}} + E_y^{TM_{pq}}\right)_{free-space} \cdot \left(\mathbf{H}_t(z = 0+)\right) dxdy \tag{2.24}$$

Equation (2.21) produces an expression for coefficients $(A_{mn} + B_{mn})$ in terms of coefficients $E, F, \overline{E}, \overline{F}$. Coefficients $(\overline{A}_{mn} - \overline{B}_{mn})$ can be related to coefficients $E, F, \overline{E}, \overline{F}$ from equation (2.22). Equation (2.23) provides an expression for $(E_{pq} + F_{pq})$ in terms of $A, B, \overline{A}, \overline{B}$, and equation (2.24) gives $(\overline{E}_{pq} + \overline{F}_{pq})$ in terms of $A, B, \overline{A}, \overline{B}$. B and $\overline{B}$ can be eliminated using the relation of equation (2.7) and F and $\overline{F}$ are the known incident TE and TM fields; therefore, there are four equations and four unknowns which can be solved using standard matrix methods.

Once the solution for the scattered field at the cavity aperture is found, the scattered far field can be written as [1]:

$$\mathbf{E}(\mathbf{r}) = \frac{e^{-jkr}}{r} \left[A_\theta \hat{a}_\theta + A_\phi \hat{a}_\phi\right] \tag{2.25}$$

where

$$\begin{bmatrix} A_\theta \\ A_\phi \end{bmatrix} = \tfrac{jk_o}{2\pi} e^{jk_o \cos\theta r} \times \\ \int_{-\frac{c}{2}}^{\frac{c}{2}} dx \int_{-\frac{d}{2}}^{\frac{d}{2}} dy \begin{bmatrix} E_x \cos\phi + E_{y\,\sin\phi} \\ \cos\theta \left(-E_x \sin\phi + E_y \cos\phi\right) \end{bmatrix} e^{jk_o \sin\theta(\cos\phi x + \sin\phi y)}. \tag{2.26}$$

The radar cross section is calculated from

$$RCS = \lim_{r\to\infty} 4\pi r^2 \left|\frac{E_s}{E_i}\right|^2, \tag{2.27}$$

where E_s is the scattered field and E_i is the incident field. The amplitude of the incident field is taken to be unity. Using equations (2.25) and (2.27) the radar cross section due to the cavity is [1]:

$$RCS = 4\pi \left|\mathbf{A}\right|^2. \tag{2.28}$$

In general, the RCS is determined for a range of incident angles (θ, ϕ) and frequencies. A sequential code contains double loops over aspect angle and frequency while the parallel implementation steps through the range of incident angles but simultaneously calculates the RCS over the range of frequencies. The following sections will discuss parallel implementation of the theory described above.

3. Parallel Concepts

There are two broad categories of parallelism to be considered from which other subcategories could be defined: data parallelism and control parallelism. Machines employing data parallelism perform operations simultaneously on large data sets while those employing control parallelism run different parts of the algorithm simultaneously on different processors. Two other terms used to describe the type of parallelism performed by a machine are fine-grained versus coarse-grained parallelism. In fine-grained parallelism one elemental portion of a problem resides on each processor whereas in coarse-grained parallelism several elemental portions may be grouped onto a single processor.

Since the need for greater computational speed is the primary reason for considering parallel processing, it is expedient to have some convenient measure of comparison between the performance of a parallel algorithm and its sequential counterpart. One measure of performance is speed-up and is defined as the time required for sequential program execution divided by the time required for execution on N processors. Other parameters of interest for evaluation are scalability and portability. Scalability refers to the effect on speed-up as more or less processors are used as the size of a problem increases or decreases. Portability indicates how readily a code can be executed on different machine architectures.

The particular architecture of a parallel computer plays an important role in determining scalability, portability, and speed-up of a particular algorithm. This is because the inherent parallelism in a problem is usually suited for implementation on one of two basic architecture classifications: Single Instruction Multiple Data (SIMD) or Multiple Instruction Multiple Data (MIMD). SIMD machines are usually preferable for problems with a high degree of inherent data parallelism, while MIMD machines are typically used for problems that are inherently control parallel. The reader is referred to the literature for a more detailed discussion of these classifications [6].

The parallelized RCS cavity code was executed on a MasPar MP-1 massively parallel computer consisting of 4,096 processor elements (PEs) connected in a two-dimensional toroidal wrap topology [7]. It is a fine-grained, data-parallel processing system that operates in a SIMD fashion. For the MP-1 used here, each PE has 16 KBytes of memory and they are arranged in a square matrix having 64 columns and 64 rows with toroidal wrap[8].

There are two programming languages available for use on the MP-1: MasPar Parallel Application Language (MPL) and MasPar Fortran (MPF). MPL is based on Kernighan and Ritchie C. MPF implements Fortran 77 as well as a portion of Fortran 90 programming features [9]. The parallel implementation of the sequential RCS Cavity code described here was accomplished using the MPF programming language.

4. Implementation of Parallel Structures

First, the inherent parallelism in the algorithm must be identified. The two main outer DO loops in this application are over frequency and aspect angle, respectively. From the previous discussion, it can be seen that this problem is ideally suited for a SIMD machine such as the MP-1 because it clearly contains a high degree of data parallelism. We chose frequency to be the data parallel variable which eliminated the frequency DO loop from the algorithm. Every scalar variable depending on frequency became a vector, and every array depending on frequency increased by one dimension.

Second, the sequential code is compiled and executed without modification on the front end of the parallel computer to make sure it runs without error before any attempt at parallelization has begun. It is very useful when debugging the parallel code as it serves to generate intermediate data values. Next COMMON blocks that contain both scalar and array variables must be identified in order to separate them into blocks containing either scalars or arrays exclusively. The reason for this is that any arrays used in a Fortran 90 manner (i.e., operated on in parallel) must be allocated to the DPU whereas all other scalars and arrays reside on the front end. In addition, when using Fortran 90 constructs on COMMON block arrays the *nofecommon* compiler directive must be used or an error will occur at compile time. When *nofecommon* is used, all numeric and logic arrays in COMMON blocks are placed on the DPU while character arrays and scalars are placed on the front end. For this reason, it is important that arrays used only in a Fortran 77 context not be placed in COMMON blocks because each time they are used the compiler must move each element of the array from the PEs to the front end, significantly slowing execution of the program. Also, EQUIVALENCE statements are not allowed for arrays used in a Fortran 90 manner. At this point it is useful to briefly describe how the compiler maps data to the 64 x 64 PE Array in MPF.

The first dimension in a two-dimensional array will be mapped in the x direction, and the second dimension will be mapped in the y direction. If a dimension in either x or y exceeds 64, the elements will be layered in memory on the PEs. For an array of rank three or higher, the third dimension and above will be layered in memory. A one-dimensional array is mapped in a raster-scan fashion: successive elements wrap around the array from top row to bottom row. If the number of array elements exceeds the number of PEs, the wraparound proceeds into successive layers of memory. Thus, care should be taken when dimensioning arrays to make the most efficient use of the PE Array.

Although most of the parallelism was achieved by executing the code for all frequencies of interest simultaneously, further parallelism was obtained by converting some of the Fortran 77 contructs to Fortran 90. For instance, initialization of all arrays was done in parallel through a single assignment statement rather than a DO loop as shown in Table 1.

Another very important Fortran 90 feature is the WHERE statement. It must replace any IF statements in the sequential code when the variable in the logical expression has become an array. IF statements operate on scalars, but WHERE statements may operate on each element of an array. In addition, all array variables defined within a WHERE block must be conformable. Table 2 shows a portion of the code in which WHERE statements replaced IF statements. In the sequential code the variables *test* and *hcon* were scalars. However, in the parallel code these variables became vectors because they were frequency dependent. Thus, the expression $TEST > 10$ might be true for some frequencies and not for others. For those

PEs whose variable *test* was greater than 10, the corresponding variable *hcon* was given a value determined by the first assignment statement. The *hcon* variable on all other PEs would be assigned values according to the assignment statement in the ELSE WHERE portion of the block.

Sequential Code	Parallel Code
implicit complex(h)	implicit complex(h)
dimension hn5(5,5)	dimension hn5(64,5,5)
dimension hn6(5,5)	dimension hn6(64,5,5)
dimension hn7(5,5)	dimension hn7(64,5,5)
dimension hn8(5,5)	dimension hn8(64,5,5)
do 5 i = 1,5	hn5 = h1
do 6 j = 1,5	hn6 = h0
	hn7 = h0
hn5(i,j) = h1	hn8 = h1
hn6(i,j) = h0	
hn7(i,j) = h0	
hn8(i,j) = h1	
6 continue	
5 continue	

Table 1: Initialization Statements

Sequential Code	Parallel Code
implicit complex(h)	implicit complex(h)
dimension hgi(5,5)	dimension hgi(64,5,5)
real test	dimension hcon(64)
	real test(64)
if(test.gt.10.) then	where(test > 10.)
hcon = cmplx(1.E+10)	hcon = cmplx(1.E+10)
else	else where
hcon =	hcon=
cexp(cmplx(hgi(i,j)))	exp(cmplx(hgi(:,i,j)))
endif	end where

Table 2: WHERE construct versus IF statement.

An interesting problem arose when rewriting the subroutine which uses the Gauss-Jordan method to solve the system of linear equations. Table 3 shows the partial pivoting portion of the routine that searches down the column for the pivot value.The problem here was that a separate matrix for each frequency of interest had to be evaluated simultaneously, and it was very unlikely that the pivot value would occur in the same row for each matrix. In addition, the IF statement used in the sequential code was invalid here because the variable for the pivot value, *anorm*, was no longer a scalar since there was a different pivot value for each frequency. Another problem was that *anorm* was not conformable to the arrays being assigned values within the WHERE block for exchanging rows. Thus it was necessary to define two conformable arrays *anorm1* and *anorm2* by using the intrinsic function SPREAD. SPREAD is a transformational function that creates an array of rank one greater than the source array. In this way the values of one-dimensional *anorm* were copied to two-dimensional *anorm1* and *anorm2* which were then conformable to the arrays being operated on in the WHERE block. Another interesting problem in this routine occurred in the statement that checked for a matrix singularity. The sequential code checks the scalar variable *anorm* to see if it is equal to zero. If it is, an error message is written to the screen and program execution halts. We implemented this check in parallel by using the intrinsic functions MINVAL and MINLOC. MINVAL returned the lowest value in the array *anorm*. If the lowest value returned was equal to zero; i.e., a singular matrix occurred for at least one frequency, an error message was written to the screen giving the location of the array element where the singular matrix occurred (using MINLOC) and program execution terminated.

This section has highlighted some of the modifications made to the sequential code so that it would execute in parallel on the MP-1. Although some of the specific details are peculiar to the MP-1, the general methodology is applicable to a wide range of parallel machines. The next section provides a comparison of run times for the sequential versus parallel code.

```
                         Parallel Code

implicit complex(h)
dimension ha1(64,40,80)
real anorm(64), anorm1(64,80), anorm2(64,80)
complex xc(64,80)

do i = 1,12 ! COLUMNS
   if(i.ne.n) then
do j = i+1,12! ROWS

anorm = abs(real(ha1(:,j,i))) + abs(aimag(ha1(:,j,i)))
anorm1 = spread(anorm,dim=2,ncopies=80)
anorm = abs(real(ha1(:,i,i))) + abs(aimag(ha1(:,i,i)))
anorm2 = spread(anorm,dim=2,ncopies=80)

where(anorm1.gt.anorm2)
   xc = ha1(:,i,:) !EXCHANGE ROWS
   ha1(:,i,:) = ha1(:,j,:)
   ha1(:,j,:) = xc
end where

end do

!CHECK TO SEE IF PIVOT IS TOO SMALL
```

Table 3: Parallel version of Gauss-Jordan partial pivoting routine.

5. Timing Results

In this section we compare execution times for the sequential and parallel versions of the code discussed above. The sequential code was executed on a SPARCstation IPX. It should be noted that there is no unique way to compare the performance of the two machines [10]. Based on manufacturer's specifications, we estimate that for integer operations this workstation is about 20 times faster than a single processor of the MasPar whereas for floating point operations the ratio is about 60. Since the present application involves both integer and floating point operations, we will assume in the remainder of this paper that the ratio of the speeds is 40.

For timing and output comparisons of the sequential versus parallel code, we considered a total of 24 cavity modes and 1,849 free-space modes (to provide adequate convergence of the solution). The frequency range in both cases was from 9 to 11 GHz with the number of steps taken ranging from 1 to 255 for the sequential code and from 1 to 266 for the parallel version. Only one aspect angle was considered for this timing information as aspect-angle-dependent variables were not parallelized. Figure 2 plots the runtime in seconds versus the number of frequencies for the parallel version. Figure 3 plots the same variables for the sequential code. It can be seen that for a single frequency the sequential code ran faster than the parallel code. This is, in part, due to the fact that the sequential code was executed on the SPARCstation IPX. However, it is interesting to note that even with the slower individual processor speed on the MP-1, at 35 frequencies the parallel version executed faster than the sequential code. In addition, Figure 3 shows a linear increase in run time versus number of frequencies for the sequential code while Figure 2 shows an almost constant run time versus frequency for the parallel version with discrete jumps at multiples of 64 frequencies. A fit describing the timing data for the sequential code is $T_s(N_f) = 4.9 \cdot N_f$ where N_f is the total number of frequencies considered. Likewise, a fit describing the timing data for the parallel code is $T_p(N_f) = 93 + 20 \cdot \left\lceil \frac{N_f}{64} \right\rceil$. The constant term reflects the overhead associated with setting up the arrays. The discrete jumps in the parallel timings are due to the layering in memory that occurs (as explained in Section 4) each time the number of frequencies considered exceed the physical dimensions of the PE grid in the x direction (the first dimension of all frequency-dependent array variables was frequency). Timing information was obtained for three of the more computation-intensive subroutines in the program using the modified code. These subroutines were identified by looking at the percentage of time spent in each program segment. Subroutines *summa* and *sig* match the fields at the cavity aperture and calculate the matrix elements by summing over the cavity and free-space modes. Subroutine *cgjr* utilizes the Gauss-Jordan method to solve a system of linear equations with complex coefficients. Figure 4 shows that each graph retains the general shape of Figure 2 as would be expected for frequency-dependent computations.

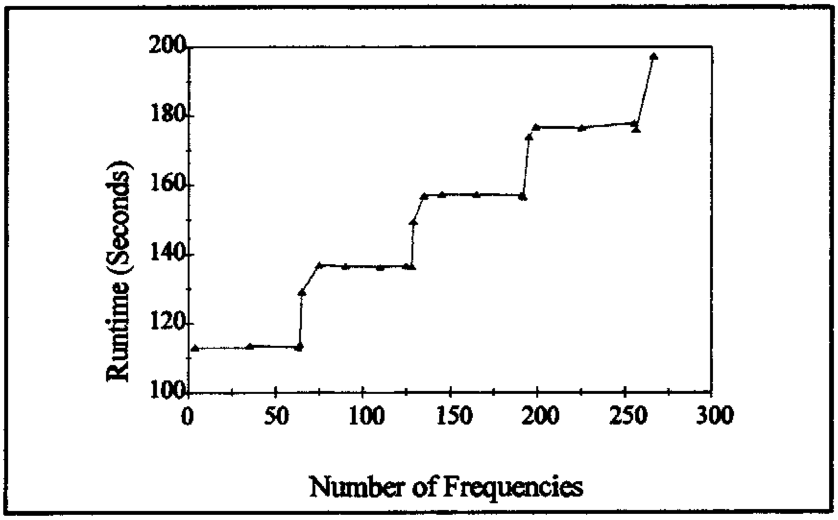

Fig. 2: Runtime vs. Number of Frequencies (Parallel Code)

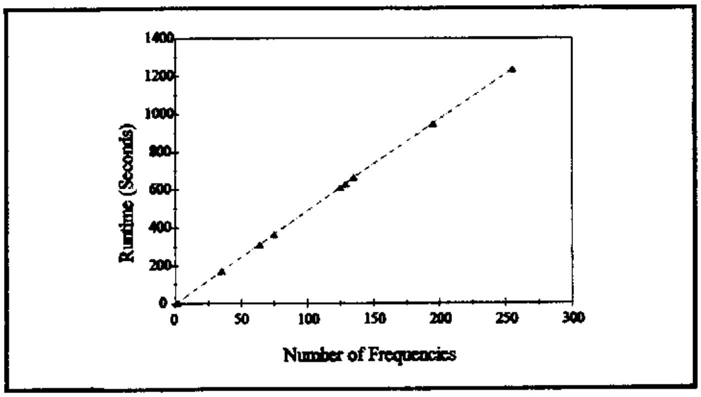

Fig. 3: Runtime vs. Number of Frequencies (Sequential Code)

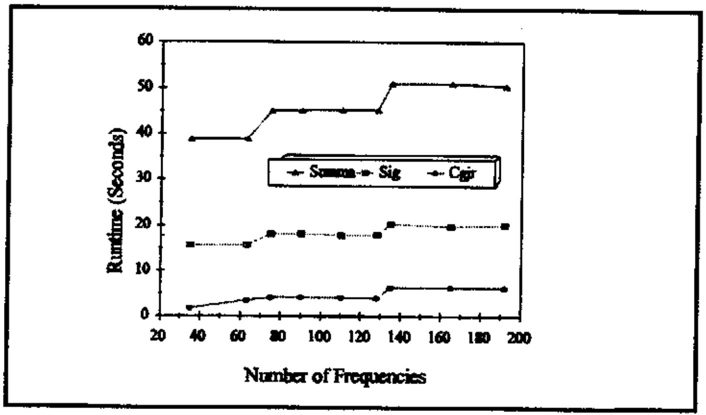

Fig. 4: Runtime vs. Number of Frequencies (Individual Subroutines in Parallel Code)

As indicated in Section 3, speed-up, scalability, and portability are three common measures of a code's performance. Figure 5 is a plot of speed-up versus number of frequencies. Recall that speed-up is defined as the ratio of runtime on a single processor versus runtime on N processors. The speed-up achieved in this case ranged from 1.5 for 35 frequencies to 6.95 for 255 frequencies. It is important to remember that the sequential code was timed on the SPARCstation IPX. Thus, from the previous discussion, a speed-up of 1.5 and 6.95 would translate to 60 and 278, respectively, if the parallel processors operated at the same speed. As indicated in Figure 5, there was a slight decrease in speed-up as the boundary of the PE grid was reached at each multiple of 64 frequencies. This is most likely due to increased router communication at the PE grid boundary and is a common feature in this type of application [11][12].

Speed-up continued to increase as the number of frequencies increased, indicating a scaling of code performance versus problem size for the portion of the code that was parallelized. Execution time for 64 frequencies, 20 aspect angles, 24 cavity modes, and 1,849 free-space modes was 52.5 hours. This execution time does, however, compare favorably to the 150 hours that would be required for the sequential code.

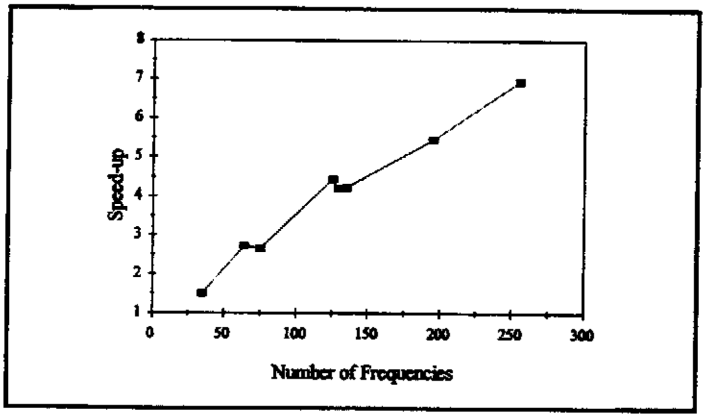

Fig. 5: Speed-up of Parallel to Sequential Code

6. Conclusions

While significant speed-up was achieved by exploiting the parallelism inherent in the frequency-dependent portion of the algorithm, performance could in principle be further improved by parallelizing the code over aspect angle as well. An attempt was made to do this, but the amount of memory required quickly exceeded that available on the MP-1 due to the fact that many more scalar variables became arrays and arrays that were both frequency and aspect angle dependent increased by yet one more dimension. However, as parallel computers with increased memory space become available further speed enhancement is likely since parallelization methodologies are of general validity (not restricted to the MP-1). Moreover, it is likely that more inherently parallel algorithms could be developed by carefully rethinking problems with a massively parallel platform in mind. For instance, the number of modes required for an accurate solution is proportional to the size of the wavelength cross section in wavelength[1]. Thus, execution time for both the sequential and parallel codes discussed here becomes prohibitive for waveguides greater than approximately three wavelengths in size. However, by rethinking the problem to parallelize execution over modes as well as frequency and aspect angle, the modal analysis technique might still be a valid solution method for larger waveguides. Nevertheless, as demonstrated here, significant increases in code performance can already be realized by modifying existing sequential codes for use on massively parallel machines.

We have already mentioned that portability is an important issue in considerations of parallel programs. A code that has been extensively fine tuned for a specific parallel computer may be highly efficient, but it will be difficult to maintain and update and may not easily be adapted to other architectures. In this context it is important to note that the present code has been written in a language that is essentially the Fortran 90 standard and thus can be ported with relative ease to other platforms supporting Fortran 90. Moreover, new Fortran standards are currently under development, including Fortran D and High Performance Fortran languages. The former provides facilities for automatic parallelization by the compiler. Thus Fortran will continue to be one of the prime languages for parallel computation and will most likely not be surpassed in terms of portability at least for the foreseeable future. A second issue that needs to be addressed is the code's dependence on the specific interconnection network of the parallel machine such as, in the case of the MasPar, a two-dimensional mesh with toroidal wrapping. It is known [6] that this topology may be embedded with dilation 1 in the three-dimensional mesh and the hypercube. Thus the present implementation can be used without any major changes on these other commonly encountered architectures. Most compilers will automatically allocate processors according to the desired topology ("logical" grid as opposed to "physical" grid).

In summary, execution time for the sequential code is impractical for obtaining wideband data when an adequate number of modes is considered for accurate results. However, since the code has been adopted by the U.S. Electromagnetic Code Consortium it was of interest to determine whether significant speed-up could be achieved by parallelizing the code. The algorithm is well suited for SIMD implementation since it possesses a high degree of data parallelism evident in the outer loops over frequency and aspect angle. Even though memory restrictions precluded parallelizing the algorithm over both frequency and aspect angle to achieve even greater speed-up, this kind of effort will only gain in importance as more and more powerful machines with faster processors and increased memory space enter the market place.

7. Acknowledgements

The authors would like to thank S. W. Lee for his invaluable assistance in the completion of this work. The authors would also like to acknowledge G. G. Dominick for his help in preparing the paper and the U.S. Department of Energy for fellowship support provided to L. D. Vann through the Computational Science Graduate Fellowship Program administered by Ames Laboratory.

References

[1] Lee, S.W., H. Ling, and L. Lin. *"RCS of Circular Waveguide Cavity."* Final Report for the period November 1989 through May 1990, AD-B154 226, Defense Electromagnetic Analysis Company, 5 Moraine Court, Champaign, IL 61821-5263, March 1991. Prepared for Headquarters Ballistic Missile Organization/MYET.

[2] Harrington, R. F. *Time-Harmonic Electromagnetic Fields.* New York: McGraw-Hill, 1961.

[3] Balanis, C. A. *Advanced Engineering Electromagnetics.* New York: John Wiley & Sons, 1989.

[4] Kong, J. A. *Electromagnetic Wave Theory.* New York: John Wiley & Sons, Inc., 1990.

[5] Mittra, R., and S. W. Lee. *Analytical Techniques in the Theory of Guided Waves.* New York: The Macmillan Company, 1971.

[6] Kumar, V., A. Grama, A. Gupta, and G. Karypis. *Introduction to Parallel Computing - Design and Analysis of Algorithms.* Redwood City: Benjamin/Cummings, 1994.

[7] Hwang, K. *Advanced Computer Architecture - Parallelism, Scalability, Programmability.* New York: McGraw-Hill, 1993.

[8] Blank, T.*"The MasPar MP-1 Architecture."* Proceedings of the IEEE Compcon Spring 1990, IEEE, February 1990.

[9] Metcalf, M. and J. Reid. *Fortran 90 Explained.* Oxford: Oxford University Press, 1990.

[10] Hennessy, J. L., and D. A. Patterson. *Computer Architecture: A Quantitative Approach.* San Mateo: Morgan Kaufmann, 1990.

[11] Wille, L. T., J. L. Rogers, C. P. Burmester, and R. Gransky. *"Towards First-principles Theories of Materials and Biological Systems - The Need for Massive Parallelism."* Future Generation Computer Systems, **10**, 331 (1994).

[12] David, R. S. and L. T. Wille. *"Massively Parallel Finite-Difference Time-Domain Methods For Electromagnetic Scattering Problems."* Proceedings of the Applied Computational Electromagnetics Society 1994 Meeting, March 1994, Vol. I, pp. 495-502.

[13] Personal communication with Dr. S. W. Lee, August 1993.