2007

A LOCALLY CORRECTED NYSTRM METHOD FOR SURFACE INTEGRAL EQUATIONS: AN OBJECT ORIENTED APPROACH

Bryan James Guernsey
University of Kentucky, bryanguernsey@gmail.com

ABSTRACT OF THESIS

A LOCALLY CORRECTED NYSTRÖM METHOD FOR SURFACE INTEGRAL EQUATIONS: AN OBJECT ORIENTED APPROACH

Classically, researchers in Computational Physics and specifically in Computational Electromagnetics have sought to find numerical solutions to complex physical problems. Several techniques have been developed to accomplish such tasks, each of which having advantages over their counterparts. Typically, each solution method has been developed separately despite having numerous commonalities with other methods. This fact motivates a unified software tool to house each solution method to avoid duplicating previous efforts. Subsequently, these solution methods can be used alone or in conjunction with one another in a straightforward manner. The aforementioned goals can be accomplished by using an Object Oriented software approach. Thus, the goal of the presented research was to incorporate a specific solution technique, an Integral Equation Nyström method, into a general, Object Oriented software framework.

KEYWORDS: Electromagnetics, Integral Equations, Nyström Method, High Order Solution Method, Object Oriented Software

Bryan James Guernsey

July 23, 2007

A LOCALLY CORRECTED NYSTRÖM METHOD FOR SURFACE INTEGRAL EQUATIONS: AN OBJECT ORIENTED APPROACH

By

Bryan James Guernsey

Dr. Robert J. Adams
Director of Thesis
Dr. Stephen D. Gedney
Co-Director of Thesis
Dr. YuMing Zhang
Director of Graduate Studies
July 23, 2007

RULES FOR THE USE OF THESES

Unpublished theses submitted for the Master's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgments.

Extensive copying or publication of the thesis in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this thesis for use by its patrons is expected to secure the signature of each user.

Name	Date

THESIS

Bryan James Guernsey

The Graduate School

University of Kentucky

2007

A LOCALLY CORRECTED NYSTRÖM METHOD FOR SURFACE INTEGRAL EQUATIONS: AN OBJECT ORIENTED APPROACH

THESIS

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in the
College of Electrical Engineering
at the University of Kentucky

By

Bryan James Guernsey

Lexington, Kentucky

Director: Dr. Robert J. Adams, Professor of Electrical Engineering
Co-Director: Dr. Stephen D. Gedney, Professor of Electrical Engineering

Lexington, Kentucky

2007

Table of Contents

List of Tables

List of Figures

Introduction:

Research areas in Computational Physics are ever expanding as computing technology continues to grow. An important goal of these research efforts is to solve very large, complex problems in a fast, efficient, and accurate manner. Consider the development of a new GPS (Global Positioning System) satellite. This is a very complex problem (certainly beyond the scope of an analytical solution) with thousands if not millions of unknowns. Researchers have developed and will continue to improve upon techniques to model these satellites such that they function properly when put into orbit.

One approach to modeling problems of the type posed above is to use an Integral Equation (IE) method. IE methods seek to pose integral based solutions to the underlying physical equations which govern the proposed problem. In a computational approach, these equations are then discretized to yield a numerical solution which can be found through successive computations.

IE methods are not the only approach used to solve these problems; others include Finite Element Methods (FEM) and time domain methods such as the Finite Difference Time Domain (FDTD). Each of these methods has advantages over the others but they all have several commonalities. Classically, new simulation codes are developed for each new solution technique. These new codes come at the cost of reworking common bonds between methods. Thus, a general framework which houses all of these solution techniques would be advantageous. This can be accomplished through Object Oriented coding.

This thesis will focus on one area of Computational Physics, Electromagnetics, and a specific IE method used to solve related problems, The Locally Corrected Nyström Method. First, a review of the governing physical equations and properties of Electromagnetic IE problems will be presented. This will be followed by an overview and subsequent discussion of the Locally Corrected Nyström Method. Further motivation for an Object Oriented design will be presented along with an overview of the Nyström

implementation. Finally, the presented methods will be validated by a series of numerical examples and performance results.

I. Review of Surface Integral Equations

1.0) Maxwell's Equations [1]

The fundamental laws that govern Electromagnetic (EM) phenomena are given by a set of vector equations known as Maxwell's equations, named after the famous Scottish mathematical physicist James Clerk Maxwell. These equations written in time domain, differential form are:

$$\nabla \times \vec{E} = -\vec{M}_i - \frac{\partial \vec{B}}{\partial t} \tag{1.0.1}$$

$$\nabla \times \vec{H} = \vec{J}_i + \vec{J}_c + \frac{\partial \vec{D}}{\partial t} \tag{1.0.2}$$

$$\nabla \cdot \vec{D} = \rho_v \tag{1.0.3}$$

$$\nabla \cdot \vec{B} = \rho_m \tag{1.0.4}$$

where $\vec{E}$ is the vector electric field in [V/m], $\vec{H}$ is the vector magnetic field in [A/m], $\vec{D}$ is the electric flux density in [C/m], $\vec{B}$ is the magnetic flux density in [Wb/m], $\vec{M}_i$ is the impressed magnetic current density in [V/m^2], $\vec{J}_i$ is the impressed electric current density in [A/m^2], $\vec{J}_c$ is the conduction electric current density in [A/m^2], ρ_v is the electric charge density in [C/m], and ρ_m is the magnetic charge density in [Wb/m]. Equations (1.0.1) and (1.0.2) are Faraday's and Ampere's Law respectively and equations (1.0.3) and (1.0.4) are the electric and magnetic forms of Gauss' Law.

The constitutive relations which supplement the above equations are as follows:

$$\vec{D} = \varepsilon * \vec{E} \tag{1.0.5}$$

$$\vec{B} = \mu * \vec{H} \tag{1.0.6}$$

$$\vec{J}_c = \sigma * \vec{E} \tag{1.0.7}$$

where ε is the time varying permittivity of the media in [F/m], μ is the time varying permeability of the media in [H/m], σ is the time varying conductivity of the media in [S/m] and where * represents a convolution.

This thesis will discuss Surface Integral Equations (SIEs) in the frequency domain, rather than the time domain. Thus the above equations need to be written in Time Harmonic form. Assuming an $e^{j\omega t}$ time dependence, equations (1.0.1-1.0.4) are written:

$$\nabla \times \vec{E} = -\vec{M}_i - j\omega\vec{B} \tag{1.0.8}$$

$$\nabla \times \vec{H} = \vec{J}_i + \vec{J}_c + j\omega\vec{D} \tag{1.0.9}$$

$$\nabla \cdot \vec{D} = \rho_v \tag{1.0.10}$$

$$\nabla \cdot \vec{B} = \rho_m \tag{1.0.11}$$

where ω is radian frequency of the EM waves. The constitutive relations remain the same for the frequency domain, however the convolution simply becomes a multiplication.

One final relation that needs to be presented is known as the Continuity Equation. The Continuity Equation is derived by taking the divergence of Ampere's Law, (1.0.9):

$$\nabla \times \vec{H} = \vec{J}_i + \vec{J}_c + j\omega\vec{D}$$
$$\nabla \times \vec{H} = \vec{J}_{ic} + j\omega\vec{D}$$
$$\nabla \cdot \left(\nabla \times \vec{H}\right) = \nabla \cdot \left(\vec{J}_{ic} + j\omega\vec{D}\right)$$

using the vector identity $\nabla \cdot \left(\nabla \times \vec{H}\right) = 0$, the above becomes

$$\nabla \cdot \left(\vec{J}_{ic} + j\omega\vec{D}\right) = 0$$
$$\nabla \cdot \vec{J}_{ic} = -\nabla \cdot j\omega\vec{D}$$

then applying Gauss' Law,

$$\nabla \cdot \vec{J}_{ic} = -j\omega\rho_v \tag{1.0.12}$$

Equation (1.0.12) is the final form for the Continuity Equation.

1.1) Boundary Conditions [1]

When EM waves propagate between two different media they experience discontinuities at the interface. In order to properly preserve Maxwell's equations at these interfaces

special treatment is needed in the form of boundary conditions. These boundary conditions dictate field behavior at material and conducting interfaces and are a natural consequence of Maxwell's equations. The derivation shown here follows from [1].

General Material Interface

Consider Figure 1.1.1 below, an interface between two materials with constitutive parameters ε_1, μ_1, σ_1 and ε_2, μ_2, σ_2. The derivation of the electric field boundary condition originates from Faraday's Law in integral form. Taking the surface integral of both sides of equation (1.0.8) yields:

$$\iint_S \left(\nabla \times \vec{E}\right) \cdot d\vec{s} = -\iint_S \vec{M}_i \cdot d\vec{s} - j\omega \iint_S \vec{B} \cdot d\vec{s} \tag{1.1.1}$$

Applying Stokes Theorem to the left hand side reduces (1.1.1) to,

$$\begin{aligned} &\iint_S \left(\nabla \times \vec{A}\right) \cdot d\vec{s} = \oint_C \vec{A} \cdot d\vec{l} \quad (Stokes'\ Theorem) \\ &\oint_C \vec{E} \cdot d\vec{l} = -\iint_S \vec{M}_i \cdot d\vec{s} - j\omega \iint_S \vec{B} \cdot d\vec{s} \end{aligned} \tag{1.1.2}$$

Equation (1.1.2) is the relation from which we will derive the electric field boundary condition.

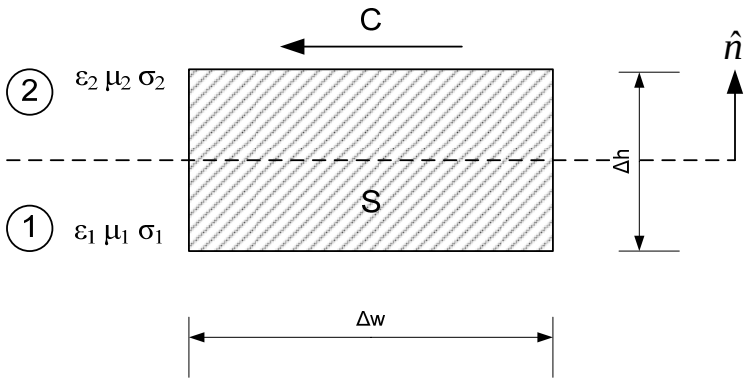

Figure 1.1.1 Material Interface for Field Boundary Conditions

As the height of the rectangle Δh becomes smaller the overall area S also becomes smaller. Therefore, in the limit as Δh goes to zero the contribution of the magnetic flux

surface integral term in (1.1.2) becomes negligible. In the same limit as height approaches zero the left hand side of (1.1.2) becomes:

$$\oint_C \vec{E} \cdot d\vec{l} = \vec{E}_1 \cdot \hat{a}_w \Delta w - \vec{E}_2 \cdot \hat{a}_w \Delta w \tag{1.1.3}$$

The magnetic current $\vec{M}_i$ is confined to a very thin layer between the two media. Because $\vec{M}_i$ is strictly a surface current as the height approaches zero the surface integration has a non-zero contribution to (1.1.2):

$$\begin{aligned} &\lim_{\Delta h \to 0} \iint_S \vec{M}_i \cdot d\vec{s} = \lim_{\Delta h \to 0} \left[\vec{M}_i \cdot \left(\hat{a}_w \times \hat{a}_h \right) \Delta w \Delta h \right] \\ &\lim_{\Delta h \to 0} \left[\vec{M}_i \cdot \left(\hat{a}_w \times \hat{a}_h \right) \Delta w \Delta h \right] = \vec{M}_s \cdot \left(\hat{a}_w \times \hat{a}_h \right) \Delta w \end{aligned} \tag{1.1.4}$$

Equating (1.1.3), (1.1.4), and applying some basic vector identities yields the final result for the electric field boundary condition:

$$\begin{aligned} &\vec{E}_1 \cdot \hat{a}_w \Delta w - \vec{E}_2 \cdot \hat{a}_w \Delta w = -\vec{M}_s \cdot \left(\hat{a}_w \times \hat{a}_h \right) \Delta w \\ &\left(\vec{E}_1 - \vec{E}_2 \right) \cdot \hat{a}_w = -\vec{M}_s \cdot \left(\hat{a}_w \times \hat{a}_h \right) \\ &\text{Realizing, } \hat{a}_h = \hat{n} \\ &\hat{n} \times \left(\vec{E}_2 - \vec{E}_1 \right) = -\vec{M}_s \end{aligned} \tag{1.1.5}$$

The boundary condition for the magnetic field can be derived in the same manner as above but rather beginning with Ampere's Law in integral form. Following the same steps yields the magnetic field boundary condition:

$$\hat{n} \times \left(\vec{H}_2 - \vec{H}_1 \right) = \vec{J}_s \tag{1.1.6}$$

If the boundary had no sources (i.e. $\vec{J}_s$ and $\vec{M}_s$ = 0), equations (1.1.5) and (1.1.6) dictate a classic result in that the electric and magnetic fields are tangential continuous across a boundary interface. A discontinuity in the tangential fields would necessitate a surface current to be present.

Now that the boundary condition on the fields has been established the discussion will continue with the boundary condition for the fluxes. The integral form of Gauss' Law can be derived by taking the volume integral of equation (1.0.10):

$$\iiint_V \nabla \cdot \vec{D} \, dv = \iiint_V \rho_v dv \tag{1.1.7}$$

The Divergence Theorem can then be used to simplify the left hand side of (1.1.7):

$$\begin{aligned} &\iiint_V \nabla \cdot \vec{A} dv = \oiint_S \vec{A} \cdot d\vec{s} \ \ (\textit{Divergence Theorem}) \\ &\oiint_S \vec{D} \cdot d\vec{s} = \iiint_V \rho_v dv \end{aligned} \tag{1.1.8}$$

Now refer to the cylindrical cutout between the two regions in Figure 1.1.2. There are two surface areas of interest, namely A and Ao. As the height of the cylinder vanishes the contribution of Ao to the surface integral in (1.1.8) is negligible. Assuming there are no charges or sources on the boundary, (1.1.8) can be rewritten as,

$$\oiint_A \vec{D} \cdot d\vec{s} = \vec{D}_2 \cdot \hat{n} A - \vec{D}_1 \cdot \hat{n} A = 0$$

Dividing both sides by the area yields the electric flux boundary condition:

$$\hat{n} \cdot \left(\vec{D}_2 - \vec{D}_1 \right) = 0 \tag{1.1.9}$$

A similar expression for the magnetic flux boundary condition can be derived from the magnetic form of Gauss' Law:

$$\hat{n} \cdot \left(\vec{B}_2 - \vec{B}_1 \right) = 0 \tag{1.1.10}$$

If there are charges along the surface of the interface the right hand side of (1.1.8) collapses simply into surface charges and the flux boundary conditions become,

$$\begin{aligned} \hat{n} \cdot \left(\vec{D}_2 - \vec{D}_1 \right) &= \rho_e \\ \hat{n} \cdot \left(\vec{B}_2 - \vec{B}_1 \right) &= \rho_m \end{aligned} \tag{1.1.11}$$

where ρ_e and ρ_m are the electric and magnetic surface charge densities on the interface.

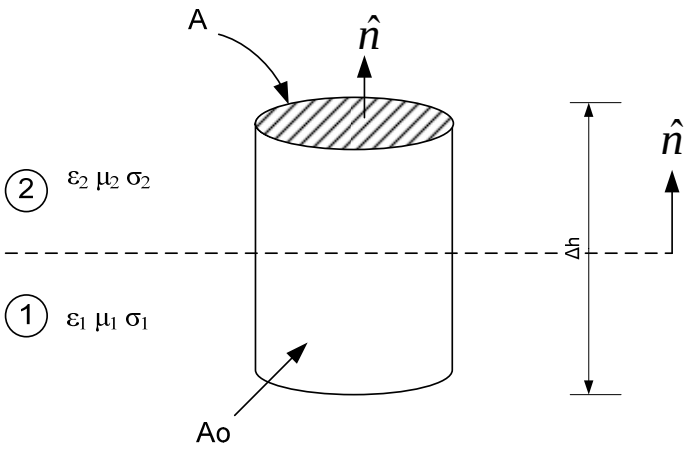

Figure 1.1.2 Material Interface for Flux Boundary Conditions

Perfectly Conducting Interface

There are two other interfaces to note before continuing the discussion of SIEs, a Perfect Electric Conductor (PEC) and a Perfect Magnetic Conductor (PMC). For a material to be a PEC it must have infinite electrical conductivity. As a consequence of having infinite electrical conductivity a PEC cannot support magnetic charge or magnetic current. Therefore,

$$\vec{M}_s\,|_{PEC} = 0, \rho_m\,|_{PEC} = 0$$

Similarly, a PMC cannot support electric charge or electric current:

$$\vec{J}_s\,|_{PMC} = 0, \rho_e\,|_{PMC} = 0$$

A property of both PECs and PMCs is that the total field within the conductor is zero. Assuming that region 1 is a PEC and utilizing the above properties, the boundary conditions can be written,

$$\begin{aligned} \hat{n}\times\vec{E}_2 &= 0 \\ \hat{n}\times\vec{H}_2 &= \vec{J}_s \\ \hat{n}\cdot\vec{D}_2 &= \rho_e \\ \hat{n}\cdot\vec{B}_2 &= 0 \end{aligned} \qquad (1.1.12)$$

Finally, if region 1 is a PMC then the boundary condition can be written,

$$\begin{aligned} \hat{n}\times\vec{E}_2 &= -\vec{M}_s \\ \hat{n}\times\vec{H}_2 &= 0 \\ \hat{n}\cdot\vec{D}_2 &= 0 \\ \hat{n}\cdot\vec{B}_2 &= \rho_m \end{aligned} \tag{1.1.13}$$

1.2) Equivalence Principle [1, 2]

A basic EM principle used in the derivation of SIEs is known as Huygen's Equivalence Principle. Consider the problem posed by Figure 1.2.1 of an arbitrary target illuminated by an incident EM wave. The problem is divided into two different regions, separated by the fictitious surface S. Region 1 is a homogeneous space with constitutive parameters ε_1 μ_1. Region 2 is an inhomogeneous space which can contain an arbitrary number of targets. For this example Region 2 contains both a PEC and material target. The incident EM wave is located in Region 1 and produces fields E_1, H_1 in Region 1 and E_2, H_2 in Region 2.

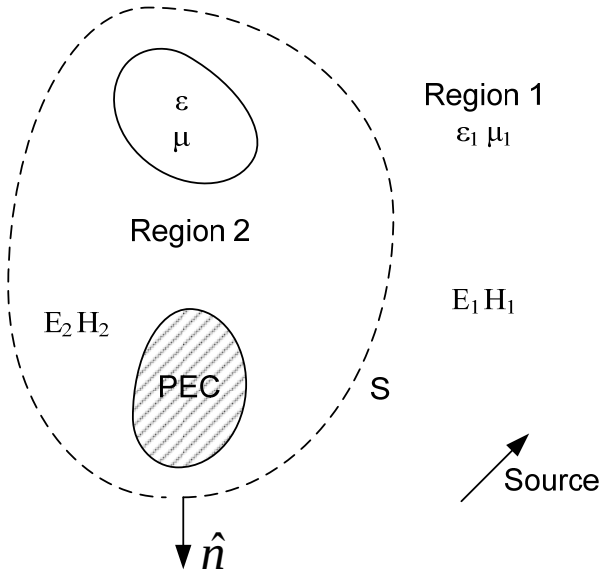

Figure 1.2.1 Equivalence Principle

Now consider the intermediate problem posed by Figure 1.2.2. Equivalent currents have been placed on the surface S. These currents satisfy the following relations:

$$\vec{J}_s = \hat{n} \times \vec{H}_1$$
$$\vec{M}_s = -\hat{n} \times \vec{E}_1$$

where $\hat{n}$ is the outward surface normal to S. Equivalence theory states that the combination of the original source and the equivalent sources produce the exact same fields in Region 1 as the original problem, namely $\vec{E}_1$ and $\vec{H}_1$. However, as a consequence of these equivalent sources the fields in Region 2 have been nullified via the extinction theorem. In other words, the fields in Region 2 are zero because the equivalent sources radiate negative the incident field within Region 2.

Since the fields in Region 2 are nullified by these equivalent currents the inhomogeneities can be modified in an arbitrary manner without affecting the fields in Region 1. A convenient choice would be to remove all the inhomogeneities and replace Region 2 with the same material parameters as Region 1 leaving both regions with the same homogeneous properties (Figure 1.2.3). This leaves an equivalent exterior problem with both the original incident EM source and the equivalent surface currents radiating in homogeneous space.

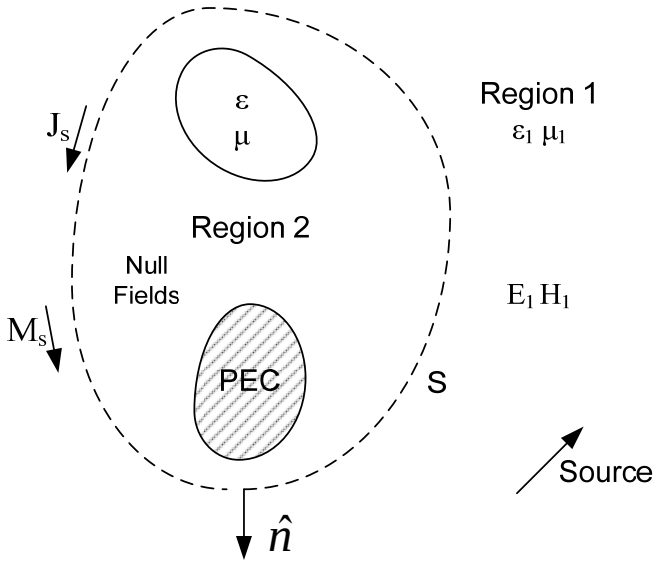

Figure 1.2.2 Intermediate Equivalence Principle

Thus, using Equivalence, the problem has been simplified to solving for the unknown surface currents $\vec{J}_s$ and $\vec{M}_s$ rather than the fields in both regions in the presence of the inhomogeneities. This principle is of practical interest because for typical problems (e.g. EM wave hitting an airplane) we are mostly concerned with the scattered field reflected from the target rather than the fields within the target.

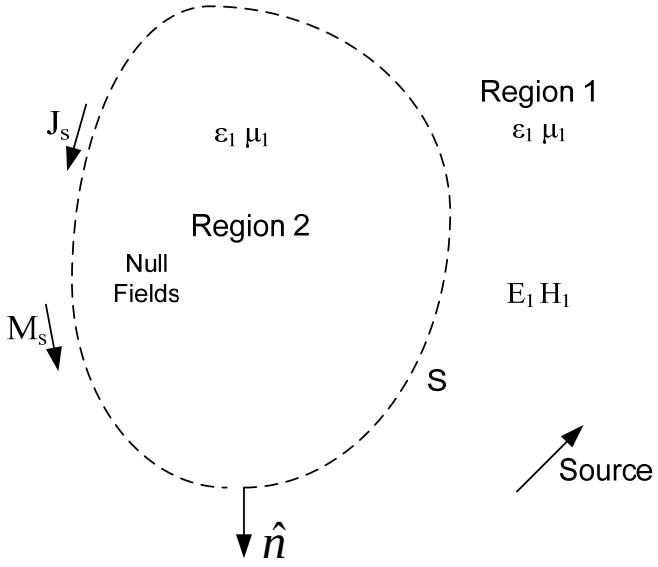

Figure 1.2.3 Equivalent Exterior Problem

1.3) Derivation of Surface Integral Equations [3, 4]

1.3.1) Electric Field Integral Equation

Now that some basic EM principles have been presented we can continue on to the discussion of Surface Integral Equations. The first SIE of interest is the Electric Field Integral Equation (EFIE). For simplicity let's first consider the EFIE on a PEC target.

Figure 1.3.1a depicts an arbitrarily shaped PEC target illuminated by an EM wave propagating in free space. We are interested in the scattered field off this target; from Equivalence, we know that the total field exterior to the target is equal to the sum of the incident and scattered fields:

$$\vec{E}^{tot} = \vec{E}^{inc} + \vec{E}^{scat} \tag{1.3.1}$$

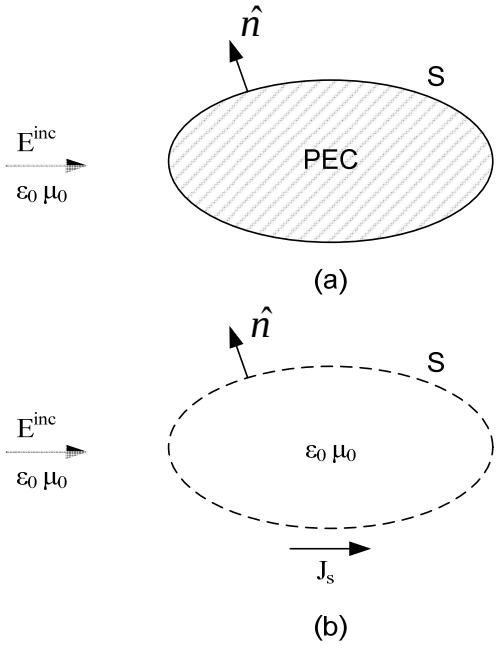

Figure 1.3.1 Scattering by PEC and Equivalent Problem

Figure 1.3.1(b) shows the equivalent problem of $\vec{E}^{inc}$ and $\vec{J}_s$ radiating in freespace. In order to solve equation (1.3.1) we need to move the problem to a location where the total field is known. According to equation (1.1.12) the total tangential electric field must be zero on the surface of a PEC. Thus (1.3.1) can be rewritten as,

$$\hat{n}\times\vec{E}^{inc}\,|_S = -\hat{n}\times\vec{E}^{scat}\,|_S$$
$$\text{or} \tag{1.3.2}$$
$$\hat{t}\cdot\vec{E}^{inc}\,|_S = -\hat{t}\cdot\vec{E}^{scat}\,|_S$$

where $\hat{t}$ is the unit tangent to the surface S. $\vec{J}_s$ radiates $\vec{E}^{scat}$. Their relationship is governed by the magnetic vector potential $\vec{A}$. From [1] $\vec{E}^{scat}$ can be written:

$$\vec{E}^{scat} = -jk_0\eta_0\vec{A} + \frac{\eta_0}{jk_0}\nabla\nabla\cdot\vec{A}$$
$$\vec{A}(\vec{r}) = \int_S \vec{J}_s\cdot G(\vec{r},\vec{r}\,')ds' \tag{1.3.3}$$
$$G(\vec{r},\vec{r}\,') = \frac{e^{-jk_0R}}{4\pi R} \;\; where\; R = \left|\vec{r}-\vec{r}\,'\right|$$

$G(\vec{r},\vec{r}\,')$ is known as the Freespace Green's Function and describes how a point source at $\vec{r}\,'$ radiates to an observation point $\vec{r}$. Combining equations (1.3.2) and (1.3.3) yields the EFIE:

$$\begin{aligned}\hat{t}\cdot\vec{E}^{inc} &= \hat{t}\cdot\left[-jk_0\eta_0\vec{A}+\frac{\eta_0}{jk_0}\nabla\Phi_e\right]\\ \Phi_e(\vec{r}) &= \nabla\cdot\vec{A} = \nabla\cdot\int_S \vec{J}_s(\vec{r}\,')\frac{e^{-jk_0|\vec{r}-\vec{r}\,'|}}{4\pi|\vec{r}-\vec{r}\,'|}ds'\end{aligned} \tag{1.3.4}$$

where η_0 is the freespace wave impedance and k_0 is the freespace wave number. $\Phi_e(\vec{r})$ is known as the electric scalar potential.

1.3.2) Magnetic Field Integral Equation

The next SIE of interest is the Magnetic Field Integral Equation (MFIE). Again consider the problem depicted in Figure 1.3.1. However, rather than solving for the electric field we will focus on the solution for the magnetic field. Again according to Equivalence we have the relation in the region outside of the PEC target,

$$\vec{H}^{tot} = \vec{H}^{inc} + \vec{H}^{scat} \tag{1.3.5}$$

Let's again move the problem to the surface of the target. The magnetic field boundary condition on a PEC is given by equation (1.1.12). Thus (1.3.5) can be written as,

$$\begin{aligned}\hat{n}\times\vec{H}^{tot}|_S &= \hat{n}\times\vec{H}^{inc}|_S + \hat{n}\times\vec{H}^{scat}|_S\\ \vec{J}_s &= \hat{n}\times\vec{H}^{inc}|_S + \hat{n}\times\vec{H}^{scat}|_S\end{aligned} \tag{1.3.6}$$

In this case, $\vec{J}_s$ again radiates $\vec{H}^{scat}$ and is governed by the magnetic vector potential $\vec{A}$. $\vec{H}^{scat}$ can thus be written [1],

$$\vec{H}^{scat} = \nabla\times\vec{A}$$

Plugging into (1.3.6),

$$\vec{J}_s = \hat{n}\times\vec{H}^{inc}|_S + \hat{n}\times\nabla\times\vec{A}|_S$$

The right hand side of (1.3.6) needs some special attention in the form of a Principle Value Integral (PVI) [5]. The problem in evaluating this integral lies in the fact that $\vec{H}$ is dual valued at the surface S. The details of this evaluation will not be discussed here, however the final result of the MFIE from [3, 5] becomes,

$$\hat{n}\times\vec{H}^{inc}\,|_S=\frac{1}{2}\vec{J}_s+\hat{n}\times\int_S\vec{J}_s\times\nabla G(\vec{r},\vec{r}\,')ds'$$

$$where\ \nabla G(\vec{r},\vec{r}\,')=\left(1+jk_0R\right)\cdot\frac{e^{-jk_0R}}{4\pi R^2}\cdot\frac{\vec{R}}{R} \tag{1.3.7}$$

$$and\ R=\left|\vec{r}-\vec{r}\,'\right|,\ \vec{R}=\vec{r}-\vec{r}\,'$$

1.3.3) Combined Field Integral Equation

The above EFIE and MFIE derivations are applicable to arbitrary targets, both open (e.g. thin PEC strip) and closed (e.g. sphere, dielectric slab) for the EFIE and simply open for the MFIE because of its dual valued nature. However, when solving problems involving closed targets both the EFIE and MFIE operators experience a break down effect known as Interior Resonance. Interior Resonance stems from the fact that at discrete frequencies the internal and external boundary conditions are simultaneously satisfied. This leads to a dual valued problem which has no unique solution. The reader will be referred to Chapters 5 and 6 of [2] for a more in depth proof of Interior Resonance.

There are several remedies for this phenomena one of which is known as the Combined Field Integral Equation (CFIE). For the same problem geometry the EFIE and MFIE operators experience resonance at different frequencies. The CFIE takes advantage of this fact by taking a linear combination of the EFIE and MFIE operators. The weight of each operator is chosen such that the CFIE has a unique solution at all frequencies. The CFIE operator is commonly written as [2],

$$CFIE=\alpha EFIE+(1-\alpha)\eta MFIE \tag{1.3.8}$$

where α is a scaling coefficient between 0 and 1. The wave impedance, η, is applied to the MFIE operator to ensure that both operators have the same scaling.

1.3.4) Integral Equation for Homogeneous Dielectric Bodies

Now that the SIEs for PEC targets have been discussed our focus will shift to dielectric bodies. The most commonly used SIE for EM scattering from dielectric targets is known as the PMCHWT formulation (named after Poggio, Miller, Chang, Harrington, Wu, and Tsai) [3, 6, 7]. Consider the problem of Figure 1.3.4 of a dielectric body illuminated by an incident EM wave.

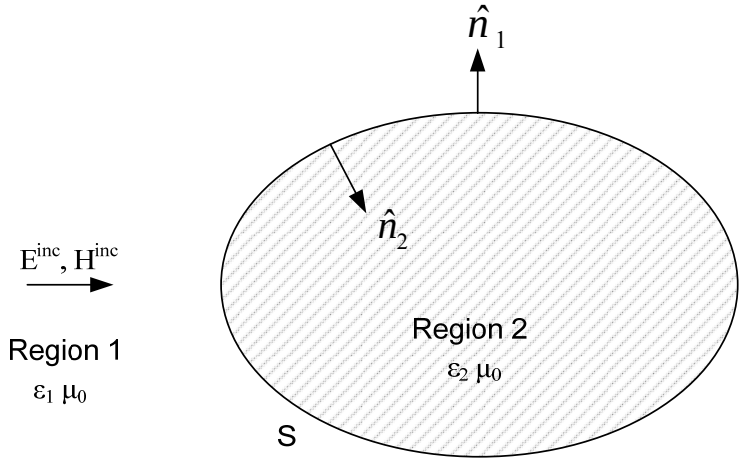

Figure 1.3.4 Scattering from a Homogeneous Dielectric Body

Again the dielectric body in Figure 1.3.4 will be replaced with equivalent currents radiating from the surface. However, dielectric bodies can support both $\vec{J}_s$ and $\vec{M}_s$ surface currents. Subsequently, the formulations for the scattered field in each region are different. $\vec{E}^{scat}$ and $\vec{H}^{scat}$ are now more generally written as,

$$\begin{aligned}
\vec{E}^{scat} &= -j\omega\mu\vec{A} - \frac{j}{\omega\varepsilon}\nabla\nabla\cdot\vec{A} - \nabla\times\vec{F} \\
\vec{H}^{scat} &= \nabla\times\vec{A} - j\omega\varepsilon\vec{F} - \frac{j}{\omega\mu}\nabla\nabla\cdot\vec{F} \\
\vec{A} &= \int_S \vec{J}_s\cdot G(\vec{r},\vec{r}\,')ds' \\
\vec{F} &= \int_S \vec{M}_s\cdot G(\vec{r},\vec{r}\,')ds'
\end{aligned} \tag{1.3.9}$$

where $\vec{F}$ is the dual to $\vec{A}$, the electric vector potential. Using these relations we can write the total field in Region 1:

$$\begin{aligned}
\vec{E}^{tot} &= \vec{E}^{inc} + \vec{E}^{scat} \\
\hat{n}_1\times\vec{E}^{tot} &= \hat{n}_1\times\vec{E}^{inc} + \hat{n}_1\times\vec{E}^{scat} \\
\hat{n}_1\times\vec{E}^{tot} &= \hat{n}_1\times\vec{E}^{inc} + \hat{n}_1\times\left(-j\omega\mu_0\vec{A}_1 - \frac{j}{\omega\varepsilon_1}\nabla\nabla\cdot\vec{A}_1 - \nabla\times\vec{F}_1\right)
\end{aligned} \tag{1.3.10}$$

The total field in Region 2 is now written as,

$$\begin{aligned}
&\vec{E}^{tot} = \vec{E}^{scat} \\
&\hat{n}_2 \times \vec{E}^{tot} = \hat{n}_2 \times \vec{E}^{scat} \\
&\hat{n}_2 \times \vec{E}^{tot} = \hat{n}_2 \times \left(-j\omega\mu_0 \vec{A}_2 - \frac{j}{\omega\varepsilon_2} \nabla\nabla \cdot \vec{A}_2 - \nabla \times \vec{F}_2 \right)
\end{aligned} \tag{1.3.11}$$

The boundary condition between the two regions is that the tangential electric field is continuous.

$$\begin{aligned}
&\hat{n}_1 \times \left(\vec{E}_1^{\,tot} - \vec{E}_2^{\,tot} \right) = 0 \\
&\hat{n}_1 \times \vec{E}_1^{\,tot} = \hat{n}_1 \times \vec{E}_2^{\,tot}
\end{aligned}$$

With the knowledge that $\hat{n}_1 = -\hat{n}_2$ equations (1.3.10) and (1.3.11) can be combined:

$$\begin{aligned}
&\hat{n}_1 \times \vec{E}^{inc} + \hat{n}_1 \times \vec{E}_1^{\,scat} = \hat{n}_2 \times \vec{E}_2^{\,scat} \\
&\hat{n} \times \vec{E}^{inc} = -\hat{n} \times \vec{E}_1^{\,scat} - \hat{n} \times \vec{E}_2^{\,scat} \\
&\hat{n} \times \vec{E}^{inc} = \hat{n} \times \int_S j\omega\mu_0 \vec{J}_s \left(G_1 + G_2 \right) - \vec{M}_s \times \nabla \left(G_1 + G_2 \right) + \frac{j}{\omega\varepsilon_1} \left[\nabla\nabla \cdot \left(\vec{J}_s G_1 \right) + \frac{\varepsilon_1}{\varepsilon_2} \nabla\nabla \cdot \left(\vec{J}_s G_2 \right) \right] ds' \\
&\textit{where, } G_i = \frac{e^{-jk_i R}}{4\pi R}, \; k_i = \omega\sqrt{\varepsilon_i \mu_0}
\end{aligned} \tag{1.3.12}$$

where the subscript has been dropped on the surface normal which is assumed to point into the region that contains the incident EM wave. The magnetic field follows the same argument as above. The final form for the magnetic field is written as,

$$\hat{n} \times \vec{H}^{inc} = \hat{n} \times \int_S j\omega\varepsilon_1 \vec{M}_s \left(G_1 + \frac{\varepsilon_2}{\varepsilon_1} G_2 \right) + \vec{J}_s \times \nabla \left(G_1 + G_2 \right) + \frac{j}{\omega\mu_0} \left[\nabla\nabla \cdot \left(\vec{M}_s G_1 \right) + \nabla\nabla \cdot \left(\vec{M}_s G_2 \right) \right] ds' \tag{1.3.13}$$

Equations (1.3.12) and (1.3.13) represent the final form for the dielectric or PMCHWT SIE formulation.

1.3.5) Hybrid Dielectric / PEC Body Integral Equation

Before continuing the discussion on Hybrid SIE's it is convenient to define compact versions of the integral operators given above. In EFIE, MFIE, and PMCHWT formulations there are two basic integral operators; they are defined as follows:

$$L(\vec{J}_s) = \hat{n} \times \int_S \vec{J}_s \cdot G(\vec{r}, \vec{r}\,') ds' + \hat{n} \times \int_S \nabla\nabla \cdot \left(\vec{J}_s \cdot G(\vec{r}, \vec{r}\,') \right) ds'$$
$$K(\vec{J}_s) = \frac{1}{2} \vec{J}_s + \hat{n} \times \int_S \vec{J}_s \times \nabla G(\vec{r}, \vec{r}\,') ds' \tag{1.3.14}$$

The L and K operators are valid for magnetic surface currents as well and appear in the same form by simply replacing $\vec{J}_s$ with $\vec{M}_s$.

Now consider the problem in Figure 1.3.5 of a PEC strip above a dielectric slab. There are two regions of interest in this problem, Region 1 which contains the PEC strip and Region 2 which contains the dielectric slab. We know from the boundary conditions and equivalence that Region 1 supports only electric current, $\vec{J}_{pec}$. Region 2 will support both electric and magnetic currents, $\vec{J}_\varepsilon$ and $\vec{M}_\varepsilon$.

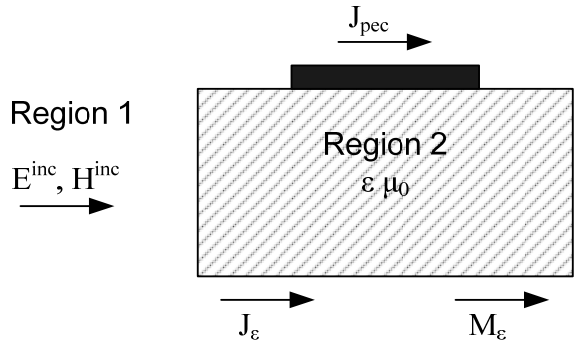

Figure 1.3.5 Hybrid PEC Dielectric Scattering Problem

Formulating the integral equations that govern the scattered fields in each region follows from the PMCHWT derivation. In compact form, the fields in Region 2 are written,

$$0 = L(\vec{J}_\varepsilon) + K(\vec{M}_\varepsilon)$$
$$0 = L(\vec{M}_\varepsilon) + K(\vec{J}_\varepsilon) \tag{1.3.15}$$

Realizing that the PEC can only support $\vec{J}$, the fields in Region 1 can be written in a similar manner:

$$\vec{M}^{inc} = 0 = L(\vec{J}_\varepsilon) + K(\vec{M}_\varepsilon) + L(\vec{J}_{pec})$$
$$\vec{J}^{inc} = L(\vec{M}_\varepsilon) + K(\vec{J}_\varepsilon) + K(\vec{J}_{pec}) \tag{1.3.16}$$

1.4) Kernels

The final focus of this review section will be on the integral operators defined in equations (1.3.14). For the purposes of this thesis, the quantities internal to the integrals of these operators are known as Kernels. The *L* and *K* operators have three basic types of kernels, each with their own properties.

First consider the *K* operator. The Kernel present in the *K* operator is of the form:

$$\vec{f} \times \nabla G(\vec{r}, \vec{r}\,')$$

Notice that as the source and observation points, $\vec{r}\,'$ and $\vec{r}$, begin to coincide the Kernel approaches an infinite value. Functions that exhibit this behavior are known as singular functions. This particular Kernel has a $\frac{1}{R^2}$ singularity because of the derivative on the Green's Function. Because of this singularity special care needs to be taken when solving the discrete form of the *K* operator. Note that the *K* operator has a purely diagonal term outside the integral. Thus the *K* operator is known as a Second Kind Integral Operator. This diagonal term helps to improve conditioning for discrete forms of the *K* operator.

The *L* operator has two types of Kernels. They are as follows:

$$\vec{f} \cdot G(\vec{r}, \vec{r}\,') \qquad \text{(i)}$$

$$\nabla\nabla \cdot \left(\vec{f} \cdot G(\vec{r}, \vec{r}\,') \right) \qquad \text{(ii)}$$

Kernel (i) is a smoothing Kernel and has a $\frac{1}{R}$ singularity. Kernel (ii) is hypersingular and poorly behaved because of its $\frac{1}{R^3}$ singularity. The *L* operator is a first kind integral equation because of the combination of the smoothing and hypersingular pieces. Thus, discrete forms of the *L* operator tend to be very poorly conditioned.

II. Method of Weighted Residuals

Section I was devoted to deriving explicit forms for SIEs on arbitrary targets. These SIEs can be solved exactly for simple geometries such as spheres using techniques such as the Mie Series [8]. However, the ultimate goal of SIEs is to solve problems of intricate shapes such as an aircraft. Once the target becomes more complex the exact solution is increasingly difficult to achieve. To alleviate this problem the Method of Weighted Residuals (also known as Method of Moments or MoM) will be used to discretize the continuous problem domain into finite regions on which an approximate solution can be reached.

MoM can be divided into four main steps [9]:

1) Discretization of the Problem Domain
2) Selection of Interpolation Functions
3) Formulation of the System of Equations
4) Solution to the System of Equations

First, the geometry must be discretized into simple geometrical pieces, usually triangles or quadrilaterals for surfaces and tetrahedra or hexahedra for volumes. These elements are used to approximate the exact geometry to a desired degree of accuracy. The process of discretizing the target geometry is commonly referred to as meshing.

The interpolation functions are chosen to represent the current which is induced on these targets from the inbound sources. The degree of these functions determine, in part, the accuracy of the approximation.

A system of equations arises from unknown coefficients placed with each interpolation function as one cannot know a priori how the current will behave on a given element. N elements and N unknown coefficients leads to an N by N system of equations which can be solved in a direct manner using LU decomposition [10].

Each of the four steps in MoM will be discussed in more detail in the subsequent sections.

2.0) Meshing

The first and possibly most important step in MoM is domain discretization or meshing. The manner in which the problem domain is meshed has a significant impact on solution time and accuracy. As noted above, the continuous problem domain is sub divided into a set number of smaller subdomains or elements using basic geometrical shapes. The elements are assumed to be fitted polyhedra. For example, triangular elements are fitted if each edge is shared by two or fewer triangles (open structures do not necessarily have every edge shared). Below in Figure 2.0 is a cube whose surface has been meshed with both quadrilaterals and triangles.

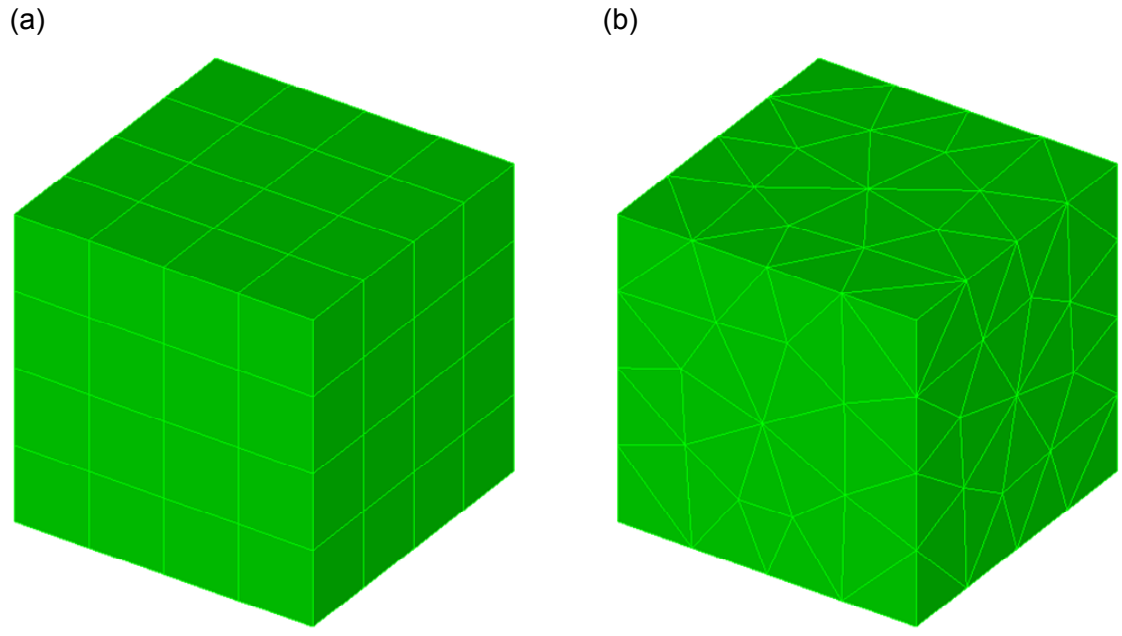

Figure 2.0 Meshed Cube. 2.0(a) is meshed with quadrilaterals, 2.0(b) with triangles.

In general, there is a relationship between the number of meshed elements and solution accuracy; that is if the number of meshed elements increases the solution generally becomes more accurate. For linear elements there is a linear relationship between

discretization error and element size [9], $error: O(h),\ h = cell\ size$. However, this relationship comes at the penalty of increased computational time. For a direct LU decomposition the number of operations to reach a solution is on the order of the number of elements cubed, $O(N^3)$. Thus an increase in the number of elements significantly impacts computational time. Careful consideration needs to be taken when meshing to compromise between solution accuracy and computational time.

2.0.1) High Order Meshing

As noted above there is a fundamental relation between mesh density, solution accuracy, and solution time. One technique used to increase solution accuracy without sacrificing as much time is the use of high order meshing elements (Figure 2.0.1). For smooth targets, these high order elements, also known as curvilinear elements, can improve discretization error based on the polynomial order of the element, $error: O(h^p),\ p = polynomial\ order$. Thus when compared to linear elements you can achieve an equivalent discretization error using fewer high order elements. This allows for improved accuracy without increasing computational costs.

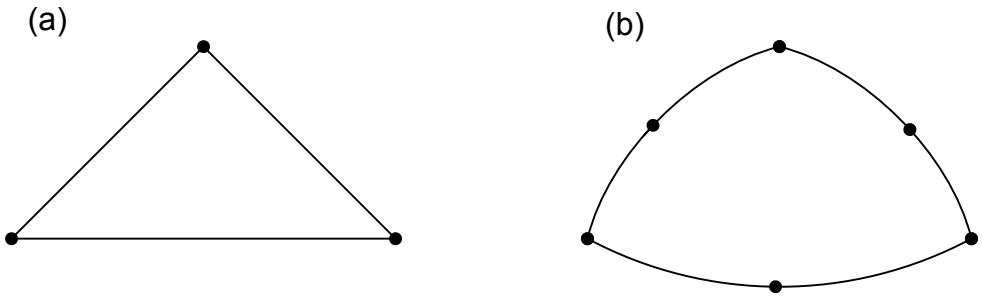

Figure 2.0.1 Meshing Elements. 2.0.1(a) depicts a linear triangular element, 2.0.1(b) a quadratic triangular element. Note that the second order element has curved edges and more nodes.

Further discussion on curvilinear elements and general curvilinear coordinate systems can be found in [9] and [11].

2.1) Interpolation Functions [9]

Interpolation Functions represent an approximation to the actual solution within each meshed element. These are known functions with varying polynomial degree which are only supported locally. Interpolation Functions can be chosen to span multiple mesh elements on which they have nonzero values. Outside of the defined local support the Interpolation Functions are zero.

Consider the boundary value problem (BVP),

$$L\phi = f \tag{2.1.1}$$

where L is an arbitrary operator, f is a known forcing function or source, and ϕ is a function which resides within a domain Ω bound by Γ. ϕ satisfies the classic Dirichlet and/or Neumann boundary conditions on Γ:

$$\begin{aligned} &\phi = \gamma_1 |_\Gamma \text{ , } \gamma_1 \text{ constant} \qquad \text{(Dirichlet)} \\ &\frac{\partial \phi}{\partial n} = \gamma_2 |_\Gamma \text{ , } \gamma_2 \text{ constant} \quad \text{(Neumann)} \end{aligned} \tag{2.1.2}$$

This BVP can be solved using MoM. ϕ is expanded into a summation of known interpolation functions weighted by unknown constant coefficients.

$$\phi \approx \sum_{j=1}^{N} \phi_j c_j \text{ , } \phi_j \in \Omega \tag{2.1.3}$$

ϕ_j are the known interpolation functions of polynomial order p that span Ω bound by Γ. c_j are the unknown constant coefficients which need to be solved for. These are also referred to as Degrees of Freedom or DOFs. Plugging (2.1.3) back into (2.1.1) yields,

$$\sum_{j=1}^{N} c_j L\phi_j \simeq f \tag{2.1.4}$$

2.2) Formulation of the System of Equations

We have now expanded (2.1.1) into (2.1.4) which has *N* unknowns. In order to solve the system we need to first formulate *N* equations to yield a unique solution. First, let's define the inner product as follows [9]:

$$\begin{aligned} \langle \Psi, \Phi \rangle &= \int_{\Omega} \Psi^a \Phi \, d\Omega \text{ (scalar form), } a \text{ denotes a conjugate transpose} \\ \langle \vec{\Psi}, \vec{\Phi} \rangle &= \int_{\Omega} \vec{\Psi}^a \vec{\Phi} \, d\Omega \text{ (vector form)} \end{aligned} \tag{2.2.1}$$

Now take equation (2.1.4) and take the inner product with some testing function ϕ_i:

$$\sum_{j=1}^{N} c_j \langle \phi_i, L\phi_j \rangle_{\Omega} = \langle \phi_i, f \rangle_{\Omega} \; ; \; i = 1, N \tag{2.2.2}$$

Repeating this procedure for *N* testing functions results in an *N* by *N* system of equations:

$$\begin{pmatrix} \langle \phi_1, L\phi_1 \rangle & \dots & \langle \phi_1, L\phi_N \rangle \\ \dots & \dots & \dots \\ \langle \phi_N, L\phi_1 \rangle & \dots & \langle \phi_N, L\phi_N \rangle \end{pmatrix} \begin{pmatrix} c_1 \\ \dots \\ c_N \end{pmatrix} = \begin{pmatrix} \langle \phi_1, f \rangle \\ \dots \\ \langle \phi_N, f \rangle \end{pmatrix} \tag{2.2.3}$$

Define the compact form of this system as,

$$\bar{\bar{K}}\bar{c} = \bar{b} \tag{2.2.4}$$

The formulation presented follows a Galerkin approach in which the expansion functions ϕ_j are known as the basis functions and ϕ_i are known as the testing functions. The choice of these functions will be discussed in more detail with relation to the Nyström method. However, for Galerkin approaches both the basis and testing functions are represented by the same interpolation functions.

2.3) Solution to system of Equations

The solution of (2.2.4) takes the form,

$$\bar{c} = \bar{\bar{K}}^{-1}\bar{b} \tag{2.3.1}$$

The inverse of the *K* matrix is effectively implemented using an LU decomposition (This thesis will be concerned with direct solution methods rather than iterative approaches which can be found in [10]). Once the coefficients have been found an approximation to

the current on each patch can be made simply by multiplying the patch basis function by the weighting coefficient. These current approximations can then be used in postprocessing to yield other quantities of interest.

III. Nyström Method [12-14]

As noted in previous sections, the goal of this thesis is to solve SIEs on arbitrary targets using MoM. There are many techniques to accomplish this task. Our focus will now shift to the underlying method used throughout the remainder of this thesis, the Nyström Method. It is worth noting that the Nyström Method has been shown to be equivalent to a quadrature sampled Moment Method [14].

A conventional Nyström Method serves as a simple and efficient way to discretize integral equations with non-singular kernels. Consider the following IE:

$$\phi(x) = \int_S \varphi(x')G(x-x')ds' \tag{3.1.1}$$

where S is a smooth surface, $\phi(x)$ is a known forcing function evaluated at observation point x on S, $G(x-x')$ is the kernel, and $\varphi(x')$ is the function of interest at the source point x'. Along with this IE consider a quadrature rule given by,

$$\begin{aligned} &\int_S f(x)ds \cong \sum_{n=1}^{N} \omega_n f(x_n) \\ &x_n = x(u_n) \end{aligned} \tag{3.1.2}$$

An example of a quadrature rule which satisfies (3.1.2) would be a Gauss-Legendre rule with weights ω_n and abscissa points u_n [15]. The Nyström discretization of $\phi(x)$ on S is preformed simply by evaluating $\phi(x)$ at each quadrature point on S. That is,

$$\varphi_n = \varphi(x_n) \tag{3.1.3}$$

The discretized form of (3.1.1) can then be achieved by applying (3.1.2) and (3.1.3) to form the matrix relation:

$$\phi_m = \sum_{n=1}^{N} \omega_n(m)G(x_m - x_n)\varphi_n \;;\; m = 1, N \tag{3.1.4}$$

Assuming that $\phi(x)$ and $G(x-x')$ are regular functions (non-singular), S is smooth, and a high order quadrature scheme is used then the solution to (3.1.4) represents a high order approximation to the function of interest $\varphi(x')$. The error in this approximation is the same as the underlying quadrature rule.

As noted in section 1.4 the kernels used in the integrals for wave scattering problems are singular, some even hypersingular. These properties spoil the high order modeling of the kernel at vanishing separation of the source and observation points. Therefore the Nyström scheme has to be modified to incorporate these singularities while still maintaining its high order accuracy.

3.1) Locally Corrected Nyström Method (LCN)

Conventional Nyström Methods are capable of discretizing and solving non-singular IEs to high order accuracy. Wave scattering problems have kernels which are undefined when the source and observation points coincide. However, Nyström Methods can still be applied to problems with singular kernels by modifying the underlying quadrature rule near the singularity. The high order quadrature rule used in the conventional Nyström Method provides a very accurate solution for smooth functions, or for singular functions at distances away from the singularity. The Locally Corrected Nyström Method (LCN) uses the same high order quadrature rule away from the singularity but also employs "local corrections" to maintain high order accuracy near the singular point.

Again consider the problem in (3.1.1) however let the kernel be singular. (3.1.4) can be we written as,

$$\phi_m = \sum_{m=1}^{N}\sum_{n=1}^{N} \omega_n(m) G(x_m - x_n)\varphi_n \tag{3.1.5}$$

LCN now divides the problem into two regions, a region near and far from the singular point. Thus (3.1.5) becomes,

$$\phi_m = \sum_{m\in far}\sum_{n=1}^{N} \omega_n G(x_m - x_n)\varphi_n + \sum_{m\in near}\sum_{n=1}^{N} \tilde{\omega}_n(m)\varphi_n \tag{3.1.6}$$

where $\tilde{\omega}_n(m)$ represents the modified quadrature weights for the specialized rule at the singularity. The evaluations in the far region are straight forward and follow directly from the previous section. The specialized quadrature rule for the near region serves to fix the underlying quadrature rule so that the integration can maintain its high order properties despite the integrand being singular.

In order to solve for these modified quadrature weights a set of basis functions $B_m(r)$ are defined on the surface or patch of interest. The details of these functions will be discussed in subsequent sections. However, they are typically chosen in a similar fashion to the underlying quadrature rule. Thus the near term can be expanded,

$$\sum_{n=1}^{N} \tilde{\omega}_n(m) B_n(r_m) = \int_S G(x_m - x_n') B_n(r_m') ds' \tag{3.1.7}$$

Repeating (3.1.7) for $m = 1, N$ leads to a linear system of equations:

$$L_s \bar{\omega} = \bar{\kappa}_m \tag{3.1.8}$$

where L_s is a matrix local to the patch with elements $[L_s]_{n,j} = B_n(r_j)$, $\bar{\kappa}_m$ is a vector with entries $\kappa_m = \int_S G(x_m - x_n') B_n(r_m') ds'$ which can be evaluated to a desired precision using adaptive quadrature. The solution of (3.1.8) yields the modified quadrature weights which then can be applied to the near term in (3.1.6). With the application of the modified weights (3.1.6) now represents a high order discrete solution to the integral equation posed in (3.1.1) with a singular kernel.

3.2) Basis Functions

Now that the LCN method has been presented we will shift focus to specific details on applying this method to a scattering problem. First let's assume an arbitrary target has been discretized into curvilinear patches which model the surface contour to high order. The next step in the MoM procedure would be the selection of basis functions. As noted in section 3.1, the selection of basis functions for LCN mirrors that of the underlying quadrature rule. More specifically, the basis functions are modeled in the same polynomial space as the quadrature rule. For the purposes of this thesis Legendre polynomials will be chosen as the representative space for the quadrature rule and the basis functions.

The p^{th} order basis on a quadrilateral is defined as follows [16, 17]:

$$
\begin{aligned}
& P_n(u^1)P_m(u^2)\frac{\vec{a}_1}{\sqrt{g}} \;; \quad n \in 0, p \\
& \qquad\qquad\qquad\qquad\quad\; m \in 0, p\text{-}1 \\
& P_l(u^1)P_k(u^2)\frac{\vec{a}_2}{\sqrt{g}} \;; \quad l \in 0, p\text{-}1 \\
& \qquad\qquad\qquad\qquad\quad\; k \in 0, p \\
& \sqrt{g} = \vec{a}_1 \times \vec{a}_2 \cdot \hat{a}_n
\end{aligned}
\tag{3.2.1}
$$

where $P_j(u)$ are the j[th] order Legendre polynomials. Following Stratton's notation in [11] (u^1, u^2) are the local curvilinear coordinates of the quadrilateral cell, $\vec{a}_i$ are the local unitary vectors, $\hat{a}_n$ is the surface normal, and $\sqrt{g}$ is the surface Jacobian.

There are a few things to note about the basis set defined in (3.2.1). First, the set is mixed order because of the differing degrees on the two Legendre polynomials. One is complete to order p and the other complete to order $p+1$. As noted in [16, 17] a mixed order basis set better serves scattering problems because it correctly models edge singularities which result from open or sharp surfaces. A polynomial complete set, one which the Legendre polynomials all have the same order, would lead to spurious solutions when applied to a problem with edge singularities. Both a polynomial complete set and a mixed order set would solve problems involving smooth scatterers however the mixed order set is chosen because it applies to a wider range of problems.

The second item of note is that there are two distinct basis functions in (3.2.1), one which lives along the unitary vector $\vec{a}_1$ and the other along $\vec{a}_2$. When using this basis set in conjunction with LCN an appropriate testing procedure must follow. A straight forward approach from [16] uses separate, mixed order quadrature rules for each of the two test vectors. This results in a square system of local corrections which can be readily solved.

3.3) Treatment of Self and Near Self Interactions

3.3.1) Duffy Transform

As noted above, the LCN needs local corrections in order to maintain high order accuracy. There are two distinct corrections which need to be discussed, those for the

self terms and near self terms. Self terms occur when the basis and testing functions lie on the same patch which presents a problem because of the singularity in the kernel. Near self terms are not singular however they lie within a region of rapid variation in the kernel and thus need special treatment.

The singularity in the self terms can be handled by a technique known as Duffy Transform [18]. Consider the kernel $\int_S G(\vec{r},\vec{r}\,')ds'$ being integrated over the quad depicted in Figure 3.3a with a singular point at the cell center, $\vec{r}_n^{\,c}$.

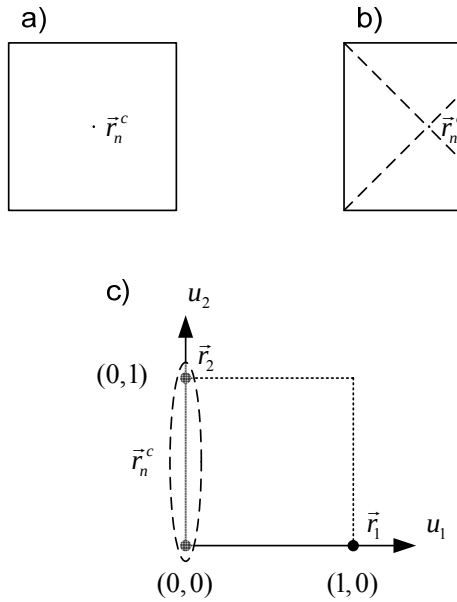

Figure 3.3 a) Quadrilateral with singular point at cell center. b) Triangulated Quad. c) Degenerate Quad mapped into unitary space.

The first step in the Duffy Transform is to triangulate the cell at the singular point (Figure 3.3b). Thus the integration becomes a sum over the four segmented triangles. Each of these triangles is then considered to be a degenerate quad meaning that one side of the quad has collapsed into a point on the triangle. The degenerate quad is then mapped into the unitary space: the former vertices $\vec{r}_1,\vec{r}_2$ are mapped to (1,0) and (0,1) respectively

while $\vec{r}_n^{\,c}$ is mapped to edge $u_1 = 0$ (Figure 3.3c). Once the mapping is complete, the integral over each triangle can be written:

$$\begin{aligned}
&\int_{\Delta} G(\vec{r},\vec{r}\,')ds' = A_{\Delta i}\int_0^1\int_0^1 \frac{e^{-jkR(u_1,u_2)}}{4\pi R(u_1,u_2)}u_1 du_1 du_2 \\
&R(u_1,u_2) = \left|\vec{r}(u_1,u_2) - \vec{r}_n^{\,c}\right| \\
&\vec{r}(u_1,u_2) = (1-u_1)\vec{r}_n^{\,c} + u_1(1-u_2)\vec{r}_1 + u_1u_2\vec{r}_2 \\
&A_{\Delta i} = \text{ area of ith triangle}
\end{aligned} \tag{3.3.1}$$

The process of triangulating the singular quad and mapping to the unitary space effectively removes the $O(1/R)$ singularity. The resulting integral in (3.3.1) can then be found using a Gauss Legendre quadrature rule or a Lin Log rule for faster convergence. Integrands which contain singularities of higher order need to be modified before Duffy can be used. If the higher singularity can not be reduced to $O(1/R)$ then another technique must be employed for the self term.

3.3.2) Octree Decomposition [19, 20]

As noted above, the near self interactions need to be "locally corrected" in order to maintain the high order accuracy of the Nyström Method. Section 3.1 outlined how to apply these local corrections in a manner which preserves the high order properties of the Nyström Method, however, the question remains as to the cutoff point between near and far interactions. A technique used to make this distinction is known as an Octree Decomposition.

To best illustrate an Octree Decomposition we will first look at its two dimensional counterpart, the Quadtree Decomposition. Consider the problem depicted in Figure 3.3.2.1 of an arbitrary two dimensional scattering target. To begin a Quadtree decomposition first enclose the target by a fictitious square whose dimension is equal to that of the largest linear dimension of the scatterer. This is sometimes referred to as the root box. Next, create the first level of the Quadtree by dividing the root box into four equal pieces, denoted in Figure 3.3.2.1 by the blue dotted lines. This results in four equally sized groups at level one and is in fact how the Quadtree gets its name.

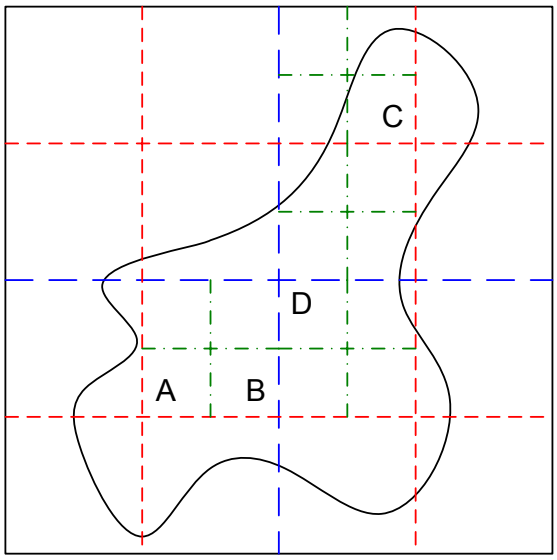

Figure 3.3.2.1 Quadtree Decomposition. The arbitrary target is decomposed into three Quadtree levels.

The next step in a Quadtree decomposition follows similarly from the first, divide each of the level one groups into four level two groups denoted by the red dotted lines in the figure. Thus at level two we now have sixteen total groups. This process continues for a specified number of levels until it is determined that subsequent divisions are no longer necessary. Consider the level three division in Figure 3.3.2.1, denoted by the green dashed dotted lines. Note that not every level two group was divided because the group either did not contain any information (e.g. top left level two group) or not enough information to warrant the division (e.g. bottom right level two group).

Next, consider the level three groups A, B, and C. Groups A and B are known as touching near neighbors because they share a common edge. In the context of the Nyström method, interactions between sources in region A and observers in region B would need to be locally corrected following the procedure in section 3.1. Conversely, interactions between group A and group C would simply be preformed via the point based reactions outlined in section 3.0. Thus, A and C would be considered a far interaction due to their separation in the Quadtree decomposition.

An Octree Decomposition mirrors that of the Quadtree but with an added dimension. That is, three dimensional targets would be enclosed in a cube and divided into eight

equal groups during each division. Thus, each group in an Octree can have up to twenty seven touching near neighbors (including the self term) as opposed to only nine in a Quadtree decomposition. Again, Octree near neighbor reactions would need to incorporate the local corrections to maintain high order accuracy. Figure 3.3.2.2 shows an example of an Octree decomposition.

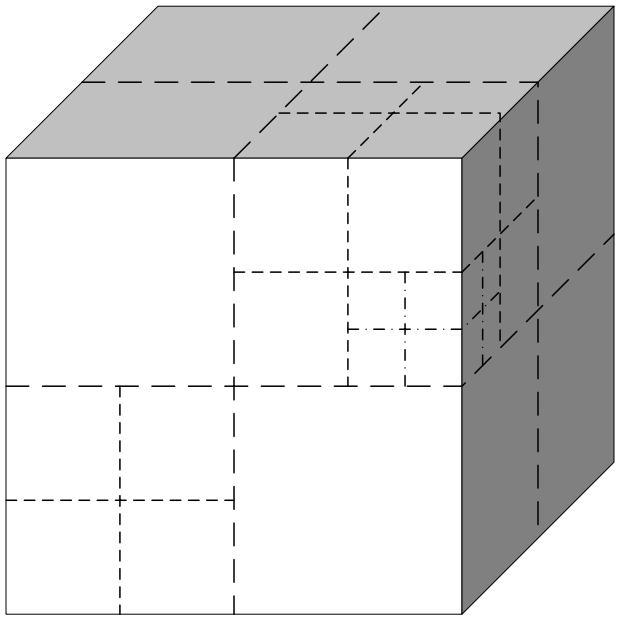

Figure 3.3.2.2. Octree Decomposition.

For additional accuracy, the local corrections can be expanded beyond the near neighbor groups to include the non-touching near neighbor groups. These groups are defined as children of my parents touching near neighbors. Thus from Figure 3.3.2.1 group D would be a non-touching near neighbor of group A. The motivation for including the non-touching near neighbor groups for local corrections is because EM kernels can be nearly singular in close proximity to the self patch. There are several other EM techniques which use Octree Decompositions, an example being the Fast Multipole Method (FMM) [21].

3.4) Nyström EFIE and MFIE [14]

Now that the basics of the LCN have been presented we will now apply these techniques in conjunction with the MoM process to yield the Nyström specific versions of the discrete EFIE and MFIE operators for PEC surfaces. Again consider the EFIE from equation (1.3.4) which will be rewritten here as:

$$-\vec{t}\cdot\vec{E}^{inc}(\vec{r})=\vec{t}\cdot\left[-jk_0\eta_0\int_S\vec{J}(\vec{r}\,')G(\vec{r},\vec{r}\,')ds'-j\frac{\eta_0}{k_0}\nabla\int_S\nabla G(\vec{r},\vec{r}\,')\cdot\vec{J}(\vec{r}\,')ds'\right] \quad (3.4.1)$$

where $\vec{t}$ is a vector tangential to S at $\vec{r}$. Expanding the current in (3.4.1) with the basis functions defined in (3.2.1) yields the discrete EFIE:

$$\begin{aligned}-\vec{a}_{j_{q_m}}\cdot\vec{E}^{inc}(\vec{r}_{q_m})=-j\frac{\eta_0}{k_0}\sum_{p=1}^{N_p}\sum_{k=1}^{N_k}\left[k_0^2\vec{a}_{j_{q_m}}\cdot\int_{S_p}\vec{J}_k^{\,i}(\vec{r}\,')G(\vec{r}_{q_m},\vec{r}\,')ds'\right]-\\ j\frac{\eta_0}{k_0}\sum_{p=1}^{N_p}\sum_{k=1}^{N_k}\left[\vec{a}_{j_{q_m}}\cdot\nabla\int_{S_p}\nabla G(\vec{r}_{q_m},\vec{r}\,')\cdot\vec{J}_k(\vec{r}\,')ds'\right]\end{aligned} \quad (3.4.2)$$

where the test vector $\vec{a}_{j_{q_m}}$ is a unitary vector evaluated at the quadrature abscissa point $\vec{r}_{q_m}$ and N_p, N_k are the number of curvilinear patches and quadrature points per patch respectively. Note that in practice both of the derivatives will be placed on the Green's function due to the choice of point basis functions. The relation in (3.4.2) will be used for the far interactions however the second term will need to be rewritten for the near interactions as it is hypersingular and not integrable via Duffy Transform.

From [12], we can rewrite the second term on the right hand side of (3.4.2) as:

$$\begin{aligned}&\vec{a}_{j_{q_m}}\cdot\nabla\int_{S_p}\nabla G(\vec{r}_{q_m},\vec{r}\,')\cdot\vec{J}_k(\vec{r}\,')ds'=\\&+\int_{S_p}\nabla G(\vec{r}_{q_m},\vec{r}\,')\cdot\left[\vec{a}_{j_{q_m}}\nabla_{||}'\cdot\vec{J}_k^{\,i}(\vec{r}\,')-\vec{K}_{q_m}(\vec{r}\,')\right]ds'\\&-\oint_{C_p}\left(\hat{e}_p'\cdot\vec{J}_k^{\,i}(\vec{r}\,')\right)\left(\vec{a}_{j_{q_m}}\cdot\nabla G(\vec{r}_{q_m},\vec{r}\,')\right)dl'\\&-\oint_{C_p}\hat{e}_p'\cdot\vec{K}_{q_m}(\vec{r}\,')G(\vec{r}_{q_m},\vec{r}\,')dl'\end{aligned} \quad (3.4.3)$$

where C_p is the contour bounding S_p, $\hat{e}_p$ is the outward normal to C_p tangential to S, and $\vec{K}_{q_m}(\vec{r}\,')$ is defined as: $\vec{K}_{q_m}(\vec{r}\,') = \frac{\vec{a}_j(\vec{r}\,')}{\sqrt{g'}}\left(\sqrt{g}\nabla_{\|}\cdot\vec{J}_k^i(\vec{r}\,')\right)|_{\vec{r}=\vec{r}_{q_m}}$. Now, the surface integral in (3.4.3) can be shown to have a $O(1/R)$ singularity which can be efficiently evaluated to desired accuracy using Duffy Transform. The contour integrals lie on the boundaries of S. Since the quadrature points will always be contained in S these integrals will never be singular and thus can be evaluated via adaptive quadrature without further treatment.

The discrete Nyström MFIE follows in a similar manner. Consider the MFIE from (1.3.7) written here as:

$$\hat{a}_n \times \vec{H}^{inc}(\vec{r}) = \frac{1}{2}\vec{J}(\vec{r}) - \hat{a}_n \times \nabla \times \int_S G(\vec{r},\vec{r}\,')\vec{J}(\vec{r}\,')ds' \tag{3.4.4}$$

Applying the same basis functions as those for the EFIE, (3.4.4) can be written,

$$\begin{aligned}\vec{a}_{j_{q_m}} \cdot \left(\hat{a}_n \times \vec{H}^{inc}(\vec{r}_{q_m})\right) = \frac{1}{2}\sum_{k=1}^{N_k}\vec{a}_{j_{q_m}}\cdot\vec{J}_k^i(\vec{r}_{q_m}) - \left(\hat{a}_n \times \hat{a}_{j_{q_m}}\right)\cdot \\ \sum_{p=1}^{N_p}\sum_{k=1}^{N_k}\int_{S_p}\left(\vec{J}_k^i(\vec{r}\,')\times\vec{R}\right)\left(\frac{1}{R}\frac{\partial}{\partial R}G(R)\right)ds'\end{aligned} \tag{3.4.5}$$

where the unit normal $\hat{a}_n$ is evaluated at $\vec{r}_{q_m}$, $\vec{R} = \vec{r}_{q_m} - \vec{r}\,'$, and $R = \left|\vec{R}\right|$. It can be shown that the integrand in (3.4.5) has a $O(1/R)$ which again can be found in a controllable manner with Duffy Transform.

3.5) Advantages over Galerkin Methods [12]

Now that sufficient background has been presented for SIEs and the Nyström method in particular we will now discuss its advantages over standard Galerkin approaches. The classical method for solving SIEs comes from [4]. Rao, Wilton, and Glisson (RWG) developed their divergence conforming basis functions over triangular patches. RWG basis functions are said to be divergence conforming because they preserve normal continuity of current across edges; this is a consequence of the basis having a divergence complete to zero-th order. RWG functions are rooftop functions which span two

triangular patches joined by a shared edge. In the context of a MoM procedure, choosing RWG basis functions for both the test and basis sets results in a Galerkin procedure.

Elimination of Multipatch Basis

The first major advantage of Nyström over a standard Galerkin approach using RWG basis functions is the elimination of multipatch basis functions. Multipatch basis functions are typically used to strictly enforce continuity which helps solution accuracy as well as ease of implementation. For low order geometry models, the error in not enforcing continuity is on the same order as the error in the discretization. Thus for low order RWG basis functions the enforcement condition becomes necessary as to help eliminate a significant source of error. Conversely, for a high order model the error in not enforcing continuity does not have as significant an impact because continuity is achieved as a natural consequence of properly solving the integral equation. Also, the discretization error can more easily be made insignificant for high order schemes. The decrease in modeling error can also be reduced in the context of a Galerkin, RWG procedure, however, high order extensions for RWG basis functions have been shown to be difficult to implement [22].

Junction issues are another set of problems which can be avoided by eliminating multipatch basis functions. Typical EM problems of interest involve objects which are composed of both material and conducting regions. Junctions, or basis functions which connect a material and conducting patch, need non-trivial special treatment. Nyström methods avoid these issues because basis functions live only on one patch which can be associated with either a material or conducting region but not both. These junction issues also impact fast solution methods, such as the FMM, which take advantage of grouping techniques to do long range interactions [21]. For these methods, multipatch basis would either need to be split which increases complexity or the group sizes would need to increase which decreases efficiency.

Faster Precomputation

Because of its point interaction nature, the Nyström method is faster at filling the system matrix than standard Galerkin methods. Galerkin methods require N^2 numerical double integrations in order to fill the system matrix. Nyström requires less than N^2 kernel evaluations and only $O(N)$ calculations of the local correction coefficients. This fact is validated by the data presented in [12]. A comparison was made between a high order Nyström method, a high order Galerkin method, and a low order Galerkin solver known as the Fast Illinois Solver Code (FISC). Each method was run under comparable conditions for different sized PEC spheres; that is each code was setup to have an equivalent number of unknowns per wavelength. An MFIE formulation and a standard LU solver were used in all cases. The results are summarized below in Table 3.5.

Nyström vs Galerkin Performance on PEC Spheres

Scattering code	Radius (λ)	No. of unknowns	Setup time (s)	Solve time (s)	RMS error (dB)
FastScat (Nyström)	0.9	600	74	36	0.35
FastScat (Galerkin)	0.9	600	972	88	0.07
FISC (Galerkin)	0.9	600	83	42	1.28
FastScat (Nyström)	1.8	2400	539	2742	0.26
FastScat (Galerkin)	1.8	2400	8177	3395	0.05
FISC (Galerkin)	1.8	2430	873	2255	0.61
FastScat (Nyström)	2.7	5400	1953	31735	0.097
FastScat (Galerkin)	2.7	5400	38803	36152	0.021
FISC (Galerkin)	2.7	5880	8230	28795	0.723

Table 3.5. Nyström vs Galerkin Performance [12].

Note that for the 2.7λ case the Nyström setup time was nearly 20 times faster than the high order Galerkin method and over 4 times faster than FISC while yielding comparable accuracy.

Memory Reduction:

One final advantage of note is Nyström's ability to reduce memory costs for iterative solvers. For an iterative solver, storing the full system matrix costs $O(N^2)$. This can be reduced with Nyström to $O(N)$ by only storing the local correction matrices. Unsaved portions can be reproduced if need be in a fast and efficient manner because they are simply point evaluations of the kernel. In the context of the FMM, memory costs can be reduced from $O(N^{5/4})$ in the single level case to $O(N\log(N))$ in the multilevel case [12].

IV. Object Oriented Design [23, 24]

4.0) Motivation

The following sections will switch gears toward the software engineering efforts of the Nyström Implementation. The Nyström method as described in previous sections is relatively straightforward to implement and has been used previously by several sources [12, 14]. In fact, the University of Kentucky already uses Nyström methods in its own Material Scattering (Mscat) code. However, as research in the EM area continues to grow the need for a common software framework becomes apparent. All forms of discrete EM solvers, whether an integral or differential based method in the time or frequency domain, share some common bonds which typically are re-written each time a new code is developed: geometry modeling and information storage is an example of a link between all methods. The degrees of commonality depend on the implemented method but moving existing code into a united framework would allow for reuse without "reinventing the wheel." New EM software technologies would come online faster as common pieces would simply be reused rather than written from scratch.

As a consequence of these commonalities, the University of Kentucky decided to create a universal framework for all EM codes and research, entitled General Electromagnetic Framework or GEMF. The goal of GEMF is to unite all existing codes and current research under one umbrella so that new techniques can build off previous efforts. GEMF would not only allow for code reuse with new research but also ease improvements to previous technologies and enable hybridized techniques: a combined Integral Equation and Finite Element method for example. This concept seems very simplistic and straightforward however only recently has it been used in practice. The EIGER code [25] was developed under this paradigm as are current codes at the University of Illinois [26].

In order to achieve its goal, the GEMF code must exhibit several key qualities: reusability, flexibility, and maintainability. First and foremost, the GEMF code must be setup such that old code can easily be reused. Without this property the main, underlying

goal of GEMF is compromised. Also, GEMF must be flexible to allow for new technologies to be integrated. Finally, since GEMF will house several technologies it must be easy to maintain. If the code is not designed properly and documented extensively pieces could potentially be lost and thus effort would be wasted. These necessary conditions are typical advantages and natural consequences of an Object Oriented (OO) design.

4.1) Classes and Objects

Now that the motivation for an OO design has been presented a definition of OO design and subsequent discussion is needed. Meyer defines OO design as, "the construction of software systems as structured collections of abstract data type implementations" [23]. At first glance this definition may seem confusing so for a more complete understanding let us first discuss two features of OO design, Classes and Objects.

Consider the following example: inventorying cars on a dealer lot. Each car can be defined by the following attributes: make, model, color, year. Let us define a Car class:

```
class Car
{
      string      make;
      string      model;
      string      color;
      int         year;
}
```

The Car class serves as a blueprint from which a specific Car type can be created. A specific instance of the Car class would be called a Car object, say for example a 2006 gray Honda Civic. The object has information which uniquely describes a particular instance of a class. Thus each car on the lot is a specific instance or object of the Car class. Now returning to Meyer's definition, OO systems are built as a collection of classes. Each class provides a list of services which are accessed in no particular order. A good class design is general so that different system level designs can reuse the services provided by each class. The process of combining classes into an OO system typically is achieved in a bottom-up manner.

4.1.1) Design Approach and Potential Problems

Typical OO systems begin by considering the top level of abstraction, or the class from which everything else will be derived. In the context of GEMF, the highest level is known as the GEMF System level. Each subsequent system type is derived from this GEMF System base class. Figure 4.1.1 diagrams the hierarchy of GEMF Systems.

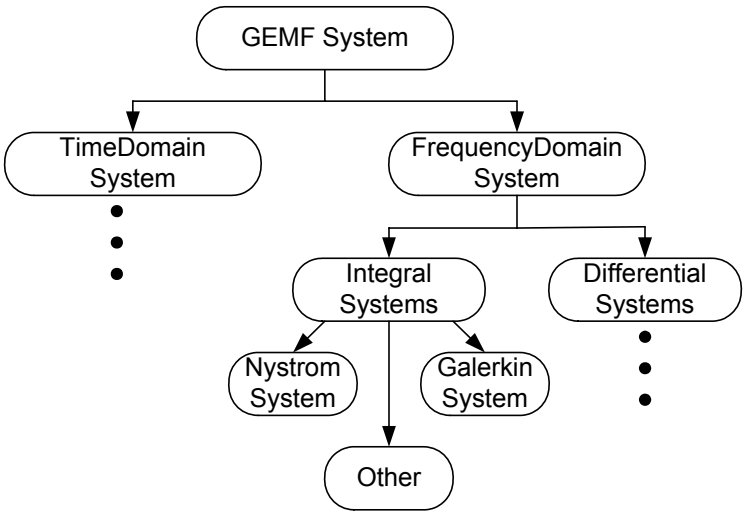

Figure 4.1.1 GEMF System hierarchy

The system level hierarchy was designed to be as general as possible and to incorporate all the potential system types. Once the hierarchy is established each system type can then be filled in a bottom-up manner. This is a feasible approach however initial designs are potentially difficult because such a broad range of problems need to be considered. Consequently, efficiency may be sacrificed for generality in that functions can no longer be geared toward a specific system type but must be applicable to subclassed system types.

4.2) Reusability

As noted above, reusability is an essential piece of OO system design. Meyer defines reusability as, "the ability of software products to be reused, in whole or part, for new applications" [23]. To illustrate the concept of reusability we will discuss the geometry

hierarchy within GEMF. Since all of these system types use a similar philosophy for meshing a CellMap class was created to store all the common information between system and analysis types (see Figure 4.2). Information such as node lists, edge and face maps, and topology links (Volume, Face, Edge, Node) can all be accessed from the CellMap.

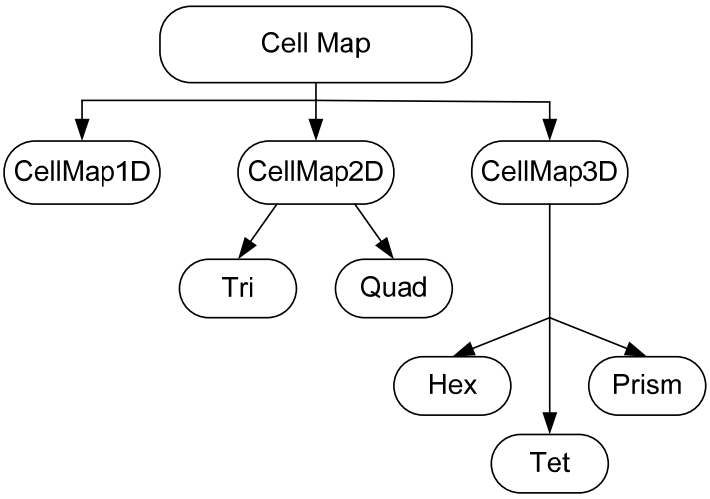

Figure 4.2 CellMap Hierarchy

Note that the CellMap is subclassed into three distinct types, CellMap3D, 2D, and 1D. This allows for general use with 3D, 2D, or 1D meshes without needless extra storage. For the surface Nyström implementation in GEMF only the 2D and 1D levels need to be used. However, if a 3D Finite Element method were introduced the CellMap3D would be needed but could be accessed without changing the CellMap structure. Thus, the above CellMap class can be reused regardless of the system or analysis requirements.

4.2.1) Inheritance

An important property that allows classes like CellMap to be reused is called inheritance. Again consider Figure 4.2. Note that both the CellMap3D and 2D are subclassed into different types, Quads and Tris for CellMap2D, and Hexes, Tets, and Prisms for CellMap3D. This again allows for general meshing and gives two options at the 2D level and three options at the 3D level. For simplicity let's consider the two subclasses of CellMap2D, Tris and Quads. Both the Tri and Quad subclasses will inherit all common functions and data from their parent class CellMap2D (e.g. nodelist, dimension). They

will then have the ability to add their own features such as edge node mapping (which of course would be different for Tris and Quads). Thus, the Tri and Quad classes are known as subclasses or descendants of the CellMap2D class. The same inheritance argument applies for the subclasses of CellMap3D.

4.3) Flexibility

Reusability and inheritance provide some means toward flexibility in an OO system. Another additional means of flexibility is known as dynamic binding. Dynamic binding is considered the most important aspect of runtime flexibility. Again we can consider the CellMap hierarchy of Figure 4.2 to illustrate. For each CellMap type, a position vector can be established given a local coordinate point of interest:

```
virtual R3 rPositionVector(const GEMFFLOAT *lc, SilvesterArgs &args)
const = 0;
```

The explicit calculation of position vectors depends on the CellMap type thus the method is defined at the base class level as a virtual function which then will be implemented specifically at the leaf or child class level. Subsequently, QuadCellMap2D and TetCellMap3D will have different implementations of the same `rPositionVector` function. Dynamic binding ensures that at runtime the correct version of this same function is called depending on the current CellMap type.

Dynamic binding offers another flexibility: Since the caller in the Nyström System only knows about CellMaps and calls the `rPositionVector` routine only via the base class, there is no need to change the Nyström System implementation when another CellMap type is added (e.g. OctagonCellMap2D). Dynamic binding will still choose the correct version of `rPositionVector` at runtime. This property of dynamic binding is very important to OO system design. Classically new code could only use older code. Now with dynamic binding old code can use new code without any retrofits!

4.4) Maintainability

Within the context of maintainability, software engineers are mostly concerned with the amount of effort needed to implement changes in user requirements, changes in data formatting, bug fixing, etc. In fact, according to [23] an estimated 70% of software costs are devoted to maintenance. Consequently, a quality piece of software cannot afford to neglect the aspects of maintainability. Fortunately, OO system design ensures that these maintenance procedures are relatively simple and straightforward to perform. All of the aforementioned topics are foundations of maintainability. Changes due to user requirements or data formatting should be easy to apply following the paradigms of reusability and flexibility: existing source code would only need to be modified in one place or (if the user decides not to change existing code) a new inheritance branch would incorporate the changes. Inheritance also helps to ensure that bugs are localized and not spread over the entire source code tree.

V. GEMF Implementation

Now that a basic motivation for OO system design has been presented we will specifically discuss the Nyström implementation within GEMF. The platform chosen for GEMF was Visual C++. This is one of the more widely used programming platforms and is conducive to OO design. There are many classes and functions which are used within the Nyström framework (too many to discuss in the context of this thesis), subsequently, we will focus on the main classes and functionality of the Nyström method: NyströmSystem, Background and Kernel Classes, and NyströmFillManager.

5.1) NyströmSystem

The NyströmSystem class houses all of the Nyström specific analysis data and functions. As depicted in Figure 4.1.1 NyströmSystem is a subclass of the more general Frequency Domain System. NyströmSystem is the brains behind Nyström analysis in that it handles all of the upper level duties such as SystemCreate, SystemFill, and SystemSolve. Because the Nyström method is applicable to a wide variety of analysis and solver types GEMF has a hierarchy of NyströmSystem types. Further discussion of the NyströmSystem classes will begin by discussing this hierarchy while moving toward the specific branch implemented for this thesis.

5.1.1) Nyström Hierarchy

In an effort to preserve the reusability and flexibility aspects of OO design, the NyströmSystem class has a hierarchy of analysis types, depicted below in Figure 5.1.1. Currently there are three different Nyström analysis types: DenseNyström, FMMNyström, and LOGOSNyström.

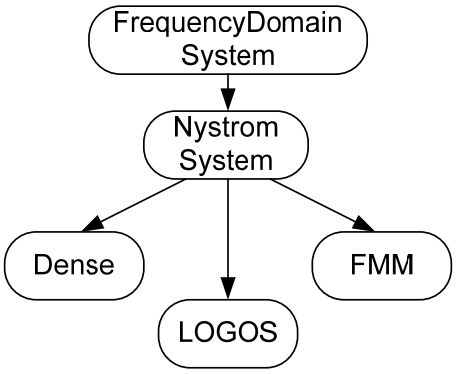

Figure 5.1.1 NyströmSystem Hierarchy.

The DenseNyström class was motivated by a standard direct LU solution method and will be discussed in subsequent sections. FMMNyström and LOGOSNyström are fast iterative and direct methods respectively. They each require a unique filling and storage scheme and thus have their own branch in the Nyström hierarchy.

5.1.2) Input Information

A common link not only between the NyströmSystem types but FrequencyDomainSystem types is the input mesh and simulation information. Currently, this information is read in by a series of function calls at the FrequencyDomainSystem level: `ReadMaterialParameters(),ReadMeshDefinition(),ReadUserControlData()`

The MaterialParameters file contains information such as the number of material surfaces in the mesh (e.g. dielectric, PEC). MeshDefinition contains the output and some basic connectivity data supplied by the meshing tool, such as node and element lists as well as element type (e.g. Quad, Tri, Hex). UserControlData provides simulation specific information such as frequency and default error tolerance.

Three specific functions within the input file context of FrequencyDomainSystem worthy of a brief discussion are `CreateNewBackGNDandKernel, CreateNewCellInfo, and CreateNewElement.` Creating a new Background occurs during the processing of the MaterialParameters file. Each new material type (e.g. dielectric, PEC) has different characteristics and thus a new Background object is created for each material type. The

CellInformation class houses data specific to cells within the mesh. For example, dielectric materials can support both electric and magnetic currents while PEC materials can only support electric current. Therefore, dielectrics and PECs will have separate CellInfo objects which will contain information about its supported current type (amongst other information). Finally, a new element is created with its corresponding node list for each entity specified by the MeshDefinition file.

In an effort to preserve reusability, the creation of both new backgrounds and elements are handled in a switch statement. Consider the following code excerpt:

```
void GEMF::FreqDomainSystem::CreateNewElement(GEMFSYSID elemType,
GEMFSYSID cellInfoID, GEMFSYSID order, std::vector<GEMFSYSID> &nodeIDs)
{
	switch(elemType)
	{
	case LINEAR_QUAD:
		CreateLinearQuadCell(elemType, cellInfoID, order, nodeIDs);
		break;

	case LINEAR_TRI:
		UnsupportedCellTypeWarning(elemType);
		break;

		.
		.
		.

	default:
		UnsupportedCellTypeWarning(elemType);
		break;
	}
}
```

Each time a new element is created its type is passed to this function and the corresponding element is created via the switch statement. Currently there is only support for Linear Quads; the unsupported types will greet the user with a warning message until the correct creation functions come online.

5.1.3) DenseNyströmSystem

As noted above, DenseNyströmSystem is responsible for the higher level system functions: SystemCreate, SystemFill, SystemSolve. SystemCreation creates all of the

objects needed for a Nyström analysis: system matrices, DOF information and storage, generation of an Octree.

Once SystemCreation prepares all of the objects and memory, SystemFill begins computing the overall system matrix. This is done by looping over the Octree structure and filling local element blocks. These local blocks are then assembled into the global matrix via the DOF information established in the system creation. The SystemFill is at such a high level of abstraction that it mainly consists of function calls to other classes (e.g. FillManager) that house the meat of the filling process. Finally, SystemSolve takes the global system matrix and performs a direct LU factorization. The factorized form can then be used to solve multiple right hand side (RHS) forcing vectors.

5.2) Background and Kernel Classes

As mentioned above, the Background and Kernel classes are associated with materials. They both have the class hierarchy depicted in Figure 5.2. Kernels react two cells through the Background's Greens Function. The material information needed in the Kernel evaluations is stored on the Background objects. The actual evaluations are preformed by the Kernel class. Currently only the Homogeneous Media Backgrounds and Kernels are available but the class hierarchy has been established following the above OO paradigm.

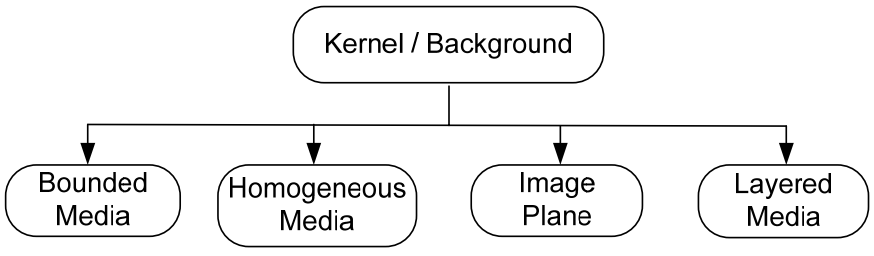

Figure 5.2 Kernel and Background Class Hierarchy

Kernel evaluations are at the core of all frequency domain integral equation methods. Thus, the functions within the Kernel class must be as efficient as possible. Timing

analysis and profiling will be discussed in the results section however one topic worth mentioning is how the Kernel class handles multiple source and field reactions.

Four possible types of source field reactions exist in integral equation methods regardless of the material parameters in the mesh (PEC, Dielectric, etc.): Electric Field due to Electric Current E(J), Electric Field due to Magnetic Current E(M), Magnetic Field due to Electric Current H(J), Magnetic Field due to Magnetic Current H(M). These source field blocks are determined solely by the source and field patches (field patch determines the supported field types, source patch determines the radiating sources). The Kernel evaluations perform all reaction blocks simultaneously to allow common pieces to be reused. Consider the following code excerpt:

```
if(isE)
{
        if(isJ)
        {
                .
                .
                .
        }
        if(isM)
        {
                .
                .
                .
        }
}
```

```
if(isH)
{
        if(isJ)
        {
                .
                .
                .
        }
        if(isM)
        {
                .
                .
                .
        }
}
```

The above *if* blocks determine which source field reactions to compute (Booleans are set prior to Kernel function calls). Thus, only the necessary reactions are computed and common calculations can be efficiently reused.

5.3) NyströmFillManager

The last and perhaps most important section of the Nyström implementation is the NyströmFillManager. As noted above, the FillManager controls all aspects of the local element filling process. Since there are two possible types of reactions for Nyström (near and far reactions) the NyströmFillManager has two subclasses: NearFillManager and

FarFillManager. NyströmFillManager also has ties to the NyströmLEMData class which stores local element data and contains copies of the DOF information used in the assembly. The FillManager hierarchy is diagramed below in Figure 5.3.

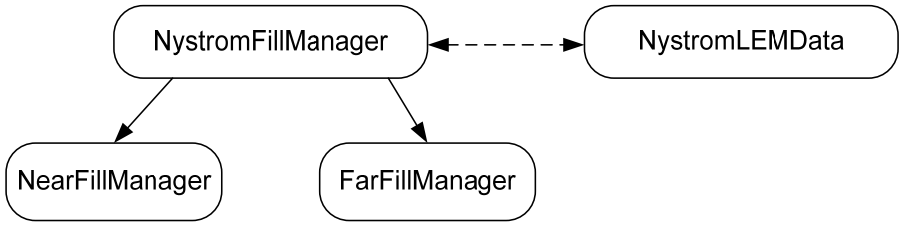

Figure 5.3 NyströmFillManager Hierarchy and tie to NyströmLEMData

The Near and Far FillManagers perform three main tasks: SetupCalculator, SetExternalFunctions, ComputeLEMRow.

SetupCalculator

SetupCalculator uses the idea of "Factories" to establish the numerical integrator used in the filling process. The integrator factory is a way to properly point the NyströmFillManager to the correct integrator desired by the integrand filling functions. The integrator factory requires a topology type (e.g. QuadFace), integration order, and two Booleans which tell the factory whether it should create a singular integrator and/or an adaptive integrator.

```
NumericalIntegrator* CreateNumericalIntegrator(TOPOTYPES &topoType,
                                               int const &order,
                                               bool &isAdaptive,
                                               bool &isSingular);
```

SetExternalFunctions

SetExternalFunctions uses the same factory ideas as SetupCalculator to establish a pointer to the correct integrand filling function. Creating the external function requires the function type and pointer to where the function is located. For Nyström, each

possible integrand function lives on the NyströmFillManager. There are three possible types: IntegrandFunctionNonSelf, IntegrandFunctionSelfPatch, and IntegrandFunctionSelfBoundary.

```
GEMF::ExternalFunction* CreateIntegrandFunction
                        ( const int &intID,
                          NyströmFillManager* const &NyströmFillMgr = 0
                        );
```

ComputeLEMRow

Finally, ComputeLEMRow takes the numerical integrator and external function discussed above and computes the integration. Since these were established using the factory paradigm the integration should be directed to the correct integrand function within the NyströmFillManager.

```
virtual int ComputeIntegration(ExternalFunction* const &integrand,
                                    GEMF::GEMFFLOAT* &resultVector,
                                    const int &vectorLength,
                                    const int &datatype =
                                    GEMF::REAL_DATA_TYPE) = 0;
```

NyströmFillManager Duties

The remaining tasks of the FillManager are preformed at the parent class level. The current source field pairs are passed in via NyströmSystem. Once the FillManager has access to the current source and field cells then decisions on supported field and current types can be made. As noted above the integrand filling functions live and are calculated in the parent level FillManager. The final task performed by the FillManager is computing the Local Element Matrix. This function calls and assembles LEM rows calculated at the child class level. These are then stored on the LEMData structure and passed back to NyströmSystem to be assembled into the global system matrix.

VI. Numerical Results

Now that sufficient background has been presented, our focus will shift toward validating the Nyström implementation within GEMF. There are several benchmarks to compare against. Firstly, as noted above, Nyström methods have previously been utilized in Kentucky's Mscat code (written in Fortran). The code structure for both the GEMF and Mscat codes are the same so in theory their results should be identical. In practice, they have a small amount of error due to numerical round off and finite data precision. Analytical techniques such as the Mie Series can be used for validating canonical geometries (e.g. Cylinders, Spheres). One final validation tool is an RWG Galerkin based IE solver developed in EE 625: *Computational Electrodynamics* at the University of Kentucky. The Galerkin based solver will help prove accuracy for non-canonical geometries. The numerical validation presented is for the EFIE only. Validations for the remaining SIE types will be discussed below in the Future Work section.

The specific cases to be studied are: 1λ radius PEC Sphere, $\frac{3}{4}\lambda$ edge PEC Box, and an open $\frac{3}{4}\lambda$ edge PEC Box; all simulations were run using a base frequency of 1 GHz ($\lambda \approx 0.3m$). Before presenting the results for these test cases we must first briefly discuss a universal measure for scattering problems, Radar Cross Section.

6.0) Radar Cross Section [2]

A typical measure of interest in scattering problems is known as the Radar Cross Section (RCS). RCS is a measure of how a target scatters power in a given direction relative to an isotropic scatterer (one which uniformly scatters power). Before we define the RCS we first must describe the radiated far fields. In the far field, the scattered field becomes an outward traveling plane wave:

$$\vec{E}^{scat} \approx \hat{\theta} E_{\theta}^{scat} + \hat{\phi} E_{\phi}^{scat}$$

$$E_{\theta}^{scat} \approx -jk_0\eta_0 A_{\theta} + \frac{\partial F_{\phi}}{\partial r} \quad E_{\phi}^{scat} \approx -jk_0\eta_0 A_{\phi} - \frac{\partial F_{\theta}}{\partial r}$$

Applying the far field approximation, the scattered fields can be written in terms of their Cartesian projections:

$$E_{\theta}^{scat}(r,\theta,\phi) \approx -jk_0 \frac{e^{-jk_0 r}}{4\pi r} \begin{bmatrix} \cos\theta\cos\phi A_x(\theta,\phi) + \cos\theta\sin\phi A_y(\theta,\phi) - \sin\theta A_z(\theta,\phi) \\ -\sin\phi F_x(\theta,\phi) + \cos\phi F_y(\theta,\phi) \end{bmatrix}$$

$$E_{\phi}^{scat}(r,\theta,\phi) \approx -jk_0 \frac{e^{-jk_0 r}}{4\pi r} \begin{bmatrix} -\sin\phi A_x(\theta,\phi) + \cos\phi A_y(\theta,\phi) \\ +\cos\theta\cos\phi F_x(\theta,\phi) + \cos\theta\sin\phi F_y(\theta,\phi) - \sin\theta F_z(\theta,\phi) \end{bmatrix}$$

$$\vec{A}(r,\theta,\phi) = \eta_0 \frac{e^{-jk_0 r}}{4\pi r} \iint_S J(x',y',z') e^{jk_0(x'\sin\theta\cos\phi + y'\sin\theta\sin\phi + z'\cos\theta)} ds'$$

$$\vec{F}(r,\theta,\phi) = \frac{e^{-jk_0 r}}{4\pi r} \iint_S M(x',y',z') e^{jk_0(x'\sin\theta\cos\phi + y'\sin\theta\sin\phi + z'\cos\theta)} ds'$$

Now the RCS can be defined via the scattered far fields:

$$\sigma_{\theta}(\theta,\phi) = \lim_{r\to\infty} 4\pi r^2 \frac{|E_{\theta}(r,\theta,\phi)|^2}{|E^{inc}(0,0,0)|^2} \quad \sigma_{\phi}(\theta,\phi) = \lim_{r\to\infty} 4\pi r^2 \frac{|E_{\phi}(r,\theta,\phi)|^2}{|E^{inc}(0,0,0)|^2}$$

Rewritten in terms of the far field vector potentials:

$$\sigma_{\theta}(\theta,\phi) = \frac{k_0^2}{4\pi} \frac{|\cos\theta\cos\phi A_x(\theta,\phi) + \cos\theta\sin\phi A_y(\theta,\phi) - \sin\theta A_z(\theta,\phi) - \sin\phi F_x(\theta,\phi) + \cos\phi F_y(\theta,\phi)|^2}{|E^{inc}(0,0,0)|^2}$$

$$\sigma_{\phi}(\theta,\phi) = \frac{k_0^2}{4\pi} \frac{|-\sin\phi A_x(\theta,\phi) + \cos\phi A_y(\theta,\phi) + \cos\theta\cos\phi F_x(\theta,\phi) + \cos\theta\sin\phi F_y(\theta,\phi) - \sin\theta F_z(\theta,\phi)|^2}{|E^{inc}(0,0,0)|^2}$$

Observation angles co-located with the incident field angle yields a MonoStatic RCS, different observation and incident angles yields a BiStatic RCS.

Since there are two incident field projections and two far field projections there are four RCS measures:

$$\sigma_{\theta,\theta}(\theta,\phi) = \sigma_{V,V}(\theta,\phi) = \sigma_{\theta} \text{ due to } E_{\theta}^{inc}$$

$$\sigma_{\theta,\phi}(\theta,\phi) = \sigma_{V,H}(\theta,\phi) = \sigma_{\theta} \text{ due to } E_{\phi}^{inc}$$

$$\sigma_{\phi,\theta}(\theta,\phi) = \sigma_{H,V}(\theta,\phi) = \sigma_{\phi} \text{ due to } E_{\theta}^{inc}$$

$$\sigma_{\phi,\phi}(\theta,\phi) = \sigma_{H,H}(\theta,\phi) = \sigma_{\phi} \text{ due to } E_{\phi}^{inc}$$

6.1) Validation

Scattering from a PEC Sphere

The first validation case was a 1λ radius PEC sphere simulated at a frequency of 1 GHz ($\lambda \approx 0.3m$). Low order geometry modeling was used along with a constant basis order (12 DOFs per patch). Both the GEMF and Mscat codes were ran while increasing the total number of patches. These results were then compared with the Mie Series solution [27]. Bistatic RCS results are depicted below in Figures 6.1.1, 6.1.2, and 6.1.3.

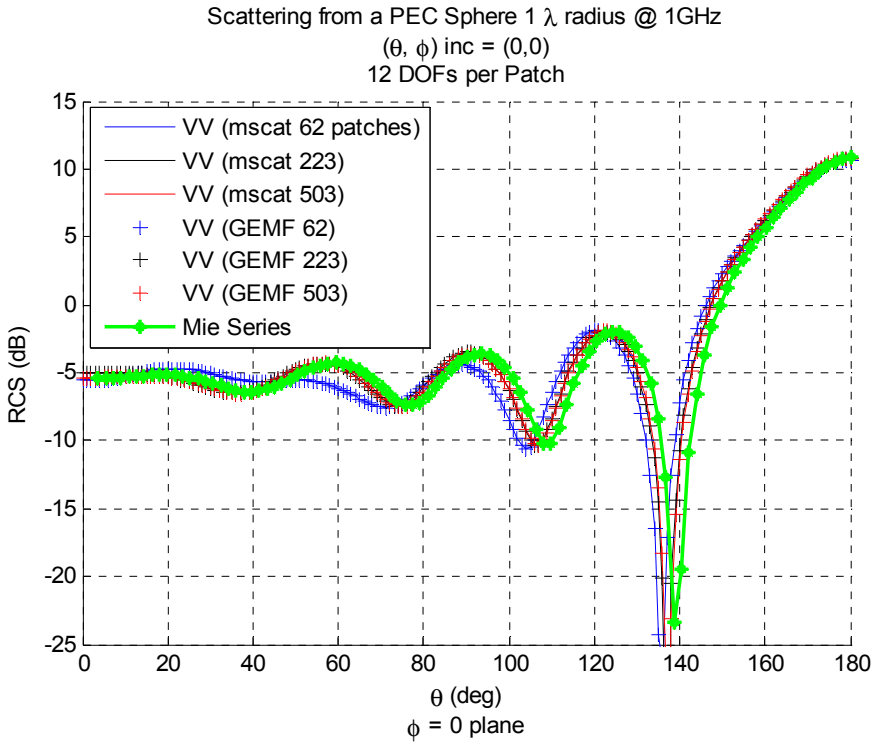

Figure 6.1.1 Scattering from a PEC Sphere 1λ radius @ 1GHz (0.3m) Co-Pole Term. $\sigma_{VV} : \phi = 0°$

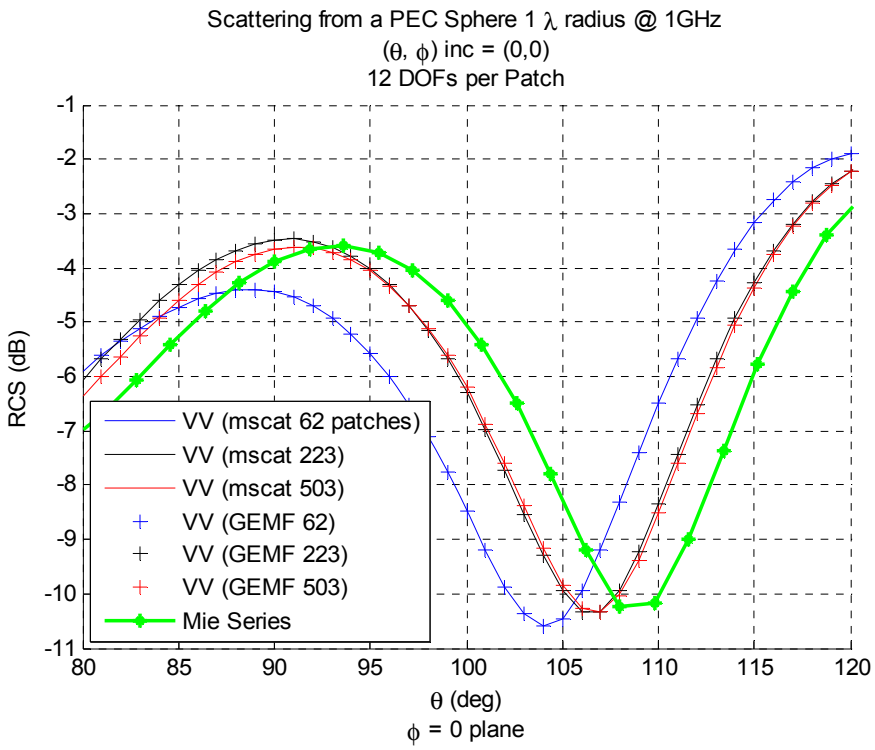

Figure 6.1.2. Enhanced View of PEC Scattering from a Sphere.

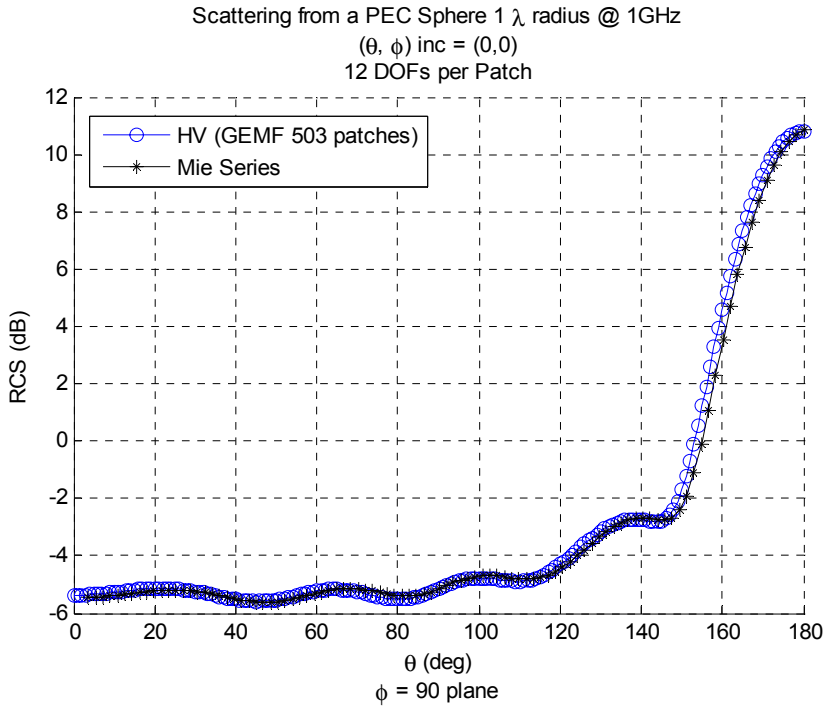

Figure 6.1.3 Scattering from a PEC Sphere Cross Pole Term. $\sigma_{HV} : \phi = 90°$

As you can see from the above plots as the total number of patches are increased both the Mscat and GEMF codes converge to the Mie Series solution. This convergence is limited by the underlying geometric discretization error; thus in order to fully take advantage of the Nyström method we should switch to curvilinear patches to help eliminate geometric error. There is one final note concerning the PEC sphere test case, accuracy of GEMF vs Mscat. Note that both codes appear to have identical answers. In fact, the worst case error observed during debugging and testing was 0.1% relative error. The error observed was from the System Matrix *Z* in a Forbenius Norm sense. That is,

$$\frac{\left\| Z_{mscat} - Z_{GEMF} \right\|_{fro}}{\left\| Z_{mscat} \right\|_{fro}} < 0.1\%$$

Scattering from a PEC Box

The second validation case was a $\frac{3}{4}\lambda$ edge PEC box at 1 GHz. These results were compared to the aforementioned RWG Galerkin solver from EE 625. Note that the incident field was aimed at the corner of the box, $(\theta^{inc}, \phi^{inc}) = (45°, 45°)$. This was to help validate both the incident field excitation and the RCS calculations. BiStatic RCS results are depicted below in Figures 6.1.4 though 6.1.7, MonoStatic RCS in Figure 6.1.8.

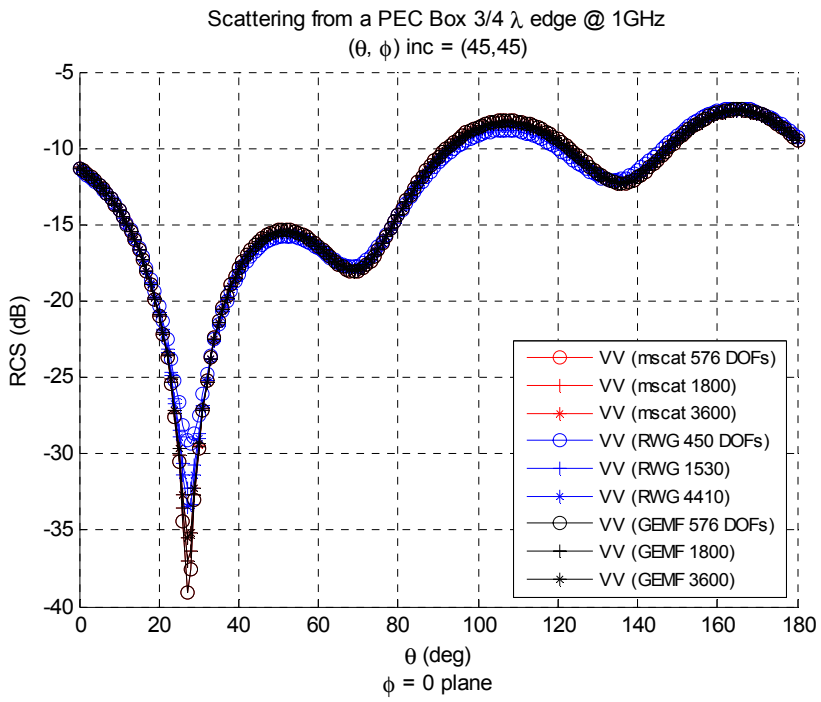

Figure 6.1.4 Scattering from a PEC Box ¾ λ edge @ 1GHz. Co-Pole Term. $\sigma_{VV} : \phi = 0°$

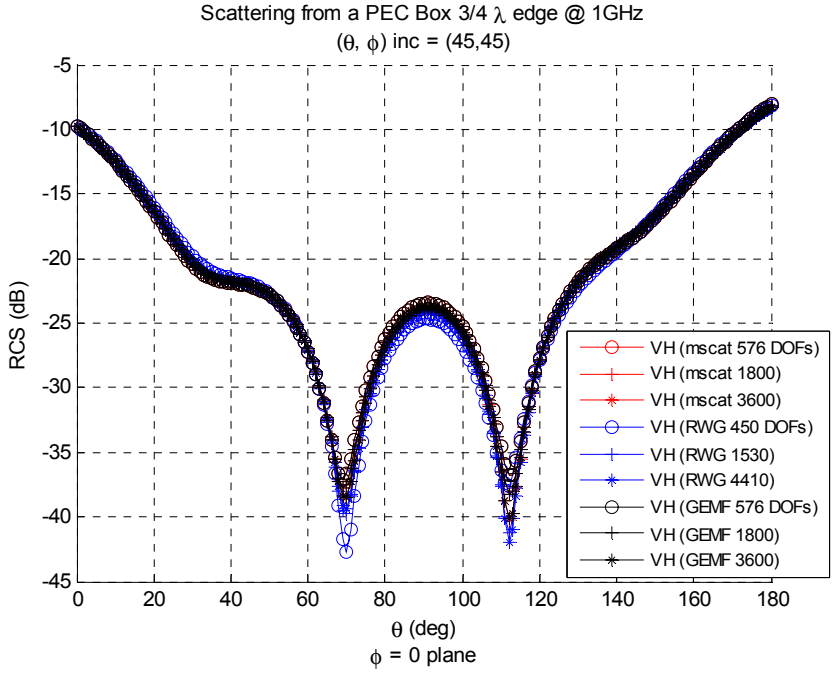

Figure 6.1.5 Scattering from a PEC Box ¾ λ edge @ 1GHz. Cross-Pole Term. $\sigma_{VH} : \phi = 0°$

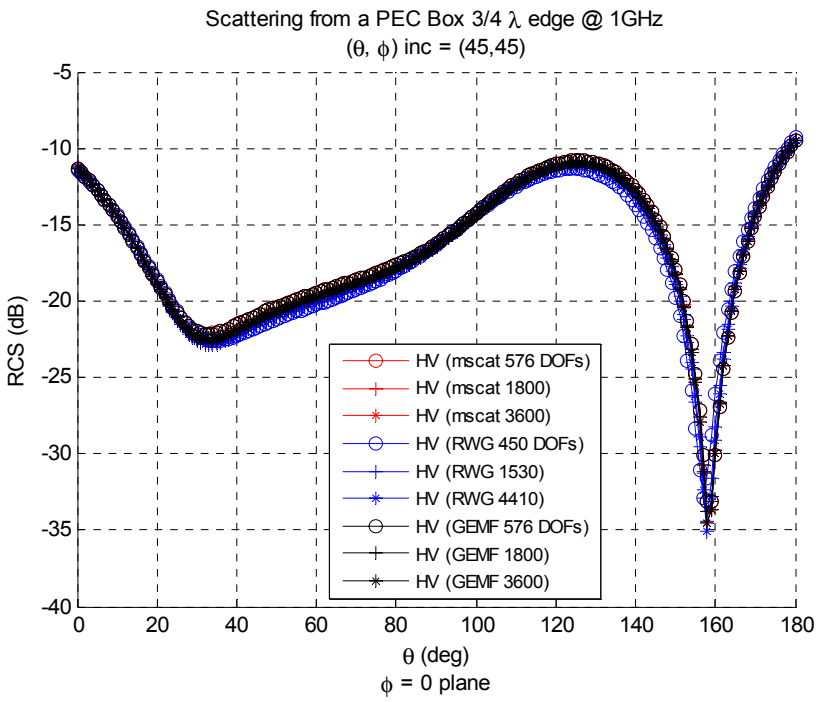

Figure 6.1.6 Scattering from a PEC Box ¾ λ edge @ 1GHz. Cross-Pole Term. $\sigma_{HV} : \phi = 0°$

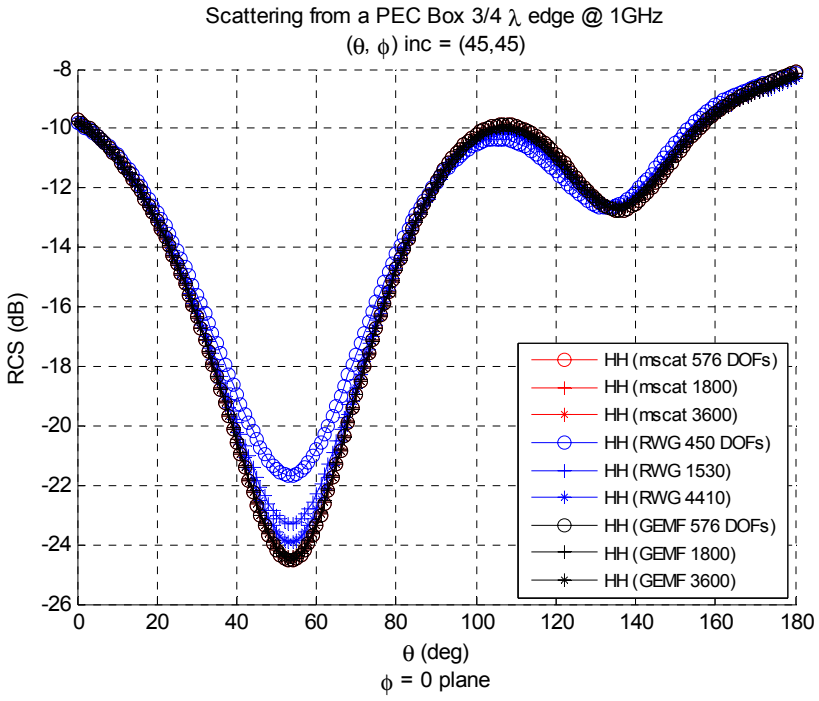

Figure 6.1.7 Scattering from a PEC Box ¾ λ edge @ 1GHz. Co-Pole Term. $\sigma_{HH} : \phi = 0°$

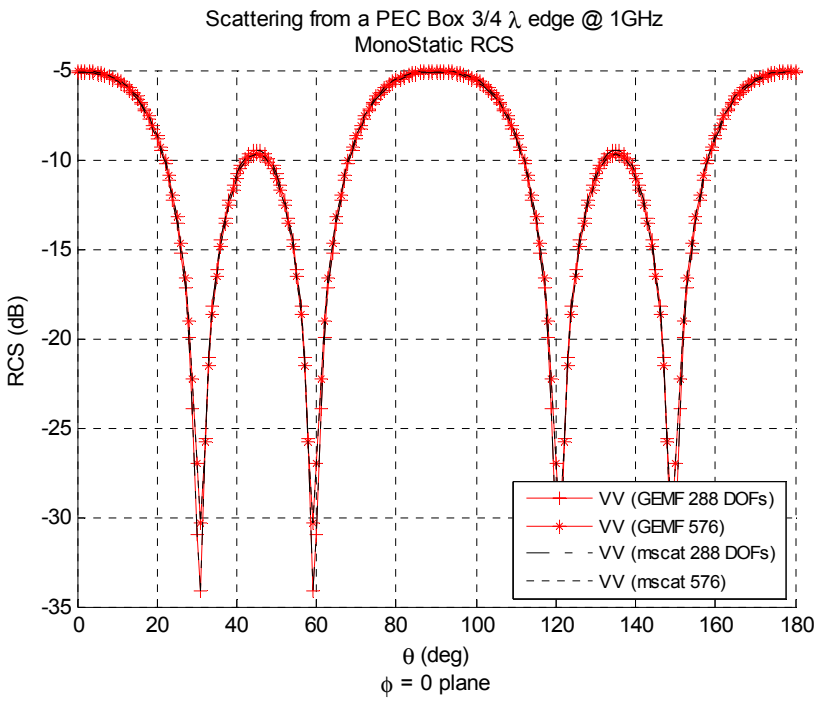

Figure 6.1.8 Scattering from a PEC Box ¾ λ edge @ 1GHz. MonoStatic RCS.

Again, from the results above we can see that all 3 codes converge to the correct answer. Take note of the convergence in Figure 6.1.7. Since there is no discretization error the Nyström code converges faster than the RWG code because of its high order nature. Because of its low order nature the RWG code requires geometric refinement to achieve the correct results while Nyström simply requires a polynomial refinement.

Scattering from an Open PEC Box

The final validation case was an open $\frac{3}{4}\lambda$ edge PEC box at 1 GHz. This problem is more difficult than the previous because the open box allows for an increase in multiple scattering; energy can bounce around inside the box before being scattered back away from the target. The incident field was directed into the opening rather than at a corner. The BiStatic results are below in Figures 6.1.9 and 6.1.10, MonoStatic results in Figure 6.1.11.

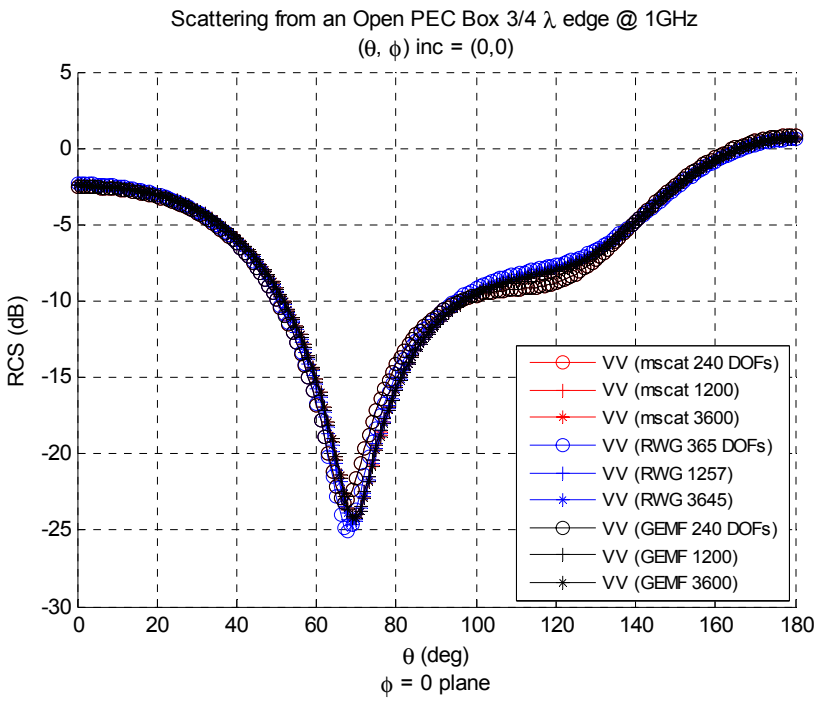

Figure 6.1.9 Scattering from an Open PEC Box ¾ λ edge @ 1GHz. Co-Pole Term. $\sigma_{VV} : \phi = 0°$

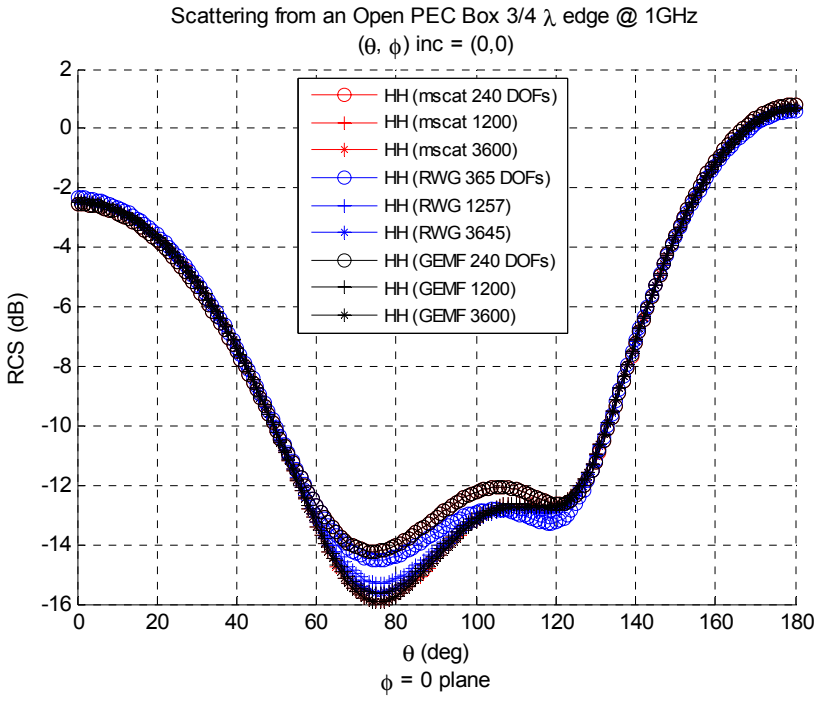

Figure 6.1.10 Scattering from an Open PEC Box ¾ λ edge @ 1GHz. Co-Pole Term. $\sigma_{HH} : \phi = 0°$

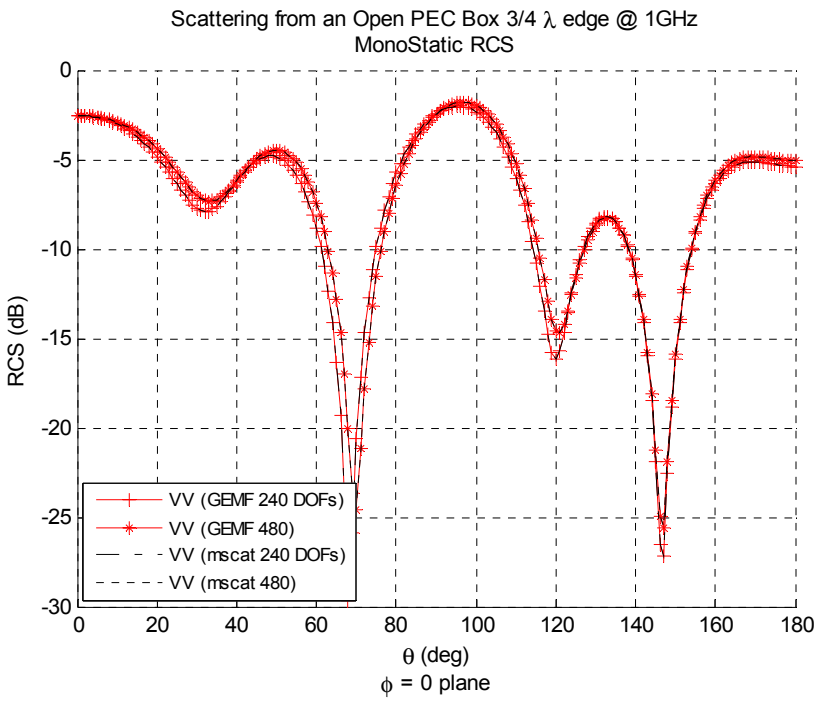

Figure 6.1.11 Scattering from an Open PEC Box ¾ λ edge @ 1GHz. MonoStatic RCS.

6.2) Performance Analysis

As noted previously, there are potential efficiency tradeoffs when using an OO paradigm to develop software. Since the underlying code structure in both Mscat and Nyström within GEMF are similar it seems reasonable to assume their performance would follow suit. Subsequently, a timing study was conducted to determine which code performed better during the system matrix filling and solve routines. Both codes were given the same problem under two different scenarios: variable patches with constant DOFs per patch, and constant number of patches with variable DOFs per patch (the geometry for each test was the same PEC sphere from section 6.1). Below, Figures 6.2.1 and 6.2.2 depict timing results between the Fortran and C++ implementations of the Nyström code.

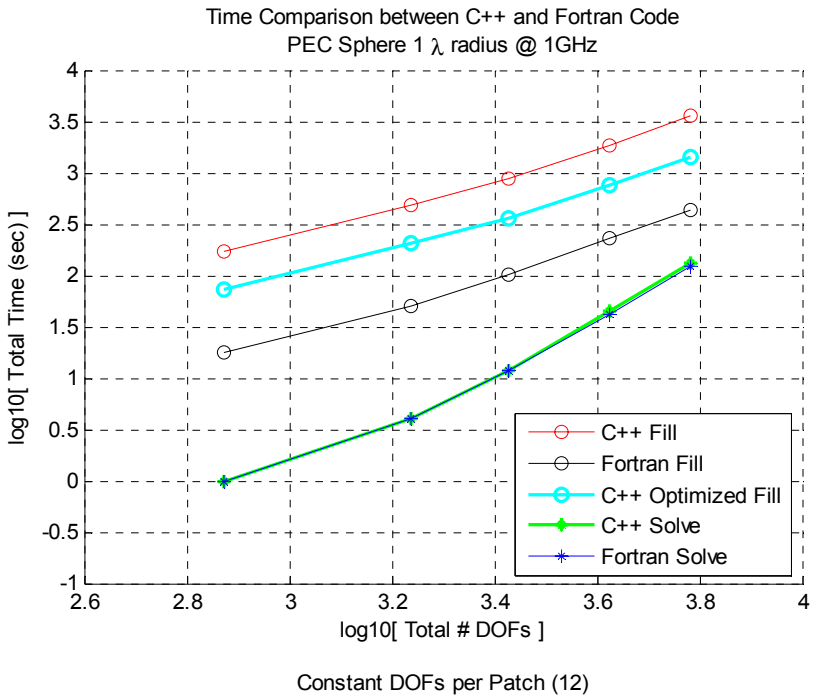

Figure 6.2.1. Timing Comparison Between C++ and Fortran Codes with a Constant Number of DOFs per Patch (12) and Variable Number of Patches.

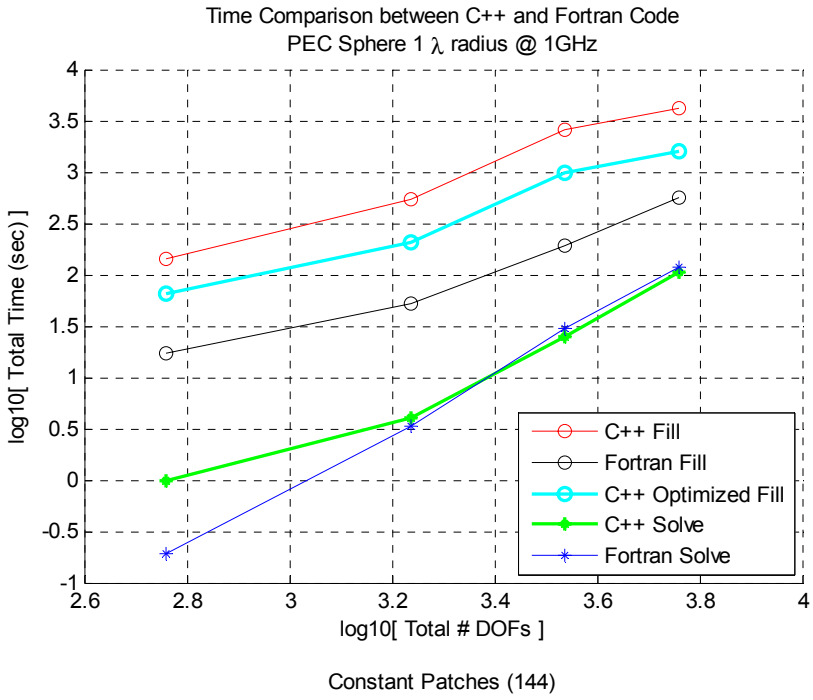

Figure 6.2.2. Timing Comparison Between C++ and Fortran Codes with a Constant Number of Patches (144) and Variable Number of DOFs per Patch.

First, consider the example of variable patches with constant patch DOFs. Note that both the C++ and Fortran codes scale at approximately the same rate. This is expected as the code structures are similar. The Fortran version of the Nyström code is a factor of 8 faster than the C++ version in the fill time. As expected, the solve times were nearly identical; this is simply because both codes use the same LAPACK libraries to calculate the system solution. The second example reinforced all of these points. Fortran was still around a factor of 8 faster. The overall solution times were slightly higher; this was most likely due to the adaptive integrators taking longer to converge because of the increase in near singular terms.

Possible Causes

Some preliminary work has been done to determine the root cause of the speed differences between the Fortran and C++ codes. Using Intel's VTune profiling tool it was thought that the use of C++ Standard Template Libraries (STL) caused the performance difference. Specifically, the use of the STL vector class.

The vector class is a special type of array. Rather than dynamically allocating memory as would be needed to create an array during runtime, vectors can yield the same results simply by using the push_back command. This allows the user to add elements to a vector without using the dynamic memory allocation needed for runtime arrays (in reality, C++ performs the necessary memory allocation behind the scenes).

One can easily see that if the vector is not initialized to the desired size it is possible for C++ to spend unnecessary time allocating memory each time a new element is added to the vector. Thus one possible solution to the performance drop in C++ versus Fortran would be to preallocate all the vectors to their appropriate size. Another alternative would be to use dynamic arrays rather than vectors. The latter option was originally vetoed because using vectors eased code implementation.

These assumptions were verified by preallocating memory used for vectors within C++. These changes caused a factor of 2 performance increase. These results are displayed along with the original timing comparison in Figures 6.2.1 and 6.2.2.

VII. Future Work

Substantial effort has been put forth to create a Nyström surface integral formulation within GEMF. However, there are several remaining tasks to bring this analysis branch of GEMF to completion. The OO paradigm has been at the forefront in creating the Nyström formulation so in theory the remaining pieces should fit seamlessly into the existing framework. In reality, there will be some stumbling blocks and possible code reorganization. Thus, the remaining sections seek to outline these remaining tasks and the details associated therein. Please note that the following sections do not include all of the potential future work (e.g. implementing fast solvers) rather some of the more basic yet still important tasks.

7.0) Additional Surface Integral Formulations

As discussed in the review section of this thesis there are several other SIE methods beyond the Electric Field Integral Equation: 2 formulations for PEC targets, namely the MFIE and CFIE, and the PMCHWT for hybrid PEC and material targets. The core Kernel calculations within GEMF were designed with all of these formulations in mind. For a general target you will have the following reactions:

$E(J)$ Electric Field due to Electric Current Source
$E(M)$ Electric Field due to Magnetic Current Source
$H(J)$ Magnetic Field due to Electric Current Source
$H(M)$ Magnetic Field due to Magnetic Current Source

During numerical integration, the integrand functions have built in Boolean checks to determine the desired Kernel reaction based on the current source and field patch (see Section 5.2). Since the EFIE was the chosen method for preliminary study only the *E(J)* Kernel blocks were validated. The remaining blocks were implemented but not debugged. These additions will specifically impact the following functions within the HomogeneousMediaKernel class: ComputeTdotGdotB, ComputeGradGdotB, ComputeTdotMixedPGFdotB, ComputeGV.

One additional adjustment needs to be made in order to correctly develop the remaining SIE's. Currently, the testing vectors used in the MoM procedure were chosen to be the field patch unitary vectors. For the EFIE, this actually results in the most accurate testing procedure. However, for the MFIE we need to change the testing vectors to the reciprocal unitary vectors ($\hat{n} \times unitary$) to ensure accuracy. The CFIE follows suit with the MFIE as it is simply a linear combination of the two. Ideally the code will automatically chose the correct testing vector based on the formulation desired. The field testing vectors are set in the CalculateFieldTestVectors routine inside the NyströmFillManager.

7.1) High Order Geometry Modeling

The Nyström method is a high order method due to the combination of high order geometry modeling along with high order basis functions. Removing one of these two components reduces the overall effectiveness of the method (as seen above in the numerical results). GEMF is setup to handle arbitrary geometry models. These calculations mainly impact CellMaps. Again, while the code structure is in place high order geometries have yet to be validated.

One additional portion of the code which may need some modification by the addition of high order geometries is the input file readers and subsequent mesh creation. Creating node and edge lists will need to be verified since each input element will have more than the standard number of nodes (e.g. 8 for a 2^{nd} order quad).

7.2) Extension to Volume Integral Formulations

Much like the previous two tasks the extension to Volume Integral Equations (VIE) should be fairly straight forward. Again, VIE's will follow a similar code structure as the current SIE's. However, they will touch different portions of the geometry and integration tools.

7.3) Creating the GEMF Toolbox

Perhaps one of the most important on going efforts will be the expansion and maintenance of GEMF. Eventually, GEMF will be a toolbox from which each subsequent analysis type can access common functions (e.g. integrators, geometry). Currently there are two methods under GEMF, the presented Nyström method and a Discrete Galerkin Finite Element Time Domain (DGFETD) method. These two methods are very unique and thus interact with the GEMF toolbox in vastly different manners. This uniqueness helped draw out faulty designs in the preliminary version of the toolbox. Each new analysis method will no doubt raise new issues with the GEMF toolbox however these initial growing pains have hopefully alleviated potential future problems.

A short term goal to aid in the maintenance of GEMF is the development of a GEMF code repository (Figure 7.3). The GEMF repository will house all of the current analysis types, each having their own module. The code in every module will be unique to that analysis type. Each then will have access to the global GEMF toolbox.

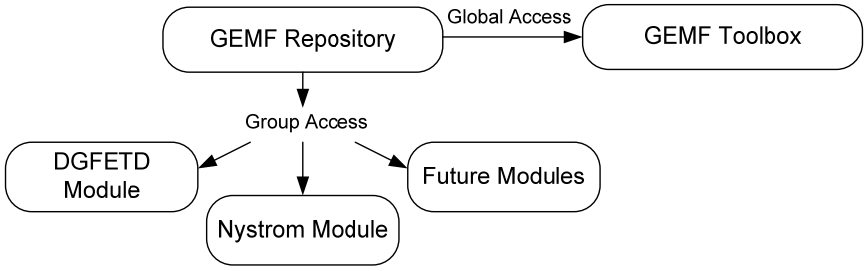

Figure 7.3 GEMF Code Repository

Segregating the code into modules has several advantages. The most important advantage is that everyone will have access to one common GEMF toolbox so there will not be several different revisions existing at one time. Also, through the use of Concurrent Versions System (CVS) tools read and write access can be set for each individual module. This will allow the owner of the code, University of Kentucky, to allow different development partners have access only to particular portions of the code.

There are however a few disclaimers in setting up the repository in the above manner. If the GEMF toolbox ever needed an invasive code change (e.g. changing functions, moving code up or down the inheritance branches) it will be the responsibility of the owner to ensure that each module remains unaffected or receives the necessary changes. For this to be possible, each module will need extensive unit tests and sample problems which can be executed and compared to previous results to validate the changes.

7.4) Performance Improvements

Previously, preliminary effort has been put forth to improve the performance of the implemented Nyström method in C++. These efforts resulted in a factor of 2 increase, however, the overall performance was still a factor of 4 slower than the Fortran version. Subsequently, further investigation is needed to reduce the performance gap.

Again with the aid of Intel's VTune performance analyzer the system fill time was profiled. After the initial modifications, the function call rPositionVector consumes 41% of the overall fill time. This call is expected to be a significant part of the fill process but the overall percentage is believed to be too high. Drilling down a few levels in the profiling tool yields a possible cause: vector class memory management.

The SilvesterArgs class was created to be a storage container for Sylvester polynomial calculations; these are the interpolation polynomials used for CellMap calculations. SilvesterArgs consists of 2 and 3 dimensional vectors and are used throughout the code as temporary memory storage containers (631 instances!). It is believed that these 2 and 3D vectors yield too much overhead and slow down the code due to poor memory management. There are a few potential solutions. First, replace the temporary SilvesterArgs memory containers with WorkSpace arrays. WorkSpace was developed after SilvesterArgs and provides a clean and efficient way to handle dynamic memory and temporary storage containers. Another alternative would be to re-write the SilvesterArgs class to use WorkSpace arrays rather than vectors. This could potentially save time because there are so many instances of SilvesterArgs throughout the code.

References

[1] C. A. Balanis, *Advanced Engineering Electromagnetics*. New Jersey: John Wiley and Sons, 1989.

[2] A. F. Peterson, S. L. Ray, and R. Mittra, *Computational Methods for Electromagnetics*. New York: IEEE Press, 1998.

[3] A. J. Poggio and E. K. Miller, "Integral Equation Solutions of Three Dimensional Scattering Problems," in *Computer Techniques for Electromagnetics*. vol. 7, R. Mittra, Ed. New York: Pergamon Press, 1973.

[4] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic Scattering by Surfaces of Arbitrary Shape," *IEEE Transactions on Antennas and Propagation,* vol. AP-30, 1982.

[5] S. Gedney, "Evaluating the PVI of the MFIE," Lexington, 2006.

[6] Y. Chang and R. F. Harrington, "A Surface Formulation for Characteristic Modes of Material Bodies," *IEEE Transactions on Antennas and Propagation,* vol. 25, pp. 789-795, 1977.

[7] T. K. Wu and L. L. Tsai, "Scattering from Arbitrarily Shaped Lossy Dielectric Bodies of Revolution," *Radio Science,* vol. 12, pp. 709-718, 1977.

[8] W. C. Chew, *Waves and Fields in Inhomogeneous Media*. New York: IEEE Press, 1995.

[9] J. Jin, *The Finite Element Method in Electromagnetics*. New York: John Wiley and Sons, 2002.

[10] G. H. Golub and C. F. V. Loan, *Matrix Computations*. Baltimore: Johns Hopkins University Press, 1996.

[11] J. A. Stratton, *Electromagnetic Theory*. New York: McGraw-Hill, 1941.

[12] L. F. Canino, J. J. Ottusch, M. A. Stalzer, J. L. Visher, and S. M. Wandzura, "Numerical Solution of the Helmholtz Equation in 2D and 3D Using a High-Order Nyström Discretization," *Journal of Computational Physics,* vol. 146, pp. 627-663, 1998.

[13] S. D. Gedney, "High-Order Method of Moments Solution of the Scattering by Three Dimensional PEC Bodies using Quadrature Based Point Matching," *Microwave and Optical Technology Letters,* vol. 29, pp. 303-309, 2001.

[14] S. D. Gedney, "On Deriving a Locally Corrected Nyström Scheme From a Quadrature Sampled Moment Method," *IEEE Transactions on Antennas and Propagation,* vol. 51, 2003.

[15] R. L. Burden and J. D. Faires, *Numerical Analysis*, 7 ed. Pacific Grove, CA: Brooks Cole, 2001.

[16] S. D. Gedney, A. Zhu, and C.-C. Lu, "Study of Mixed Order Basis Functions for the Locally Corrected Nyström Method," *IEEE Transactions on Antennas and Propagation,* vol. 52, pp. 2996-3004, November 2004.

[17] A. F. Peterson and F. Caliskan, "The Need for Mixed Order Representations with the Locally Corrected Nyström Method," *IEEE Antennas and Wireless Propagation Letters,* vol. 2, pp. 72-73, 2003.

[18] M. G. Duffy, "Quadrature over a pyramid or cube of integrands with a singularity at the vertex," *SIAM Journal of Numerical Analysis,* vol. 19, pp. 1260-1262, December 1982.

[19] Wikipedia, "Octree," 2007.

[20] R. Finkel and J. L. Bentley, "Quad Trees: A Data Structure for Retrieval on Composite Keys," *Acta Informatica,* vol. 4, pp. 1-9, 1974.
[21] W. C. Chew, J. Jin, E. Michielssen, and J. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*. Norwood, MA: Artech House, 2001.
[22] S. M. Wandzura, "Electric Current Basis Functions for Curved Surfaces," *Electromagnetics,* vol. 12, p. 77, 1992.
[23] B. Meyer, *Object Oriented Software Construction*. New York: Prentice Hall, 1988.
[24] R. Hannemann, "Modeling and Imaging of Elastodynamic Wave Fields in Inhomogeneous Anisotropic Media an Object Oriented Approach," in *Electrotechnology*. vol. PhD Hessen, Germany: Kassel University, 2001, p. 150.
[25] W. A. Johnson, R. E. Jorgenson, R. M. Sharpe, J. B. Grant, D. R. Wilton, and J. W. Rockway, "EIGER: A New Generation of Computational Electromagnetic Tools," in *Dept. of Energy Tri-Lab Computation Conference* Livermore, CA, 1995.
[26] J. Xiong, L. Sun, Z. Qian, and W. C. Chew, "Organizing Integral Equation for Complex Structure for Object Oriented Programming," *Antennas and Propagation Society International Symposium, IEEE,* pp. 2921-2924, 2006.
[27] V. Demir, A. Elsherbeni, D. Worasawate, and E. Arvas, "A Graphical User Interface for Plane Wave Scattering from a Conducting, Dielectric, or a Chiral Sphere," *IEEE Antennas and Propagation Magazine,* vol. 46, pp. 94-99, Oct. 2004.

VITA

Bryan James Guernsey

Birth Place: Pittsburgh, PA
Date: January 26, 1983

Degree(s): BSEE Virginia Polytechnic Institute and State University
May 2005

Position(s): Aerospace Corporation
Associate Member of Technical Staff
University of Kentucky
Graduate Research Assistant
Nanosonic Inc.
Research Engineer
General Electric Industrial Systems Division
Reliability Engineer

Honor(s): RCTF Scholarship
University of Kentucky

Publication(s):

The Discontinuous Galerkin Finite Element Time Domain Method (DGFETD): A High Order, Globally-Explicit Method for Parallel Computation. Stephen Gedney, Chong Luo, Bryan Guernsey (University of Kentucky); J. Alan Roden, Robert Crawford, Jeffery A. Miller (Aerospace Corporation). 2007 International Symposium on Electromagnetic Compatability.