THESIS

RANDOMIZED HIERARCHICAL SEMI-SEPARABLE STRUCTURES FOR PARALLEL DIRECT DOUBLE-HIGHER-ORDER METHOD OF MOMENTS

Submitted by

Nabeel Moin

Department of Electrical and Computer Engineering

In partial fulfilment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Fall 2017

Master's Committee:

Advisor: Branislav Notaros

Ali Pezeshki
Xinfeng Gao

ABSTRACT

RANDOMIZED HIERARCHICAL SEMI-SEPARABLE STRUCTURES FOR PARALLEL DIRECT DOUBLE-HIGHER-ORDER METHOD OF MOMENTS

As technology grows more and more rapidly, the need for large-scale electromagnetics modelling arises. This includes software that can handle very large problems and simulate them quickly. The goal of this research is to introduce some randomized techniques to existing methods to increase the speed and efficiency of Computational Electromagnetics (CEM) simulations.

A particularly effective existing method is the Surface Integral Equation (SIE) formulation of the Method of Moments (MoM) using Double Higher Order (DHO) modelling. The advantage of this method is that it can typically model geometries with fewer unknowns, but the disadvantage is that the system matrix is fully dense. In order to counter this drawback, we utilize Hierarchical Semi-separable Structures (HSS), a data-sparse representation that expresses the off-diagonal blocks of the matrix in terms of low rank approximations. This improves both the speed and memory efficiency of the DHO-MoM-SIE.

Of the three steps of HSS (construction, factorization, and solving), the one with the most computational cost is construction, with a complexity of $O(rN^2)$, where N is the size of the matrix and r is maximum rank of the off-diagonal blocks. This step can be improved by constructing the HSS form with Randomized Sampling (RS). If a vector can be applied to the system matrix in $O(N^{1.5})$ time, which we accomplish by means of the Fast Multipole Method (FMM) then the HSS construction time is reduced to $O(r^2N^{1.5})$. This work presents the theory of the above methods. Numerical validation will also be presented.

ACKNOWLEDGEMENTS

I would first like to thank my research advisor, teacher, and mentor, Professor Branislav Notaros, for guiding me through this project. I would also like to thank my committee members, Professor Pezeshki and Professor Gao.

Next, I would like to thank my fellow graduate students and lab-mates: Sanja, Pranav, Aaron, and Cameron. Their wealth of knowledge, their entertaining conversations, and their company during the countless mid-morning coffee breaks helped keep me sane through the long weeks.

Lastly, and most importantly, I would like to thank my family. Without their endless support, from my earliest memories to the hectic final stages of writing this thesis, I would not have been able to accomplish nearly as much.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

A. Background

Although Maxwell's Equations have been around for nearly a century and a half [1], the real challenge is solving them. Unfortunately, analytical solutions for electromagnetic scattering only exist for very simple shapes exhibiting a high degree of symmetry, such as perfect spheres [2]. Yet problems that involve electromagnetics, such as antenna design, radar, satellite communication, and optics rarely have such symmetries.

The field of Computational Electromagnetics (CEM) solves Maxwell's Equations approximately by employing a myriad of numerical methods. These methods can consist of procedures such as the discretization of the object into smaller, simpler objects; discretizing time; applying boundary conditions; approximating derivatives with differences or integrals with summations; etc [3].

Frequency-domain CEM methods can be classified into two broad categories [4]: Integral Equation (IE) methods [5-9] and (Partial) Differential Equation (PDE or DE) methods [10-11]. Each class has its strengths and weaknesses: PDE methods, such as the Finite Element Method (FEM), produce sparse system matrices, but require discretization of the empty space around the object, whereas IE methods such as the Method of Moments (MoM) discretizes only the object itself, but does not, in general, have any sparsity pattern [5]. The method that we examine in this work is the Method of Moments, and as such we will be concerned with overcoming the challenges of solving a fully dense system of linear equations.

The fully dense system matrix is not a problem for small problem sizes. However, since the matrix solving step has a computational complexity of $O(N^3)$ for an $N \times N$ matrix, for very large problems the computational cost becomes prohibitively high [12]. A particularly effective strategy of countering this weakness is known as Hierarchical Semi-Separable Structures (HSS) [13]. This involves making low-rank approximations to the off-diagonal blocks of the system matrix by expressing them as a product of orthonormal and triangular matrices via a modified version of the well-known QR decomposition. With this structure in place, the solving complexity decreases drastically to $O(rN)$ where r is the maximum rank of the off-diagonal block [14]. The tradeoff is that the HSS compression step is $O(rN^2)$. This is the new bottleneck of the MoM-HSS process, and hence the overall complexity of the method is $O(rN^2)$.

The goal of this research is to modify an existing MoM solver that uses HSS in order to decrease the overall complexity. We aim to employ the technique of Randomized Sampling (RS) in order to do so [15-17]. This technique rapidly speeds up the HSS compression step. It can decrease the complexity to $O(r^2N)$. The method entails "sampling" the columns of the system matrix before the QR-like decomposition by multiplying by a series of random, Gaussian vectors.

The speedup of this method is dependent on being able to perform a faster matrix-vector multiplication than the classical $O(N^2)$ for dense matrices. So, we pre-fill the matrix using the Fast Multipole Method (FMM) which allows for $O(N^{1.5})$ matrix-vector multiplications [18-23]. FMM algorithms involve approximating the interaction between collections of particles (in our case, basis functions) that are very far away from each other with the interaction between equivalent, aggregated sources. As a result, the sections of the system matrix corresponding to well-separated basis functions are expressed as the outer product between vectors. Without HSS or RS, this method is typically used in conjunction with an iterative solver in order to exploit the

fast multiply, although there is some recent work on developing FMM codes that solve the matrix directly [24-25].

B. Organization

The first few chapters of this thesis detail the theoretical aspects of the various components of the method. Chapter 2 of this thesis will cover the central component of the method, the Method of Moments with the Surface Integral Equations (SIE) formulation. Double-Higher-Order (DHO) modelling will also be discussed. Chapter 3 briefly outlines the HSS theory, including the matrix structure, the compression step, and a comparison between the complexities of its associated algorithms. Chapter 4 will detail the RS process and its benefit to HSS. Chapter 5 will cover the steps in FMM method.

Chapter 6 will present numerical results. Agreement with analytical results and classical MoM results will be demonstrated. Additionally, an empirical measurement of the complexity and scalability will be presented. Finally, Chapter 7 will summarize the findings of the thesis as well as suggesting some directions of future study.

CHAPTER 2: THE METHOD OF MOMENTS

One of the most common techniques used to analyze frequency-domain CEM problems is the Method of Moments (MoM, sometimes referred to as the Boundary Element Method, BEM) with the Surface Integral Equations (SIE) formulation. We employ this method to numerically solve for the electromagnetic scattering from an arbitrary conducting body. The method can be modified very slightly to be made to analyze homogenous or piecewise inhomogeneous dielectric bodies as well, but this work will only consider metallic structures. See [26] for more detail on the SIE formulation that includes dielectric bodies.

The MoM formulation can be summarized as such [3]: the method starts by discretizing the input object or surface that is being analyzed into small, two-dimensional patches. A boundary condition relates the incident electric field (also an input to the method) to the electric field that is scattered from this object. The scattered field can in turn be expressed as integrals of the currents on the surface of the object (hence, SIE), which are defined locally on each patch in terms of basis functions with unknown weights.

In order to turn the single boundary condition equation with many unknown weights (an underdetermined problem) into a system of many equations, the technique of Galerkin testing is employed. During this step, we take the inner product between the boundary equation with each of the basis functions to produce a linear system of equations.

We employ Double Higher Order (DHO) modelling: using higher order geometric elements and higher order polynomial basis and testing functions. This approach allows for the analysis of large structures with fewer unknowns than typical methods that use low order geometric modelling and basis functions.

A. Surface Integral Equations

This section presents a brief overview of the MoM with SIE formulation. Suppose a metallic structure is excited by an impinging time-harmonic wave with field intensity $\mathbf{E}^{inc}$ and frequency ω. Surface currents $\mathbf{J}_S$ will be induced on the surface of the conductor, which will then re-radiate a scattered field $\mathbf{E}^{scat}$. We model the metallic surface with Perfect Electric Conductor (PEC) which is a non-physical material with infinite conductivity. In that case, the incident and scattered fields must satisfy the following boundary condition on the surface of the conductor, S:

$$\left(\mathbf{E}_{tang}^{inc} + \mathbf{E}_{tang}^{scat}(\mathbf{J}_S)\right)_{|r\in S} = 0 \tag{2.1}$$

Or equivalently:

$$\mathbf{E}_{tang}^{scat}(\mathbf{J}_S)_{|r\in S} = -(\mathbf{E}_{tang}^{inc})_{|r\in S} \tag{2.2}$$

The dependence of the scattered electric field on the surface current density can be seen by first expressing the electric field in terms of the electric scalar potential Φ and the magnetic vector potential $\mathbf{A}$:

$$\mathbf{E}(\mathbf{J}_S) = -j\omega\mathbf{A} - \nabla\Phi \tag{2.3}$$

and then expressing the potentials in equation (1.3) in terms of the surface current density:

$$\mathbf{A} = \mu_0 \int_S \mathbf{J}_S g \, dS \qquad \text{and} \qquad \Phi = \frac{j}{\omega\varepsilon_0} \int_S (\nabla_S \cdot \mathbf{J}_S)\, g dS \tag{2.4}$$

where μ and ε are the permeability and permittivity of free space, respectively, j is the imaginary unit equal to $\sqrt{-1}$, and g is the Green's function for an unbounded homogenous medium given by:

$$g = \frac{e^{-j\omega\sqrt{\varepsilon\mu}R}}{4\pi R} \tag{2.5}$$

The surface S is discretized into smaller patches (also commonly referred to as "elements") that are typically some kind of triangular or quadrilateral shape. This will be referred to as the "meshing" step. We then express the unknown surface current density on each patch as a weighted sum of predetermined basis functions, $\mathbf{f}_i$:

$$\mathbf{J}_S = \sum_{i=1}^{N} a_i \mathbf{f}_i(\mathbf{r}) \tag{2.6}$$

where a_i are the unknown weighting coefficients and N is the number of basis functions that we are using to describe the surface current density.

This leaves us with N unknowns but only one equation, (2.2). So, the system is "tested" in which we multiply both sides of equation (2.2) by a testing function (or weighting function) and integrate over the domain of the testing function, S_w:

$$\int_{S_w} w\, E^{scat}(J_S) dS_w = -\int_{S_w} w E^{inc} dS_w \tag{2.7}$$

The most common choice for the testing functions are the basis functions themselves, in which case it is referred to as "Galerkin" testing. Combining equations (2.2) through (2.7) and applying some vector identities, we obtain the following expression for the i^{th} testing equation:

$$-j\omega\mu \int_{S_i}\int_{S}\left(\mathbf{f}_i \cdot \sum_{i=1}^{N} a_i \mathbf{f}_i(\mathbf{r})\right) g dS dS_i + \frac{j}{\omega\varepsilon}\int_{S_i}\int_{S}\left((\nabla\cdot\mathbf{f}_i)\cdot\left(\sum_{i=1}^{N} a_i \nabla\cdot\mathbf{f}_i(\mathbf{r})\right)\right) g dS dS_i$$

$$= -\int_{S_i} \mathbf{f}_i E^{inc} dS_i \tag{2.8}$$

With some assumptions on the above integrals, the summations can be exchanged with the integrations, resulting in the following MoM system equation:

$$[Z]\{a\} = \{V\} \tag{2.9}$$

where [Z], an $N\times N$ matrix, is referred to as the "Galerkin impedance matrix" or "system matrix" whose entries are given by:

$$Z_{ij} = -j\omega\mu \int_{S_i}\int_{S_j}\left(\mathbf{f}_i \cdot \mathbf{f}_j\right) g\, dS_j dS_i + \frac{j}{\omega\varepsilon}\int_{S_i}\int_{S_j}\left((\nabla\cdot\mathbf{f}_i)\cdot\left(\nabla\cdot\mathbf{f}_j\right)\right) g\, dS_j dS_i \tag{2.10}$$

{a}, an $N\times 1$ vector, contains the unknown weighting coefficients from equation (2.6) and is referred to as the "unknowns" or "currents" vector, and {V}, an $N\times 1$ vector, is referred to as the "excitation" vector and is calculated as:

$$V_i = -\int_{S_i} \mathbf{f}_i E^{inc} dS_i \tag{2.11}$$

B. Double Higher Order

In the meshing step above, the structure is generally subdivided into smaller, simple shapes such as flat triangles or flat quadrilaterals. These linear geometric approximations are limited in how well they can model the surface. In our meshing, we use curved elements that are defined by

a series of interpolation nodes, $\mathbf{r}_{kl}$, and then interpolated by Lagrange polynomials, L_k. The elements can by described analytically by the following parametric equation:

$$\mathbf{r}(u,v) = \sum_{k=0}^{K_u} \sum_{l=0}^{K_v} \mathbf{r}_{kl} L_k(u) L_l(v), \qquad -1 \le u, v \le 1 \tag{2.12}$$

Additionally, the basis functions are commonly chosen to be low order (linear or constant), resulting in few unknowns per element. The downside is that the structure must be discretized into a larger number of elements. Moreover, the lower order approach is restricted in the current patterns it can produce. For those reasons, we utilize higher order basis functions, where the surface current density is approximated as:

$$\mathbf{J}_S(u,v) = \sum_{m=0}^{N_u} \sum_{n=0}^{N_v-1} a_{u,mn} \mathbf{f}_{u,mn}(u,v) + \sum_{m=0}^{N_u-1} \sum_{n=0}^{N_v} a_{v,mn} \mathbf{f}_{v,mn}(u,v) \tag{2.13}$$

and the basis functions are given by

$$\mathbf{f}_{u,mn}(u,v) = \frac{P_m(u) v^n}{J(u,v)} \boldsymbol{a}_u(u,v) \tag{2.14}$$

$$\mathbf{f}_{v,mn}(u,v) = \frac{P_n(u) v^m}{J(u,v)} \boldsymbol{a}_v(u,v) \tag{2.15}$$

$$P_k(u) = f(x) = \begin{cases} 1-u, & k = 0 \\ 1+u, & k = 1 \\ u^k - 1, & k \ge 2, even \\ u^k - u, & k \ge 3, odd \end{cases} \tag{2.16}$$

where the unitary vectors $\boldsymbol{a}_u$ and $\boldsymbol{a}_v$ and the Jacobian J are given in terms of $\mathbf{r}$ from equation (2.12) by

$$\boldsymbol{a}_u(u,v) = \frac{\partial \mathbf{r}(u,v)}{\partial u}, \qquad \boldsymbol{a}_v = \frac{\partial \mathbf{r}(u,v)}{\partial v}, \qquad \text{and} \tag{2.17}$$

$$J(u,v) = |\boldsymbol{a}_u(u,v) \times \boldsymbol{a}_v(u,v)| \tag{2.18}$$

Note that our basis functions in (2.14) and (2.15) are not orthogonal. This means that the system

CHAPTER 3: HIERARCHICAL SEMI-SEPERABLE STRUCTURES

A. HSS Preliminaries

In this chapter, the definitions and algorithms associated with Hierarchical Semi-separable Structures (HSS) will be discussed [13]. HSS is a matrix representation that expresses a matrix in terms of low-rank approximations to off-diagonal blocks in a hierarchical manner, very similar to Hackbusch's $\mathcal{H}$-Matrices [27-29]. There are numerous advantages to using HSS such as memory efficiency and computational speedup, and it is especially applicable to CEM problems due the asymptotic behavior of the Green's function. Furthermore, since HSS is an algebraic method, it can easily be implemented in existing solvers.

The initial step of constructing an HSS representation of an nxn matrix A is to partition the row indices (or equivalently, the column indices) into 2^{k-1} disjoint subsets l_i:

$$\bigcup_{1}^{2^{k-1}} l_i = S \qquad \text{and} \qquad l_i \cap l_j = \emptyset, \qquad i \neq j \tag{3.1}$$

$$S = \{1,2,3,\dots,n\}$$

where n is the size of matrix A, and k is the HSS level. These sets are then arranged as the leaves in a proper binary tree where the leaf nodes are l_i and each non-leaf node is the union of its two children. The tree is then "post-ordered" such that the right child of the nth node is the (n-1)th node. An example of a 4-level post-ordered HSS tree can be seen in figure 3.1.

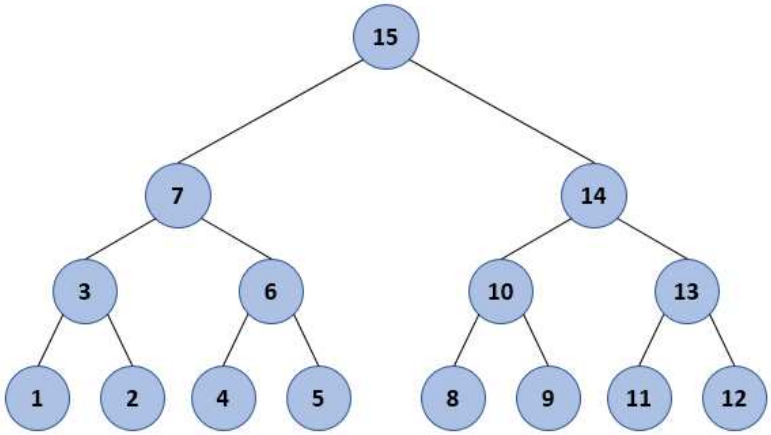


Figure 3.1: An example of a post-ordered, four-level HSS tree

B. Rank-revealing QR decomposition

Before we proceed with the details of the construction of the HSS matrix, the rank-revealing QR (RRQR) decomposition must be discussed. This factorization method is based on the Gram-Schmidt process, similar to the well-known QR decomposition, but whereas QR desires an exact factorization $A = QT$, the goal of the RRQR is instead to find an approximate factorization $A \approx QT$. In the case of the QR decomposition of a matrix A of size $M \times N$, the factor sizes are $M \times r$ and $r \times N$ for Q and T, respectively, where r is the rank of matrix A. In contrast, the factors from RRQR are of size $M \times r_\varepsilon$ and $r_\varepsilon \times N$ for Q and T, where r_ε is referred to as the approximate rank or "ε-rank" of matrix A. This process will allow us to express the off-diagonal blocks of a matrix in terms of low-rank approximate factorizations.

The procedure for the RRQR with an $M \times N$ input matrix A and input tolerance ε is as follows:

1. Find the column of A with the largest Euclidian norm. We call this the jth column, a_j
2. Swap the ith column, a_i, with a_j (where i is the iteration counter)
3. Normalize a_i to produce $\hat{a}_i$. Store the norm of a_i as the ith diagonal entry of T: $t_{ii} = \|a_i\|_2$; and the normalized column $\hat{a}_i$ as the ith column of Q: $q_i = \hat{a}_i$. If $t_{ii} \leq \varepsilon t_{11}$, then the procedure terminates and the ε-rank of A is said to be $r_\varepsilon = (i-1)$
4. Calculate the remaining elements of the ith row of T by projecting a_i onto the columns of A: $t^H = q^H[a_{i+1}\, a_{i+2} \ldots a_N]$
5. Update the matrix A as $A = A - q_i t_i^H$

There are a few features of the RRQR factorization that are worth noting. As a consequence of using the Gram-Schmidt orthogonalization, the columns of Q are orthonormal, so $Q^H Q = I$. Additionally, the matrix T is upper triangular. These are facts that are exploited later in the HSS factorization step [13].

C. HSS construction

The HSS form is generated recursively through a series of RRQR decompositions. At the leaf level of the HSS tree, the diagonal blocks, D_i, are extracted and saved as dense blocks:

$$D_i = A_{t_i \times t_i} \tag{3.2}$$

where × denotes the Cartesian product between the two index sets. Then, the remainder of each block-row is extracted and factorized using a RRQR decomposition (covered in the previous section) in a process referred to as "row compression":

$$A_{t_i \times (S \setminus t_i)} \approx U_i\, A^R_{\hat{t}_i \times (S \setminus t_i)} \tag{3.3}$$

Here, the notation $S\backslash t_i$ refers to the set difference between the full index set S and the leaf index set t_i so that $A_{t_i \times S\backslash t_i}$ is the off-diagonal block row corresponding to the row index set t_i. The matrix on the right-hand-side of the equation A^R denotes the matrix A after it has undergone Row compression. Note also that the post-QR row index set has been replaced with $\hat{t}_i$ to indicate that there is a different index set corresponding to the new matrix A^R. These two factors, U_i and $A^R_{t_i \times S\backslash t_i}$ are stored, and the original matrix A is discarded at the end of the leaf-level row compression step. An example of a leaf-level row-compression step can be seen in figure 3.2.

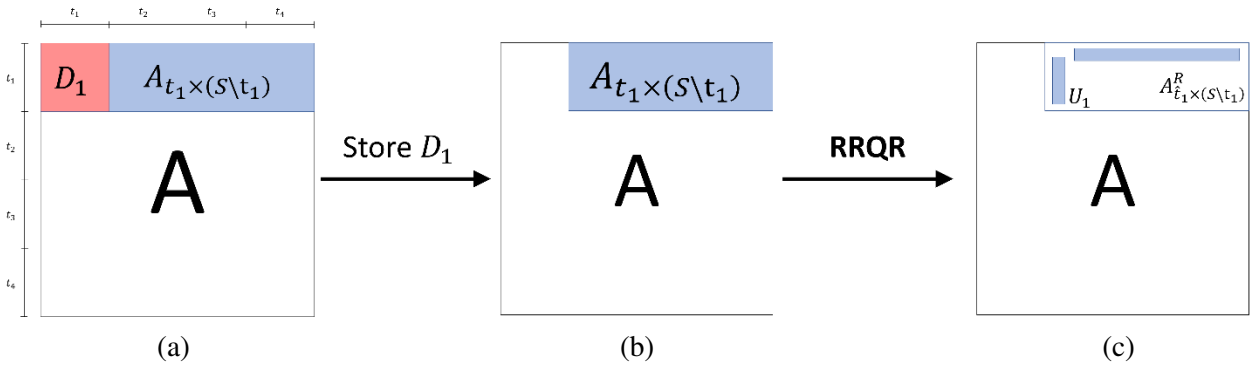


Figure 3.2: An example of leaf-level HSS compression. (a) the block-row is separated into the diagonal block, D_1, and the off-diagonal, $A_{t_1 \times (S\backslash t_1)}$. (b) the diagonal block is stored, and the off-diagonal block undergoes RRQR. (c) after RRQR, the factors U_1 and $A^R_{\hat{t}_1 \times (S\backslash t_1)}$ are stored and the original off-diagonal block is discarded.

Once the leaf-level nodes have undergone row compression, the rest of the tree is traversed in a bottom-up fashion. At each non-leaf node i, the off-diagonal block is constructed by concatenating the appropriate section of each of the generators of its children and subsequently factorized using the RRQR decomposition:

$$\begin{pmatrix} A^R_{\hat{t}_{c1}\times(S\setminus t_i)} \\ A^R_{\hat{t}_{c2}\times(S\setminus t_i)} \end{pmatrix} \approx \begin{pmatrix} R_{c1} \\ R_{c2} \end{pmatrix} A^R_{\hat{t}_i\times(S\setminus t_i)} \tag{3.4}$$

where the subscript $c1\ and\ c2$ refer to node i's left and right children, respectively. Note that, although the two generators that node i inherits are already row compressed, there may be some additional rank-deficiency in the concatenated matrix, so this step is necessary. The portions of A^R that are not forwarded are kept by the children nodes:

$$A_{t_{c1}\times t_{c2}} \approx U_{c1}\, A^R_{\hat{t}_{c1}\times t_{c2}} \qquad and \qquad A_{t_{c2}\times t_{c1}} \approx U_{c2}\, A^R_{\hat{t}_{c2}\times t_{c1}} \tag{3.5}$$

An illustration of the forwarding and row compression process for non-leaf nodes can be seen in figure 3.3

Once row compression has been completed up to the root level, the HSS form undergoes an analogous "column compression" process. First, at each node, a new index set $\bar{t}_i$ is defined which contains the indices of the rows chosen for the basis in the row-compression RRQR step.

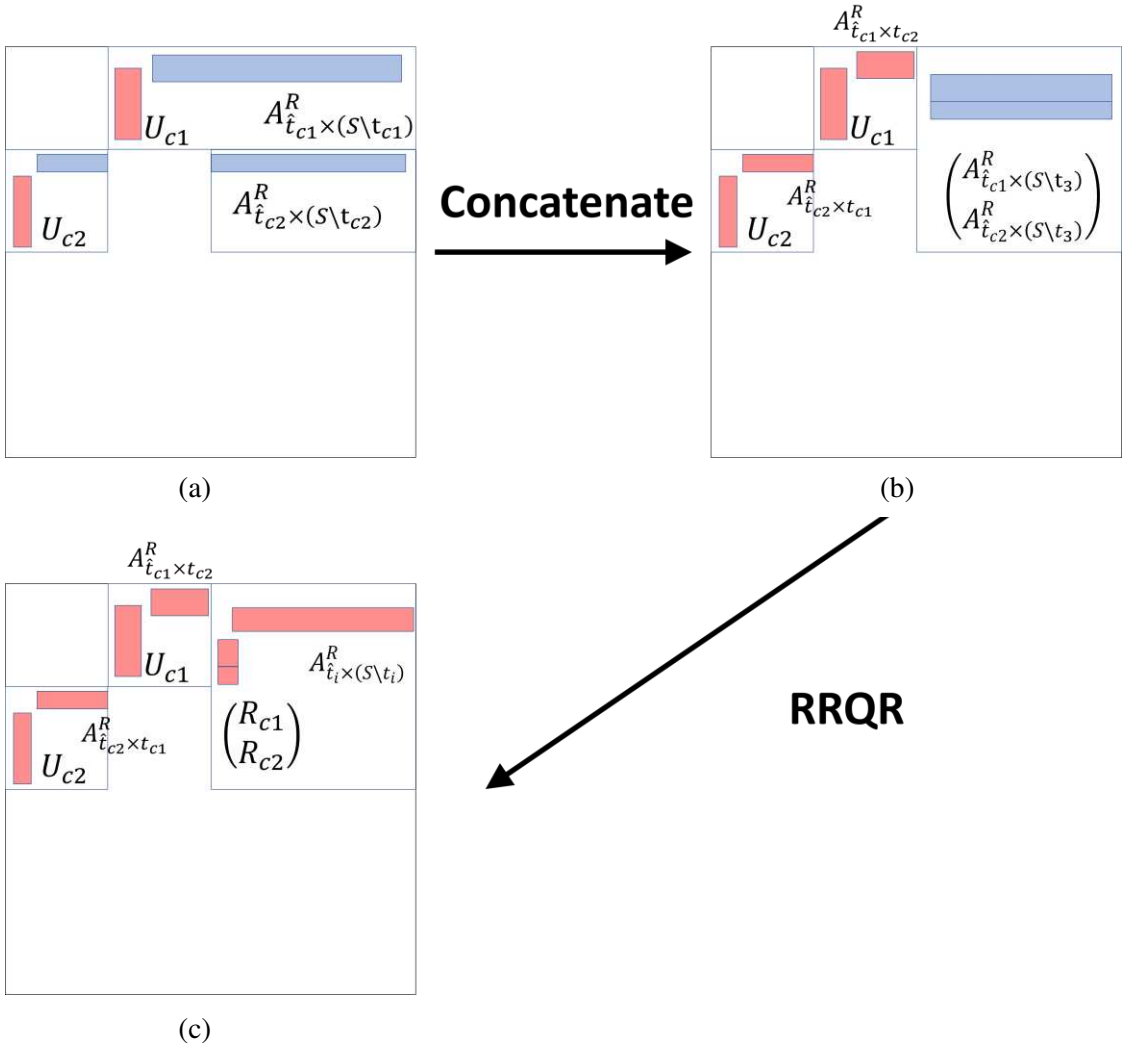


Figure 3.3: An example of non-leaf-level HSS compression, where red represents a final generator and blue indicates that the matrix will go through some additional compression (a) the node inherits the generators of its children (b) the off-diagonal block for this node is constructed from the A^R generators of its two children c1 and c2, and the remainder is kept by the children. (c) After performing RRQR on this concatenated block, the generators R and A^R are stored. Note that this is a depiction of a 3-level HSS tree, and if it was a four-level or higher tree, the A^R generator would not be final.

At the leaf level, column compression is done with another RRQR:

$$A^R_{(\hat{S}\setminus\hat{t}_i)\times t_i} \approx A^{RC}_{(\hat{S}\setminus\hat{t}_i)\times\tilde{t}_i} V_i^H \tag{3.6}$$

where $\hat{S}$ is the union of all row-compressed index sets, $\hat{t}_i$, and A^{RC} is the matrix after it has undergone both row and column compressions. The generators $A^{RC}_{\tilde{t}_i\times\tilde{t}_i}$ and V_i^H are then stored. Just

as in the row compression step, each non-leaf node inherits the generators of its children and then undergoes an RRQR decomposition:

$$\left(A^{RC}_{(\hat{S}\setminus\hat{t}_i)\times\tilde{t}_{c1}}\ A^{RC}_{(\hat{S}\setminus\hat{t}_i)\times\tilde{t}_{c2}}\right) \approx A^{RC}_{(\hat{S}\setminus\hat{t}_i)\times\tilde{t}_i}\begin{pmatrix}W_{c1}\\W_{c2}\end{pmatrix}^H \tag{3.7}$$

Note the notational difference between t_i (original index sets, no accent), $\hat{t}_i$ (row index set after row compression, hat), and $\tilde{t}_i$ (column index set after column compression, tilde). The final generator B is obtained for each child node as the portions of A that are not forwarded to the parent node:

$$B_{ci} = A_{\hat{t}_{c1}\times\tilde{t}_{c2}} \quad \text{and} \quad B_{c2} = A_{\hat{t}_{c2}\times\tilde{t}_{c1}} \tag{3.8}$$

The task of computing the HSS factorization has been shown to have an asymptotic complexity of $O(rN^2)$ where N is the size of the matrix and r is the maximum ε-rank of an off-diagonal block [31].

D. HSS Factorization

Once the HSS form has been constructed, the next step is to perform a ULV factorization on the HSS form in preparation for matrix solving. Finally, the system is solved by starting at the most compressed (root) level and eliminating downward through the HSS tree. This matrix Since this work focuses primarily on the matrix filling and compression stages of the MoM process, the factorization and solving algorithms of HSS will be omitted, and the reader can consult [13] for further detail.

The HSS ULV factorization step has been shown to have an asymptotic complexity of $O(r^2N)$ and the solving step $O(rN)$ where N is the size of the system matrix and r is the maximum ε-rank encountered when producing the HSS form. As r is bounded by N, and in practice is typically much smaller than N, the computational bottleneck of the overall HSS process lies in the compression step, with a complexity of $O(rN^2)$. While this is still significantly better than the $O(N^3)$ complexity of a solution using classical Gaussian elimination, it is not as fast as some iterative methods which can have a complexity of $O(N^2)$ [31]. Hence, the focus of this research has been modifying the compression step in an attempt to speed up the HSS construction step.

CHAPTER 4: RANDOMIZED SAMPLING

This chapter discusses the approach of Randomized Sampling (RS), pioneered by P. G. Martinsson in [15-17]. This approach aims to reduce the computational complexity of the HSS construction step by analyzing only a small, randomly sampled subset of a matrix instead of the whole matrix. This chapter also introduces a matrix decomposition method known as Interpolative Decomposition (ID) which integrates with Randomized Sampling better than the traditional RRQR decomposition.

A. Interpolative Decomposition

Interpolative Decomposition is a matrix factorization procedure that is very similar to RRQR that was presented in Chapter 3. The goal is to obtain a factorization for a matrix A in the following form

$$A = A(:, J)\, X \tag{4.1}$$

where J is a set of column indices and $A(:, J)$ denotes a collection of the columns of A indexed by J [32].

As one can see, the main difference between the RRQR and ID is the column space basis matrix of the factorization. The matrix that supplies the basis for the column space in RRQR (the "Q" matrix) is a normalized, orthogonalized basis that does not necessarily share any columns in common with the original matrix. On the other hand, in ID, the matrix that forms the basis for the column space for A is extracted directly from the original matrix A. So, while ID loses the property

of orthonormality, it saves on time of performing the additional computations required to produce the T matrix in the RRQR decomposition.

The procedure for performing an ID of an $M \times N$ matrix A is as follows.

1. Find the column of A with the largest Euclidian norm. We call this the jth column, a_j
2. Record the value j in a list of indices J and then swap the ith column, a_i, with a_j
3. Normalize a_i to produce $\hat{a}_i$. Store the norm of a_i as the ith diagonal entry of T: $t_{ii} = \|a_i\|_2$. If $t_{ii} \leq \varepsilon t_{11}$, then the procedure terminates and the ε-rank of A is said to be $r_\varepsilon = (i-1)$
4. Calculate the remaining elements of the ith row of T by projecting a_i onto the columns of A: $t^H = \hat{a}^H[a_{i+1}\, a_{i+2} \ldots a_N]$
5. Update the matrix A as $A = A - \hat{a}_i t_i^H$
6. Repeat steps 1 through 5 until desired accuracy is achieved or matrix columns are exhausted. Up to this point, the process is essentially equivalent to an RRQR.
7. Partition the T matrix into $[T_1\ T_2]$ where T_1 is an $r \times r$ matrix formed by the columns of T indexed by J and T_2 is the leftover columns. Produce the matrix factor X as $X = [I_r\ T_1^{-1}T_2]$ where I_r is the $r \times r$ identity matrix.
8. Extract the r columns of A corresponding to the columns indexed by J to obtain the second factor, $A(:, J)$. The factorization is now $A \approx A(:, J)X$

B. Randomized Sampling

In the case of very large electromagnetics problems, the matrices for which we must compute an RRQR decomposition or ID become very large even though they may only be subsets

of the entire system matrix. More importantly, the size of the matrix is oftentimes much larger than the (approximate) rank-reduced factorization. Therefore, it is desirable to find a method that expedites the matrix decomposition step by using prior knowledge of the ε-rank deficiency.

Consider the following naïve attempt at solving this problem: we could simply extract the first r columns of the matrix A and perform a decomposition on those columns to obtain an approximate basis for the column space. Better yet, we could take the r columns with the largest norms and use those columns for the decomposition. Unfortunately, the problem with the above approach is that the columns are not guaranteed to be linearly independent.

The above reasoning gives rise to the RS algorithm. Instead of using a subset of the columns of A as the input to the matrix decomposition, we use a series of sampling vectors, ω_i whose entries are independently drawn from a Gaussian distribution. Each matrix-vector product $A\omega_i$ will yield a vector that is a randomly weighted linear combination of the columns of A. By successively sampling the matrix in this manner, we obtain a collection of vectors that are in the column space of A, but with the added benefit of being much less likely to have linear dependence. Hence, if we use this new collection of vectors as the input for a matrix decomposition, it will produce an approximate basis for the column space.

The RS algorithm is detailed below:

1. Generate a sampling vector ω_i of length N (the number of columns in A) whose entries are randomly chosen from a Gaussian distribution. Make r such vectors and concatenate them in the matrix Ω.
2. Form the matrix product $A_S = A\Omega$ (the subscript "S" refers to the "sampling" that has been done on the matrix)

3. Perform an interpolative decomposition on A_S^H to obtain the factors J and X such that $A_S^H = A_S^H(:, J)X$ or equivalently $A_S = X^H A_S(J, :) = X^H A(J, :)\Omega$.
4. Hence, the factorization for A is $A = X^H A(J, :)$

Several points should be noted. Firstly, the matrix A is rarely exactly rank r, and as such the span of the r columns of matrix A_S may not match the range of the original matrix A. To diminish this effect, an "oversampling" parameter, p, is chosen (usually about 10) and instead of generating r random samples of the columns, we generate $r + p$ random samples. If we preform ID on the oversampled matrix, we will obtain a factorization where the first r basis vectors that we choose are likely to be the dominant r columns of A.

As this process relies on randomly generated numbers, it has some probability of failure. In [16], it has been shown that the factorization produced by the above algorithm will satisfy

$$\|A - X^H A(J, :)\| \leq \left[1 + 11\sqrt{r + p}\,\sqrt{\min\{M, N\}}\right]\sigma_{r+1} \tag{4.2}$$

where σ_{r+1} is the (r+1)th largest singular value of A with a probability that depends only on the oversampling parameter, p, and does not depend on N or M or any other property of A. In [16], it was shown that the failure of producing the desired factorization in equation (4.2) is

$$6 \cdot p^{-p} \tag{4.3}$$

The superexponential convergence of the failure probability is one of the major advantages to using the RS algorithm: the probabilistic aspect is basically negligible for an appropriate choice of p.

Lastly, the rank, r, is rarely known prior to performing the RS/ID process. Hence, the number of sampling vectors $r + p$ is not known. The strategy that we have employed in the context

of HSS is to do a full (with $\min\{M, N\}$ sampling vectors) RS/ID decomposition during the first block compression in the HSS tree, record the resulting ε-rank, r_1, and use $r_1 + p$ sampling vectors for the remaining compressions.

CHAPTER 5: THE FAST MULTIPOLE METHOD

Regarded as one of the Top 10 Algorithms of the 20th century [33], the Fast Multipole Method (FMM, also known as the Fast Multipole Approximation, FMA) is a technique that allows for the rapid computation of long-range interactions [20]. It does so by calculating interactions between well-separated groups of unknowns instead of calculating every Galerkin impedance from equation (2.10). This is analogous to sending letter through the postal system, where letters are collected at post offices (aggregation), transported in bulk (translation), and then distributed to individual recipients (disaggregation).

The strength of FMM lies in its ability to perform matrix-vector multiplications (MVM) with the system matrix in $O(n^{1.5})$ time, as opposed to the classical $O(n^2)$ time, where n is the dimension of the square matrix. This merit is commonly used in tandem with iterative solving methods where the system matrix must be serially applied to a vector, i.e. Krylov subspace methods [34]. In this work, however, the fast MVM of FMM is used instead to rapidly sample sub blocks of the system matrix for the RS algorithm presented in the previous chapter.

A. FMM Geometrical Preprocessing

FMM seeks to reduce computation time and memory usage by grouping the unknowns, classifying the pairs of groups as "Near-Field" interactions or "Far-Field" interactions, then filling the appropriate section of the matrix with the appropriate method. The first step in the process is to collect the elements (and thus the unknowns) into groups, called "domains." Typically, the

space that contains the unknowns is uniformly gridded into volumetric cells and the collection of elements that fall within one of these cells becomes an FMM domain.

Next, for each pair of domains, the distance between the centers of the cells is calculated, and by comparing the distance to some predetermined threshold (in terms of the wavelength of the excitation, usually about 2 to 5λ) the interaction is either classified as "near-field" (if the distance is smaller than the threshold) or as "far-field" (if the distance is greater than the threshold).

B. Green's Function Expansion

Before proceeding with the matrix filling procedure, we must present the major mathematical tool for the FMM approximation. The approximation arises from the following expansion of the Green's function from equation (2.10) using Gegenbauer's Addition Theorem [35]:

$$\frac{e^{jk|\mathbf{x}-\mathbf{x}'|}}{|\mathbf{x}-\mathbf{x}'|} = \frac{e^{jk|\mathbf{r}+\mathbf{r}_\mathrm{d}|}}{|\mathbf{r}+\mathbf{r}_\mathrm{d}|} = \frac{jk}{4\pi}\int e^{j\mathbf{k}\cdot\mathbf{r}_\mathrm{d}} \sum_{l=0}^{L} j^l(2l+1)h_l^{(1)}(k|\mathbf{r}|)P_l(\hat{\mathbf{k}}\cdot\hat{\mathbf{r}})\, d\hat{k} \tag{5.1}$$

where j is the imaginary unit, $j = \sqrt{-1}$, k is the wavenumber of the incident plane wave excitation, $\hat{\mathbf{k}}$ is the integration variable over all possible directions on a sphere, and the functions $h^{(1)}$and P are spherical Hankel functions of the first kind and Legendre polynomials, respectively. The relationship between the position vectors $\mathbf{x}$ and $\mathbf{x}'$and displacement vectors $\mathbf{r}$ and $\mathbf{r}_\mathrm{d}$ can be seen in figure 5.1.

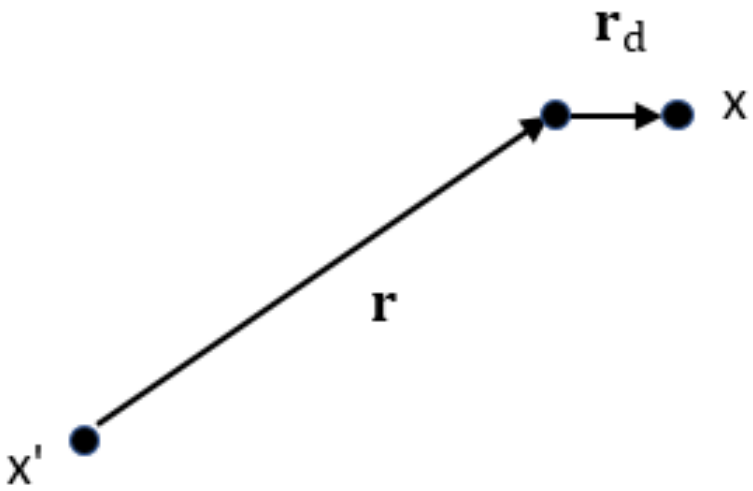


Figure 5.1 A diagram showing the conversion from positions $\mathbf{x}$ and $\mathbf{x}'$ to displacements $\mathbf{r}$ and $\mathbf{r}_d$

Note that, in order for the equality in equation (5.1) to hold true, the series must be summed to infinity, $L \to \infty$. This is obviously not numerically possible, so we instead truncate the sum at some finite L. It has been shown in [36] that to have a relative truncation error less than $10^{-\beta}$, the series should be truncated at:

$$L = kD + 1.8\beta^{\frac{2}{3}}(kD)^{\frac{1}{3}} \tag{5.2}$$

where D is the largest value of $\mathbf{r}_d$, which will be taken to be the size of the discretization cells of the FMM domain grouping step.

Note that the expression for the Gegenbauer's Addition Theorem has no explicit dependence on $\mathbf{x}$ or $\mathbf{x}'$. As such, the main advantage to using this expansion is the ability to decouple the integrals for calculating the Galerkin impedances from equation (2.10) into three parts: the basis integral, the testing integral, and the Green's function integral. This will allow us

to essentially "pre-factorize" sections of the impedance matrix by expressing them as multiplication between vectors, as will be shown in the following section.

C. FMM Near-Field and Far-Field Interactions

Once the elements have been grouped and each pair of domains has been classified as either near-field or far-field, the matrix is ready to be filled. The sections of the system matrix corresponding to near-field interactions are filled densely by calculating the Galerkin impedances using equation (2.10). These parts of the matrix are the same as if they had been computed in the MoM-SIE formulation of Chapter 2.

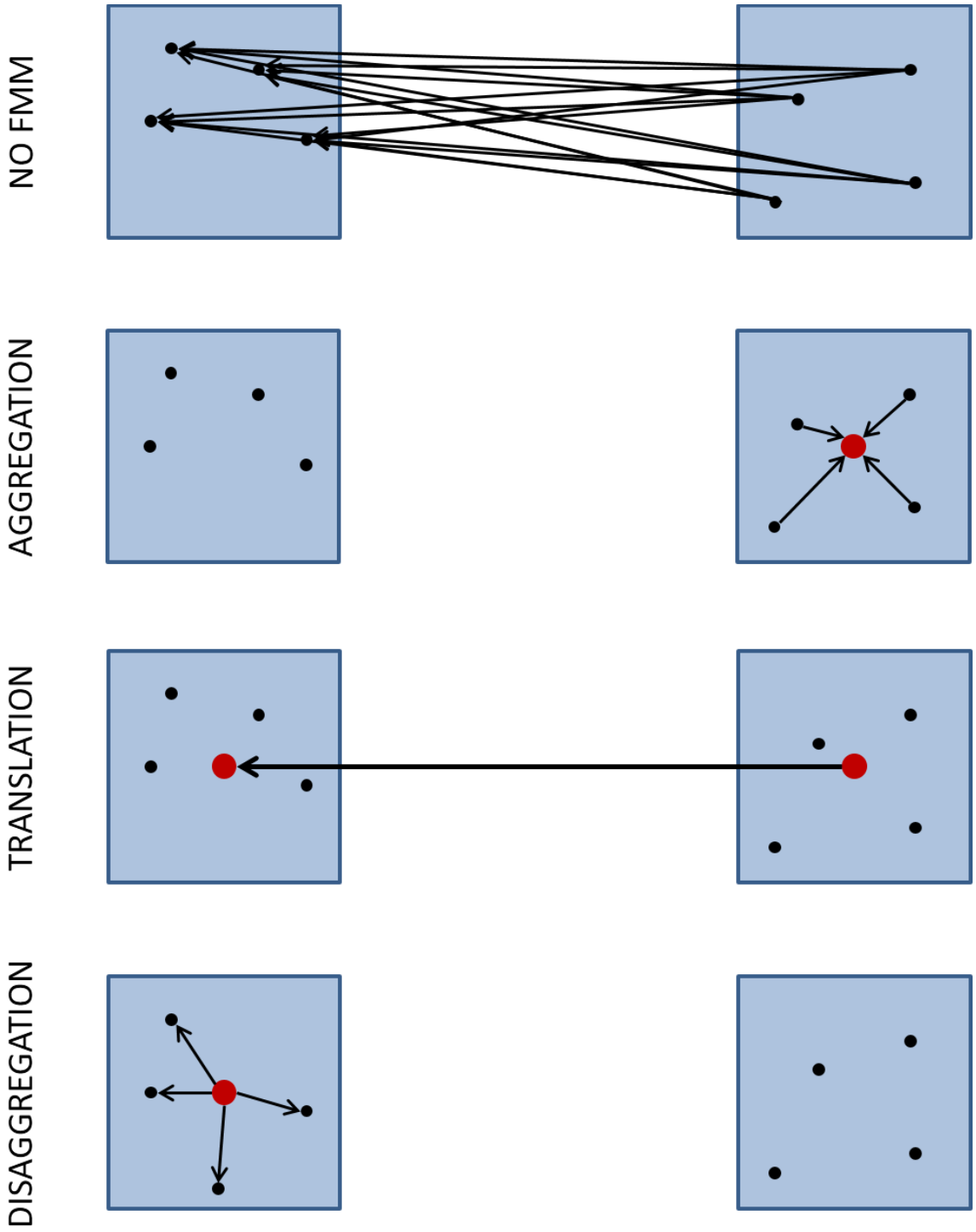


Figure 5.2 FMM near-field matrix filling. Instead of calculating the interaction between every pair of basis functions (black circles), FMM calculates interactions between well-separated groups. The first step of the FMM process, aggregation, involves calculating an equivalent source (red circle) at the center of the source domain (right). Then, in the translation step, the effect of that equivalent source on an equivalent testing point in the testing domain (left). Lastly, disaggregation distributes the equivalent testing onto the basis functions in the testing domain.

On the other hand, the matrix blocks corresponding to far field interactions are not filled explicitly, but instead approximated as a series of outer products between vectors. We first present the physical interpretation of FMM. Suppose we wish to calculate testing on the unknowns in FMM domain m due to sources in domain m'. The first step is to collect the source unknowns to the center of domain m'. This step, referred to as "aggregation," is analogous to collecting letters at a post office in the postal service analogy from above. It is performed by computing only the integral over the basis unknown in equation (2.10), with a phase term from equation (5.1). This can be represented as four row-vectors: three vectors for the three components of the first vector-valued integral, referred to as $\boldsymbol{R}_V$ and one row-vector for the second, divergence integral, $\boldsymbol{R}_D$. The four row-vectors are horizontally concatenated and kept in the matrix $\boldsymbol{R}$ whose elements are given by:

$$R^*_{V,j}(\boldsymbol{k}) = \int_{S_j} \mathbf{f}_j(\mathbf{r}) \cdot e^{-j\mathbf{k}\cdot(\mathbf{r}_{m'}-\mathbf{r})} d^2\mathbf{r} \tag{5.3}$$

$$R^*_{D,j}(\boldsymbol{k}) = \int_{S_j} \Big(\nabla \cdot \mathbf{f}_j(\mathbf{r})\Big) \cdot e^{-j\mathbf{k}\cdot(\mathbf{r}_{m'}-\mathbf{r})} d^2\mathbf{r} \tag{5.4}$$

where $\mathbf{r}_{m'}$ is the center of source FMM domain, m'. These vectors are computed for various values of $\boldsymbol{k}$ corresponding to the spherical integration directions of expression in equation (5.1).

The next step is translation, where the effect of the equivalent source term at the center of domain m' is calculated on an equivalent testing point at the center of domain m. In our post office analogy, this would be similar to the mass transportation of letter, e.g. between cities. Here, we utilize the multipole expansion of the Green's function from section *B*:

$$T_L(\mathbf{k}, \mathbf{r}_{mm'}) = \sum_{l=0}^{L} (-j)^l (2l+1) h_l^{(1)}(k|\mathbf{r}_{mm'}|) P_l(\hat{\mathbf{k}} \cdot \hat{\mathbf{r}}_{mm'}) \tag{5.5}$$

Where $\mathbf{r}_{mm'}$ is the distance between the two FMM domains:

$$\mathbf{r} = \mathbf{r}_m - \mathbf{r}_{m'} \tag{5.6}$$

Lastly, the third step, disaggregation, consists of distributing the field influence on the domain center to the rest of the group. This is done by computing only the testing half of the integrals in equation (2.10). We store these in the vector $\boldsymbol{C}$, which itself is a concatenation of the vector $\boldsymbol{C}_V$ and $\boldsymbol{C}_D$, the vector integral and the divergence integral summands on the Galerkin impedance. Due to reciprocity, this step appears very similar to the aggregation step. The corresponding part of the postal analogy is delivering mail to individual recipients in a different city. The entries of $\boldsymbol{C}_V$ and $\boldsymbol{C}_D$ are given by

$$C_{V,i}(k) = \int_{S_i} \mathbf{f}_i(\mathbf{r}) \cdot e^{-j\mathbf{k}\cdot(\mathbf{r}_m - \mathbf{r})} d^2\mathbf{r} \tag{5.7}$$

$$C_{D,i}(k) = \int_{S_i} \left(\nabla \cdot \mathbf{f}_i(\mathbf{r})\right) \cdot e^{-j\mathbf{k}\cdot(\mathbf{r}_m - \mathbf{r})} d^2\mathbf{r} \tag{5.8}$$

Once the factors from the aggregation, translation, and disaggregation steps have been computed, the submatrix that encompasses the basis functions from FMM domain m' and the testing functions from FMM domain m can be numerically computed as such:

$$Z_{ij} = \sum_{p=1}^{P} \left[\kappa_V C_{V,i}(\boldsymbol{k}_p) T_L(\boldsymbol{k}_p, \mathbf{r}_{mm'}) R_{V,j}(\boldsymbol{k}_p) + \kappa_D C_{D,i}(\boldsymbol{k}_p) T_L(\boldsymbol{k}_p, \mathbf{r}_{mm'}) R_{D,j}(\boldsymbol{k}_p)\right] \tag{5.9}$$

where κ_V and κ_D are the scaling coefficients from equation (2.4):

$$\kappa_V = -j\omega\mu \quad \text{and} \quad \kappa_D = \frac{j}{\omega\varepsilon} \tag{5.10}$$

and P is the total number of Gauss-Legendre integration points over the sphere in order to compute the integral from equation (5.1).

CHAPTER 6: NUMERICAL RESULTS

In this chapter, a number of numerical examples will be presented. We aim to demonstrate the agreement in the scattering patterns with solutions from other sources, such as an "MoM-only" code that does not employ any of the approximations discussed in this work (HSS, RS, FMM) and analytical solutions. Furthermore, we show an empirical measurement of the complexity of the MoM-HSS-RS-FMM method as compared to regular MoM-HSS with respect to the size of the system matrix.

A. Cube array

The first geometry that will be analyzed is a small array of cubes. This is one of the simplest geometries that can be generated using quadrilateral patches. It serves as a basic test case that shows that the method is working properly, and no critical error was made in the implementation.

The model consists of nine identical cubes each with edge length 1λ, where lambda is the free-space wavelength of the excitation. The cubes are arranged in a three-by-three grid as can be seen in figure 6.1 and are excited by a theta-polarized incident plane wave at 300MHz (λ=1 meter). This setup also allows us to test the FMM near/far field matrix filling since parts of the cube array are well-separated.

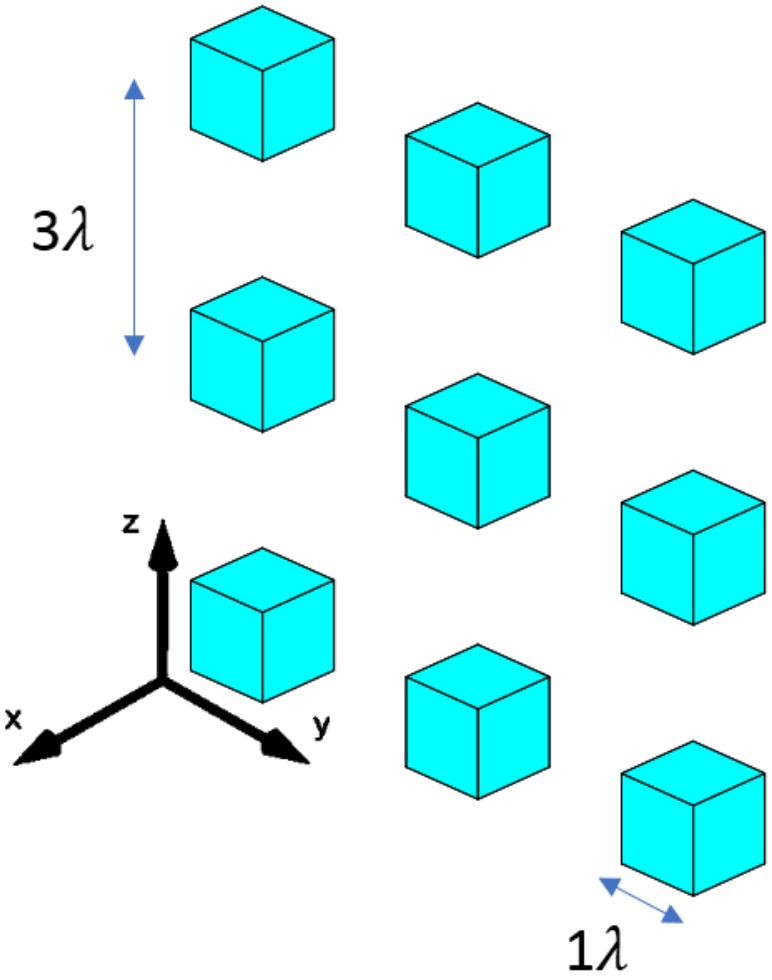


Figure 6.1 An array of cubes. The wavelength λ is 1 meter (f=300MHz)

The bistatic radar cross section (RCS) of the cubes array is calculated from [37] as:

$$\sigma(\theta,\phi) = \lim_{r\to\infty} 4\pi r^2 \left(\frac{|\mathbf{E}^{scat}(r,\theta,\phi)|}{|\mathbf{E}^{inc}|} \right)^2 \tag{6.1}$$

where θ is the polar angle (measured from the positive z-axis) and ϕ is the azimuthal angle. This is a measure of how much power is scattered in each direction. "Bistatic" here means that the

direction of excitation and measurement direction are not necessarily the same. A cut along $\theta = 90^o$ (the xy-plane) of the RCS can be seen in figure 6.2.

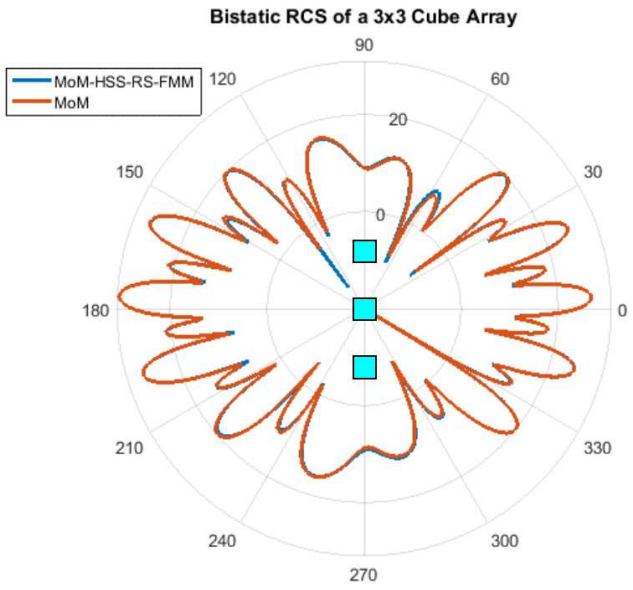
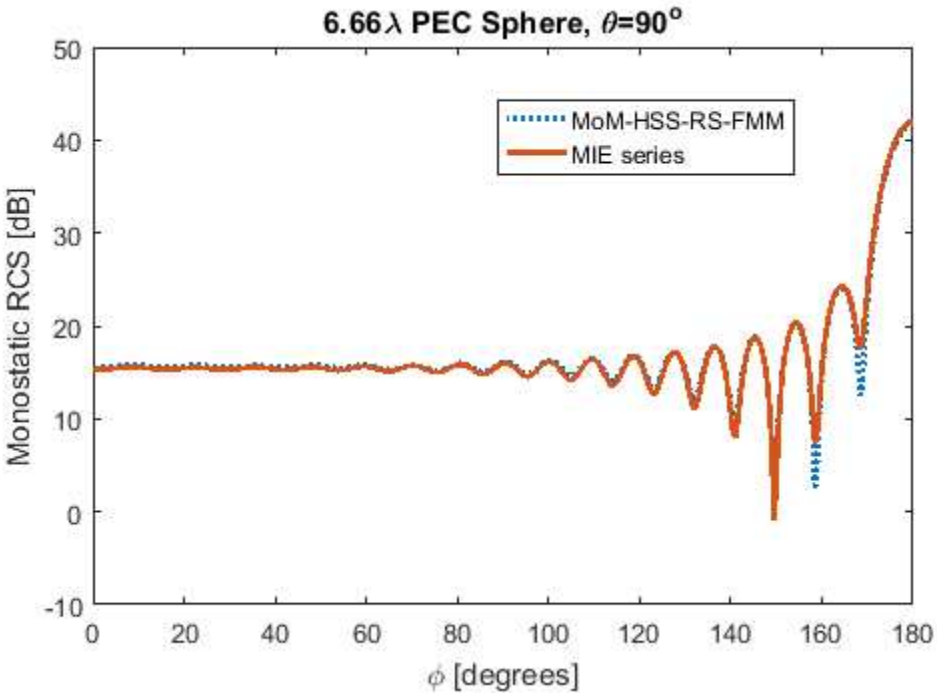


Figure 6.2 Bistatic RCS for the cube array, $\theta = 90^o$.

In this plot, we compare the technique to a standard MoM solver that is running the same model but employing none of the approximations of the MoM-HSS-RS-FMM method. We see good agreement between the two solutions.

B. Sphere

The next example is a metallic sphere with diameter 6.67λ. This example was chosen to demonstrate agreement between the MoM-HSS-RS-FMM method and theory, since analytical solutions exist for spheres via the MIE series [2]. The sphere was modelled with 225 elements and 24,300 unknowns and excited by a θ-polarized plane wave. Bistatic RCS cuts can be seen in figure 6.3a and 6.3b.



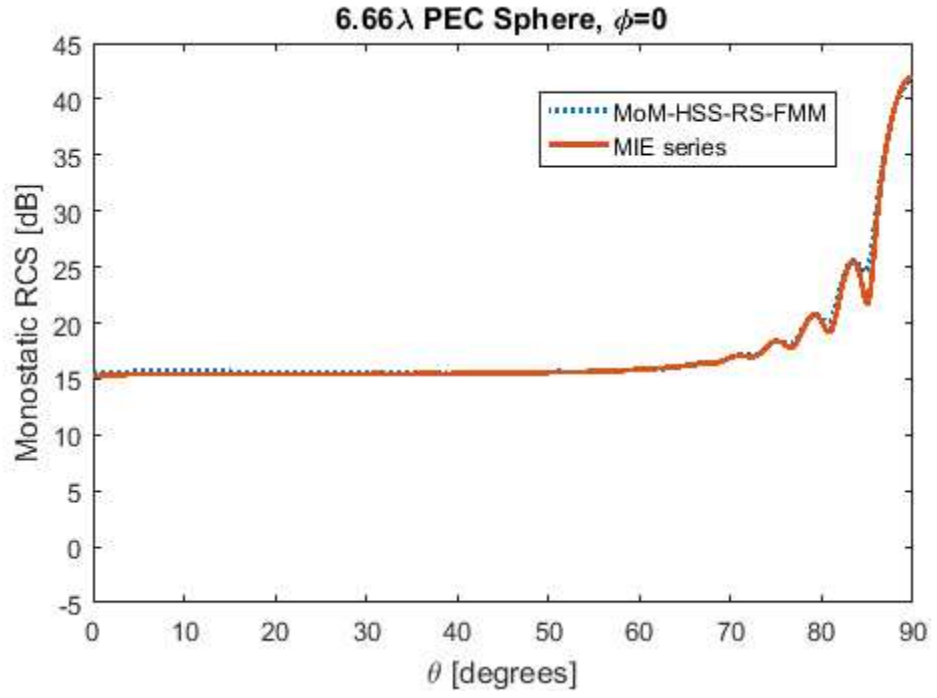


Figure 6.3 (a) Bistatic RCS of a PEC Sphere, $\theta = 90^o$. (b) Bistatic RCS of a PEC Sphere, $\phi = 0$.

We can see relatively good agreement in both of these angular sweeps. However, the effect of the various approximations in MoM-HSS-RS-FMM can be seen in the slight discrepancy between the sharp peaks of the RCS pattern.

C. Complexity

Lastly, the complexity was empirically determined by running a sphere array with different levels of discretization and timing each run with a fixed maximum rank, r. The times were plotted against the size of the problem (in terms of unknowns) on a log-log plot, and the slope, m, of that line will give us the coefficient of the polynomial complexity $O(N^m)$. The sphere array was run using both MoM-HSS (figure 6.4) and MoM-HSS-RS-FMM (figure 6.5) with problem sizes ranging from 7,000 to 93,000.

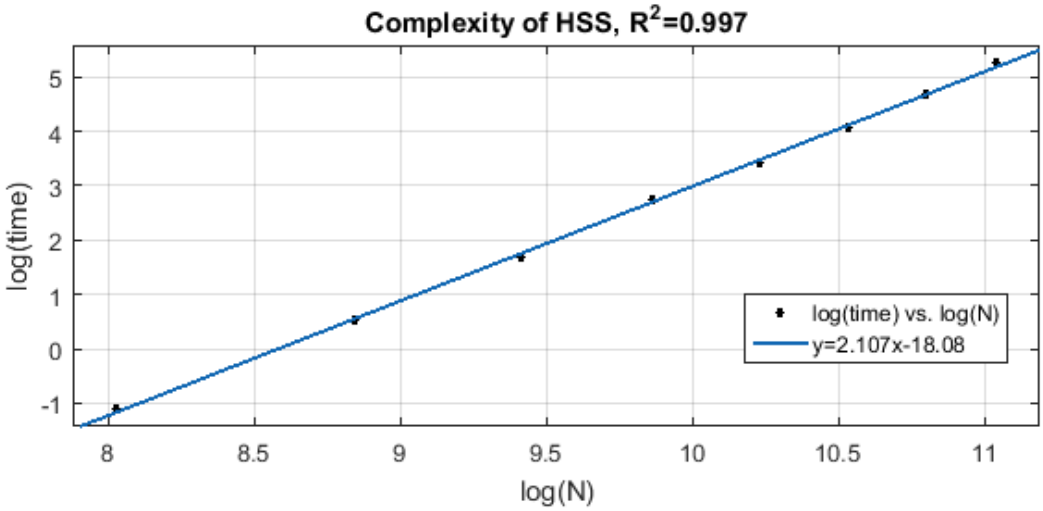


Figure 6.4 complexity of MoM-HSS

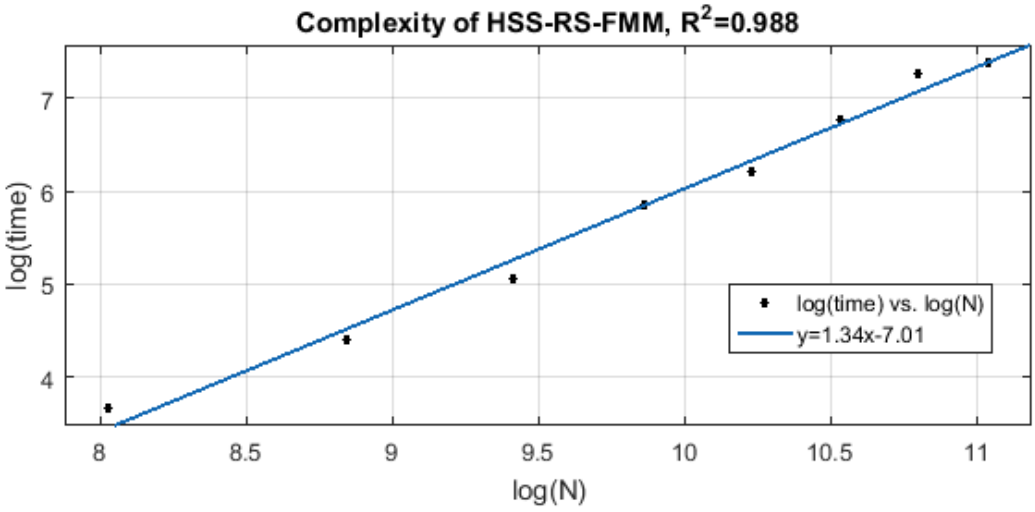


Figure 6.5 Complexity of MoM-HSS-RS-FMM

From these results, we can see that the measured complexity of MoM-HSS is approximately $O(N^{2.107})$ which is very close to the theoretical $O(rN^2)$. The measured complexity of MoM-HSS-RS-FMM was $O(N^{1.34})$ which is actually slightly better than the expected $O(r^2N^{1.5})$.

CHAPTER 7: CONCLUSIONS

A. Summary

The MoM is a powerful method of analyzing the electromagnetic scattering of arbitrary bodies. The main weakness of the MoM, the non-sparsity of its matrix, can be overcome using the matrix structure HSS. Although this speeds up computation, the method has a bottleneck in its construction step. To combat this issue, we have introduced the method of RS. This method utilizes randomized methods to quickly compute the many matrix factorizations that are necessary for the HSS construction step. The speedup of the RS procedure requires a method of performing fast matrix-vector multiplications, for which we have introduced FMM. FMM expresses sections of the matrix implicitly as a series of pre-factorized vector outer products. This work has presented the integration of RS and FMM into pre-existing MoM-HSS software to make MoM-HSS-RS-FMM.

The newly developed method has been compared against both analytical solutions and solutions from pure MoM software. In both cases, there was significant agreement in the solutions. Additionally, the complexity of MoM-HSS-RS-FMM was compared to that of the previous MoM-HSS method. The complexity was found to be significantly better in the MoM-HSS-RS-FMM case.

B. Future Work

There are many directions that future research could take to improve the current method. The most prudent continuation of this research is to change the FMM matrix filling into Multi-

Level FMM (MLFMM) [38]. The MLFMM is very similar to FMM except whereas FMM only has one iteration of grouping the unknowns into domains, MLFMM has multiple levels of groups of varying fineness that are hierarchically defined. This method can increase the speed of the matrix-vector multiplication from FMM's $O(N^{1.5})$ to $O(N \log N)$ and can thus speed up the compression step from the current method's $O(r^2 N^{1.5})$ to $O(r^2 N)$.

Another worthwhile modification is the Null Field Method (NFM), which can improve accuracy of MoM-HSS-RS-FMM [39-40]. This method involves changing the local basis functions of MoM, which are defined over patches, into global basis functions, that are defined over groups of patches. This effectively causes a change of basis in the system matrix in order to introduce blocks in the system matrix that are identically zero. NFM focuses on increasing the accuracy of HSS by minimizing the amount of information that is lost during the compression step.

REFERENCES

[1] E. T. Whittaker, *A history of the theories of aether and electricity, Vol. I – The Classical Theories*. London: Nelson, 1951-1953.

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

[3] J. M. Jin, *Theory and Computation of Electromagnetic Fields*. Hoboken, NJ: Wiley, 2010.

[4] B. M. Notaros, "Higher order frequency-domain computational electromagnetics," *Special Issue on Large and Multiscale Computational Electromagnetics*, *IEEE Trans. Antennas Propagation.*, vol. 56, no. 8, pt. I, pp. 2251–2276, Aug. 2008.

[5] R. F. Harrington, *Field Computation by Moment Methods,* ser. Electromagnetic Waves. Piscataway, NJ:IEEE Press, 1993.

[6] B. D. Popovic and B. M. Kolundzija, "Analysis of metallic antennas and scatterers" in *IEE Electromagnetic Wave Series*. London, U.K.: IEE 1994, no. 38.

[7] B. M. Kolundzija and A. R. Djordjevi, *Electromagnetic Modeling of Composite Metallic and Dielectric Structures*. Norwood, MA: Artech House 2002.

[8] A. F. Peterson, *Mapped Vector Basis Functions for Electromagnetic Integral Equations*. San Rafael, CA: Morgan & Claypool 2006.

[9] W. C. Chew, J. M. Jin, E. Michielssen, and J. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*. Norwood, MA:Artech House 2001.

[10] J. M. Jin *The Finite Element Method in Electromagnetics*, 2nd ed. New York: Wiley 2002.

[11] J. L. Volakis, A. Chatterjee, and L. C. Kempel, *Finite Element Method for Electromagnetics*. New York: IEEE Press 1998.

[12] A. Heldring, J. M. Ruis, J. M. Tamayo, J. Parron, and E. Ubeda, "Fast direct solution of method of moments linear system," *IEEE Trans. Antennas Propag.,* vol. 58, no. 3, pp. 1015-1016, Mar.2010.

[13] A. B. Manić, A. P. Smull, F. H. Rouet, X. S. Li and B. M. Notaroš, "Efficient Scalable Parallel Higher Order Direct MoM-SIE Method With Hierarchically Semiseparable Structures for 3-D Scattering," in *IEEE Transactions on Antennas and Propagation*, vol. 65, no. 5, pp. 2467-2478, May 2017.

[14] S. Wang, X. S. Li, J. Xia, Y. Situ, and M. V. de Hoop, "Efficient scalable algorithms for solving dense linear systems with hierarchically semiseparable structures," *SIAM J. Sci. Comput.*, vol. 35, no. 6, pp. C519‑C544, 2013.

[15] P. G. Martinsson, "A fast randomized algorithm for computing a hierarchically semiseparable representation of a matrix" *SIAM J. Matrix Anal. Appl.* vol. 32, No. 4, pp. 1251–1274, 2011.

[16] N. Halko, P.G. Martinsson, and J.A. Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. 2009. Technical Report 2009-05.

[17] Per-Gunnar Martinsson, "Compressing Rank-Structured Matrices via Randomized Sampling," *SIAM Journal on Scientific Computing* Vol 38, No. 4, 2016.

[18] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *J. Comp. Physics*, vol. 73, pp. 325-348, Dec. 1987.

[19] V. Rokhlin, “Rapid Solution of Integral Equations of Scattering Theory in Two Dimensions," *J. Comput. Phys*, vol. 86, no. 2, pp. 414-439, February 1990.

[20] R. Coifman, V. Rokhlin, and S. Wandzura, “The Fast Multipole Method for the Wave Equation: A Pedestrian Prescription," *IEEE Antennas Propagat. Mag*., vol. 35, no. 3, pp. 7-12, June 1993.

[21] C.C. Lu and W.C. Chew, “A Fast Algorithm for Solving Hybrid Integral Equation," *IEE Proceedings-H*, vol. 140, no. 6, pp. 455-460, December 1993.

[22] B. Dembart and E. Yip, “A 3D Moment Method Code Based on Fast Multipole," *Digest of the 1994 URSI Radio Science Meeting*, p. 23, Seattle, Washington, June 1994.

[23] J.M. Song and W.C. Chew, “Fast Multipole Method Solution Using Parametric Geometry," *Micro. Opt. Tech. Lett.*, vol. 7, no. 16, pp. 760-765, November 1994.

[24] P.-G. Martinsson and V. Rokhlin, "A fast direct solver for boundary integral equations in two dimensions," *J. Comput. Phys.*, vol. 205, pp. 1-23, 2005.

[25] E. Corona, P.-G. Martinsson, and D. Zorin, "An O(N) direct solver for integral equations on the plane," *Appl. Comput. Harmon. Anal.*, vol. 38, pp. 284-317, 2015.

[26] M. Djordjevic and B. M. Notaros, “Double higher order method of moments for surface integral equation modeling of metallic and dielectric antennas and scatterers,” *IEEE Trans. Antennas Propag*., vol. 52, no. 8, pp. 2118‑2129, Aug. 2004.

[27] W. Hackbusch, ”A sparse matrix arithmetic based on H-matrices. Part I: Introduction to H-matrices,” *Computing*, vol. 62, no. 2, pp. 89‑108, 1999.

[28] W. Hackbusch and B. N. Khoromskij, "A sparse H-matrix arithmetic. Part II: Application to multi-dimensional problems," *Computing*, vol. 64, no. 1, pp. 21–47, Jan. 2000.

[29] W. Hackbusch, L. Grasedyck, and S. Börm, "An introduction to hierarchical matrices," *Math. Bohem.*, vol. 127, no. 2, pp. 229–241, 2002.

[30] A. B. Manic, F.-H. Rouet, X. S. Li, and B. M. Notaroš, "Efficient EM scattering analysis based on MoM, HSS direct solver, and RRQR decomposition," in *Proc. IEEE Int. Symp. Antennas Propag.*, Vancouver, BC, Canada, Jul. 2015, pp. 1660–1661.

[31] S. Wang, X. S. Li, J. Xia, Y. Situ, and M. V. de Hoop, "Efficient scalable algorithms for solving dense linear systems with hierarchically semiseparable structures," *SIAM J. Sci. Comput.*, vol. 35, no. 6, pp. C519–C544, 2013.

[32] H. Cheng, Z. Gimbutas, P. Martinsson, and V. Rokhlin, "On the compression of low rank matrices," *SIAM Journal of Sci. Comput.*, Vol. 26, pp. 1389-1404, 2005.

[33] B. Cipra, "The Best of the 20th Century: Editors Name Top 10 Algorithms," *SIAM News*, vol. 33, no. 4, p. 1, 2000.

[34] Y. Saad and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 3, pp. 856–869, Jul. 1986.

[35] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. New York: Dover Publications, 1965.

[36] J. Song and W. C. Chew, "Error Analysis for the Truncation of Multipole Expansion of Vector Green's Functions," *IEEE Microw. Wireless Compon. Lett.*, vol. 11, no. 7, pp. 311–313, Jul. 2001.

[37] C. A. Balanis, *Advanced Engineering Electromagnetics*. New York: Wiley, 1989, pp. 894–896.

[38] M. Zhou, O. Borries, and E. Jørgensen, "Design and optimization of a single-layer planar transmit-receive contoured beam reflectarray with enhanced performance," *IEEE Trans. Antennas Propag.*, 2014.

[39] T. N. Killian, S. M. Rao, and M. E. Baginski, "Electromagnetic scattering from electrically large arbitrarily-shaped conductors using the method of moments and a new null-field generation technique," *IEEE Trans. Antennas Propag.*, vol. 59, pp. 537–545, Feb. 2011.

[40] S. M. Rao and M. S. Kluskens, "A New Power Series Solution Approach to Solving Electrically Large Complex Electromagnetic Scattering Problems," *Appl. Comput. Electromagnetics Society Journal*, Vol. 31, No. 9, pp 1009-1019, Feb. 2016.

LIST OF ABBREVIATIONS

BEM: Boundary Element Method

CEM: Computational Electromagnetics

DHO: Double Higher Order

FEM: Finite Element Method

FMA/FMM: Fast Multipole Approximation/Method

HSS: Hierarchical Semi-Separable Structures

ID: Interpolative Decomposition

IE: Integral Equation

MoM: Method of Moments

PDE: Partial Differential Equation

PEC: Perfect Electric Conductor

RCS: Radar Cross-Section

RS: Randomized Sampling

SIE: Surface Integral Equations