RIGOROUS ANALYSIS OF COMPOSITE FINITE ARRAY STRUCTURES

by

Rickie William Kindt

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2004

Doctoral Committee:

Professor John L. Volakis, Chair
Professor Gabriel Rebeiz
Professor Andrew E. Yagle
Research Scientist Valdis V. Liepa

ABSTRACT

RIGOROUS ANALYSIS OF COMPOSITE FINITE ARRAY STRUCTURES

by

Rickie William Kindt

Chair: John L. Volakis

A particularly challenging problem for computational electromagnetics is the rigorous numerical analysis of electrically large and highly coupled systems. The main challenge lies in the fact that excessive amounts of time and storage are required for accurate analysis of structures in this size class (on the order of 100λ or more), particularly when a high level of detail exists. A popular topic in this research area is the large finite (antenna) array. Large finite arrays require extensive amounts of computational resources to model, making the task difficult. Thus, large arrays are typically analyzed using low-cost approximate methods. These methods can be limited in accuracy, depending upon the size of the finite array and how edge effects are treated. Further, approximate models are typically not adequate for the task of measuring isolation and coupling between multiple array systems, or even arrays in the environment of other structures. Hence, existing methods are typically either too computationally expensive or not rigorous enough to address large finite arrays. The work in this thesis is designed to fill the gap between these two bodies of research, developing a general analysis approach for array-type structures that overcomes the bottlenecks of the rigorous and expensive exact methods, without resorting to the approximations imposed by high-frequency methods.

This thesis begins by developing an existing rigorous method for arbitrary three-dimensional structure analysis (the finite element-boundary integral method) into a decomposition method for finite arrays. The proposed technique combines a domain decomposition approach on near-zone interactions with a far-zone decomposition (fast multipole method) on distant interactions, reducing matrix storage requirements to a small and fixed amount for any sized array problem in the same class. This analysis approach is then extended to accomplish simultaneous solution of multiple and highly coupled systems of both the array and non-array type, as might be encountered in a realistic platform-supported antenna structure. In this thesis, all the underlying methods necessary for this combined task are developed in detail and shown to provide excellent results with minimal computational resources.

DEDICATION

Dedicated to the teachers, family, and friends who challenged and inspired me.

ACKNOWLEDGEMENTS

I would like to thank my advisor, John L. Volakis for his guidance during my graduate studies. I also want to thank my committee members, Professors Gabriel Rebeiz, Andrew E. Yagle, and Valdis V. Liepa, for what they have taught me, and thinking that my work was worth their time and consideration.

At the Naval Research Laboratory, I would like to thank Dr. Mark Kragalott and W. Mark Dorsey for providing antenna designs and measurement data for this thesis, and also Dr. William P. Pala Jr. for supporting my research.

At the University of Michigan, I would like to thank all of the past and present Volakis group members, particularly Dr. Kubilay Sertel, Dr. Mike Carr, and Dimitris Psychoudakis for their companionship over the years. Thanks also to my colleague Bernhard Schoenlinner for providing antenna designs to test my code with.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

A particularly interesting and challenging problem in the field of computational electromagnetics (CEM) is the rigorous frequency domain analysis of electrically large and highly coupled systems. The challenge lies in the fact that excessive amounts of time and storage are required for accurate analysis of structures in this size class (on the order of 100λ or more), especially when a high level of detail exists. A popular problem in this research area is the large finite (antenna) array. Like any electrically large structure, large finite arrays require extensive amounts of computational resources to model, making the task difficult. Composite array structures, potentially consisting of multiple arrays and other nearby structures, are particularly difficult to analyze due to the combination of strong intra-array coupling, complex materials, and the high tessellation densities required for detailed antenna and circuit structures. One example in this target class might be the radar at the front end of an aircraft, designed to perform in the presence of a covering radome and absorbing materials. A similar example might be a highly detailed array under a non-commensurate frequency selective surface (FSS), or perhaps even an antenna array that is conformal to an airplane fuselage. On the largest scale, one might want to consider the case of a multi-functional communication and RADAR tower on a ship, featuring multiple large antenna arrays cluttered with nearby

scattering structures such as cannon turrets or parked aircraft on the ship deck. Accurate coupling models in this class of problems are still lacking, where sensitive communication systems require as much as 90dB isolation from nearby radiating structures to function properly.

A great deal of research has gone into rigorous analysis methods for accurate modeling of three-dimensional structures at microwave frequencies. However, while powerful modeling tools already existed by the late 1990s, for very large structures these tools lack practicality. This lack prompted the introduction of 'fast methods' to specifically address the challenges of larger structures. A good example of a 'fast method' is the fast multipole method (FMM) [1]. The FMM is able to reduce the cost of conventional integral equation methods, typically having storage requirements of $O(N^2)$ for a matrix system of size N, down to a more manageable $O(N^{1.5})$ or better. Even so, the FMM lacks the necessary refinement to deal with very large structures such as complex finite arrays, dozens of cubic wavelengths or more in scale. In this case, even $O(N^{1.5})$ storage requirements are more than many systems can handle. Since even the popular fast methods are too computationally expensive for exact modeling of large-scale structures, approximate methods are typically used instead.

On the high frequency end of the computational spectrum, asymptotic methods have been shown to achieve great success in the analysis of infinite as well as large finite array structures. Array structures have been extensively explored in the literature, using high frequency methods and even full-wave methods, typically based on an infinitely periodic cell with some emphasis on edge treatment [2-6]. Another more recent approach to large array analysis combines a truncated discrete Fourier transform (TDFT) with the

method of moments (MoM), for an asymptotic high frequency analysis of large arrays with reduced unknown counts [7]. Approximate methods have been shown to generate satisfactory results for computations such as far-zone patterns. However, for isolation/coupling-type problems, a prime example being two large finite arrays in close proximity, separated by structural features supporting radiation as well as possible surface wave interference, the accuracy of these asymptotic methods may not be enough. In particular, edge treatment for arrays will fail in the presence of nearby and highly coupled structures where the assumption is the existence of only the finite array environment. Hence, asymptotic models alone may not be adequate for the task of measuring isolation or coupling between multiple array systems, a problem addressed this thesis.

The isolation modeling of multiple electrically large structures can potentially be carried out as a piecemeal process, combining a physical optics (PO) approach with exact methods. In this case, the most appropriate analysis tools are applied to the individual systems independently, and then linked in a decoupled hybrid fashion. Though adequate for weakly coupled systems, such approximations typically cannot provide the accuracy necessary for realistic isolation studies of highly coupled systems. The decoupled domain decomposition approach will fail for proximal systems with a high degree of coupling, particularly when current flows directly between the systems, such as through a surface-wave mode of coupling. Further, setting the accuracy issue aside, when systems are highly coupled, the decoupled iterative system solution required for this type of analysis is not likely to converge. This type of piecemeal analysis is often quite useful, but can only be applied correctly when the coupling is weak enough. Clearly, for

simultaneous solution of large and highly coupled systems a rigorous analysis tool is needed.

From the perspective of this thesis, existing analysis methods for array-to-array coupling are either too computationally expensive (rigorous/exact), or not rigorous enough (asymptotic). In this thesis, an alternative hybrid approach is presented, capable of rigorous and simultaneous solution of large-scale coupling problems. The proposed approach exploits existing capabilities in computational electromagnetic (CEM) research, implementing the refined accuracy of rigorous frequency domain methods to address large-scale array-type problems with an analysis cost approaching that of high frequency methods. The content of this thesis is intended to fill the gap between conventional exact methods and asymptotic methods. Specifically, the methods developed here overcome the bottlenecks of the rigorous and expensive exact analysis approaches, without resorting to the approximations imposed by high-frequency techniques (assuming field or current distributions) in order to achieve accurate analysis of large, highly coupled systems.

To address the need for a rigorous formulation for analyzing three-dimensional structures of arbitrary shape (possibly consisting of metallic as well as inhomogeneous or anisotropic materials), a suitable implementation of the hybrid finite element-boundary integral (FE-BI) method is developed in the first part of the thesis. Hybrid FE-BI methods have received much attention in the literature for the rigorous and effective analysis of three-dimensional, inhomogeneous structures [3, 5, 6, 8-12], and are considered to be well established, reliable CEM tools. Though the emphasis is on rigorous structural analysis, using as few discretized elements as possible to exactly

model a structure is vitally important for large structure analysis, as large unknown counts have a cumulative effect on overall storage. The FE-BI method used here employs curvilinear hexahedral basis elements, enabling rigorous analysis of arbitrary, curved-surface structures with fewer unknowns compared with a flat-facet approach to geometric representation [13, 14].

The remainder of the thesis is structured as follows: In Chapter 2, the underlying FE-BI formulation is developed and verified for a representative set of problems. Smaller example structures are considered in this chapter, as the exact treatment of the domain boundary with integral equations comes at increasingly large storage cost as the electrical size of the structure increases, making conventional FE-BI alone an inadequate approach for the analysis of electrically large structures. The primary bottleneck of the conventional FE-BI approach lies in the roughly $O(N^2) = O(n^2 m^2)$ (n=array elements, m=element unknowns, N=nm) storage requirements for the system matrix.

An excellent way to overcome the storage issue is via the fast multipole method (FMM) [1]. FMM analysis achieves significant memory reductions by interacting distant portions of the geometrical structure (where rigor can be relaxed) with approximations of the free-space Green's function. This can be done in such a way that the results are exactly the same (numerically) as the conventional FE-BI approach to the same problem, but with reduced storage and computational cost. Typically, a controlled degree of approximation is allowed in the solution for the sake of expediency. For distant interactions, this approximation is allowable in the context of a numerical process that is iterative over solution error, and hence already approximate in nature. In Chapter 3 of this thesis, the FMM approach to CEM analysis is developed and shown to allow larger

array modeling as compared to conventional FE-BI alone. Though effective, it is demonstrated that FMM does not provide a significant enough improvement over conventional FE-BI for convenient analysis of large array problems. The main shortcoming is that FMM does not exploit existing redundancies found in finite array structures – a critical exploit that must be adopted for effective large array analysis.

Conceivably, it is possible to exploit structural redundancies in the near-zone environment of finite arrays for computational and storage savings. Using a near-zone domain decomposition approach to specifically exploit the translational symmetry of finite array structures, the array decomposition method (ADM) is developed for FE-BI in Chapter 4. ADM is designed to take advantage of redundant coupling paths in the finite array environment for storage and solution-time savings without compromising accuracy – i.e. the integrity of the rigorous, first-principle integral equations is preserved. ADM achieves $O(nm^2)$ storage due to the block-Toeplitz nature of the resulting system of equations, with $O(n\log(n)m^2)$ solution complexity due to explicit fast Fourier transform (FFT) solution acceleration. Further, the method allows a special preconditioning approach for arrays that results in a relatively small number of iterations for solution convergence. In the development of ADM, the speed and accuracy of the method is examined for several array-type problems.

To enhance the capability and scope of problems addressable via ADM, a generalized multi-dimensional solution approach is developed in Chapter 5 for partitioning structures more complex than simple linear and planar arrays. By defining dimensions as vectors, using spacing and direction (translation) parameters, it is possible to decompose array interactions across multiple dimensions via a Toeplitz property. As a

practical example, it is demonstrated how multi-dimensional analysis can transform a coupling study of two identical arrays into a single multi-dimensional array analysis problem with significantly better storage savings.

While ADM is a fast and elegant solution method for finite array structures, the $O(nm^2)$ storage requirements still place an upper limit on the size of the array that can be analyzed. That is, the storage requirements of ADM increase linearly as the element count n increases. Ideally, one would prefer a finite array method to have storage requirements proportional to the storage costs required of a single element analysis. To achieve this, a near-zone decomposition (ADM) is combined with a far-zone decomposition approach (FMM) to form the hybrid array decomposition-fast multipole method (AD-FMM) in Chapter 6. This dual decomposition approach results in roughly $O(m^2)$ near-zone matrix storage for any sized array of the same element. AD-FMM maintains the excellent convergence properties associated with the physically based block-diagonal matrix system preconditioning of the standalone ADM algorithm. Further, far-zone coupling interactions, comprising the majority of large intra-array interactions, are explicitly accelerated via the FFT. The most unique feature of this distinguished analysis method for finite arrays is that it does not have a matrix storage bottleneck – a limiting feature of all other rigorous analysis methods for finite structures. Indeed, the bottleneck for AD-FMM is the vector storage of the field and excitation coefficients, a cost shared by all exact methods though never before a storage bottleneck. It will be demonstrated that much larger problems can be solved using the AD-FMM approach as compared to the methods in the previous chapters, even on desktop computers.

As a potential obstacle, if m is large (the array element size), ADM can still be quite costly. Expanding upon the benefits introduced by multi-dimensional analysis, the array elements themselves can also be decomposed into smaller divisions, allowing the $O(m^2)$ near-zone matrix storage requirements to be further reduced. This has little to no effect for elements smaller in scale than a wavelength, but is especially important for large array elements (large m), particularly for elements that are several wavelengths long. This decomposition reduces the total near-zone storage requirements via AD-FMM to a cost proportional to λ^2 for any sized array and array element. With this critical development, AD-FMM is used to simulate practical finite arrays with validations of finite array measurements.

For future work, a final challenge is to bring the aforementioned methods together in a way that multiple systems can interact and be solved simultaneously, with maximum exploitation of redundant interactions. In Chapter 8, a multi-dimensional, multi-system analysis method for array structures is presented. In this approach, multiple systems are constructed from a pool of common dimensions. Dimensional pooling allows self-system decomposition of each array-type structure, with cross-system decomposition on elements of different arrays sharing a common dimension. From an organizational perspective, the matrix systems describing the coupling between two systems will have a Toeplitz property that can be exploited for cross-system storage reduction and acceleration via the FFT. The benefits of this type of decomposition will be demonstrated for several composite array configurations.

In the simple and elegant forms of the ADM and AD-FMM in chapters 4-8, the domains of elements are interacted only through coupling via the free-space Green's

function. That is, both the ADM and the AD-FMM model array systems as though there were a crack, or bad electrical contact between elements of the array. Under many circumstances, this choice may match well with structural conditions. However, if it is necessary to model a good connection between adjacent elements, a current junction (field continuity) condition must be imposed. This condition potentially disturbs the Toeplitz property of the matrix system, and must be treated diligently. As a proposed solution, in Chapter 9, bridge systems are introduced as a means of enforcing Toeplitz properties in array systems with connected elements. In combination with a multi-system analysis approach, bridge systems effectively address the issue inter-domain connectivity in an efficient, yet considerably more complicated manner that preserves Toeplitz properties. With this final piece of the puzzle in place, it is possible to achieve accurate results for practical finite array systems requiring good electrical contacts between adjacent array elements.

CHAPTER 2

CONVENTIONAL FINITE ELEMENT-BOUNDARY INTEGRAL METHOD

The first step in developing methods to accurately analyze large composite array structures is to first ensure that the tools are built upon solid fundamentals. In this thesis, a hybrid formulation combining the finite element method (FEM) with a surface or boundary integral equation (SIE or BIE) approach is employed as the underlying formulation. This hybrid combination is typically referred to as the finite element-boundary integral (FE-BI) method, and has been proven reliable many times over in the literature [2, 5, 6, 8, 9, 12-16]. The finite element method can easily handle arbitrary material combinations, including lossy, lossless, inhomogeneous, magnetic, and anisotropic materials, as well as embedded metallic structures [8, 16]. The FEM for three-dimensional analysis is fairly standard in its formulation, with the main difference amongst implementations being the way in which boundary conditions at the volumetric limits are enforced. FEM is most robust when the fields at the volumetric boundary are matched to radiating surface currents modeled via integral equations. In this hybrid approach, the FEM formulation is used to model the fields within the volumetric region, and the integral equations exactly match the FEM fields at the boundary of the volumetric domain with radiating surface currents satisfying the Sommerfeld radiation condition external to the structure. The FE-BI approach allows simultaneous solution for fields

both internal and external to the volumetric region. Hence, FE-BI is ideal for real-world structural analysis, easily modeling arbitrary three-dimensional structures in free-space, as well as multiple structures in the presence of each other. From an application perspective, the FE-BI formulation is general enough to solve scattering, radiation, as well as microwave circuit-type problems.

For the most part, the chosen formulations (FEM and BIE) are independent of the geometric elements used to tesselate the geometry. To achieve accurate geometric representations, it is preferable that geometric features be modeled exactly or as near to exact as possible, suggesting the need for basis elements that conform to curved surfaces. For the purposes of this thesis, curvilinear hexahedral basis elements were chosen. It has been demonstrated on numerous occasions that this choice of basis element results in fewer unknowns to achieve accurate results compared with a flat-faceted approach for curved geometries [11, 17]. In this thesis, it is critically important to achieve accurate modeling with as few unknowns as possible, since ultimately the analysis of very large array structures will only compound the cost of excessive discretization. The details and consequences of the chosen formulations and geometric representations are examined in this chapter.

2.1 Finite Element Method Formulation

The employed FEM formulation is quite standard, and can be easily derived from the differential form of the time-harmonic wave equation. For details on how the FEM

formulation is generated, the reader is referred to several sources on the subject matter [8, 13, 16]. For inhomogeneous media, the common resulting derivation is given by

$$\int_V \left[\left(\nabla \times \bar{E} \right) \bullet \bar{\bar{\mu}}_r^{-1} \bullet \left(\nabla \times \bar{E} \right) - k_0^2 \bar{E} \bullet \bar{\bar{\varepsilon}}_r \bullet \bar{E} \right] dV - jk_0 \eta_0 \int_S \left(\bar{E} \times \bar{H} \right) \bullet \hat{n} dS = -jk_0 \eta_0 \int_{V_i} \bar{E} \bullet \bar{J}^i dV \quad (2.1.1)$$

In this equation, $\bar{\bar{\varepsilon}}_r, \bar{\bar{\mu}}_r$ are the relative constitutive tensors for a general anisotropic media, whereas k_0 and η_0 are the wave number and characteristic impedance of free-space, respectively. The bounding surface *S* encloses the volumetric structure *V,* modeled internally via the FEM (see Figure 2.1). The first integral term on the left represents the relations of the vector electric fields within the structure, denoted by $\bar{E}$, whereas the second term represents the relation of the fields at the boundary domain, and serves to relate $\bar{E}$ and $\bar{H}$ on *S*, as well as impose the radiation condition external to the FEM region. The right hand side integral term is a generalization of impressed current excitations of volumetric extent V_i, within the FEM region, which will be used later to excite electromagnetic structures internally. For a completely enclosed system in which the tangential electric fields are zero on the FEM boundary, (2.1.1) can be used to generate a unique solution internal to the bounding surface *S*. However, typically one is interested in the field solution external to the structure as well. In order to facilitate this, it is most robust to enforce boundary conditions on the FEM using an integral equation formulation for radiating currents ($\bar{J}, \bar{M}$) on the bounding surface.

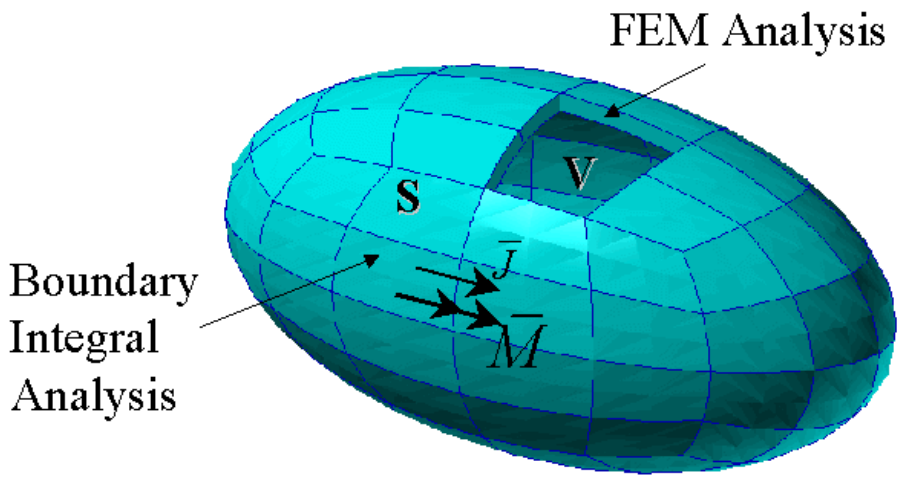


Figure 2.1. Illustration of an arbitrary volumetric structure for FE-BI analysis.

2.2 Boundary Integral Formulation

For the type of problems addressed here, the structure is treated as a closed volume V, terminated in an enclosing surface S, as depicted by the arbitrary structure in Figure 2.1. The FEM is used to enforce electric field continuity within the volume of the structure. In order to generate a unique solution both inside and outside the volumetric region, it is necessary to enforce boundary conditions at the surface of the volume using integral equations. The integral equations chosen to enforce a unique solution can take many forms, depending on the boundary conditions available. For example, one may choose to enforce the tangential component of the incident electric fields $\bar{E}^{inc}$ as boundary conditions, resulting in what is commonly referred to as the electric field integral equation (EFIE) formulation. Alternatively, one may choose to enforce the tangential component of the magnetic fields $\bar{H}^{inc}$ as boundary conditions, resulting in what is commonly referred to as the magnetic field integral equation (MFIE) formulation.

However, at frequencies corresponding to internal resonance frequencies of the enclosing surface shell, each of the aforementioned methods can generate incorrect solutions (even though they may not be physically possible). That is, at certain frequencies, both MFIE and EFIE may be corrupted by spurious modes and not give a unique solution. For a completely arbitrary structure, it is not possible to have a priori knowledge of where the resonant frequencies are, and it is thus preferable to have a generalized means of dealing with all structures without foreknowledge of troublesome frequencies.

The potential for hybrid FEM-integral equation methods to suffer from an internal resonance issue is a common criticism. Many methods have been devised for dealing with the internal resonance issue, a short survey of which is given in [18]. The approach presented here exploits the fact that eigenvalues corresponding to the internal resonance frequencies for the EFIE differ from those of the MFIE. It has been reported that using a linear combination of the MFIE and EFIE will generate a unique (and correct) result [18, 19]. This composite formulation is referred to as the combined field integral equation (CFIE). The potential for the CFIE formulation to generate a correct solution is, of course, dependent on both the MFIE and EFIE formulations supporting the actual 'desired' solution at the given frequency. In other words, if either of the MFIE or EFIE formulations fail to give the correct result (for reasons aside from the internal resonance issue), then the CFIE will not give the correct result either. Thus, the CFIE approach is risky in the sense that a correct solution to a generalized problem relies on two formulations being accurate, verses just one for the EFIE or MFIE alone. Later in this chapter, it will be suggested that the MFIE formulation potentially suffers from a number

of weaknesses that must be adequately addressed in order to have confidence in the solution of a generalized problem involving completely arbitrary structures.

Whereas numerous integral equation formulations exist for dealing with specialized problems such as layered media, in this work the FEM deals with arbitrary internal field conditions and the robust free-space Green's function handles arbitrary coupling conditions external to all volumetric regions. This approach is general enough to correctly deal with the situations where the specialized Green's functions are typically employed, and also deal well with more arbitrary three-dimensional problems. This makes it possible to model a large class of structures without having to specialize the code to achieve the same accurate solutions.

The two main operators used in constructing the boundary integral equations can be most concisely stated as [12]

$$L(\bar{w}(\bar{r}')) = jk_0 \int_S [\bar{w}(\bar{r}') + \frac{1}{k_0^2} \nabla\nabla' \bullet \bar{w}(\bar{r}')] g(\bar{r}, \bar{r}') dS' \tag{2.2.1}$$

and

$$K(\bar{w}(\bar{r}')) = -(1 - \Omega / 4\pi)(\hat{n} \times \bar{w}(\bar{r}')) + \int\!\!\!\!\!\!-_S [\bar{w}(\bar{r}') \times \nabla g(\bar{r}, \bar{r}')] dS' \,. \tag{2.2.2}$$

The integral in (2.2.2) with the dash signifies that a singularity condition exists and must be treated accordingly. The variable $\bar{w}(\bar{r}')$ used in the operator equations has been generalized to represent either electric or magnetic source currents, and will be used as a common vector basis function (defined later) to construct these field quantities. The Ω term represents the solid angle subtended by the testing location, and will be equal to 2π for smooth surfaces. The kernel $g(\bar{r}, \bar{r}')$ used in this FE-BI implementation is the usual scalar free-space Green's function given as

$$g(\bar{r},\bar{r}') = \frac{e^{-jk_0|\bar{r}-\bar{r}'|}}{4\pi|\bar{r}-\bar{r}'|}. \tag{2.2.3}$$

In future chapters, specialized kernels based on finite array analysis will be substituted into the formulation for (2.2.3).

The various integral equation formulations are constructed by inserting the correct choice of electric and magnetic currents into (2.2.1) and (2.2.2). The EFIE formulation is constructed as the combination of the *L*() operator applied to the electric currents $\bar{J}(\bar{r}')$ and the *K*() operator applied to the magnetic currents $\bar{M}(\bar{r}')$. Specifically, for an incident field excitation $\bar{E}^{inc}$, the relation is given by

$$L(\eta_0\bar{J}(\bar{r}')) - K(\bar{M}(\bar{r}')) = \bar{E}^{inc}. \tag{2.2.4}$$

The MFIE is simply constructed as the dual of this equation, and is given by

$$K(\eta_0\bar{J}(\bar{r}')) + L(\bar{M}(\bar{r}')) = \eta_0\bar{H}^{inc}. \tag{2.2.5}$$

Though it is straightforward to enforce the relation of the two formulations using duality, it will be shown that testing should be carried out differently for the EFIE and MFIE to achieve a well-conditioned system of equations.

To generate a system of equations that gives a unique solution, the currents must be tested over the entire surface *S* of the enclosing structure. This is done in such a way as to construct a linear $m \times m$ matrix system, where *m* represents the unknown field coefficients used to construct the field solution. In later chapters, when dealing with array structures, the notation *N*=*nm* will be used, where *m* represents the array element unknowns, *n* is the number of array elements, and thus *N* gives the total number of unknowns. This chapter deals with single structures, or equivalently single element

arrays (n=1), and hence the special case N=m. To generate an $m \times m$ system of equations, (2.2.4) and (2.2.5) must be evaluated over the FEM domain boundary with an appropriate testing function. The chosen testing procedure employs the same basis functions $\overline{w}(\overline{r})$, that are used to construct the surface currents. This choice of testing function is commonly referred to as Galerkin's method [20]. The EFIE and MFIE formulations can be tested with $\overline{w}_t(\overline{r})$ or $\hat{n} \times \overline{w}_t(\overline{r})$, or a combination of both. The best choice is to test the EFIE equation (2.2.4) with $\overline{w}_t(\overline{r})$ and the MFIE equation (2.2.5) with $\hat{n} \times \overline{w}_t(\overline{r})$, a choice that can be shown to generate the most diagonal matrix equations. Applying the testing procedure to (2.2.4) and (2.2.5) gives have the new relations

$$\int_S \overline{w}_t(\overline{r}) \bullet L(\eta_0 \overline{J}(\overline{r}'))dS - \int_S \overline{w}_t(\overline{r}) \bullet K(\overline{M}(\overline{r}'))dS = \int_S \overline{w}_t(\overline{r}) \bullet \overline{E}^{inc} dS \qquad (2.2.6)$$

and

$$\int_S \hat{n} \times \overline{w}_t(\overline{r}) \bullet K(\eta_0 \overline{J}(\overline{r}'))dS + \int_S \hat{n} \times \overline{w}_t(\overline{r}) \bullet L(\overline{M}(\overline{r}'))dS = \eta_0 \int_S \hat{n} \times \overline{w}_t(\overline{r}) \bullet \overline{H}^{inc} dS \,. \quad (2.2.7)$$

Manipulating (2.2.6) and (2.2.7) into a set of equations that can be easily implemented requires expanding the expressions for the EFIE and MFIE by substituting (2.2.4) and (2.2.5) into (2.2.6) and (2.2.7), giving

$$\begin{aligned} &\int_S \overline{w}_t(\overline{r}) \bullet \overline{E}^{inc} dS = \\ &(1 - \Omega / 4\pi) \overline{w}_t(\overline{r}) \bullet \int_S \hat{n} \times \overline{M}(\overline{r}')dS - \int_S \overline{w}_t(\overline{r}) \bullet \oint_S \overline{M}(\overline{r}') \times \nabla g(\overline{r}, \overline{r}')dS'dS + \\ &jk_0 \int_S \overline{w}_t(\overline{r}) \bullet \int_S \eta_0 \overline{J}(\overline{r}') g(\overline{r}, \overline{r}')dS'dS + jk_0 \int_S \overline{w}_t(\overline{r}) \bullet \int_S [\frac{1}{k_0^2} \nabla\nabla' \bullet \eta_0 \overline{J}(\overline{r}')] g(\overline{r}, \overline{r}')dS'dS \end{aligned} \qquad (2.2.8)$$

for the EFIE, and

$$\eta_0 \int_S \hat{n} \times \bar{w}_t(\bar{r}) \bullet \bar{H}^{inc} dS =$$

$$jk_0 \int_S \hat{n} \times \bar{w}_t(\bar{r}) \bullet \int_S \bar{M}(\bar{r}') g(\bar{r}, \bar{r}') dS' dS + jk_0 \int_S \hat{n} \times \bar{w}_t(\bar{r}) \bullet \int_S [\frac{1}{k_0^2} \nabla \nabla' \bullet \bar{M}(\bar{r}')] g(\bar{r}, \bar{r}') dS' dS \quad (2.2.9)$$

$$-(1 - \Omega / 4\pi) \int_S n \times \bar{w}_t(\bar{r}) \bullet [n \times \eta_0 \bar{J}(\bar{r}')] dS + \int_S n \times \bar{w}_t(\bar{r}) \bullet \int\!\!\!\!\!\!- _S \eta_0 \bar{J}(\bar{r}') \times \nabla g(\bar{r}, \bar{r}') dS' dS$$

for the MFIE. These expressions are still too general for direct implementation. Ultimately, it is necessary to apply a series of identities, manipulations, and the theorems in order to transform the EFIE and MFIE expressions into a form that can be readily implemented. After applying the various required transformations, the final forms of the EFIE and MFIE formulations (as used in this thesis) are

Electric Field Integral Equation

$$\int_S \bar{w}_t(\bar{r}) \bullet \bar{E}^{inc} dS =$$

$$(1 - \Omega / 4\pi) \int_S \bar{w}_t(\bar{r}) \bullet [\hat{n} \times \bar{M}(\bar{r}')] dS + \int_S \hat{n} \times \bar{w}_t(\bar{r}) \bullet \int\!\!\!\!\!\!- _S \hat{n} \times \left[\nabla g(\bar{r}, \bar{r}') \times \bar{M}(\bar{r}') \right] dS' dS \quad (2.2.10)$$

$$+ jk_0 \eta_0 \int_S \bar{w}_t(\bar{r}) \bullet \int_S g(\bar{r}, \bar{r}') \bar{J}(\bar{r}') dS' dS - jk_0 \eta_0 \frac{1}{k_0^2} \int_S \nabla \bullet \bar{w}_t(\bar{r}) \bullet \int_S g(\bar{r}, \bar{r}') \nabla' \bullet \bar{J}(\bar{r}') dS' dS$$

and

Magnetic Field Integral Equation

$$\eta_0 \int_S \bar{w}_t(\bar{r}) \bullet [\hat{n} \times \bar{H}^{inc}] dS =$$

$$(1 - \Omega / 4\pi) \eta_0 \int_S \bar{w}_t(\bar{r}) \bullet \bar{J}(\bar{r}') dS + \eta_0 \int_S \bar{w}_t(\bar{r}) \bullet \int\!\!\!\!\!\!- _S \hat{n} \times [\nabla g(\bar{r}, \bar{r}') \times \bar{J}(\bar{r}')] dS' dS +$$

$$jk_0 \int_S \bar{w}_t(\bar{r}) \bullet \hat{n} \times \int_S g(\bar{r}, \bar{r}') \bar{M}(\bar{r}') dS' dS + jk_0 \frac{1}{k_0^2} \int_S \nabla \bullet [\hat{n} \times \bar{w}_t(\bar{r})] \int_S g(\bar{r}, \bar{r}') \nabla'_s \bullet \bar{M}(\bar{r}') dS' dS \quad .(2.2.11)$$

$$+ jk_0 \frac{1}{k_0^2} \oint_C \hat{l} \bullet \bar{w}_t(\bar{r}) \int_S g(\bar{r}, \bar{r}') \nabla'_s \bullet \bar{M}(\bar{r}') dS' dl$$

For details on the necessary transformations used to achieve these expressions, the reader is referred to [8, 13, 14]. Note that these expressions have at worst $1/R^2$ singularities on account of the $\nabla g(\vec{r},\vec{r}')$ term, which is not a difficult issue for distant interactions. For instances where the source and testing locations are closer than $\lambda/10$ however, care must be taken to reduce the order of the singularities to $1/R$ in order that the functions can be accurately integrated numerically. As suggested in [21], by bringing the $\hat{n}\times$ operation under the inner integral for the terms with $\nabla g(\vec{r},\vec{r}')$ in them, it is possible to reduce the $1/R^2$ singularity to a manageable $1/R$ in the limit as the testing location approaches the source location. This procedure works for self-cells, where the source and testing locations are on the same patch, as well as cases where the source and testing surfaces are coplanar and near. However, if the surfaces are not coplanar, such as in a case involving adjacent surfaces at a sharp corner, then more careful singularity treatment may be necessary to achieve accurate solutions. For real-world applications, it is not uncommon to have multiple, separately enclosed volumes with surfaces that come close to each other or even touch, or even geometries with problematic sharp corners. In situations such as this, caution is necessary to ensure that the MFIE formulation can generate valid solutions.

2.3 Geometric Representation

The basis elements chosen to tessellate geometric structures are 27-node curvilinear hexahedral elements of the type depicted in Figure 2.2. The 12 quadratic edges forming the hexahedron are defined with three nodes. For each of the six faces of

the element, quadratic lines are drawn between the center nodes of opposing edges, running through a node at the face center, thereby bisecting each face of the element into curved quadrants. Similarly, quadratic lines are drawn between the center nodes of opposing faces, passing through a node at the center of the element interior and dividing the element into eight curved volumetric regions. These versatile basis elements allow the generation of a volumetric mesh conforming very well to actual structural features (when curves exist). Under most conditions, accurate field representations require that the mesh feature a minimum of $\lambda/10$ to $\lambda/20$ discretized edge lengths (assuming low-order field expansions). Detailed structural features that vary at a rate finer than this typically require discretization with a denser mesh, thereby providing more accurate representation of disturbed fields in these regions.

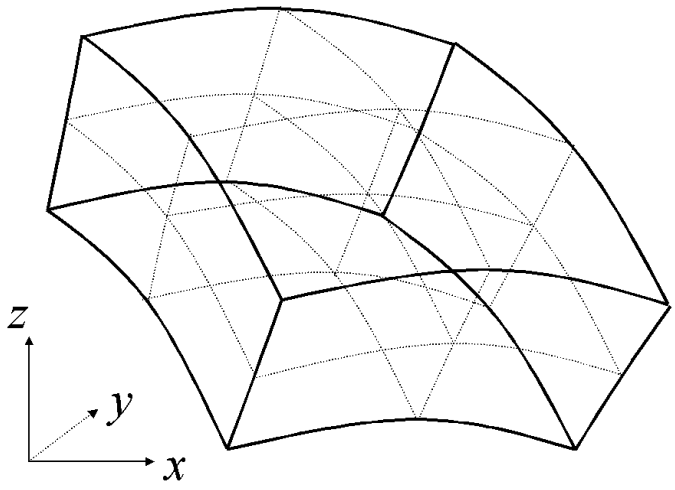


Figure 2.2. Curvilinear hexahedral elements used for geometric tessellation.

For curvilinear elements, it is not practical to carry out volumetric integration in Cartesian space. To facilitate the numerical integration procedure more conveniently,

each conformal element is mapped onto a unit cube in parametric space. Points in Cartesian space given by $\overline{r}(x,y,z)$ are mapped to parametric space via the transformation

$$\overline{r}(u,v,w)=\sum_{k=0}^{2}\sum_{j=0}^{2}\sum_{i=0}^{2}\alpha_i(u)\alpha_j(v)\alpha_k(w)\overline{r}(x,y,z)\,, \tag{2.3.1}$$

where $\alpha_n()$ are the Lagrange interpolation polynomials. Specifically,

$$\alpha_0(u)=2(u-0.5)(u-1),\ \alpha_1(u)=-4(u)(u-1),\ \alpha_2(u)=2(u)(u-0.5)\,. \tag{2.3.2}$$

In this implementation, coordinates in parametric space are expressed in terms of the covariant unitary vectors

$$a_u=\frac{\partial\overline{r}_{uvw}}{\partial u},\ a_v=\frac{\partial\overline{r}_{uvw}}{\partial v},\ a_w=\frac{\partial\overline{r}_{uvw}}{\partial w}, \tag{2.3.3}$$

which form the principle axes of the parametric unit cube, as shown in Figure 2.3. Once the conformal element has been mapped onto the unit cube in parametric space, numerical integration can then be conveniently carried out using Gaussian quadrature.

It is important to consider how the fields are represented within the discretized volume. In this implementation, edge-based vector field expansions are defined using low-order field approximations. In this model, the field contributions associated with each edge of the hexahedral element are constructed from linear shape functions of the form

$$f_0(u)=1-u,\ f_1(u)=u\,. \tag{2.3.4}$$

For a given direction in parametric space, these shape functions range in value from [0 1] over the same parametric range. The vector fields are constructed in parametric space as combinations of these linear shape functions, relative to each of the covariant axes. Explicitly, the fields in each element take the general form

$$\bar{E}(u,v,w)=\frac{1}{\sqrt{\left|G^{3D}\right|}}\begin{bmatrix}\sum_{k=0}^{1}\sum_{j=0}^{1}f_j(v)f_k(w)E^u_{jk}a_u\\ +\sum_{k=0}^{1}\sum_{i=0}^{1}f_i(u)f_k(w)E^v_{ik}a_v\\ +\sum_{j=0}^{1}\sum_{i=0}^{1}f_i(u)f_j(v)E^w_{ij}a_w\end{bmatrix}, \tag{2.3.5}$$

where each E^{uvw}_{ijk} is a field coefficient corresponding to a unique edge of the parametric cube as depicted in Figure 2.3. In (2.3.5), $\left|G^{3D}\right|$ is the determinant of the metric tensor of the transformation (2.3.1), given explicitly as

$$\begin{aligned}\left|G^{3D}\right|=&\left[\left(a_u\bullet a_u\right)\left(a_v\bullet a_v\right)\left(a_w\bullet a_w\right)+\left(a_u\bullet a_v\right)\left(a_v\bullet a_w\right)\left(a_u\bullet a_w\right)+\left(a_u\bullet a_w\right)\left(a_u\bullet a_v\right)\left(a_v\bullet a_w\right)\right]\\ &-\left[\left(a_u\bullet a_u\right)\left(a_v\bullet a_w\right)\left(a_v\bullet a_w\right)+\left(a_v\bullet a_v\right)\left(a_u\bullet a_w\right)\left(a_u\bullet a_w\right)+\left(a_w\bullet a_w\right)\left(a_u\bullet a_v\right)\left(a_u\bullet a_v\right)\right]\end{aligned}. \tag{2.3.6}$$

The basis functions used to construct the edge-based vector fields are constant along the axis associated with the edge, and vary linearly relative to the other two axes in parametric space (constant-linear-linear). Applying the reverse transformation in (2.3.1), the field expansions will then be conformal with the geometry in real space. The field expansions of (2.3.5) are substituted into the FEM equation (2.1.1) to form a solvable system of equations, a necessary step that will be addressed after surface basis functions for the integral equation formulation have been discussed.

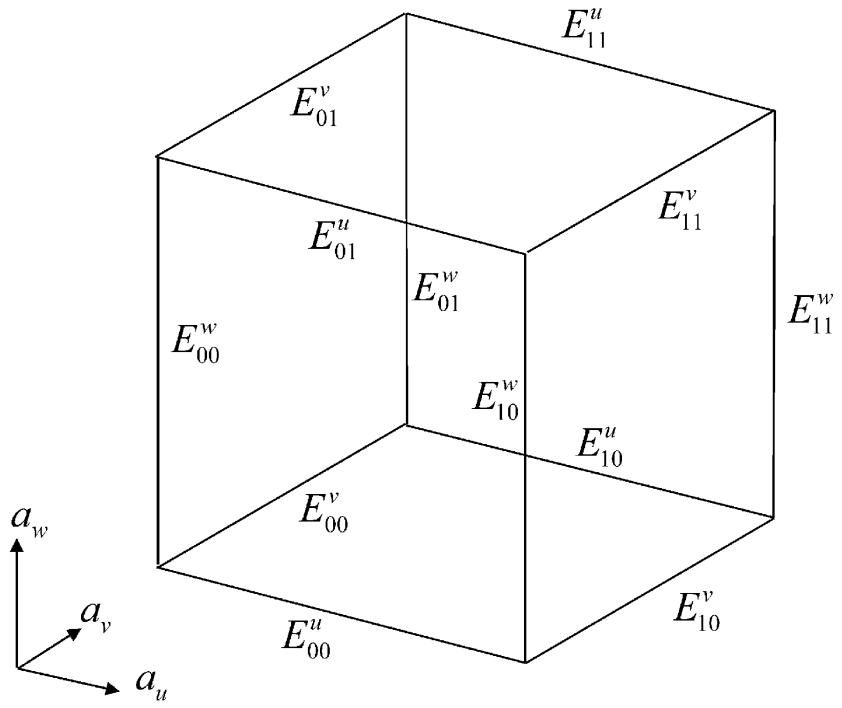


Figure 2.3. Edge-based vector expansion based on E^{uvw}_{ijk}.

In the limiting case, as the fields in the FEM region approach the bounding surface, the vector field expressions simplify to a two-coordinate variation relative to the axes associated with the element face at the surface. Specifically, the surface fields take the form

$$\overline{E}(u,v) = \frac{1}{\sqrt{\left|G^{3D}\right|}}\left[\sum_{i=0}^{1} f_i(v)E^u_i a_u + \sum_{j=0}^{1} f_j(u)E^v_j a_v\right]. \tag{2.3.7}$$

To combine the integral equation formulation with the FEM formulation, it will be necessary that the field representations used in the integral equations match that of the FEM at the bounding surface given by (2.3.7). This is facilitated most easily by using linear-constant vector edge-based field expansions for the integral equations. As a consequence of meshing the volumetric FEM region with 27-node curvilinear hexahedral

elements, the structure surface is automatically discretized with 9-node curvilinear (bi-quadratic) quadrilateral patches that can be used in association with the integral equation formulation. The curvilinear patch is mapped to a unit square in parametric space using the transformation

$$\overline{r}(u,v) = \sum_{j=0}^{2}\sum_{i=0}^{2} \alpha_i(u)\alpha_j(v)\overline{r}(x,y,z)\,, \tag{2.3.8}$$

where the same Lagrange interpolation polynomials given in (2.3.2) are used. Rather than performing integration procedures on the curvilinear surfaces in Cartesian coordinates, surface integration can be carried out in parametric space using 2D Gaussian quadrature.

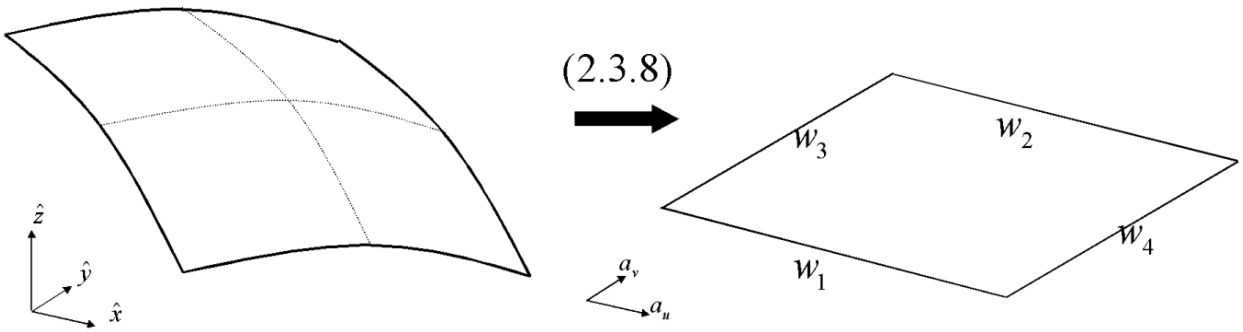


Figure 2.4. Cartesian-parametric surface transformation.

Similar to the three-dimensional model for volume elements, the patch coordinates in 2D parametric space are expressed in terms of the covariant unitary vectors

$$a_u = \frac{\partial \overline{r}_{uv}}{\partial u},\ a_v = \frac{\partial \overline{r}_{uv}}{\partial v}\,. \tag{2.3.9}$$

The transformation of the surface elements is depicted in Figure 2.4. Using parametric coordinates and the same shape functions defined in (2.3.4), it is possible to define a set of common basis functions for constructing vector surface fields. The basis functions for the surfaces are defined explicitly as

$$w_1 = \frac{1-v}{\sqrt{\left|G^{2D}\right|}} a_u,\ w_2 = \frac{v}{\sqrt{\left|G^{2D}\right|}} a_u,\ w_3 = \frac{1-u}{\sqrt{\left|G^{2D}\right|}} a_v,\ w_4 = \frac{u}{\sqrt{\left|G^{2D}\right|}} a_v . \qquad (2.3.10)$$

In these expressions, the determinant $\left|G^{2D}\right|$ is

$$\left|G^{2D}\right| = \left(a_u \bullet a_u\right)\left(a_v \bullet a_v\right) - \left(a_u \bullet a_v\right)\left(a_v \bullet a_u\right). \qquad (2.3.11)$$

A separate basis function w_j is associated with each of the four edges of the patch as shown in Figure 2.4. In parametric space, the field expansion associated with each basis on the surface will vary linearly in one dimension and remain constant in the other (linear-constant). When transformed, these shape functions correspond to conformal rooftop elements in Cartesian space, used to represent the fields and currents on the structure boundary. The electric fields on the boundary surface are expressed as

$$\bar{E}_S(\bar{r}) = \sum_j^{N_s} w_j(\bar{r}) E_j , \qquad (2.3.12)$$

whereas the magnetic fields (constructed from the same basis functions) are given by

$$\bar{H}_S(\bar{r}) = \sum_j^{N_s} w_j(\bar{r}) H_j . \qquad (2.3.13)$$

In these expressions, E_j and H_j are the unknown field coefficients for the electric and magnetic fields, respectively. These surface field expansions match exactly with the definitions for the FEM field expansions at the limit of the bounding surface. The actual number of edges N_S used to construct the field on the patch will vary, depending on the

boundary conditions. For example, if the patch resides on a PEC surface, the electric fields on the surface will be zero, and thus N_S for $\bar{E}_S(\bar{r})$ would be zero. Similarly, if the patch is adjacent to a PEC surface, one or more of the unknown coefficients for the electric fields may be zero. From these expressions, the electric and magnetic currents on the surface patches are found by the usual operations

$$\bar{J}(\bar{r}) = \hat{n} \times \bar{H}_S(\bar{r}) \tag{2.3.14}$$

and

$$\bar{M}(\bar{r}) = -\hat{n} \times \bar{E}_S(\bar{r}) . \tag{2.3.15}$$

The individual current expansions are substituted into equations (2.2.10) and (2.2.11) to form a solvable system of equations. The assembly of this system of equations is discussed in the next section.

2.4 Matrix System Assembly

To solve the hybrid FE-BI formulation developed in the preceding sections, the FEM and BI formulations must be assembled into a linear system of equations. To do so, it is necessary to express the formulation as a matrix equation consisting of matrix operators, linear field coefficients, and vector excitation coefficients. When the field expansions of (2.3.5) are substituted into the FEM equation (2.1.1), the FEM system of equations takes the form

$$\left[\int_V \left[\left(\nabla\times\overline{w}(\overline{r})\right)\bullet\overline{\overline{\mu}}_r^{-1}\bullet\left(\nabla\times\overline{w}(\overline{r})\right)-k_0^2\overline{w}(\overline{r})\bullet\overline{\overline{\varepsilon}}_r\bullet\overline{w}(\overline{r})\right]dV\right]\{E\}$$
$$-\left[jk_0\eta_0\int_S[\left(\overline{w}(\overline{r})\times\overline{w}(\overline{r})\right)\bullet\hat{n}]dS\right]\{H^s\}=-jk_0\eta_0\int_{V_i}\overline{w}(\overline{r})\bullet\overline{J}^i dV \quad . \tag{2.4.1}$$

Here, the more complicated FEM field relations of (2.3.5) have been represented with the more compact FEM basis function notation $\overline{w}(\overline{r})$. Note that the second integral term comes into play only on the surface of the structure where the fields $\overline{H}_S(\overline{r})$ reside. This equation can be assembled into a representative matrix system of the form

$$\begin{bmatrix}[A]^{II} & [A]^{IS} & 0\\ [A]^{SI} & [A]^{SS} & [B]\end{bmatrix}\begin{Bmatrix}E^i\\ E^s\\ H^s\end{Bmatrix}=\begin{Bmatrix}b^i\\ b^s\\ 0\end{Bmatrix}. \tag{2.4.2}$$

In (2.4.2), the $A^{II,IS,SI,SS}$ sub-matrix operators are sparse, with $[A]^{II}$ and $[A]^{SS}$ sparse symmetric, and $[A]^{IS},[A]^{SI}$ transposes of each other. The $[B]$ sub-matrix only comes into play when boundary conditions state that non-zero tangential electric fields exist at the surface of the structure. The excitation to the FEM system $\{b\}$ is implemented as an impressed current source of the general form

$$\{b^i\}=-jk_0\eta_0\int_{V_i}\overline{w}(\overline{r})\bullet\overline{J}^i dV \; . \tag{2.4.3}$$

A form of $\{b^i\}$ more conducive to implementation will be discussed later. The FEM matrix system (2.4.2) is not yet ready for solution, as the appropriate boundary conditions have not been imposed. As it stands, there are more unknowns than equations. In order to generate more equations, the integral equation formulation is used to enforce the boundary conditions over the FEM bounding surface. Combining the integral equation system with the FEM system generates the desired $m\times m$ system of equations.

Following a similar assembly procedure with the integral equations, the expansions for the surface fields given in (2.3.12) and (2.3.13) are first inserted into the EFIE and MFIE equations (2.2.10) and (2.2.11). Once these systems of equations are assembled, it is possible to extract the linear field coefficients and form the matrix equations. Specifically, the EFIE equation takes the form

$$\begin{bmatrix} (1-\Omega/4\pi)\int_S \overline{w}_t(\overline{r})\bullet[\hat{n}\times\overline{w}_s(\overline{r}')]dS + \\ \int_S \hat{n}\times\overline{w}_t(\overline{r})\bullet\oint_S \hat{n}\times\left[\nabla g(\overline{r},\overline{r}')\times\overline{w}_s(\overline{r}')\right]dS'dS \end{bmatrix}\{E^s\} + \begin{bmatrix} jk_0\eta_0\int_S \overline{w}_t(\overline{r})\bullet\int_S g(\overline{r},\overline{r}')\overline{w}_s(\overline{r}')dS'dS - \\ jk_0\eta_0\frac{1}{k_0^2}\int_S \nabla\bullet\overline{w}_t(\overline{r})\bullet\int_S g(\overline{r},\overline{r}')\nabla'\bullet\overline{w}_s(\overline{r}')dS'dS \end{bmatrix}\{H^s\} = \int_S \overline{w}_t(\overline{r})\bullet\overline{E}^{inc}dS \quad . \tag{2.4.4}$$

In this expression, the notation $\overline{w}_s(\overline{r})$ and $\overline{w}_t(\overline{r})$ are used to differentiate between the source and the testing functions, respectively. For matrix assembly, this expression can be more concisely written as

$$[P]^{EFIE}\{E^s\}+[Q]^{EFIE}\{H^s\}=\{b\}^{EFIE} . \tag{2.4.5}$$

The same procedure can be applied to the MFIE equation to form

$$\begin{bmatrix} jk_0\int_S \overline{w}_t(\overline{r})\bullet\hat{n}\times\int_S g(\overline{r},\overline{r}')\overline{w}_s(\overline{r}')dS'dS + \\ jk_0\frac{1}{k_0^2}\int_S \nabla\bullet[\hat{n}\times\overline{w}_t(\overline{r})]\int_S g(\overline{r},\overline{r}')\nabla'_s\bullet\overline{w}_s(\overline{r}')dS'dS \\ +jk_0\frac{1}{k_0^2}\oint_C \hat{l}\bullet\overline{w}_t(\overline{r})\int_S g(\overline{r},\overline{r}')\nabla'_s\bullet\overline{w}_s(\overline{r}')dS'dl \end{bmatrix}\{E^S\} + \begin{bmatrix} (1-\Omega/4\pi)\eta_0\int_S \overline{w}_t(\overline{r})\bullet\overline{w}_s(\overline{r}')dS + \\ \eta_0\int_S \overline{w}_t(\overline{r})\bullet\oint_S \hat{n}\times[\nabla g(\overline{r},\overline{r}')\times\overline{w}_s(\overline{r}')]dS'dS \end{bmatrix}\{H^S\} = \eta_0\int_S \overline{w}_t(\overline{r})\bullet[\hat{n}\times\overline{H}^{inc}]dS \tag{2.4.6}$$

with the more concise notation

$$[P]^{MFIE}\{E^s\}+[Q]^{MFIE}\{H^s\}=\{b\}^{MFIE}. \tag{2.4.7}$$

The [P] and [Q] operators in (2.4.5) and (2.4.7) are dense on account that every source current must be interacted with every testing current to form the $m\times m$ system of equations. As previously stated, the CFIE formulation is formed by taking a linear combination of the EFIE and MFIE formulations. Based on the stated form of (2.2.4) and (2.2.5), this is correctly done with the combination

$$(1-\alpha)MFIE+\frac{\alpha}{\eta_0}EFIE. \tag{2.4.8}$$

In this expression, α is a number in the range [0 1], chosen to give the desired ratio of EFIE to MFIE, and $1/\eta_0$ must be appended to the EFIE system to ensure consistent units between the two systems of equations. In general, the MFIE yields a better-conditioned system of equations than the EFIE, and hence a lower value of alpha typically results in faster system convergence.

Note that the integral equation formulations (2.4.5) and (2.4.7) alone are capable of generating unique solutions external to certain scattering-type targets. However, for maximum robustness, the integral equation system should be combined with the FEM system of equations to form the $m\times m$ matrix equation

$$\begin{bmatrix} A^{II} & A^{IS} & 0 \\ A^{SI} & A^{SS} & B \\ 0 & P & Q \end{bmatrix}\begin{Bmatrix} E^i \\ E^s \\ H^s \end{Bmatrix}=\begin{Bmatrix} b^i \\ b^s \\ b^e \end{Bmatrix}. \tag{2.4.9}$$

In this expression, b^e comes from $\bar{H}^{inc}$. When solved, this combined system of equations is capable of generating simultaneous solutions both internal and external to the volumetric structure. Moreover, when scaled properly, it is possible to use (2.4.9) for

solving problems with simultaneous external and internal source excitation. An example of such a problem would be a microwave circuit operating in the presence of a radiating structure, in which one is interested in evaluating the effect of the radiating source on the circuit performance [22]. The hybrid FE-BI formulation used to assemble (2.4.9) is appropriate for analyzing a wide range of microwave circuits, antenna and radiating structures, as well as scattering problems. More importantly, the formulation is an excellent candidate for the rigorous array-type coupling problems to be discussed later in this thesis.

It is worth noting that in (2.4.9), the [Q] operator appears on the matrix diagonal. This sub-matrix system contains the self-testing terms, where the field testing is performed on the same patch as the source currents. For a well-conditioned system of equations, it is imperative that the self-testing procedure produces large matrix entries, as these values are closest to the matrix diagonal and have the greatest effect on the system condition. In solving these types of problems, it is valuable to have the most well conditioned system possible, as this leads to the fastest solution time. Notice that in both (2.4.4) and (2.4.6), the self-terms of the [Q] operators contain the product $\overline{w}_t(\overline{r})\bullet\overline{w}_s(\overline{r})$, which is highly diagonal, leading to the best-conditioned system of equations. This verifies that the suggested choice of testing functions was most appropriate for the integral equation formulation testing. An improper choice of testing function would have resulted in the product $\hat{n}\times\overline{w}_t(\overline{r})\bullet\overline{w}_s(\overline{r})$, which is least diagonal when $\overline{w}_t(\overline{r})=\overline{w}_s(\overline{r})$, and leads to a poorly conditioned system of equations.

2.5 FEM Loading and Excitation

For scattering problems, system excitation is enforced at the surface of the volumetric region by the integral equations. However, it is alternatively possible to load the FEM region with excitations of the impressed source type. While it is possible to implement impressed sources as distributed surface or volumetric currents, as implied by (2.1.1), this thesis focuses on sources of the simpler lumped load type, as depicted in Figure 2.5.

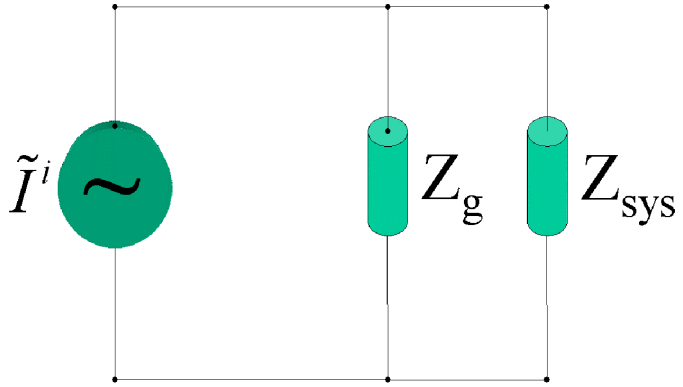


Figure 2.5. Impressed current source used for FEM loading.

Lumped-element type sources have no physical extent, and therefore do not physically disturb the system. These 'probes,' as they are commonly referred to, treat the volumetric problem as a black box Z_{sys}, to which they either deliver power to or drain power from. They are placed in parallel with discretized edges of the geometry, sharing the same voltage that exists across the common geometric edge. Probes are general enough in nature that they can be treated as active or passive ports in the geometry, with an arbitrarily assigned impedance value Z_g. When the lumped elements are connected to

the geometry, they require both an impedance matrix entry in the FEM system, as well as an entry in the system excitation (when active). The impedance matrix entry for an edge of length L has the general form

$$Z_g = jk_0\eta_0 \frac{L^2}{Z_L}, \tag{2.5.1}$$

where Z_L is the desired probe impedance [13]. Care must be taken to ensure the form of the impedance entry has been scaled to match the system of equations. By entering a very large real-valued impedance, the probe will have very little effect on the system. Conversely, adding a probe of very low impedance would have the effect of shorting out the electric field along the probed edge of the structure, hence acting as a shorting pin. The probe has a corresponding entry in the excitation vector of the form

$$b^i = -jk_0\eta_0 \left|I^i\right| e^{j\phi} L, \tag{2.5.2}$$

where $\left|I^i\right|$ is the magnitude of the source excitation, and $e^{j\phi}$ controls the phase. It is important to note that this impressed current will be flowing through the parallel combination of the source impedance and the system impedance at the probe point, as illustrated in Figure 2.5. The addition of the impedance and current excitation to the systems dictates that the unknown electric field solution will be relative to the parallel combination of the source impedance and system impedance at the point of probe entry. In other words, if one wanted to determine the input impedance of the system at the probe location, it is necessary to use a current divider rule to extract the fraction of the current entering the system. The input impedance of the system can be computed as

$$Z_{in} = \frac{E^{in} L}{\left|I^{in}\right| e^{j\phi}}, \tag{2.5.3}$$

where I^{in} is the fraction of current flowing into the FE-BI system. The effective use of these probes will be demonstrated in the results section at the end of this chapter. As a general rule, they work more effectively in combination with electrically small edges, where the approximation of a constant voltage ($E^{in}L$) across the load is a more valid assumption.

2.6 Solution of the System of Equations

The system of equations in (2.4.9) can be solved in a number of ways. In general, the solution to the system is found by inverting the matrix, or

$$\begin{Bmatrix} E^i \\ E^s \\ H^s \end{Bmatrix} = \begin{bmatrix} A^{II} & A^{IS} & 0 \\ A^{SI} & A^{SS} & B \\ 0 & P & Q \end{bmatrix}^{-1} \begin{Bmatrix} b^i \\ b^s \\ b^e \end{Bmatrix}. \tag{2.6.1}$$

However, the inversion of the matrix system comes at a very high storage and computation cost ($O(N^3)$), and is typically avoided. A good survey of solution methods for linear systems of equations can be found in [23]. The best solution approach is system dependent, but the fastest approach is typically an iterative solution procedure ($O(N^2)$). The form of (2.4.9) is non-symmetric, meaning that a simple conjugate gradient solver is not appropriate. However, bi-conjugate gradient-based solvers for non-symmetric systems are appropriate, and have been found to do a reasonably good job solving systems of equations of this type.

In (2.4.9), the FEM sub-matrix is sparse, but not well conditioned. If adequate memory resources are available, it is recommended that the FEM portion of the matrix be solved via a complete LU factorization [23]. With limited storage, larger systems will

require an iterative solution approach. The computations in this thesis were performed using two iterative solvers that were found to give the best results. Under certain conditions, it was found that an iterative solver based on the generalized minimum residual (GMRES) procedure produced the best results. However, for most tasks, it was found that a stabilized version of the bi-conjugate gradient solver (BICGSTAB(L)) converged the fastest [24].

To assist the iterative solver, it is useful to augment the solution procedure with matrix preconditioners, the use of which is described in [23]. For sparse systems of equations, an incomplete LU factorization with zero fill-in (ILU(0)) resulted in reasonable improvement to the solution process with the smallest additional storage cost. However, using a preconditioner that pre-solves the entire FEM system (selective LU factorization), and only requires the solver to iterate over the BI matrix assembly generates faster solution times. Another timesaving preconditioner combines the LU factorization of the FEM and $[Q]$ matrices (block-LU). Though fast, this preconditioner has higher storage cost than the FEM LU factorization alone.

2.7 Examples and Validations

This section presents some basic examples designed to validate the underlying FE-BI formulation. In later chapters, the reliability and validity of the base FE-BI formulation will be used as a reference for validating array-based methods.

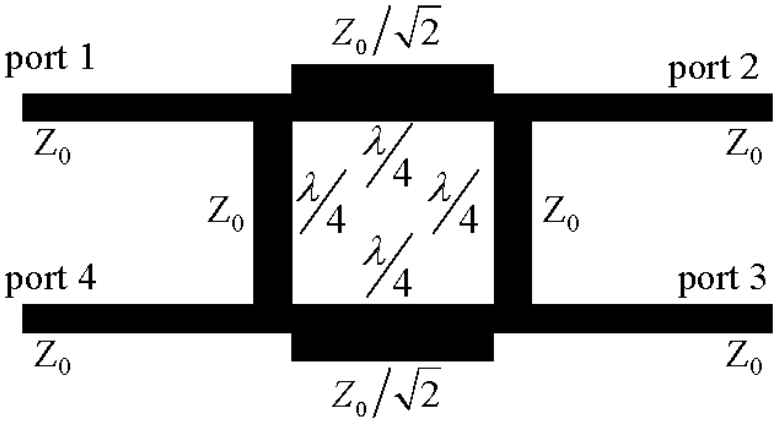


Figure 2.6. Geometry of a microstrip quadrature hybrid circuit.

The first example considered here is a microwave circuit of the type shown in Figure 2.6. This example is designed primarily to demonstrate the feasibility of the chosen formulation for circuit applications involving port-to-port coupling measurements. Further, this example validates the functionality of the FEM formulation for modeling structures using lumped-element sources and loads. The primary mode of operation for this type of circuit is characterized by a signal entering port 1 and splitting two ways, with half the power exiting from each of ports 2 and 3, port 4 isolated. Each arm of the circuit is 90-degrees in electrical length at the center design frequency, and hence the output signals at ports 2 and 3 are 90-degrees different in phase. The S-parameter sweep for this circuit is given in Figure 2.7, comparing the simulated and ideal results. The simulated results match reasonably well with the ideal curves, though definite shifts are noticeable. This is to be expected, as the ideal results assume an infinite ground plane and an ideal design (lossless transmission lines), whereas the FE-BI code has simulated a finite structure with physical bounds. This example suffices to demonstrate the port-to-port coupling capabilities of the FE-BI formulation.

For reference, the structure was modeled using a total of 168 hexahedral elements, with an associated 397 FEM unknowns and 776 BI unknowns. This number of elements and the resulting discretization is appropriate for structural analysis at the highest frequency considered in this example. The required storage for the matrix system was 10MB using COMPLEX(4) data types in FORTRAN 90. The center frequency for this design was 2GHz. The accuracy of this solution was calculated at 1.0% deviation from the exact solution using the criterion

$$err = \| \{b\} - [A]\{x\}^{est} \|_2 / \|\{b\}\|_2 . \tag{2.7.1}$$

In this expression, $\{x\}^{est}$ is the estimated solution in solving the generalized system of equations

$$[A]\{x\} = \{b\}, \tag{2.7.2}$$

which represents an arbitrary FE-BI system of the type (2.4.9). A top view of the simulated structure is shown in Figure 2.8. Notice that the mesh is very regular, with many elements of the same geometric size. A common criticism of using hexahedral elements in conjunction with FEM for CEM applications is that the regular grids associated with these elements can lead to phase error propagation in the solution. This criticism is unfounded, as the hexahedral mesh can easily be distorted to average out phase error from element to element. However, this criticism is warranted when using completely regular meshes of the type shown in Figure 2.8.

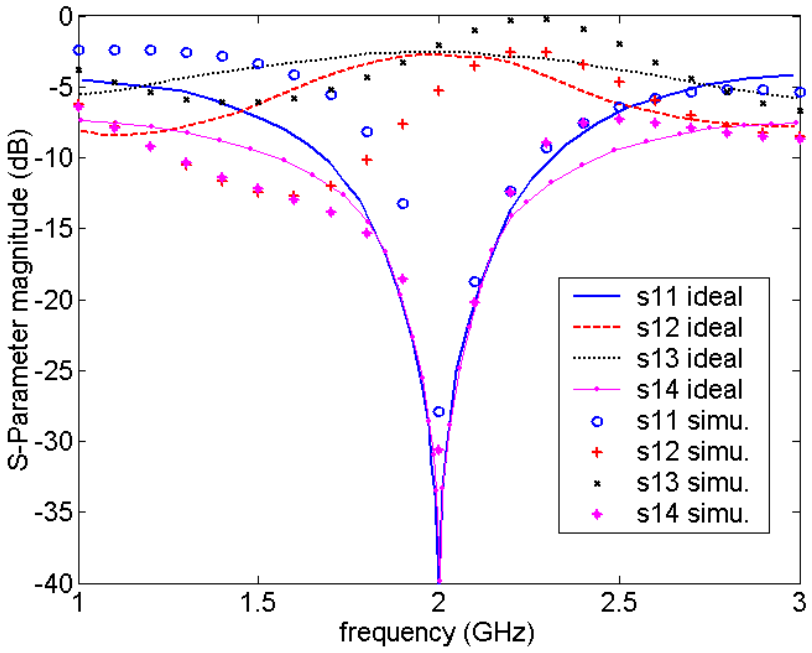


Figure 2.7. S-parameter frequency sweep for hybrid circuit.

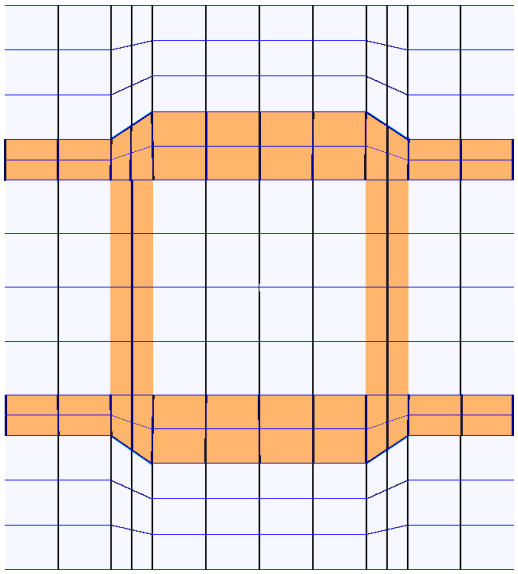


Figure 2.8. Simulated hybrid circuit design.

Table 2.1. Permutations in Matrix Preconditioning for Directional Coupler.

BICG-STAB(8)	Precond. Time	Precond. Storage	Iterations	Solver Time	Total Solve Time
No Precond.	0sec	0MB	48	24sec	24sec
ILU Precond.	5sec	7MB	14	12sec	17sec
FEM LU Precond.	0.3sec	2MB	5	3sec	3sec
Block LU Precond.	4.5sec	6MB	1	0.6sec	5sec

To compare the matrix preconditioning approaches mentioned in this chapter, the hybrid circuit example is evaluated at the center frequency of 2GHz for each of the mentioned preconditioners, and comparisons are given in Table 2.1. In this comparison, the most commonly used BICGSTAB(L) solver is employed. The advantage of pre-solving the FEM system of equations is evident, but clearly at a higher cost than ILU(0) preconditioning. The point of this exercise is to demonstrate that pre-solving the FEM portion of the system leads to faster overall solution time. For reference, the table was generated on a PC with an AMD Athlon XP 1800^{+} CPU.

The next example considered here evaluates the radiation pattern from the antenna element shown in Figure 2.9. This antenna design was used as the primary array element in a recently designed dielectric lens antenna [25]. The frequency of operation for the antenna is 24GHz, and the element is printed on Rogers Corporation RT/Duroid® 5880 substrate. The electrical permittivity of this material is $\varepsilon_r = 2.2$, and the substrate is 254 microns thick. The shown antenna structure is 3.2λ wide and 5.3λ long at 24GHz.

The measured and simulated radiation patterns are shown in Figure 2.10, and are reasonably well matched at this high frequency. For this structure, there are a number of challenges that make it more difficult to achieve good agreement between measurements and calculation. One main reason for the difference is that a portion of the circuit was excluded from the simulation – only the microstrip feed and the radial stub nearest the microstrip-to-slotline transition were included. It is believed the excluded circuit structure contributes to the anomaly on the right-hand side of the measured main beam. Multiple measurements were taken to verify the existence of this beam asymmetry. As a further difference the antenna element was simulated as a standalone, isolated element. The measured structure was actually the central element of a linear array of several antennas. Due to the size of the array, measuring the entire structure would have required too many resources for the conventional FE-BI formulation.

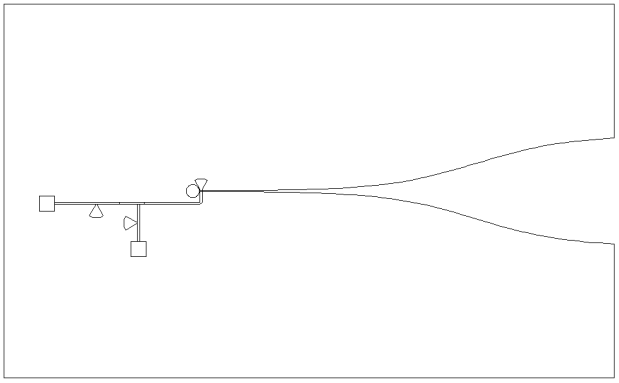


Figure 2.9. Diagram of antenna structure for example 2.

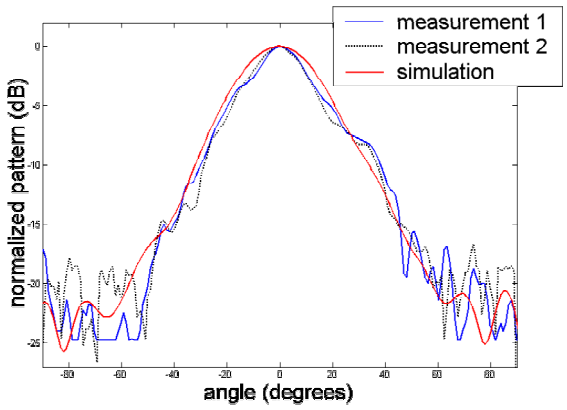
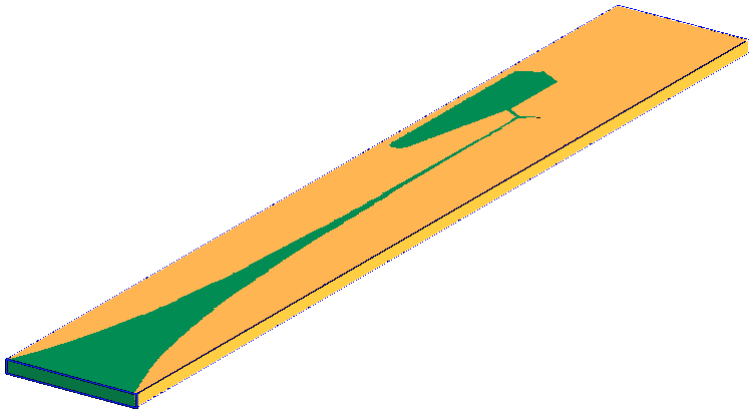
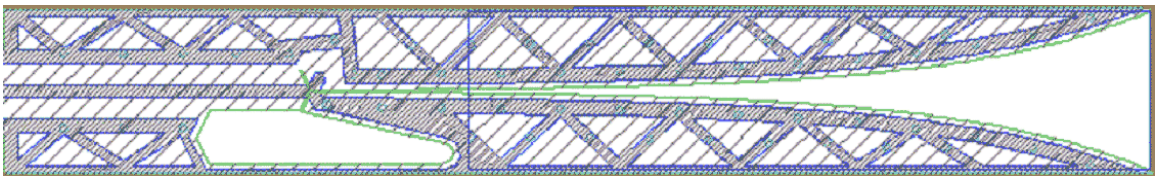


Figure 2.10. Pattern comparisons for the antenna in example 2.

For the antenna design in Figure 2.9, it is not possible to measure the input impedance of the antenna, due to the nature of the feeding circuit. To validate the impedance measurement capabilities of the formulation, another antenna structure is considered, depicted here in Figure 2.11. Figure 2.11 (a) shows the way this antenna is packaged, and a diagram of the features internal to the antenna is given in Figure 2.11 (b). The antenna measures roughly 25cm in length as shown, has a width of about 3.6cm, and is 0.3cm thick. It is designed to operate in the frequency range of 1-5GHz. This element, provided by Naval Research Labs (Pala, Kragalott, Dorsey), will be used again for array coupling studies in later chapters.



(a)



(b)

Figure 2.11. NRL antenna element layout (a) package, (b) detail.

The discretization used for this antenna element is depicted in Figure 2.12. Due to the unusual complexity of this particular structure relative to the frequency of operation, it is necessary to use edge ratios of nearly 30 to 1 in tessellating the element. Consistent use of FEM elements with the same edge ratio would result in a very large number of unknowns, making it impractical to analyze the structure with the conventional formulation developed here. The element is modeled with 2075 FEM unknowns and 2920 BI unknowns.

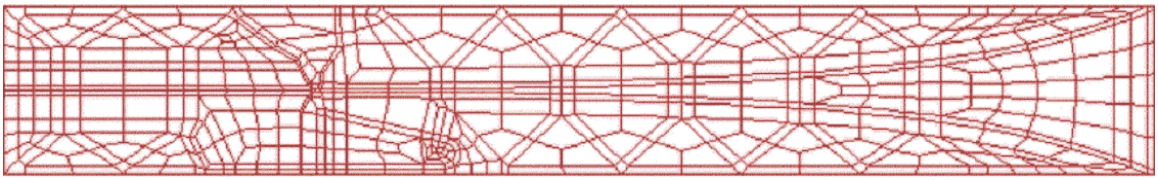
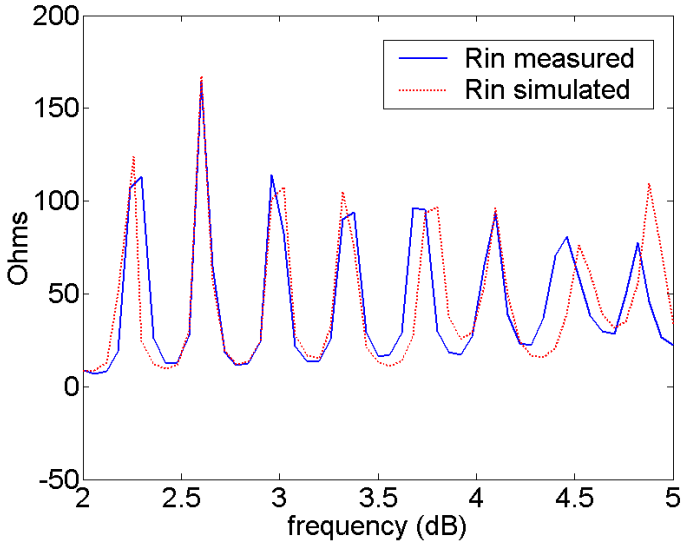
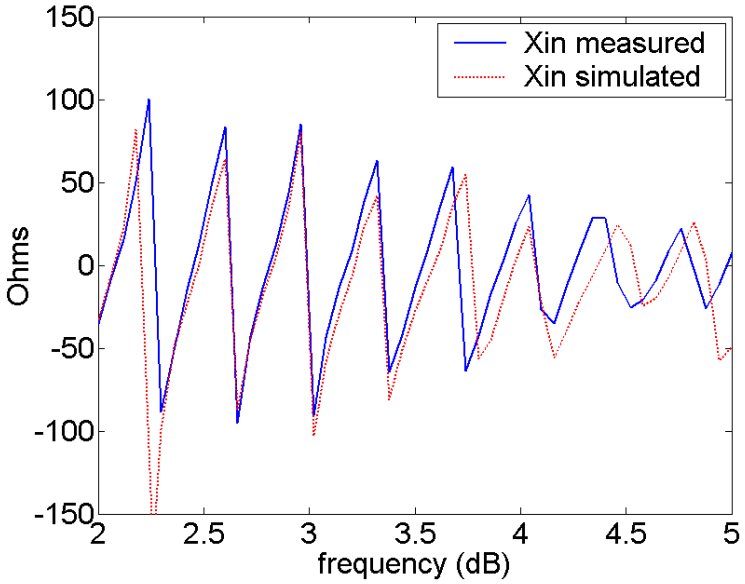


Figure 2.12. NRL antenna element discretization.

The results for the NRL antenna impedance measurement are given in Figure 2.13. While these results match reasonably well only for a small range of the overall impedance measurement, namely the range from 2.5 to 3.5 GHz, the achieved match is relatively good considering the frequency range over which a single element discretization is used for the simulation. In other words, better results could have been achieved at the higher frequencies by using a denser mesh. Note also that the impedance appears quite erratic for an element that is supposed to have broadband performance capabilities. This element achieves broadband characteristics only when embedded within the array environment. In other words, this element depends upon the coupling from neighboring array elements to achieve broadband performance characteristics. For this impedance measurement, the element was removed from the array environment and measured in isolation.



(a) real part



(b) imaginary part

Figure 2.13. NRL antenna element input impedance.

As a second test of the input impedance calculation, consider the case of a patch antenna as shown in Figure 2.14. The dimensions of the patch are 5.0cm by 3.4cm, and the material substrate has a thickness of 0.08779cm. The material has a dielectric constant of 2.17, with a loss tangent of 0.0015. The patch antenna has a resistive load of 50Ω located near one of the corners, and a coaxial feed point offset from the patch center as shown. The exact geometric design is described in [26].

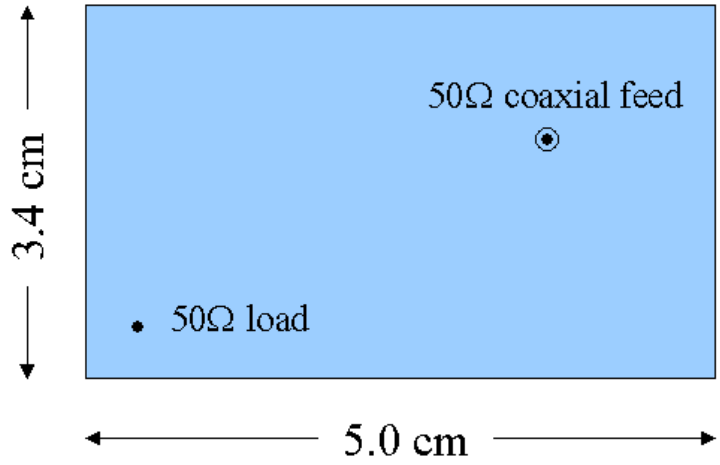
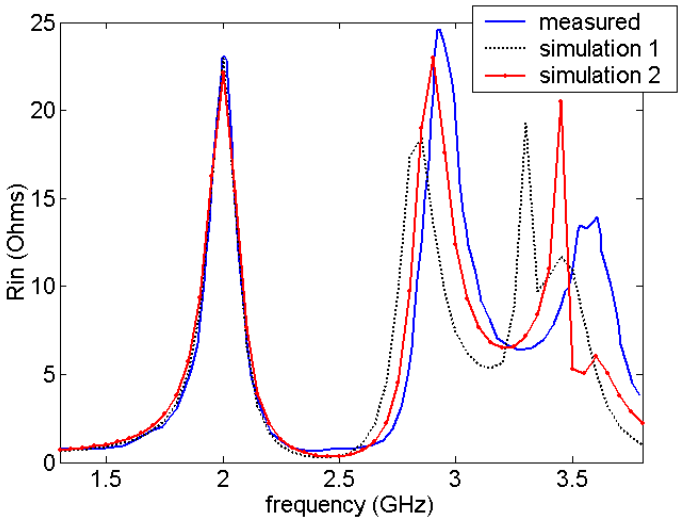
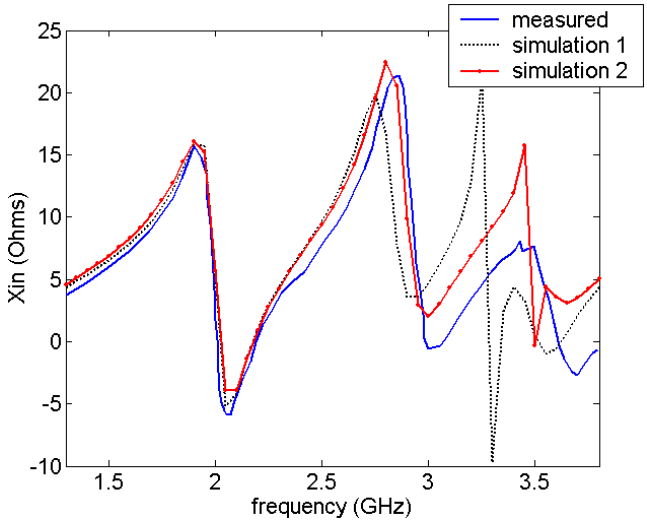


Figure 2.14. Patch antenna example geometry.



(a) Real component of patch input impedance



(b) Imaginary component of patch input impedance

Figure 2.15. Patch antenna input impedance.

For this example, the patch is analyzed twice. Simulation 1 was performed with 209 hexahedral elements, resulting in 353 FEM unknowns, 936 BI unknowns, and a matrix size of N=1289. Simulation 2 was performed with 359 hexahedral elements, resulting in 699 FEM unknowns, 1500 BI unknowns, and a matrix size of N=2259. In

Figure 2.15, the simulated values for the patch input impedance are plotted against the measured results. It is clear that the simulations and measurements for the patch antenna are in good agreement. At the higher frequencies, the denser mesh provides more accuracy, as expected.

As a demonstration of matrix condition importance, at the patch resonant frequency of 2GHz, the matrix condition was computed for various ratios of MFIE to EFIE formulations. The results are listed in Table 2.2. Clearly the results show that the MFIE formulation results in a better matrix condition number, by an order of magnitude for a problem of the tested size.

Table 2.2. Matrix Condition Study for Patch Antenna Problem.

EFIE/MFIE Ratio	Condition Number	Iterations	Solve Time No Precond.
$\alpha = 1.0$	72207	45	28sec
$\alpha = 0.5$	6202	25	15sec
$\alpha = 0.0$	5402	17	11sec

Perhaps more importantly, the effect of the FEM matrix on the overall system condition should be considered. For this system of equations, the isolated FEM matrix has a condition number of 3486, whereas the $[Q]$ matrix alone has condition numbers of 3373, 121, and 46 for $\alpha = 1.0$, $\alpha = 0.5$, and $\alpha = 0.0$ respectively. Hence, using a direct solver on the FEM portion of the system when the resources are available will significantly improve the solution speed for any value of α.

In a future section of this thesis, important coupling measurements will be made at the ports of large array structures separated by a distance. In order to validate that the formulation is capable of handling such a task, the coupling (mutual impedance) between two dipoles separated by a variable distance is considered. The coupling situation for the dipoles is depicted in Figure 2.16. The mutual coupling Z_{12} is computed by measuring the ratio of the voltage induced across the port of dipole 2, to the current entering dipole port 1 at various side-by-side separation distances. The dipoles are mostly PEC with a small dielectric gap at the center, across which the ports are placed. The coupling results are shown in Figure 2.17, and match the expected form of the mutual coupling behavior of dipoles, as reported in [27].

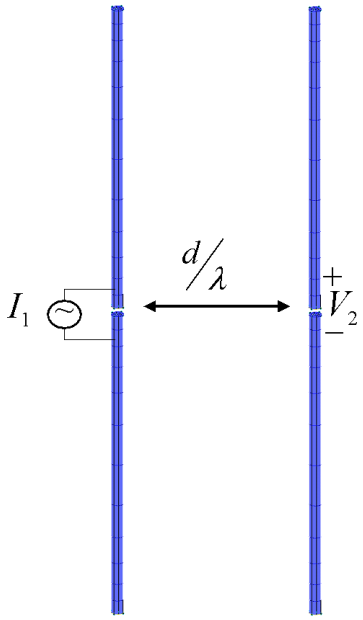


Figure 2.16. Geometry for side-by-side dipole coupling calculation.

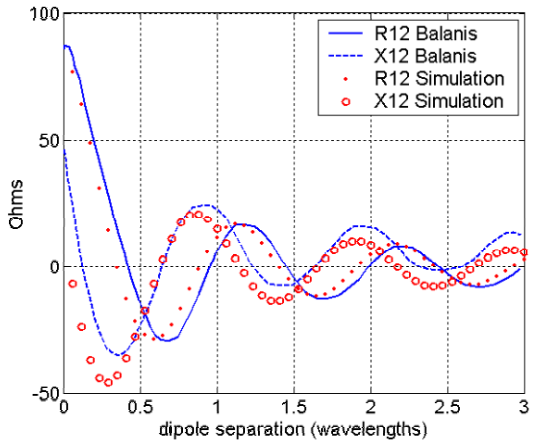


Figure 2.17. Mutual coupling for side-by-side dipoles vs. separation distance.

Having validated the formulation for a wide range of problem classes, it should be apparent that hybrid FE-BI is a robust technique for CEM analysis. At this point, it is instructive to see how well conventional FE-BI can be used for finite array analysis. This is in fact entirely necessary, as the main premise of this work is that exact methods such as conventional FE-BI are too computationally expensive for very large structural analysis, prompting the development of the advanced array methods in the later chapters of this thesis. In this example, the base formulation developed in this chapter is used to analyze several arrays of the type shown in Figure 2.19.

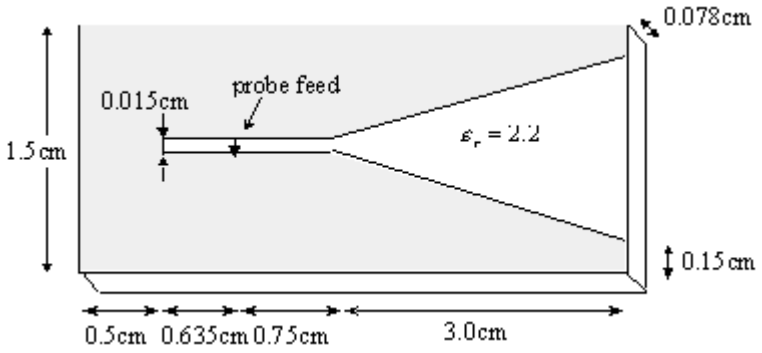


Figure 2.18. Tapered-slot antenna element for 10GHz operation.

A diagram of the element used in this array is depicted in Figure 2.18. The element is probe-fed at a distance $\lambda/4$ from the slotline short when excited at 10GHz. At 10GHz, the element is relatively small, and radiates rather poorly. However, it is necessary to construct arrays from electrically small elements when using conventional FE-BI for analysis, as will become apparent from the required storage for even small arrays. It is important to note that the intention of this example is not to design an industrial quality array, but rather to validate the potential of particular formulations to analyze similar array structures. The individual elements of the array are spaced 1.6cm apart in a two-dimensional grid. A total of 1103 unknowns were used to tessellate each element, with 507 FEM unknowns and 596 BI unknowns.

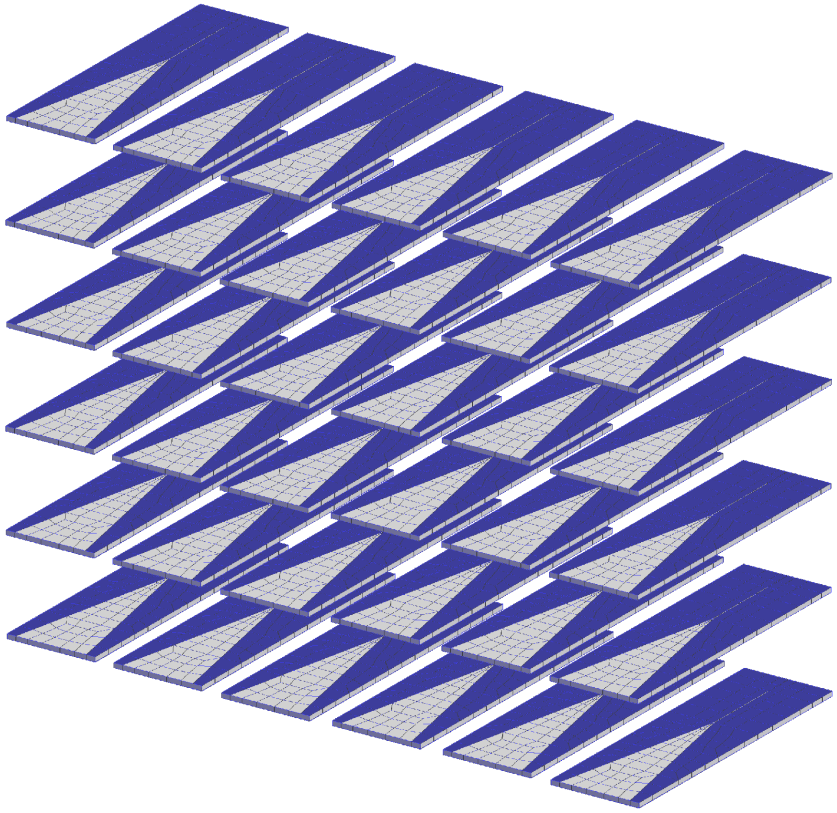


Figure 2.19. Finite tapered-slot antenna array.

Using the element of Figure 2.18, several array configurations were constructed as listed in Table 2.3. The arrays were analyzed using a 64-bit Itanium server with 16GB of shared memory. The HP rx4610 server has four 800MHz Itanium CPUs, though only a single processor was employed in the analyses appearing here. The matrix structures are populated with FORTRAN 90 based COMPLEX(4) data types, requiring 8 bytes per data element. The listed array configurations were all analyzed using the BICGSTAB(L) iterative solver with the ILU(0) matrix preconditioner. As seen from the table, a 3×3 array of this element poses no serious challenge for conventional FE-BI analysis. The 10,000 unknown problem can be solved in a few hours time using a modest amount of memory. The 6×6 array has nearly 40,000 unknowns, and requires roughly 6GB of

storage. The matrix fill time for this problem was approximately 2 hours, and the solution time was close to three days. Clearly, this size problem is near the limit for conventional FE-BI analysis. While it would be possible to squeeze a slightly larger problem into the 16GB of storage available on the server, it is clear that the problem size would only be marginally larger considering that the chosen preconditioner requires as much storage as the matrix itself, leading to the conclusion that very large structural analysis will not be possible with conventional FE-BI alone. For future reference, a 10×10 array of the same element requires too much storage to solve via conventional means (45GB), and analysis was not attempted.

Table 2.3. Array Computations for Conventional FE-BI.

Array Size	Unknown Count	Matrix Storage Requirements	Matrix Fill Time	Solve Time
3x3	9,927	368MB	7.5min.	3.7hrs.
6x6	39,708	5,880MB	1.9hrs.	3.1days
10x10	110,300	45,348MB	*16hrs.	-

*estimated data

2.8 Discussion of Limitations

From the results presented in this chapter, it should be apparent that the hybrid FE-BI is a capable method for accurate electromagnetic analysis. Also from the results presented, it should be apparent that this reliability comes at a high price – namely the storage requirements prohibit the analysis of large structures. With this conventional FE-BI approach, it was only possible to analyze a 6×6 array with 20,000 unknowns. In order to achieve accurate analysis for larger structures, it will ultimately be necessary to

take advantage of redundancies in the finite array environment. Before the exploitation of redundancies in finite array-type problems is addressed, a more general approach to reducing the cost of exact integral methods will be examined. Later in the thesis, focus will return to these general techniques for addressing specific weaknesses in the array methods. In the next chapter, the fast multipole method (FMM) is explored as an attempt to address the main shortcomings of conventional FE-BI for arbitrary structure analysis.

CHAPTER 3

THE FAST MULTIPOLE METHOD

From the details given in the preceding chapter, particularly from the demonstration of limited capability of conventional FE-BI methods to analyze finite arrays, it should be apparent that FE-BI alone is not best suited for large structure analysis. The main difficulty in using conventional FE-BI to analyze large structures is the $O(m^2)$ storage requirements of the method quickly exhaust system resources. A simple calculation shows that a matrix system with m=11,585 unknowns will require a gigabyte of storage, assuming each matrix entry is 8 bytes. It is therefore apparent that enhancements to the conventional algorithm must be made if FE-BI is to be used for even larger systems with possibly *millions* of unknowns.

To recall, the FE-BI method was initially chosen for its excellent robustness in solving problems of nearly any type with complete rigor. To avoid giving up this capability, in this chapter a powerful method for reducing the computational costs associated with FE-BI will be examined. The primary bottleneck of conventional FE-BI lies in its dense integral equation matrices, and this will be a focus for improvements. The cost of this storage stems from the rigorous near-zone terms of the matrix system used for interacting all elements in the system, both near and far. In a large structure, a significant portion of the interactions may be separated by many wavelengths. For

regions of the structure that are far separated, corresponding to roughly a wavelength separation or more, rigorous interaction can be relaxed at virtually no cost to solution accuracy when done properly. The procedure explored here for relaxing the rigor of distant interactions is called the fast multipole method, or FMM for short. The application of FMM will reduce the $O(m^2)$ storage requirements of the conventional FE-BI method down to $O(m^{1.5})$ or better. The reduction to $O(m^{1.5})$ alone reduces the cost of a 11,585 unknown problem by two orders of magnitude. In later chapters the benefits of the FMM will be exploited to achieve much greater functionality when applied in combination with a near-zone decomposition approach for array-type systems.

In this chapter, a short exposition of the fast multipole method for electromagnetic analysis is given. In this exposition, the details on the FMM will be explored with particular emphasis on implementation issues. While there will be little in the form of new contributions in this chapter, the fundamentals of the FMM, as detailed here, will be used again in later chapters in new ways for far-zone array decomposition.

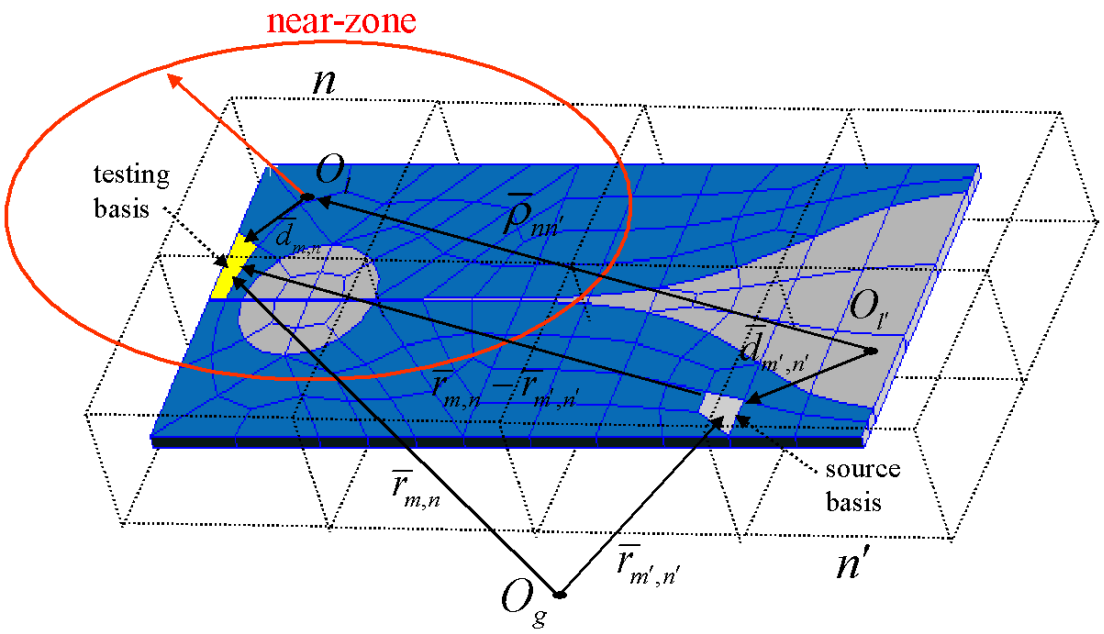


Figure 3.1. FMM cluster interaction within arbitrary system.

3.1 Fast Multipole Method Formulation

To explain how the FMM is employed in analyzing an arbitrary problem, consider the discretized structure shown in Figure 3.1. The FMM works by partitioning an arbitrary geometry into regular (or potentially irregular) regions. The chosen partitioning scheme is somewhat subjective, but follows the rudimentary criteria that the partitioned regions should be smaller than a wavelength in diameter and not overlap each other. This rule of thumb makes the FMM more efficient, and the reasons and considerations for this guideline will be made apparent shortly. Each partitioned region of the structure is given a local origin O_l. The basis elements of the geometry surface are associated with the region for which the distance to the local origin of the region is smallest. As an example, in Figure 3.1, the local regions to which source and testing basis functions are associated

are clearly depicted. All basis elements falling within the pre-defined radii of a partitioned region will be associated with that local group.

The process of grouping basis elements creates localized groups or cells that will be referred to henceforth as simply 'clusters.' The goal of this partitioning procedure is to circumvent the need for interacting individual basis elements with rigorous near-zone interactions, and instead interact *groups* of basis elements (that are far-separated) with inexpensive approximations of the coupling mechanism, i.e. the free-space Green's function. Clusters that are not far-separated will be treated in the usual way, i.e. via the rigorous near-zone integral equation formulation of the preceding chapter. In FE-BI, FMM is applied to boundary integral currents on the surface structure. This same procedure can be applied to volumetric integral equations as well [14].

Consider a source and testing location on the structure, positioned at $\overline{r}_{m',n'}$ and $\overline{r}_{m,n}$ respectively, referenced to a global origin O_g (see Figure 3.1). Let the source and testing basis elements be given local indices m' and m respectively, relative to a local clustering scheme. The testing and source clusters are denoted with the indices n and n' respectively. As a generalization, the dense impedance matrix entry for this interaction will be of the form

$$\left[a_{mn,m'n'}\right] = j\int_S\int_{S'} \overline{w}(\overline{r}_{m,n})\overline{w}(\overline{r}_{m',n'})\frac{e^{-jk_0\left|\overline{r}_{m,n}-\overline{r}_{m',n'}\right|}}{4\pi\left|\overline{r}_{m,n}-\overline{r}_{m',n'}\right|}dS'dS\,. \tag{3.1.1}$$

Rather than describing this interaction using global coordinates, it can be equivalently described using local group coordinates in combination with a vector relating the position of the source and testing cluster groups. The vector relating the position of these clusters

is given by $\bar{\rho}_{nn'}$, as indicated in Figure 3.1. Using these local coordinate relations, it is possible to rewrite the kernel in (3.1.1) as

$$\frac{e^{-jk_0\left|\bar{r}_{m,n}-\bar{r}_{m',n'}\right|}}{4\pi\left|\bar{r}_{m,n}-\bar{r}_{m',n'}\right|}=\frac{e^{-jk_0\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}}{4\pi\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}. \tag{3.1.2}$$

The advantage of this alternate expression will be made apparent below, as further details of the FMM are revealed.

The FMM, as developed here, is based primarily on two identities. These identities can be found in [28]. The first identity, based on Gegenbauer's addition theorem, allows the expansion of the kernel of the scalar free-space Green's function as

$$\begin{aligned}&\frac{e^{-jk_0\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}}{\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}=\\&-jk_0\sum_{l=0}^{\infty}(-1)^l(2l+1)j_l(k_0\left|\bar{d}_{m,n}-\bar{d}_{m',n'}\right|)h_l^{(1)}(k_0\left|\bar{\rho}_{nn'}\right|)P_l(\bar{\rho}_{nn'}\bullet\frac{\bar{d}_{m,n}-\bar{d}_{m',n'}}{\left|\bar{d}_{m,n}-\bar{d}_{m',n'}\right|})\end{aligned}. \tag{3.1.3}$$

In this substitute expression, $P_l()$ are the Legendre polynomials, $h_l^{(1)}()$ denote the spherical Hankel functions of the first kind, and $j_l()$ are the corresponding spherical Bessel functions of the first kind. The values of these functions can be found in many references on mathematical functions, such as [28]. A second identity important for the FMM development is

$$4\pi j^l j_l(k_0\left|\bar{d}_{m,n}-\bar{d}_{m',n'}\right|)P_l(\frac{\bar{d}_{m,n}-\bar{d}_{m',n'}}{\left|\bar{d}_{m,n}-\bar{d}_{m',n'}\right|}\bullet\hat{\rho}_{nn'})=\int e^{j\bar{k}\bullet(\bar{d}_{m,n}-\bar{d}_{m',n'})}P_l(\hat{k}\bullet\hat{\rho}_{nn'})dk\Omega. \tag{3.1.4}$$

Combining (3.1.3) and (3.1.4), the kernel function can be rewritten as

$$\frac{e^{-jk_0\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}}{\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}= \frac{-jk_0}{4\pi}\sum_{l=0}^{\infty}\int e^{i\bar{k}\bullet(\bar{d}_{m,n}-\bar{d}_{m',n'})}dk\Omega(-1)^l(2l+1)h_l^{(1)}(k_0\left|\bar{\rho}_{nn'}\right|)P_l(\hat{k}\bullet\hat{\rho}_{nn'}) \quad . \tag{3.1.5}$$

The important aspect of this expansion is that it decouples the primed and unprimed components of the original Green's function.

Note that a large portion of the kernel as given in (3.1.5) is dependent only on the distance vector separating clusters. It is essential at this juncture to isolate the terms in (3.1.5) that are strictly a function of the vector separating clusters, and group them into their own operator given by

$$\mathrm{T}_L[k_0\left|\bar{\rho}_{nn'}\right|,\hat{k}\bullet\hat{\rho}_{nn'}]=\sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k\left|\bar{\rho}_{nn'}\right|)P_l(\hat{k}\bullet\hat{\rho}_{nn'}))\,. \tag{3.1.6}$$

Notice that the infinite sum has been replaced with a truncated series, thereby allowing a controlled degree of accuracy and speed. The important advantage of this step is that (3.1.6) can easily be pre-computed prior to solving the system of equations at low cost, and can potentially be re-used for any groups with the same vector separation within the arbitrary structure. This lumped quantity is referred to as the translation operator. The translation operator matrix will have entries for each of the paths between the n individual clusters (n^2 total entries), as well as for K directions in k-space, the choice of which is described below. In essence, the translation matrix is used to compute the effect of the clusters on each other in the far-zone, by shifting the far-zone influence (field signatures) of each group such that they line up in k-space for the selected set of k-space directions, as a cheaper alternative to interacting all basis elements within the clusters individually in the near-zone.

In effect, the FMM uses an approximation of the kernel in (3.1.5) given by

$$\frac{e^{-jk_0\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}}{\left|\bar{\rho}_{nn'}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}\approx\frac{-jk_0}{4\pi}\int e^{j\bar{k}\bullet(\bar{d}_{m,n}-\bar{d}_{m',n'})}\mathrm{T}_L[k_0\left|\bar{\rho}_{nn'}\right|,\hat{k}\bullet\hat{\rho}_{nn'}]dk\Omega\,. \qquad (3.1.7)$$

This approximate expression is attractive in the context of exact methods that employ iterative solvers to generate approximate solutions. However, the conditions under which this approximate expression can be used are limited by some critical guidelines. Because of the interchange of summation and integration order, the number of terms L in the summation cannot be allowed to be much larger than $k_0\left|\bar{\rho}_{nn'}\right|$, or the Hankel function will oscillate wildly. However, in order for the multipole expansion to converge, the value of L must be sufficiently large, so one cannot simply choose an arbitrarily small L. That is, the value of L must be greater than the maximum value attainable by $k_0(\bar{d}_{m,n}-\bar{d}_{m',n'})$, or $k_0(D_n+D_{n'})$, where D_{n} and $D_{n'}$ are the maximum radii of the testing and source clusters respectively, such that the partial-wave expansion in (3.1.4) converges. Preferably, one would like to choose L as small as possible to minimize computational and storage costs, since smaller L values correspond to significantly fewer k-space directions. In the end, one must choose a value of L as a trade-off between accuracy and storage requirements. Conceptually, this suggests a benefit in choosing smaller cluster diameters, which consequently allows a smaller value of L. As a tradeoff, this increases the total number of clusters, which in turn increases the number of group interactions (translations) that must be pre-computed and stored. It has been reported that one can optimize the computational cost when the number of clusters is chosen proportional to the square root of the total number of basis functions, or $n=\sqrt{N}$ [1].

Once the cluster assignments have been determined, the number of terms used in (3.1.6) is found roughly by the relation [1, 29]

$$L = k_0(D_n + D_{n'}) + \alpha_L \ln(k_0(D_n + D_{n'}) + \pi). \tag{3.1.8}$$

Here, α_L is chosen to generate the desired accuracy for the application type and machine architecture, but should be approximately between 1 and 10.

There also remains the issue of clarifying when the approximation (3.1.7) can be implemented. As mentioned earlier, it is necessary that $L < k_0 |\overline{\rho}_{nn'}|$. This implies that the cluster separation $|\overline{\rho}_{nn'}|$ must be greater than L/k_0 in order to use the FMM approximation. For cluster groups separated by less than L/k_0, the near-zone interaction of the conventional FE-BI method must be used.

From a procedural perspective, the intention is to pre-compute critical structures, such at the translation matrix, prior to initiating the iterative solution procedure. This is the means by which the FMM attains a potential solution speedup. In fact, this is a necessary step, since the FMM increases the cost of the iterative solution procedure, and benefits mainly by pre-computing numerical entities prior to beginning the iterative solution. The main speedup occurs through pre-computing the entries of the translation matrix (3.1.6). Though this structure can be effectively pre-computed, unfortunately, the effect of distant clusters on each other must be updated at each iteration of the solution procedure. As a note, though the influence of clusters upon each other is carried out in *k*-space via the FMM, communication of this influence to and from the individual basis elements of the clusters must be carried out in the spatial domain through Fourier transforms. This process will be made clearer shortly. It is worth mentioning that a further advantage of pre-computing the translation operators is that for clusters on a

regular grid, the translation matrix will have a block-Toeplitz property. In other words, it will not be necessary to compute translation operators for all n^2 interactions within the partitioned structure grid – only the unique interactions, represented by unique cluster separation vectors. This allows a beneficial storage reduction. The Toeplitz property will again be exploited in later chapters when dealing with array interactions.

The FMM requires the computation of signature functions for the fields in each of the clusters. Signature functions are equivalently the Fourier transform of the basis functions, referenced to the center of the cluster to which the basis function belongs. Pre-computation of the signature function for a given cluster allows the convenient evaluation of the far-zone pattern for that cluster, simply by multiplying the field coefficients with the pre-computed signature functions. For a common basis function $\bar{w}(\bar{r})$ used to construct the fields, the signature function of cluster n is given as

$$W_n[\hat{k}, m] = \int_S \bar{w}(\bar{r}_{m,n}) e^{j\bar{k}\cdot(\bar{r}_{m,n} - \bar{\rho}_n)} dS \,. \tag{3.1.9}$$

Each of the n clusters will have a signature function sub-matrix $\left[W_{\hat{k},m}\right]_n$ that represents every basis element m within that cluster, computed for each of the k-space directions required to accurately represent the footprint of the cluster on the far-zone sphere. The number of k-space directions is proportional to the size of the cluster, and is given by approximately $2L^2$, where from before, L is the number of terms used in the summation (3.1.6). Since $\left[W_{\hat{k}}\right]_n$ is essentially the basis function mapped onto the far-zone sphere, $\left[W_{\hat{k},m}\right]_n^* \{x_m\}_n$ gives the radiation contribution of the surface currents in cluster n in the far-zone for the desired k-space directions. These values are only calculated at discrete

points or directions in *k*-space and are therefore considered diagonal (vector vs. matrix) in form. That is, information at different *k*-space directions is not communicated to each other. As usual, S is the volume/area of the geometry subtended by the currents of cluster *n*.

To demonstrate how the FMM works, consider the interaction of two clustered basis element groups as shown in Figure 3.1. For rigorous integral equation interaction of the source basis elements in cluster n' with the testing basis elements in cluster n, there will be a dense matrix of the form given in (3.1.1). This is necessarily the case, as rigorous solution requires the individual interaction of every source element with every target element of each respective cluster to achieve a full matrix that can be solved. If the clusters are far-separated, however, this rigor is not necessary. By using the approximation to the Green's function given in (3.1.7), the interaction of these groups can alternatively be represented as

$$[a_{mm'}]_{nn'} \approx \frac{k_0}{(4\pi)^2} \int \left[\int_S \overline{w}(\overline{r}_{m,n}) e^{j\overline{k}\cdot\overline{d}_{m,n}} dS \{\tau_L\}_{nn'} \int_{S'} \overline{w}(\overline{r}_{m',n'}) e^{-j\overline{k}\cdot\overline{d}_{m',n'}} dS' \right] dk\Omega . \quad (3.1.10)$$

The FMM is not actually implemented as a matrix element of the form given in (3.1.10). Rather, this expression represents an iterative solution procedure that will be described in detail later. Briefly, the two inner integrals of (3.1.10) are the pre-computed signature functions of the source and testing groups. The quantity $\{\tau_L\}_{nn'}$ in (3.1.10) represents the pre-computed translation matrix entry for interacting the testing and source groups n and n'. Essentially, the far-zone signatures of the source groups are first computed as outgoing plane waves, the source group influence is then translated in the far-zone to compute how it interacts with the testing group, and finally this influence is

transformed back into real space as incoming plane waves to individual testing locations via the outer integral. Each of the quantities within the outer integral are vectors with a length equal to the number of *k*-space directions required for the signature functions. In essence, the FMM reduces a dense matrix-vector interaction into a diagonalized vector operation, potentially resulting in solution acceleration. The result on the near-zone integral equation matrices is that they now become sparse, as depicted in Figure 3.2. The shaded regions represent the sparse data, corresponding to a 47% reduction in storage over conventional FE-BI, for which the dense matrix structure would be entirely shaded (full).

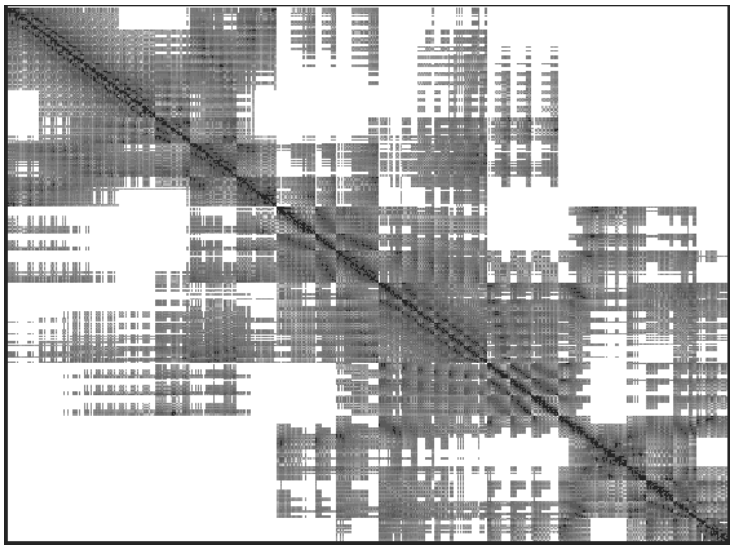


Figure 3.2. Example sparse matrix structure at 47% fill.

With conventional FE-BI, solution options include the use of direct solvers (matrix inversion, LU factorization), or indirect solvers (iterative methods). In FMM there is still the option of inverting the now sparse matrix structure of Figure 3.2, however this will no longer lead to the final solution. In order to solve the system when

the FMM has been employed, it is necessary to use an iterative solution approach. This solution procedure is described in some detail below. First, some enhancements to conventional FMM are examined in preparation for later consideration.

3.2 FMM-FFT

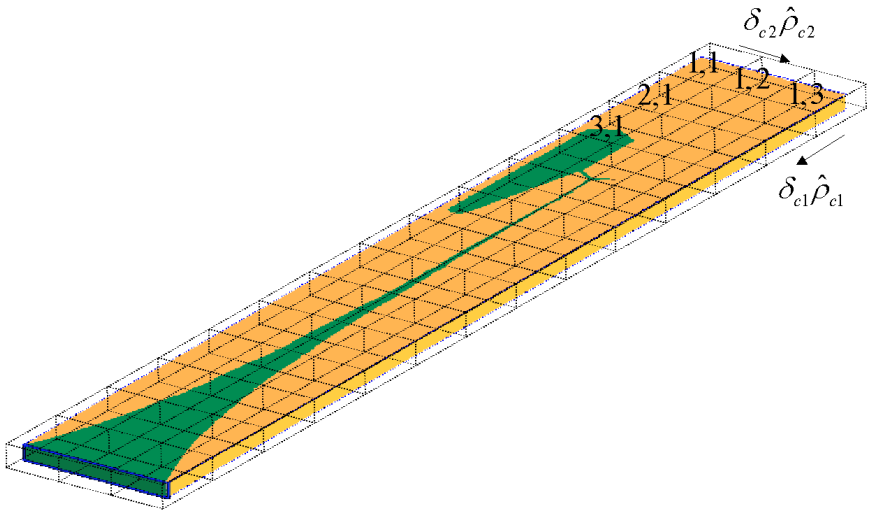


Figure 3.3. Toeplitz cluster grid for FMM-FFT.

Under certain conditions, it is possible to explicitly accelerate the FMM solution procedure using the FFT. To do this, the geometry structure must be partitioned into a grid of clusters with regular spacing and consecutive cluster numbering, as depicted in Figure 3.3. In the simplest implementation, the arbitrary structure is divided into a three-dimensional grid of cells, or clusters. Each cluster contains a number of the basis elements used to partition the structure. When the structure is divided in this way, the translation matrix (3.1.6) will have a Toeplitz property, and can be recast as

$$\mathrm{T}_L[k_0\,|\overline{\rho}|,\hat{k}\bullet\hat{\rho}] = \sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k_0\,|\overline{\rho}|)P_l(\hat{k}\bullet\hat{\rho}))\,, \tag{3.2.1}$$

where

$$|\rho| = \left|(i_{c1}-i'_{c1})\delta_{c1}\hat{\rho}_{c1} + (i_{c2}-i'_{c2})\delta_{c2}\hat{\rho}_{c2} + (i_{c3}-i'_{c3})\delta_{c3}\hat{\rho}_{c3}\right|, \tag{3.2.2}$$

and

$$\hat{\rho} = \frac{(i_{c1}-i'_{c1})\delta_{c1}\hat{\rho}_{c1} + (i_{c2}-i'_{c2})\delta_{c2}\hat{\rho}_{c2} + (i_{c3}-i'_{c3})\delta_{c3}\hat{\rho}_{c3}}{|\rho|}\,. \tag{3.2.3}$$

In these expressions, the source cluster index is given as i'_c, whereas the testing cluster is given the index i_c, for each dimension of decomposition $(c1, c2, c3)$. The cluster index for each dimension has the range $i_c = 1..n_c$, where n_c is the number of clusters in a given dimension. In the case of Cartesian coordinates, it is most simple to make the assignment $c1 = x, c2 = y, c3 = z$. In general, however, the cluster spacing is denoted arbitrarily as δ_c, and the direction of the clustering lattice is denoted as $\hat{\rho}_c$, for up to three arbitrary dimensions. Though three levels of cluster grid have been allowed for, only two are used in Figure 3.3 for the planar structure.

Toeplitz storage of the translation operators takes the form

$$\left[\mathrm{T}_L\right] = \begin{bmatrix} \ddots & \vdots & \iddots \\ \cdots & \{\tau_L\}_{(i_{c1}-i'_{c1})(i_{c2}-i'_{c2})(i_{c3}-i'_{c3})} & \cdots \\ \iddots & \vdots & \ddots \end{bmatrix}, \tag{3.2.4}$$

where each $\{\tau_L\}$ contains the *k*-space vectors for the translation between the source cluster at index $i'_{c1}, i'_{c2}, i'_{c3}$, and the target cluster at index i_{c1}, i_{c2}, i_{c3}, with a matrix index given by the difference in the source and testing indices. The overall size of $[\mathrm{T}_L]$ will be

$(2n_{c1}-1)\times(2n_{c2}-1)\times(2n_{c3}-1)\times K$, where K is the total number of unique *k*-space vectors. This quantity can be readily transformed via the FFT, and later used to accelerate the iterative FMM procedure for far-zone interaction. This procedure is described in a later section.

3.3 Multi-Level Fast Multipole Method

A popular enhancement to the FMM is the use of a multi-level version of the algorithm [14]. The multi-level FMM, or MLFMM for short, is characterized by clustering the problem at multiple levels, each consecutive level grouping the smaller clusters of the previous level into larger clusters. For example, at the highest level, the system might be broken into four quadrants of four clusters, and then at the second-highest level each quadrant is broken into four more quadrants, etc. This minimizes the number of translation operators that need to be computed at each given level. The multi-level version of the FMM algorithm reduces the computational cost from the already low $O(m^{1.5})$ down to a potential $O(m\log m)$. A potential benefit of this approach is that empty clusters at each level can be removed from the sparse translation matrix, and need not be stored. For largely arbitrary and non-symmetric geometries, this is potentially quite beneficial. The multi-level approach has not been explored in this thesis for reasons that will be made clear in later chapters when dealing with array-type problems. Suffice it to say, here the FMM-FFT has been chosen instead for implementation, as it conforms

well with the array decomposition methods explored in later chapters. However, occasionally the multi-level version of FMM will be used in comparisons.

3.4 FMM Solution Procedure

The solution of the system of equations does not differ much in terms of *computational values* when comparing the FMM and conventional FE-BI – matrix-vector product values from iteration to iteration are largely the same (ideally identical). However, from a *procedural* perspective, the methods differ significantly. The FMM portion of the procedure can be summarized in a number of concise steps. The first step is to compute the updated fields for each of the element groups *n* in the far-zone using the estimated field coefficient values $\{x\}^{est}$ of the present iteration and the pre-computed signature functions:

$$\left\{s_{\hat{k}}\right\}_n = \left[W_{\hat{k},m}\right]_n^* \{x_m\}_n^{est}. \tag{3.4.1}$$

After this process, all information about individual basis functions cannot be accessed (there is no *m* index for $\left\{s_{\hat{k}}\right\}_n$) without transformation back to spatial domain. In the next step, the fields of the source group are translated along the far-zone sphere to match up with the target element groups in terms of phase offset between the groups using the translation operator:

$$g[\hat{k},n] = \sum_{n'} \mathrm{T}_L[\hat{k},\rho_{nn'}]s[\hat{k},n']. \tag{3.4.2}$$

This is done for all source clusters and summed onto each target cluster over all *k*-space directions. Notice again that the expansion is about the element itself (index *n*) rather than individual bases within the array elements. In this notation, $g[\hat{k},n]$ represents the contribution of all source elements at the location of the testing cluster *n* in the direction $\hat{k}$. For clusters that conform to a regular grid; with consecutive cluster indexing in each dimension, this process is convolutional, and can be alternatively performed as

$$g[\hat{k},n] = \mathrm{T}_L[\hat{k},n-n'] * \{s_{n'}\}_{\hat{k}}, \tag{3.4.3}$$

assuming the general translation matrix $\mathrm{T}_L[\hat{k},n-n']$ has been stored in an equivalent block-Toeplitz format, as in (3.2.4). This procedure can be expedited through simultaneous convolutions for each of the *k*-space directions. Alternatively, when the process is convolutional, it may be possible to accelerate the interaction explicitly with the FFT, via the procedure

$$g[\hat{k},n] = FFT^{-1}\left\{FFT\left\{\left[\mathrm{T}_L\right]_{\hat{k}}^{T}\right\} FFT\left\{\{s_{n'}\}_{\hat{k}}\right\}\right\}, \tag{3.4.4}$$

where $\left[\mathrm{T}_L\right]_{\hat{k}}^{T}$ is the slice of the Toeplitz translation matrix for *k*-space direction $\hat{k}$. If using an FFT is the intention, it is most practical to pre-compute the quantity $FFT\left\{\left[\mathrm{T}_L\right]_{\hat{k}}^{T}\right\}$ prior to the matrix-vector product operation as well. If the quantity $\left[\mathrm{T}_L\right]_{\hat{k}}^{T}$ is no longer needed for computations, then $FFT\left\{\left[\mathrm{T}_L\right]_{\hat{k}}^{T}\right\}$ can be computed in its space at no additional storage cost. The size and dimension of the FFT depends on the way in which the problem was partitioned for clustering. Note that the notation shown here is specific to the form of a one-dimensional cluster grid, with the implicit generalization

being applicable to two- or three-dimensional grids. The contribution of each source element group as excitation on each target element group is then attained via the step

$$\{b_m\}_n^{far} = \int W_n[\hat{k}, m]\left\{g_{\hat{k}}\right\}_n dk\Omega . \tag{3.4.5}$$

The quantity computed in (3.4.5) is the FMM contribution to the matrix-vector product operation of the iterative solution procedure. In summary, the entire solution procedure including the near-zone interactions, can be summarized concisely as

$$\{b\} = ([A]_{near} + [A]_{far})\{x\}^{est} . \tag{3.4.6}$$

In this expression, $[A]_{near}$ represents the now sparse near-zone impedance matrix from the original formulation, and $[A]_{far}$ constitutes the above FMM procedure for calculating the interaction of distant cluster interactions.

3.4 Results and Examples

In this section, the benefits of FMM over conventional FE-BI are demonstrated. As a first example, the array configurations at the end of Chapter 2 are again analyzed. The comparisons are given in Table 3.1, and discussed below. For the analysis, the same single processor HP rx4610 Itanium server from Chapter 2 is used.

Table 3.1. Comparison of Conventional FE-BI to FMM for Sample Array Configurations.

Array Size		Standard FE-BI	FMM
3×3 9,927 Unknowns	Storage Requirements	368MB	161MB
	Fill Time	7.5min.	5.7min.
	Solve Time	3.7hrs.	6.1hrs.
6×6 39,708 Unknowns	Storage Requirements	5,880MB	802MB
	Fill Time	1.9hrs.	1.0hrs.
	Solve Time	3.1days	3.0days
10×10 110,300 Unknowns	Storage Requirements	45,348MB	3,589MB
	Fill Time	*16hrs.	11hrs.
	Solve Time	-	-

*estimated

For these comparisons, the ILU(0) preconditioner was used. This preconditioner uses the same amount of storage as the system matrix for computing its preconditioning data. Hence, for the sparse matrices associated with the FMM, ILU preconditioning with zero fill-ins can be most practical. For less sparse matrices, the ILU(0) preconditioner is more expensive, and for a completely full integral equation matrix, ILU(0) is the equivalent of a complete LU decomposition.

Starting with the 3×3 array, it is clear that the matrix storage has been significantly reduced by the FMM. However, it is also clear that the additional overhead of the FMM for this small problem has caused the solution time to increase somewhat. In fact, because of the overhead associated with the FMM algorithm, it requires a problem of the 6×6 array size before solutions speeds of conventional FE-BI and FMM accelerated FE-BI break even. That is, for the 6×6 array, both the conventional FE-BI and the FMM solution procedures took roughly 3 days.

Next, recall that with conventional FE-BI, it was not possible to attempt analysis of the 10×10 array, because the required storage was too large. With FMM, it is possible to analyze this structure, as it only requires 3.5GB of matrix storage. However, the combination of inferior matrix preconditioning and degrading matrix condition leads to a situation where even with FMM, the problem does not solve in over a week of continuous run time. That is, the solution process oscillates, diverging rather than converging. Hence, even with a reasonable answer to the matrix storage issue through the FMM, one is still faced with potential solution difficulties due to matrix condition.

This discovery prompts a brief discussion on parallel implementations of common codes. Because the condition of matrix systems tends to get worse with increasing system size, there will always be a limit on the maximum system that can be solved in a given class of problems, using a given class of iterative solution procedures, independent of system resources. It is common practice to use parallel, distributed memory networks to divide the solution procedure across multiple systems to share in the solution process. It is incorrect to surmise that this distribution process can be continued indefinitely, to address larger and larger problems with the same code. This is because there is an independent solution criterion based on the system condition that must be considered as well. To be effective, the distributed solution procedure must be combined with an improved solution approach that specifically addresses the matrix condition issue.

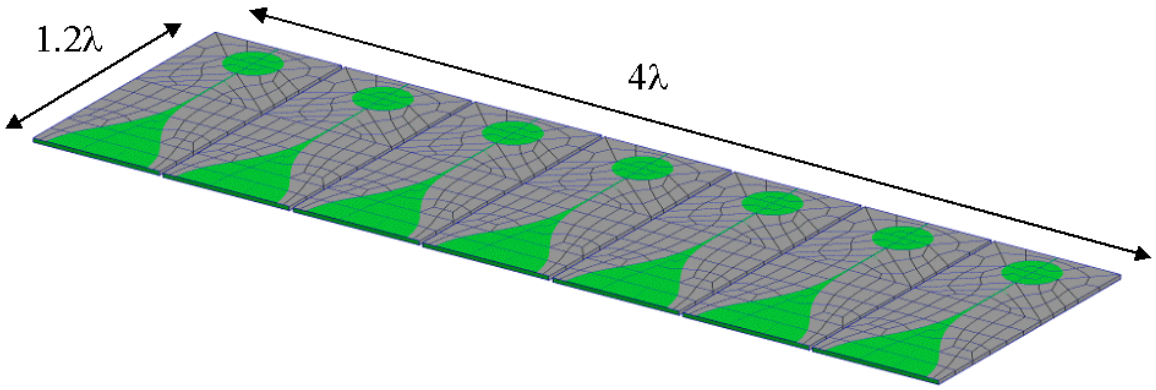


Figure 3.4. Seven-element array used for FMM comparisons.

The actual performance of the FMM will typically be problem dependent, and often implementation dependent as well. In this next example, the performance of the FMM-FFT algorithm will be evaluated. For this comparison, the seven element linear array shown in Figure 3.4 is used. The entire array structure is four wavelengths wide, with each element approximately 1.2λ in length. The unknown count is 5621, with 1617 FEM unknowns and 4004 BI unknowns. Typically, a structure of this size can expect approximately a 65% to 75% reduction in the BI storage of this problem via FMM. For the comparison, the same solver (BICGSTAB(L), L=8) from previous problems is used. These comparisons were run on a single processor AMD Athlon MP 1900+ desktop PC. The results of the comparison are listed in Table 3.2.

Table 3.2. Comparison of FMM Implementations for Seven-Element Array.

Seven-Element Array	Conventional FE-BI	Multi-level FMM	FMM-FFT
Matrix Storage	228MB	76MB	72MB
Iteration Count	32	31	31
Solution Time	5.7min.	5.8min.	2.5min.
Solution + Fill Time	6.6min.	6.6min.	3.2min.

Each of the solution approaches use the same exact matrix preconditioning, in which only the FEM portion of the matrix system is pre-solved via LU decomposition. Thus, aside from differences in numerical operation count and errors from approximating the Green's function for the FMM, the number of iterations for the solution and the residual error at each iteration should be the same for a conventional FE-BI or FMM accelerated analysis of the same problem. Notice that all the methods in Table 3.2 take roughly the same number of iterations to converge, though each method has a slightly different iteration count due to the aforementioned issues. As with the previous 3×3 array example, the MLFMM implementation has a slightly longer solution time than the conventional FE-BI, again due to the small problem and excessive overhead. However, the FMM-FFT implementation does show a solution speedup over both methods, even for this small problem.

For the FMM-FFT analysis, the structure was partitioned into a cluster grid of size $12\times1\times4$, corresponding to an FFT of size $24\times1\times8$. For this implementation, the FFT algorithm works with factors of 2, 3, 4, and 5 [30]. In general, while both the MLFMM and the FMM-FFT implementations feature $n\log n$ solution acceleration, the FMM-FFT implementation is more efficient for this type of problem, leading to greater solution

acceleration. For reference, the multi-level FMM can potentially achieve better storage savings for the same degree of accuracy. Further, the MLFMM has advantages when empty clusters exist. For this particular geometry, there are no empty clusters in the cluster grid, and hence the FMM-FFT algorithm performs ideally well. Again, the reason for interest in the FMM-FFT algorithm is that it is a natural choice to use in later chapters on more complex array-type problems.

3.5 Conclusion

In conclusion, the benefits and shortcomings of the fast multipole method for implementation within FE-BI have been discussed. The results of this chapter make it clear that advanced methods for array analysis will be necessary when the unknown counts reach into the millions, judging by the difficulty with tens of thousands of unknowns as attempted here. In the next chapter, a method specifically designed to exploit the redundancies in the near-zone interactions of finite-array problems is explored.

CHAPTER 4

ARRAY DECOMPOSITION METHOD

In the previous chapter, it was demonstrated that a generalized decomposition method such as the fast multipole method is not ideal for analyzing array-type problems when applied in an arbitrary manner. That is, while it was shown that the FMM achieved significant storage savings, the method did nothing to improve the condition of the matrix system, or shorten the solution process. Further, it stands to reason that a finite array-type problem consisting of regularly spaced, identical elements would have a great amount of redundancy that could be exploited in some way for improved analysis.

It has been recognized for some time that finite array interactions can be decomposed based on array element interactions for both solution as well as storage benefits. In this chapter, a general analysis method for FE-BI is introduced for efficiently decomposing the near-zone interactions of the finite array environment. This method decomposes the near-zone interactions of array elements based on the properties of translational symmetry and the spatial dependence of the free-space Green's function. The block-Toeplitz property of the resulting matrix system is exploited for reduced storage and solution acceleration with discrete Fourier transforms. The requirement for the proposed method is that elements be geometrically identical, insomuch that they are meshed exactly the same, and feature the same local basis element numbering. However,

excitation and impedance loading can be unique for each element. Compared to a conventional FE-BI analysis of the same problem with storage requirements of $O(n^2m^2)$, this method has $O(nm^2)$ storage requirements, where the number of array elements is given by n, and the number of unknowns per array element is given by m. The presented method preserves the framework of the FE-BI method for rigorously enforcing the coupling interactions between each of the individual array elements, and hence does not introduce any approximation into the solution. This near-zone decomposition method is referred to as the array decomposition method (ADM).

As pointed out in previous chapters, the exact treatment of the domain boundary with integral equations comes at increasingly large computational and storage requirements as the electrical size of the structure increases. In recent years, much attention has been given to fast techniques for reducing these costs. To review a bit of history, methods such as the fast multipole method (FMM) presented in the previous chapter, multilevel fast multipole method (MLFMM), as well as the adaptive integral method (AIM) have been shown to significantly reduce storage requirements of the BI matrix [1, 31-33]. These methods are only the most recent in a long history of techniques designed to accelerate conventional electromagnetic analysis, the first dating back as far as the early 1970s. Van Koughnett was perhaps the first to exploit the Toeplitz nature of array systems in CEM [34]. Around this same time, Bojarski was formulating scattering problems in *k*-space and applying fast Fourier transform (FFT) solution methods using early iterative techniques [35]. Shortly thereafter, Preis exploited Toeplitz matrix properties in conjunction with antenna problems and suggested some rapid inversion algorithms [36]. These *k*-space methods later evolved into the Spectral Iteration

Technique (SIT) in the late seventies [37-39], and the method apparently continues to be used today [4]. The conjugate gradient-FFT (CG-FFT) method was introduced later and allowed for more robust convergence properties [40-46], albeit for simplistic planar structures. To this date, the CG-FFT method continues to be one of the fastest solution approaches for planar geometries.

Like any electrically large problem, large finite arrays require massive numbers of unknowns to model rigorously, making the task difficult. Typically, large arrays are modeled using approximate methods based on infinite array analysis [2, 3, 5, 6, 47]. These methods can be limited in accuracy, depending upon the size of the finite array and how edge effects are treated. More accurate approximation methods for large array analysis have been explored recently, combining asymptotic high frequency based truncated discrete Fourier transforms (TDFT) with the method of moments (MoM) to achieve drastically reduced unknown counts and fast solution for array problems [7]. However, rigorous port-to-port coupling calculations, as well as multiple, highly coupled systems in the near-zone of each other typically cannot be addressed using these approximate methods. Thus, a demonstrated need exists for a fast and rigorous means of modeling finite arrays for realistic problems.

In this chapter, a fast method for the accurate analysis of finite arrays of three-dimensional structures is introduced. Like conventional FE-BI methods, the formulation is rigorous and thus exactly models edge effects in finite arrays. Moreover, ADM exploits the translational symmetry of finite arrays and the free-space Green's function to reduce storage requirements and accelerate solution times via the FFT. In particular, like the earlier CG-FFT method, ADM implements the FFT to speed-up the solution process

while rigorously enforcing the boundary conditions in the spatial domain. Moreover, the resulting block-Toeplitz matrix equation from the unique extended FE-BI formulation in ADM is the basis to achieve significant storage savings for the costly near-zone interactions.

For accuracy, ADM uses FE-BI to rigorously model each array element. Consequently, it is demonstrated that the LU factorization of the FE-BI sub-matrix of an isolated array element can be used as a highly efficient block-diagonal preconditioner on the overall matrix system. This feature, exclusive to the unique expansion of the FE-BI matrix equation in ADM, results in a distinct solution advantage over earlier fast methods. In addition to performing orders of magnitude faster, it is shown that ADM achieves results that are indistinguishable from conventional FE-BI methods for array-type problems, thus benefiting from the same rigorous analysis of conventional FE-BI in a fraction of the time. Perhaps more importantly, ADM significantly extends the range of problems that can be rigorously treated with limited resources, making it possible to model complex array problems using even a simple 32-bit desktop PC.

4.1 Review of Conventional FE-BI

To understand how the ADM is implemented, it is helpful to summarize the underlying FE-BI formulation which was presented in Chapter 2. For simplicity, the exposition in this chapter is based on the same reliable FE-BI formulation used in the previous chapters of the thesis. Other formulations and basis functions can be employed if desired, since ADM is not specific to the modeling technique. As in the case of

conventional FE-BI, ADM benefits from using the finite element method (FEM) to model as much of the volumetric geometry as possible, as opposed to exclusive use of integral equations methods that require significantly more matrix storage. Below, the conventional FE-BI formulation is developed in the context of a single isolated array element, and then extended to the specific case of finite arrays.

Consider the coupling paths within an arbitrary structure such as the antenna element shown in Figure 4.1, or even an entire array of elements (treated arbitrarily), as shown in Figure 4.2. In any case, the generalized impedance matrix elements from the integral equations use the same indexing, and will be of the form

$$[a_{mm'}] = jk_0 \int_S \int_{S'} \bar{w}(\bar{r}_m) \bullet \bar{w}(\bar{r}_{m'}) \frac{e^{-jk_0|\bar{r}_m - \bar{r}_{m'}|}}{4\pi|\bar{r}_m - \bar{r}_{m'}|} dS'dS \,, \tag{4.1.1}$$

where m and m' refer to the testing and source basis within the structure, respectively.

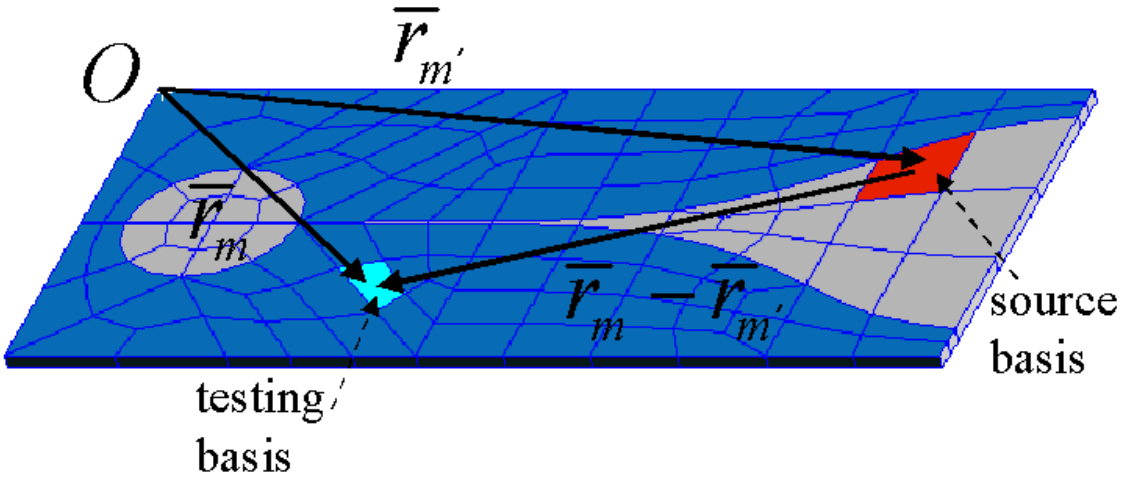


Figure 4.1. Illustration of source and testing basis interaction within a single element.

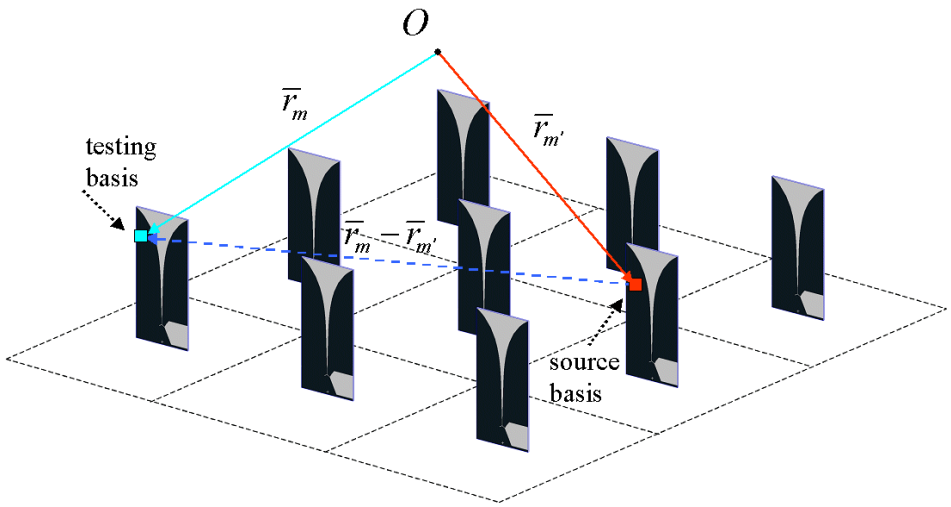


Figure 4.2. Interaction paths within array treated as arbitrary structure.

When treating array structures in this arbitrary manner, the formulation does not take geometric considerations into account, and the system of equations will always take the same general form. Recall from Chapter 2 that for this type of generalized interaction, the system of equations for conventional FE-BI can be compactly written as

$$\begin{bmatrix} A^{II} & A^{IS} & 0 \\ A^{SI} & A^{SS} & B \\ 0 & P & Q \end{bmatrix} \begin{Bmatrix} E^i \\ E^s \\ H^s \end{Bmatrix} = \begin{Bmatrix} b^i \\ b^s \\ b^e \end{Bmatrix}. \tag{4.1.2}$$

Regardless of the problem geometry, whether it is a single antenna or an array of antennas treated arbitrarily, the matrix assembly in conventional FE-BI will be organized as in (4.1.2). This is due to the generalized procedure that is used to apply the conventional FE-BI formulation to an arbitrary problem. The matrix system in (4.1.2) is non-symmetric, and convergence properties of the system can be classified as generally poor, especially for large problems, as evidenced by the results of the last two chapters. The perspective of this chapter is that an alternative assembly of the matrix operators, grouped by element interaction verses the operator type grouping of (4.1.2), will achieve

significant solution and storage benefits for array-type problems. This procedure is outlined in the next section, where the conventional FE-BI formulation is expanded for array-type problems to specifically exploit the translational symmetry of finite arrays.

4.2 Expanded FE-BI Formulation and Assembly for ADM

To implement the ADM, it is necessary to first expand the system of equations such that sub-matrix entities are grouped by element interaction, rather than operator type. Once the expansion has taken place, it will be possible to implement a beneficial decomposition of the matrix system. To facilitate this implementation, new notation based on array interactions is introduced. For simplicity, the exposition is based on linear arrays, for which the expansion is simple to visualize. This is done with the understanding that the concepts presented here, along with the notation, can easily be extended for planar and even multi-dimensional arrays. In a later chapter, the notion of multi-dimensional array analysis will be introduced to allow significantly more freedom in the approach to array-type problems and more specifically, multiple array-type problems. To begin, first consider the decomposition of a linear array. Let the indices n, n' represent the testing and source *array* elements, respectively, as depicted in Figure 4.3 for a simple two-element array. As a note, for the decomposition methods introduced here, it is necessary that the array elements be numbered consecutively in each dimension, and have a common lattice spacing for all elements in a given dimension. Within each element, basis functions are given the local index m. Because each element of the array is meshed exactly the same, the local basis function m of one array element

corresponds to the same local basis function m of all other elements in the array. For array interactions, the matrix impedance entries for integral equations can be expressed in the form

$$[a_{mm'}]_{nn'} = jk_0 \int_S \int_{S'} \overline{w}(\overline{r}_{m,n}) \bullet \overline{w}(\overline{r}_{m',n'}) \frac{e^{-jk_0|\overline{r}_{m,n} - \overline{r}_{m',n'}|}}{4\pi |\overline{r}_{m,n} - \overline{r}_{m',n'}|} dS' dS \,. \tag{4.2.1}$$

In the notation used here, the vector pointing from the global origin to the m^{th} basis function of the n^{th} array element is denoted $\overline{r}_{m,n}$. As an example, the notation $[a_{35'}]_{12'}$ implies that the fifth basis function (source) of array element two is being tested at the location of the third basis function of array element one.

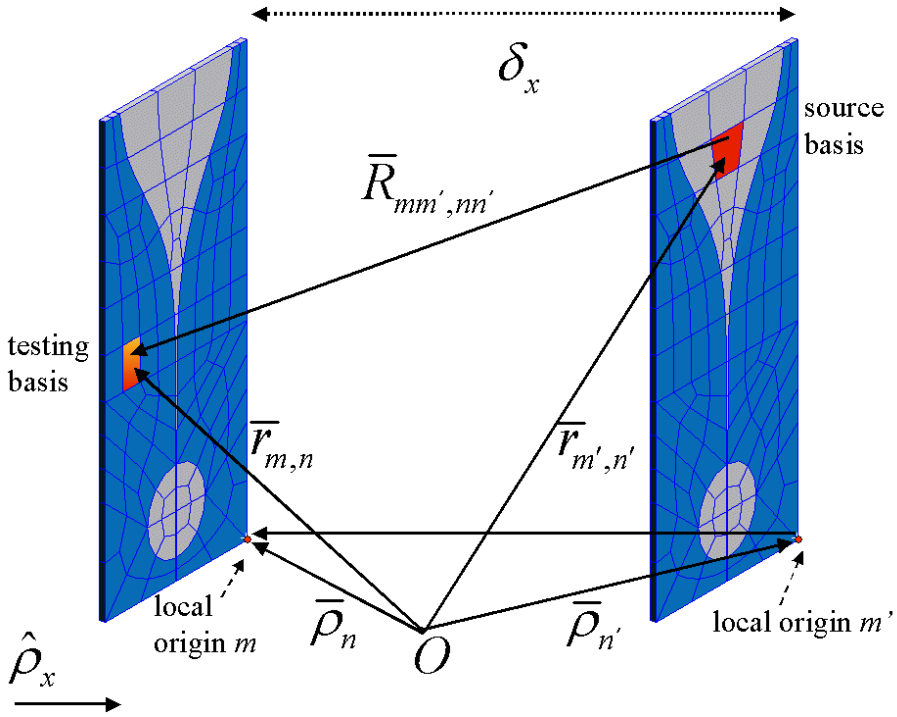


Figure 4.3. Expanded indexing for array interactions.

With this expanded notation, it is possible to reorganize the system of equations such that the coupling coefficients of the array are assembled into separate sub-matrices

for each array element interaction. Before generalizing this expanded notation, consider first the simple two-element array shown in Figure 4.3. For ADM, the expanded system of equations will be assembled in matrix notation as

$$\begin{bmatrix} \begin{bmatrix} A_{11'}^{II} & A_{11'}^{IS} & 0 \\ A_{11'}^{SI} & A_{11'}^{SS} & B_{11'} \\ 0 & P_{11'} & Q_{11'} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{12'} & Q_{12'} \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{21'} & Q_{21'} \end{bmatrix} & \begin{bmatrix} A_{22'}^{II} & A_{22'}^{IS} & 0 \\ A_{22'}^{SI} & A_{22'}^{SS} & B_{22'} \\ 0 & P_{22'} & Q_{22'} \end{bmatrix} \end{bmatrix} \begin{Bmatrix} E_1^i \\ E_1^s \\ H_1^s \\ E_2^i \\ E_2^s \\ H_2^s \end{Bmatrix} = \begin{Bmatrix} b_1^i \\ b_1^s \\ b_1^e \\ b_2^i \\ b_2^s \\ b_2^e \end{Bmatrix} \tag{4.2.2}$$

In this representation, the isolated domains of the individual array elements now form sub-matrices along the main diagonal of a larger matrix structure. Thus, all interactions within array element one (self-coupling) are represented with the upper left sub-matrix, while all interactions within array element two appear in the lower right sub-matrix. The isolated element domains are linked through cross-coupling sub-matrices, found in the upper right and lower left brackets of the matrix in (4.2.2). For example, array element two couples to array element one through the upper right sub-matrix, and array element one couples to array element two through the lower left sub-matrix. For the formulation presented here, a closed domain for each element is assumed (array elements do not touch) and thus no FEM terms appear in the mutual coupling sub-matrices. For cases where array elements share a common cell boundary, such as a continuous substrate where array elements are joined physically, it is necessary to define FEM operators to handle these junctions. Further, it is necessary to introduce bridge systems between the elements in order to preserve Toeplitz conditions. For this, a multi-system approach must

be adopted. This additional complexity will be treated in a later chapter, as it requires advanced concepts to model efficiently. In the formulation presented here, it is assumed that array elements do not touch, or if the elements do touch, the condition is treated as a crack, or bad electrical contact.

For ADM, the matrix system for an n element linear array takes the expanded form

$$\left[\begin{array}{cccc}
\begin{bmatrix} A^{II}_{11'} & A^{IS}_{11'} & 0 \\ A^{SI}_{11'} & A^{SS}_{11'} & B_{11'} \\ 0 & P_{11'} & Q_{11'} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{12'} & Q_{12'} \end{bmatrix} & \cdots & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{1n'} & Q_{1n'} \end{bmatrix} \\
\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{21'} & Q_{21'} \end{bmatrix} & \begin{bmatrix} A^{II}_{22'} & A^{IS}_{22'} & 0 \\ A^{SI}_{22'} & A^{SS}_{22'} & B_{22'} \\ 0 & P_{22'} & Q_{22'} \end{bmatrix} & \cdots & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{2n'} & Q_{2n'} \end{bmatrix} \\
\vdots & \vdots & \ddots & \vdots \\
\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{n1'} & Q_{n1'} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{n2'} & Q_{n2'} \end{bmatrix} & \cdots & \begin{bmatrix} A^{II}_{nn'} & A^{IS}_{nn'} & 0 \\ A^{SI}_{nn'} & A^{SS}_{nn'} & B_{nn'} \\ 0 & P_{nn'} & Q_{nn'} \end{bmatrix}
\end{array}\right]
\begin{Bmatrix} E^i_1 \\ E^s_1 \\ H^s_1 \end{Bmatrix} \quad \begin{Bmatrix} b^i_1 \\ b^s_1 \\ b^e_1 \end{Bmatrix}$$
$$\begin{Bmatrix} E^i_2 \\ E^s_2 \\ H^s_2 \end{Bmatrix} \quad \begin{Bmatrix} b^i_2 \\ b^s_2 \\ b^e_2 \end{Bmatrix}$$
$$\vdots \qquad \vdots$$
$$\begin{Bmatrix} E^i_n \\ E^s_n \\ H^s_n \end{Bmatrix} \quad \begin{Bmatrix} b^i_n \\ b^s_n \\ b^e_n \end{Bmatrix} \qquad (4.2.3)$$

which can be cast in more compact notation as

$$\begin{bmatrix} [a]_{11'} & [a]_{12'} & \cdots & [a]_{1n'} \\ [a]_{21'} & [a]_{22'} & \cdots & [a]_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ [a]_{n1'} & [a]_{n2'} & \cdots & [a]_{nn'} \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_2 \\ \vdots \\ \{x\}_n \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_2 \\ \vdots \\ \{b\}_n \end{Bmatrix} \quad \Rightarrow \quad [A]\{x\} = \{b\}. \qquad (4.2.4)$$

The impedance sub-matrices $[a]_{nn'}$ of (4.2.4) denote the individual coupling between the elements n and n', $\{x\} = \{\{x\}_1 \quad \cdots \quad \{x\}_n\}^T$ is a block-vector containing the vector field coefficients for each array element, and $\{b\} = \{\{b\}_1 \quad \cdots \quad \{b\}_n\}^T$ is also a block-vector containing the excitations of the array elements. In other words, each $\{x\}_n, \{b\}_n$ pair

contains the field and excitation coefficients of array element n, respectively. Notice that in this expansion, all the self-coupling terms (along matrix diagonal) are of the form

$$\begin{bmatrix} A_{nn'}^{II} & A_{nn'}^{IS} & 0 \\ A_{nn'}^{SI} & A_{nn'}^{SS} & B_{nn'} \\ 0 & P_{nn'} & Q_{nn'} \end{bmatrix}, \; n = n', \tag{4.2.5}$$

whereas the cross terms are the inter-element coupling matrices given by

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{nn'} & Q_{nn'} \end{bmatrix}, \; n \neq n'. \tag{4.2.6}$$

Hence, all the FEM information in the matrix structure now appears along the main block-diagonal of the matrix system. As will be described later, this can be exploited to reduce the number of iterations for system convergence quite significantly. The expanded system of equations is exact, and will certainly lead to the same exact solution obtained via the standard FE-BI formulation. In other words, (4.2.3) can be reorganized to the equivalent form of (4.1.2), or vice versa, and thus one can expect the same solution for ADM and conventional FE-BI. However, the organization as in (4.2.3) has unique benefits that allow ADM to achieve significant storage reduction and solution speed-ups, as described below.

Note that in (4.2.3), the system has been grouped by the physics of element interactions. This intuitive layout leads to the strongest self-coupling terms being located along the main diagonal of the matrix system with the nearest neighbor coupling terms adjacent to the main diagonal (for consecutively numbered array elements). This matrix restructuring induces a degree of symmetry, albeit 'block' symmetry, that (2.4.9) does not have. The matrix structure is block-Toeplitz, allowing for direct reduction of the dense

integral equation matrix storage from $O(n^2m^2)$ down to roughly $O(nm^2)$, with n being the number of array elements and m being the number of BI unknowns per array element. The way this decomposition is achieved is the subject of the next section.

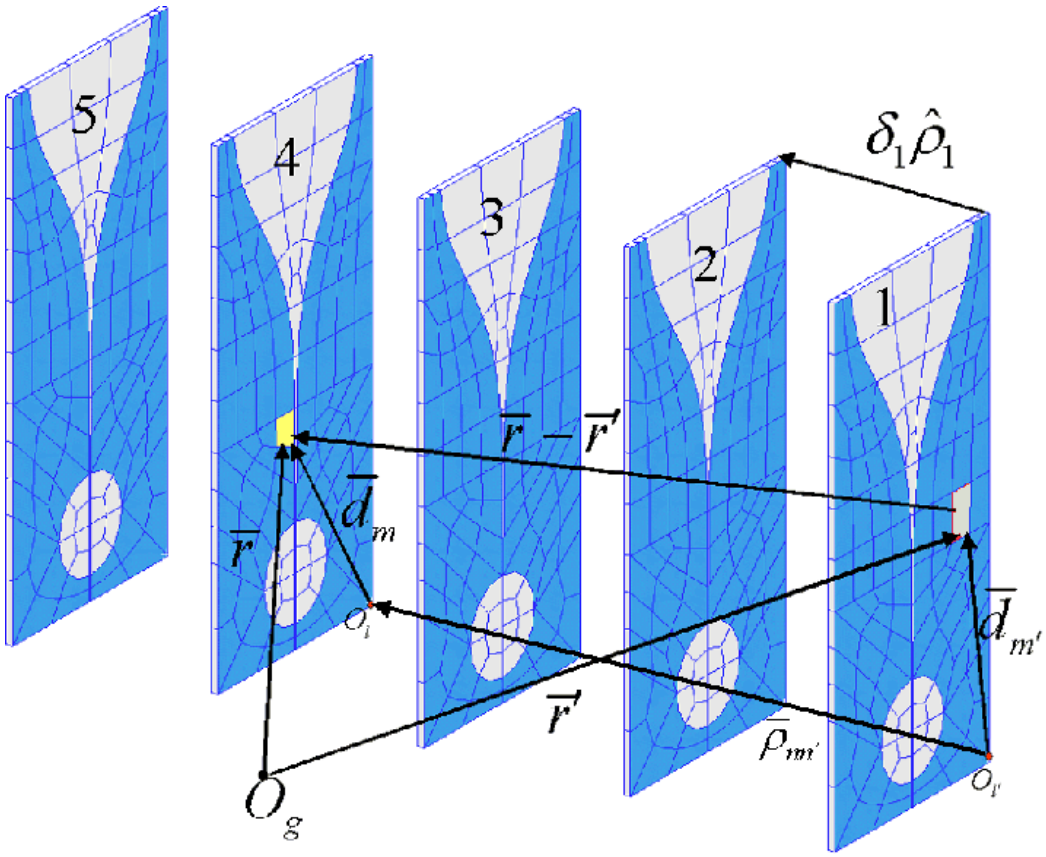


Figure 4.4 Expanded indexing for array decomposition.

4.3 Decomposition of Near-Zone Interactions

The expanded system of equations in (4.2.4) has a non-symmetric block-Toeplitz property, implying that a significant amount of the matrix information is redundant. Note that the relation in (4.2.1) is written in terms of vectors referenced to a global origin.

However, (4.2.1) can also be expressed in terms of local array element vectors and translation vectors relating the local origins of individual array elements, as depicted in Figure 4.4. Specifically, it is possible to re-write (4.2.1) as

$$[a_{mm'}]_{nn'} = jk_0 \int_S \int_{S'} \overline{w}(\overline{d}_m) \bullet \overline{w}(\overline{d}_{m'}) \frac{e^{-jk_0 \left| \overline{d}_m - \overline{d}_{m'} + \overline{\rho}_n - \overline{\rho}_{n'} \right|}}{4\pi \left| \overline{d}_m - \overline{d}_{m'} + \overline{\rho}_n - \overline{\rho}_{n'} \right|} dS' dS \,. \tag{4.3.1}$$

Consider now the consequence of a sequential numbering scheme imposed on the elements of the array. For a finite array with equally-spaced elements and consecutive numbering, it is possible to express the kernel function in terms of the difference between the source and testing array elements, or explicitly

$$g(\overline{r}_{m,n}, \overline{r}_{m',n'}) = \frac{e^{-jk_0 \left| \overline{d}_m - \overline{d}_{m'} + (i_x - i'_x)\delta_x \hat{\rho}_x \right|}}{4\pi \left| \overline{d}_m - \overline{d}_{m'} + (i_x - i'_x)\delta_x \hat{\rho}_x \right|}. \tag{4.3.2}$$

Equation (4.3.2) is the general form of the ADM kernel for finite arrays. In (4.3.2), δ_x is the fixed spacing between elements of the array, and $\hat{\rho}_x$ is the direction of the array axis (assumed along *x* for simplicity), and $\overline{d}_m$ is the vector from the local array element origin to the basis function *m* location. The pair i'_x, i_x are the sequential numbers of the source and testing elements, respectively. To be precise, (4.3.2) is only valid for linear arrays. The ADM kernel for two-dimensional arrays could equivalently be expressed as

$$g(\overline{r}_{m,n}, \overline{r}_{m',n'}) = \frac{e^{-jk_0 \left| \overline{d}_m - \overline{d}_{m'} + (i_x - i'_x)\delta_x \hat{\rho}_x + (i_y - i'_y)\delta_y \hat{\rho}_y \right|}}{4\pi \left| \overline{d}_m - \overline{d}_{m'} + (i_x - i'_x)\delta_x \hat{\rho}_x + (i_y - i'_y)\delta_y \hat{\rho}_y \right|}. \tag{4.3.3}$$

In a planar array, it is required that elements be numbered consecutively in both dimensions, and hence each element is given a pair of indices i_x, i_y, where $i_x = 1..n_x$,

$i_y = 1..n_y$, n being the number of elements in a given dimension. The kernel (4.3.2) can easily be used to create a unique and efficiently stored set of coupling coefficients.

To summarize, each element in an array of identical elements will have the same basis numbering within the local coordinate system. The coupling between any two elements of a linear array, as depicted in Figure 4.5, can then be described in terms of the identical local coordinate systems ($\overline{d}_m, \overline{d}_{m'}$), plus the separation between the local coordinate systems ($(i_d - i'_d)\delta_d \hat{\rho}_d$), for each dimension d. Consequently, the entire sub-matrix representation of the coupling from array element one to array element two, $[a]_{21'}$, will be the same as the coupling sub-matrix from element two to element three, $[a]_{32'}$, since the vector separating the origins of elements one and two ($\overline{\rho}_2 - \overline{\rho}_{1'}$) is identical to the vector separating elements three and two ($\overline{\rho}_3 - \overline{\rho}_{2'}$), or $\overline{\rho}_3 - \overline{\rho}_{2'} = \overline{\rho}_2 - \overline{\rho}_{1'} = (3-2')\delta_x \hat{\rho}_x = (2-1')\delta_x \hat{\rho}_x$. Clearly then, for sequentially numbered array elements, all unique coupling paths within the array can be represented by the difference in the array element indices, or $[a]_{nn'} = [a]_{(n-n')} = [a]_p$. In other words, for a linear array with sequential numbering of array elements, the coupling sub-matrices within $[A]$ have the property $[a]_{nn'} = [a]_{(n-n')} = [a]_p$, for any n, n' pair. Conceptually, the *sum* of the coupling seen at each element in the array is still unique, since the array is finite and the analysis methods exact. However, the isolated coupling contribution between any two elements of the array can have the same exact form. Because of this redundancy, it is possible to store only a sub-set of the terms in the matrix system without loss of information. As a side note, though one might expect a reciprocal property between $[a]_p$ and $[a]_{-p}$, in general, this is not the case when using CFIE and multiple

basis functions per array element. More specifically, the coefficients in $[a_1]$ and $[a_{-1}]$ are not *ordered* the same – the two matrices likely contain identical coefficients in a different order. For details on this aspect of the expansion, the reader is referred to Appendix A. In any case, from an implementation perspective, it is preferable to preserve both coupling matrices, such that the FFT can be used to efficiently accelerate the solution process.

The block-Toeplitz property of the ADM kernel results in only the first column and first row of coupling sub-matrices in (4.2.4) having unique values, indicated by the unshaded region in

$$\begin{bmatrix} [a]_{11'} & [a]_{12'} & \cdots & [a]_{1n'} \\ [a]_{21'} & [a]_{22'} & \cdots & [a]_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ [a]_{n1'} & [a]_{n2'} & \cdots & [a]_{nn'} \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_2 \\ \vdots \\ \{x\}_n \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_2 \\ \vdots \\ \{b\}_n \end{Bmatrix}. \tag{4.3.4}$$

The ADM kernel, as stated in (4.3.2), will only map to one of the unshaded sub-matrices of (4.3.4), and hence the shaded sub-matrices need not be computed or stored. For example, the coupling paths from source locations on array element one to testing locations on array element two $[a]_{21'}$ are the same as the paths from the corresponding source locations on array element two to testing locations on array element three $[a]_{32'}$. The same applies from element three to four $[a]_{43'}$, four to five $[a]_{54'}$, and so on. Of these, only $[a]_{21'}$ is stored, as it is in the first column of (4.3.4). The rest of the interactions get mapped to $[a]_{21'}$. Likewise, the coupling paths from array element one to

four $[a]_{41'}$ are the same as the corresponding paths from element three to six $[a]_{63'}$, and so on (only $[a]_{41'}$ is stored and mapped).

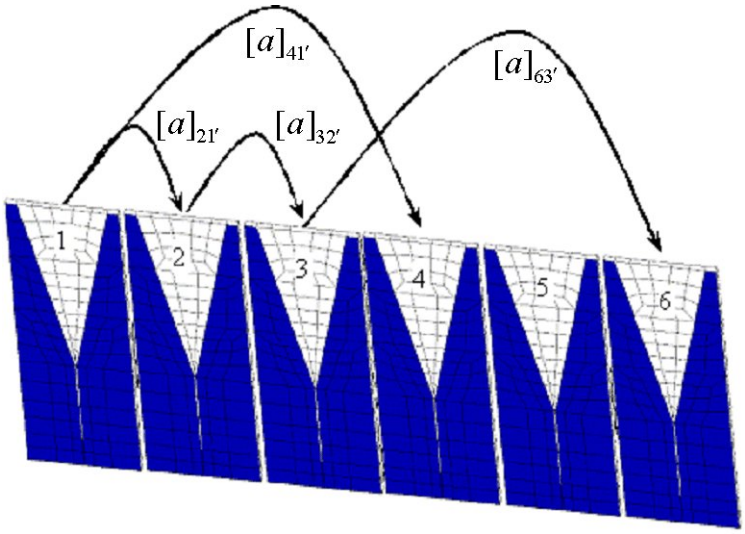


Figure 4.5. Illustration of redundant coupling paths in a 1×6 array.

In summary, for a n element linear array, the coupling terms for the entire matrix of equation (4.2.4) can be uniquely represented by a single block-vector of length $2n-1$ expressed as

$$\Pi^{1D} = \{[a]_{n-1} \quad \cdots \quad [a]_1 \quad [a]_0 \quad [a]_{-1} \quad \cdots \quad [a]_{1-n}\} \tag{4.3.5}$$

The superscript on Π indicates that the notation used in (4.3.5) is only appropriate for linear arrays. The terms in (4.3.5) are unique, and can be used to fully reconstruct (4.2.4), though there is no reason to do so.

A matrix-vector multiplication with a Toeplitz matrix is equivalently a discrete convolution operation. Consequently, the matrix-vector multiplication operation of (4.2.4) can be equivalently performed with the windowed convolution operation

$$\Pi^{1D} * \{x\} = \{b\} \tag{4.3.6}$$

Rather than representing a normal convolution of two sequences, in the general case, operation (4.3.6) represents a block-convolution operation with matrix-vector multiplications between the impedance sub-matrices of Π and the array element unknown vectors $\{x\}_n$. In other words, rather than a standard convolution with multiplication and sum operations between vector points, the block-convolution has *matrix-vector multiplication* and sum operations. To illustrate this more clearly, the matrix-vector multiplication for a two-element array as in (4.2.2) is equivalent to the expanded block-convolution operation

$$\{[a]_1 \quad [a]_0 \quad [a]_{-1}\} * \{\{x\}_1 \quad \{x\}_2\} = \{\{b\}_1 \quad \{b\}_2\}, \tag{4.3.7}$$

which can be broken down as

$$\{b\}_1 = [a]_0\{x\}_1 + [a]_1\{x\}_2 \qquad \{b\}_2 = [a]_{-1}\{x\}_1 + [a]_0\{x\}_2. \tag{4.3.8}$$

The above operation can easily be expanded to two-dimensional arrays. In this case, two indices are used to define the individual elements within the array. Thus, for a $n_1 \times n_2$ planar array, the unknowns are represented with the notation

$$\left[\{x\}\right]^{2D} = \begin{bmatrix} \{x\}_{1,1} & \{x\}_{1,2} & \cdots & \{x\}_{1,n_2} \\ \{x\}_{2,1} & \{x\}_{2,2} & & \{x\}_{2,n_2} \\ \vdots & & \ddots & \vdots \\ \{x\}_{n_1,1} & \{x\}_{n_1,2} & \cdots & \{x\}_{n_1,n_2} \end{bmatrix}, \tag{4.3.9}$$

where $\{x\}_{i_1,i_2}$ represents the unknowns on the array element at position i_1, i_2 of the array grid, with $i_1 = 1,2,...,n_1$, $i_2 = 1,2,...,n_2$. Again the Toeplitz property applies, and the coupling between the array element at location i_1, i_2 with the array element at location i_1', i_2' can be represented as $[a]_{(i_1-i_1')(i_2-i_2')} = [a]_{p_1,p_2}$. Thus, it is possible to conveniently store

the coupling terms for a planar $n_1 \times n_2$ array in a two-dimensional matrix with indexing based on the differences in the element indices as

$$\Pi^{2D} = \begin{bmatrix} [a]_{n_1-1,1-n_2} & \cdots & [a]_{n_1-1,-1} & [a]_{n_1-1,0} & [a]_{n_1-1,1} & \cdots & [a]_{n_1-1,n_2-1} \\ \vdots & \ddots & & \vdots & & \iddots & \vdots \\ [a]_{1,1-n_2} & & [a]_{1,-1} & [a]_{1,0} & [a]_{1,1} & & [a]_{1,n_2-1} \\ [a]_{0,1-n_2} & \cdots & [a]_{0,-1} & [a]_{0,0} & [a]_{0,1} & \cdots & [a]_{0,n_2-1} \\ [a]_{-1,1-n_2} & & [a]_{-1,-1} & [a]_{-1,0} & [a]_{-1,1} & & [a]_{-1,n_2-1} \\ \vdots & \iddots & & \vdots & & \ddots & \vdots \\ [a]_{1-n_1,1-n_2} & \cdots & [a]_{1-n_1,-1} & [a]_{1-n_1,0} & [a]_{1-n_1,1} & \cdots & [a]_{1-n_1,n_2-1} \end{bmatrix} \tag{4.3.10}$$

As a side note, this analysis uses the notation n_1, n_2, n_3, etc., for array dimensions in lieu of n_x, n_y, n_z to illustrate that the analyses are not limited to simple x-,y-,z- dimensions. In a manner similar to the operation in (4.3.6), for planar arrays it is possible to equivalently perform the matrix-vector multiplication with the windowed operation

$$\Pi^{2D} * \left[\{x\}\right]^{2D} = \left[\{b\}\right]^{2D} \tag{4.3.11}$$

This operation is a two-dimensional block-convolution involving matrix-vector multiplications between the coupling sub-matrices in (4.3.10) and the array element unknown vectors in (4.3.9). In the end, this operation will be performed with the FFT, so further elaboration on the block-convolution process is not necessary. The above can be expanded to treat three-dimensional grids as well, by introducing a third dimension for storage and performing three-dimensional convolutions. Visually, this is burdensome to represent. Consequently, in this chapter, consideration will be limited to the case of planar arrays. In implementation, however, ADM can be very conveniently extended to multiple dimensions.

Before considering the FFT acceleration of ADM, the savings achieved through the Toeplitz property will be considered. For ADM, there are $(2n_1-1)(2n_2-1)$ BI coupling terms (of cost $O(m_{BI}^2)$) to store for a $n_1 \times n_2$ array, as opposed to $n_1^2 n_2^2$ terms for conventional FE-BI storage. A similar property exists for one- and three-dimensional arrays. The total BI storage savings achieved by utilizing the Toeplitz property of the coupling interactions is tabulated in Table 4.1 for common array geometries. In these relations, $O(m_{BI}^2)$ is the storage cost for the boundary integral terms for a single array element. The FEM cost is considered separately below. The savings in Table 4.1 also apply to the time required to assemble the matrix (fill time), an additional cost that should not be overlooked.

Table 4.1. BI Storage Requirements for Conventional FE-BI vs. ADM.

Array Dimension	Conventional BI Storage	ADM Storage
$n \times 1$	$O(n^2 m_{BI}^2)$	$O(n m_{BI}^2)$
$n_1 \times n_2$	$O(n_1^2 n_2^2 m_{BI}^2)$	$O(n_1 n_2 m_{BI})$
$n_1 \times n_2 \times n_3$	$O(n_1^2 n_2^2 n_3^2 m_{BI}^2)$	$O(n_1 n_2 n_3 m_{BI})$

To evaluate the FEM savings, it should be pointed out that the sub-matrices along the diagonal of the main matrix in (4.2.3) represent the isolated domains of individual array elements. Consequently, since all array elements are represented identically, the sub-matrices on the diagonal have identical operator coefficients, and only one block of the matrix diagonal will be stored (because of the block-Toeplitz storage). Since the sub-

matrices along the diagonal include all the FEM terms of the system, it is possible represent the FEM portion of the entire array with the storage requirements of a single array element. Table 4.2 summarizes the FEM storage savings of ADM for various array geometries. The FEM storage cost for a single array element is proportional to $O(m_{FEM})$. Again, these savings apply to matrix fill time as well. For implementation purposes, it is more efficient to store the FEM and BI terms in separate storage structures, and only apply the FFT on the BI structures. There is no advantage in applying the FFT to the FEM terms.

Table 4.2. FEM Storage Requirements for Conventional FE-BI vs. ADM.

Array Dimension	Conventional FEM Storage	ADM Storage
$n \times 1$	$O(nm_{FEM})$	$O(m_{FEM})$
$n_1 \times n_2$	$O(n_1 n_2 m_{FEM})$	$O(m_{FEM})$
$n_1 \times n_2 \times n_3$	$O(n_1 n_2 n_3 m_{FEM})$	$O(m_{FEM})$

4.4 ADM Matrix Preconditioning

It has been suggested above that significant storage and fill time reduction is achieved by exploiting the repeatability of the array elements and the resulting block-Toeplitz matrix structure. Consideration is now given to how the solution process can be improved through matrix pre-conditioning. As pointed out earlier, the block-diagonal terms of the matrix in (4.2.4) consist of the FE-BI sub-matrices of isolated array elements. Given the relatively small size of $[a]_{11'}$ in the context of an array problem, performing an LU factorization on this sub-matrix comes at comparatively minor

computational and storage cost. In the implementation of ADM, the complete LU factorization of a single array element is used as a block-diagonal preconditioner for the larger finite array matrix system. It will later be demonstrated that this block-diagonal preconditioner, based on the physics of the problem, is highly effective in reducing solution time. It is important to note that this unique preconditioning approach is only possible due to the particular block-Toeplitz organization of the system of equations in ADM. That is, it cannot be applied to general systems of the type (4.1.2).

4.5 Matrix-Vector Product Acceleration

Having discussed storage savings and preconditioning of the matrix, the means by which the Toeplitz property of the matrix system can be used to accelerate the matrix-vector product operations is now considered. As shown in (4.3.6) and (4.3.11), the matrix-vector product can be formulated as a block-convolution operation. Since this is a discrete convolution operation, it is not necessary to increase the size of the FFT beyond the dimension of the matrix information already stored. However, it should be noted that rather than storing only $(2n-1)$ or $(2n_1-1)(2n_2-1)$ terms for linear and planar arrays, respectively, resulting in odd-dimensioned data arrays, it is possible to instead use $(2n)$ or $(2n_1)(2n_2)$ memory locations (or some other desired product of multiples) for improved FFT performance at slightly higher storage cost. In this implementation, a mixed-radix FFT is implemented that works with multiples of 2, 3, 4, and 5 [30]. Implementation aside, it is important to note that the problem at hand is formulated and

solved in the spatial domain - the discrete FFT is simply used to accelerate the matrix-vector product.

Acceleration of the matrix-vector product operations via the FFT requires three steps. First, the FFT is performed on the BI portion of the system matrix, as it appears in (4.3.5) or (4.3.10). This is a pre-computed quantity, denoted

$$\tilde{\Pi} = FFT\{\Pi\}. \tag{4.5.1}$$

For linear arrays, this FFT operation will be a one-dimensional block-FFT operation. To demonstrate the concept of a block-FFT operation, the $[a]$ terms of Π are represented with the indices $[a_{kj}]_p$, where $k, j = 1, 2, ..., m$ are the indices of the BI terms within the $[a]$ sub-matrices and $p = n-1, n-2, ..., 1-n$, the number of $[a]_p$ terms in Π of (4.3.5). The operation in (4.5.1) can be carried out with m^2 simultaneous one-dimensional FFT operations of length $(2n-1)$ on dimension p of Π. As mentioned above, it is preferable that this operation be performed on the BI terms only, so that additional storage is not required for transforming the FEM terms. Similarly, for planar arrays, it is necessary to perform a two-dimensional block-FFT on the coupling equation (4.3.10). The operation in (4.5.1) is performed only once, prior to solving the system of equations, and need not be performed with each matrix-vector product operation.

The next step is to apply the FFT to the unknown vector $\{x\}$, creating a block-vector of length $2n$-1 – the same length as $\tilde{\Pi}$, giving the spectral representation of $\{x\}$ denoted as

$$\{\tilde{x}\} = FFT\{\{x\}\}. \tag{4.5.2}$$

By representing $\{x\}$ with the indices $\{x_j\}_i$, where $i = 1,2,...,n$, the indices of the array elements and $j = 1,2,...,m$, represents the element unknowns, the operation in (4.5.2) can be carried out with m simultaneous one-dimensional FFT operations, extending the length to $2n-1$. Likewise, for a planar array of dimension $n_1 \times n_2$ a corresponding two-dimensional block-FFT must be performed on the dimensions n_1, n_2 of the matrix of unknown vectors (4.3.9), extending the data size to $[2n_1 - 1, 2n_2 - 1]$. Compared to (4.5.1), this is an inexpensive operation.

In the case of one-dimensional arrays, the next step is the consolidation operation

$$\{\tilde{z}_k\}_p = \sum_j^m \tilde{\Pi}_{kj(p-n)} \{\tilde{x}_j\}_p \,, \tag{4.5.3}$$

for each point $k = 1,2,...,m$, $p = 1,2,...,2n-1$, of $\{\tilde{x}\}$. Similarly, for planar arrays the operation

$$\{\tilde{z}_k\}_{p_1,p_2} = \sum_j^m \tilde{\Pi}_{kj(p_1-n_1)(p_2-n_2)} \{\tilde{x}_j\}_{p_1,p_2} \tag{4.5.4}$$

is performed. The final result of the matrix-vector product operation is obtained via the inverse transform operation

$$\{b\} = W(FFT^{-1}\{\{\tilde{z}\}\}) \,, \tag{4.5.5}$$

where W is a windowing function to extract the needed terms from the inverse FFT operation. The windowing function will depend on how the resulting data from the FFT operation is ordered, as this depends on the specific FFT implementation. Thus, for the BI terms, each matrix-vector product operation will consist of steps (4.5.2), (4.5.3) or (4.5.4), followed by (4.5.5). As a special note, in the case where a single expansion function per array element is employed with no FEM, ADM reduces to a limiting case

similar to the CG-FFT method, in which (4.5.1), (4.5.2), and (4.5.5) are standard point-by-point FFT operations, and $\{\tilde{z}\}$ is simply $\{\tilde{x}\}^T \tilde{\Pi}$.

As a solution cost analysis, the operation count of ADM will be examined. But first, using common array notation, the cost of conventional FE-BI can be stated as $O(n^2 m^2)$. For ADM, the block-Toeplitz matrix must be converted to the spectral domain using m^2 FFT operations of cost $2n-1$. However, this is a one-time procedure that should not be considered in the context of solution cost (consider it a one-time overhead cost). During the actual solution procedure, the data vector is converted to spectral domain with m FFT operations of cost $2n-1$. This is followed by nm^2 multiply-sum operations. Finally, there are m inverse FFT operations of cost $2n-1$. This requires $2m$ FFT operations plus nm^2 multiply-sum operations, for a cost of roughly $O(mn\log n + nm^2)$. Though it is hard to make a direct comparison of the solution cost, for array-type problems the n^2 terms are the most costly, none of which exist for the ADM. In general, ADM can be considered an $O(n\log n)$ method. However, it is more instructive to simply apply the methods to array analysis and observe the direct comparison.

4.6 Results

In applying ADM to practical problems, again consider the arrays of the TSA element shown in Figure 2.18. The tested array configurations for this section are listed in Table 4.3. The electrical volume of the arrays range from λ^3 to $400\lambda^3$, and the

unknown counts range from a few thousand up to nearly a million. In the example configurations presented here, ADM is compared to the conventional FE-BI only. In a later section, it will also be compared with FMM and other advanced array methods.

Table 4.3. Tested TSA Array Configurations.

Array Size	2×2	4×4	6×6	8×8	16×16	30×30
FEM Unknowns	2,028	8,112	18,252	32,448	129,792	456,300
BI Unknowns	2,384	9,536	21,456	38,144	152,576	536,400
Total Unknowns	4,412	17,648	39,708	70,592	282,368	992,700

As a first step, the proposed ADM is validated using the standard FE-BI method, in order to verify that the method generates the same valid results. The solutions achieved via both methods can be compared by computing the error

$$\frac{\left\|\{\boldsymbol{E}\}^{FEBI}-\{\boldsymbol{E}\}^{ADM}\right\|_2}{\left\|\{\boldsymbol{E}\}^{FEBI}\right\|_2}= \frac{\sum\sqrt{(E_x^{FEBI}-E_x^{ADM})^2+(E_y^{FEBI}-E_y^{ADM})^2+(E_z^{FEBI}-E_z^{ADM})^2}}{\sum\sqrt{(E_x^{FEBI})^2+(E_y^{FEBI})^2+(E_z^{FEBI})^2}}, \quad (4.6.1)$$

where $\{\boldsymbol{E}\}$ is the resulting electric field vector at each node in the geometry. For both methods, the same iterative solver [24] was employed to achieve the solution within a specified margin of error. In this evaluation, the tolerance was set so that convergence is achieved when the residual error $\|\{b\}-[A]\{x\}\|_2 / \left\|\{b\}\right\|_2 < 0.01$, where $\{x\}$ is the solution estimate. Using this relation, it was determined that for a 6×6 array of tapered slot antennas, the resulting error between solutions is 0.63%, or less than the one percent

solution error. For further validation, Figure 4.6 compares the radiated field patterns for the 6×6 TSA array as computed at 10GHz using the standard FE-BI formulation and ADM. As observed, the far-zone patterns obtained from ADM match identically with the conventional FE-BI results. For comparison, Figure 4.6 also includes the typical finite array approximation results, generated using the 6×6 array factor times the pattern of a TSA element in an infinite periodic array. No edge treatment has been applied to the infinite array solution. This approximation is only instructive, and as expected, is seen to lead to significant inaccuracies in the sidelobe region, since unlike ADM, it assumes the same field distributions on all elements of the array.

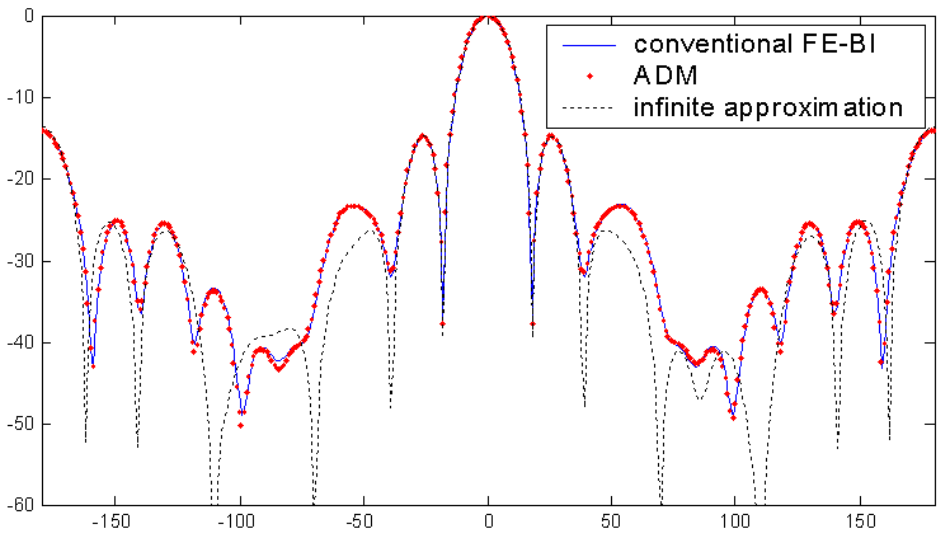


Figure 4.6. Validation of fields obtained from new formulation.

Next, the matrix fill and storage requirements of ADM are compared for the example array configurations. The results in this section were generated on an unloaded (no other processes running) 800MHz HP Itanium utilizing 16GB of shared memory and

a single processor, as in previous chapters. Although it is simple to implement parallel algorithms for both the matrix-fill and matrix-vector product operations in ADM, this capability was not enabled for the comparisons of this chapter. This avoids the additional parameter of evaluating the effectiveness of the parallel algorithms for each implementation. The large amount of shared computational resources on the Itanium makes it possible to analyze small finite arrays using conventional FE-BI. However, tapered slot antennas require significantly more unknowns per element to model accurately, and the required resources quickly escalate when approaching a 6×6 array using standard FE-BI. Consequently, a 6×6 array of the given element is the largest that can be analyzed on the Itanium with conventional FE-BI, as determined in previous chapters. Beyond this size, it is necessary to approximate the cost of the conventional FE-BI for array analysis. This estimation can be extrapolated fairly accurately using the relations of Table 4.1, since solution time is highly dependent on matrix system size. However, in reality, solvability of a matrix is not linearly based on the size of the matrix, as increasing matrix size also has an effect on the matrix condition. Table 4.4 compares the fill time and storage requirements of the standard FE-BI formulation and ADM for the example TSA array configurations. As expected, when the problem size increases, standard FE-BI methods quickly become impractical with respect to storage and fill time considerations. However, using ADM, it is possible to model TSA arrays well beyond 6×6 in size, even for arrays of electrically large TSA elements. In these examples, storage and fill time is reduced by a factor of 18 for an 8×8 array, and the savings for a 30×30 array is over two orders of magnitude. More specifically, for the 6×6 array, the storage is reduced from 6GB down to 500MB, and the fill time goes down from 47

minutes to only 4.5 minutes. In the 30×30 array case, the storage reduces from 3.8 terabytes down to only 16 gigabytes, and the fill time drops from 20 days to merely two hours. Thus, ADM is able to address problems well beyond the capability of standard FE-BI solvers, using modest resources. The *total* storage cost of ADM is even more considerable when compared with conventional FE-BI.

Table 4.4. Demonstration of Storage and Fill Time Savings.

Array Size	Storage (MB)		Fill Time	
	Standard FE-BI	ADM	Standard FE-BI	ADM
2×2	119	42	37 s	19 s
4×4	1,203	228	9.2 min	1.8 min
6×6	6,091	563	47 min	4.4 min
8×8	*19,251	1,046	*2.5 h	8.1 min
16×16	*308,020	4,469	*1.6 days	35 min
30×30	*3,807,000	16,225	*20.2 days	2.2 h

*Estimated result

The effectiveness of the proposed block-diagonal preconditioner is now evaluated. The iterative solver used in these examples computes two matrix-vector product operations per iteration [24]. A slight amount of memory and computational time is, of course, required to compute the preconditioner data. This is less than 9MB storage and 30 seconds computation time for all cases listed here. The FFT is not implemented in ADM for these results, and will be considered individually in the next section. Again, it is not possible to compute the solution times for array configurations beyond the available storage limits, and thus for arrays larger than 6×6 in size, it is necessary to extrapolate solution times. For the 6×6 TSA array (39708 unknowns), the

number of iterations reduced from 2600 down to only 10, and the corresponding solution time was reduced by a factor of 263, or more than two orders of magnitude. Beyond the 6×6, note that the solution time savings factors begin to taper off, an unexpected result, most likely due to limitations in the memory bandwidth of the Itanium. However, for the 30×30 array with nearly a million unknowns, the solution process would take an estimated *five years* for conventional FE-BI, assuming the solution procedure would even converge. This is in contrast to an estimated 82 days for ADM without FFT. A single iteration on this large problem took 18 hours for ADM without the FFT, and based on the number of iterations required for the ADM *with* FFT solution (116, see Table 4.5 below), that extrapolates to over 80 days to reach convergence.

Table 4.5. Effectiveness of Proposed Block-LU Preconditioner (no FFT).

Array Size	Iterations		Solution Time	
	Standard FE-BI	ADM no FFT	Standard FE-BI	ADM no FFT
2×2	2,488	4	50 min	7 s
4×4	2,400	7	13.8 h	2.6 min
6×6	2,600	10	3.1 days	17 min
8×8	*2,650	13	*9.8 days	1.3 h
16×16	*2,700	44	*157 days	2.6 days
30×30	*2,750	116	*5.3 years	*82 days

*Estimated result

In this section, the additional benefit of using the FFT to accelerate the matrix-vector product operations in ADM is evaluated. Two matrix-vector product operations are calculated per iteration in both cases, and the block-LU preconditioner is used in the BICGSTAB(L) iterative solver. The results are listed in Table 4.6.

Table 4.6. Evaluation of FFT Speed-ups for ADM.

Array Size	Solution Time	
	ADM Standard Matrix-Vector Product	ADM FFT Matrix-Vector Product
2×2	7 s	16 s
4×4	2.6 min	1.2 min
6×6	17 min	3.5 min
8×8	1.3 h	7 min
16×16	2.6 days	1.5 h
30×30	*82 days	16.4 h

*estimated result

Note that the overhead required to convert the matrix system to the spectral domain is not reflected in these results, as it is trivial in comparison to the overall solution times. The results clearly indicate a significant speed-up using the FFT for matrix-vector product calculations. In this case, for the 30×30 array, the solution time was accelerated from an estimated 82 days to half a day – an additional speed-up of over two orders of magnitude.

To evaluate the cumulative benefits of all ADM features, the storage and complete solution-time comparisons between standard FE-BI and ADM with block-diagonal-LU preconditioning and FFT matrix-vector product acceleration are summarized in Table 4.7.

Table 4.7. Combined Benefits of ADM.

Array Size	Storage (MB)		Total Solution Time	
	Standard FE-BI	ADM	Standard FE-BI	ADM
2×2	119	42	50 min	35 s
4×4	1,203	228	13.8 h	2.9 min
6×6	6,091	563	3.1 days	7.8 min
8×8	*19,251	1,046	*9.8 days	15.1 min
16×16	*308,020	4,469	*157 days	2.1 h
30×30	*3,807,000	16,225	*5.3 years	18.6 h

*estimated result

As seen from Table 4.7, even for the 6×6 TSA array there is an overall speed-up of nearly three orders of magnitude (8 minutes for ADM vs. 3.1 days with conventional FE-BI). For the 16×16 TSA array with 280,000 unknowns the solution time is only two hours as opposed to half a year (projected) with conventional FE-BI. As a problem size of one million unknowns is reached, as in the case of the 30×30 array, ADM achieves more than three orders of magnitude speed-up in solution time (5.3 projected years down to 18.6 hours), and the problem size is reduced from a unsolvable 3.8 terabytes to a more reasonable 16 gigabytes.

4.7 Conclusion

An extension of the FE-BI method for analyzing finite array structures was presented in this chapter. The proposed method, referred to as the array decomposition method (ADM), exploits the translational symmetry of finite arrays and the properties of the free-space Green's function to reduce storage and computation requirements without adversely affecting solution accuracy. The validity of the method was demonstrated by

comparing with the standard FE-BI formulation and the resulting storage savings were demonstrated and discussed. A preconditioning method exclusive to the matrix expansion of ADM was also proposed and demonstrated to be highly effective. The block-Toeplitz property of the matrix equation was used to accelerate the matrix-vector multiplication operations of the iterative solver with FFT methods. It was demonstrated that speed-ups of more than three orders of magnitude are achieved for practical modeling applications such as a 30×30 array (one million unknowns) of tapered-slot antennas, achieving results in hours instead of years.

ADM is appropriate for both scattering and radiation problems. In the analysis presented here, ADM is developed in conjunction with the hybrid FE-BI method, making it suitable for analyzing arrays of virtually any type of three-dimensional inhomogeneous structure. Moreover, ADM is ideal for applications that involve phased-array steering, excitation weighting, or port-to-port coupling, since it is rigorous and correctly treats edge effects in finite arrays. Perhaps most significantly, ADM significantly extends the range of applications that can be modeled on a desktop PC or workstation.

It is important to keep in perspective that the ADM presented here is only intended as a smaller piece of a much larger picture. ADM alone is rather limited in scope, being only strictly valid for freestanding arrays in free-space. Only when ADM is implemented as part of a larger picture can it truly be appreciated.

There are two main weaknesses that prevent ADM from being an ideal analysis method for finite array structures. The first weakness is that ADM is strictly a near-zone decomposition method. That is, all interactions within the finite-array environment are treated exactly using the rigor of near-zone integral equations. However, for distant

interactions within the array environment, this rigor is not necessary, nor hardly prudent. The rigor results in a linear increase in required system resources at the rate of $O(nm^2)$ for array size n. To overcome the linear increase in required resources, it is possible to neglect very far array interactions and still achieve accurate results with a controlled degree of approximation [48]. As a more viable alternative, in Chapter 6 the near-zone decomposition benefits of ADM are augmented with a far-zone decomposition approach for finite arrays. The combined near-zone/far-zone decomposition approach features the benefits of both ADM and FMM, while setting aside the main shortcomings of both methods alone. This new approach results in matrix storage requirements of $O(m^2)$ - the same as a single element analysis.

The other weakness of ADM that needs to be addressed is the $O(m^2)$ cost of the single element storage. For very large array elements, where the $O(m^2)$ cost is already prohibitive, it will not be possible to make much headway using ADM. However, in Chapter 7 it will be demonstrated that using the concepts of multi-dimensional analysis, it is possible to alleviate this cost considerably.

In the next chapter, the analysis methods presented here are extended into a generalized, multi-dimensional analysis method. Later, this development will make it possible to improve single-system analysis and also work more effectively with multiple systems.

CHAPTER 5

ARRAY DECOMPOSITION IN MULTIPLE DIMENSIONS

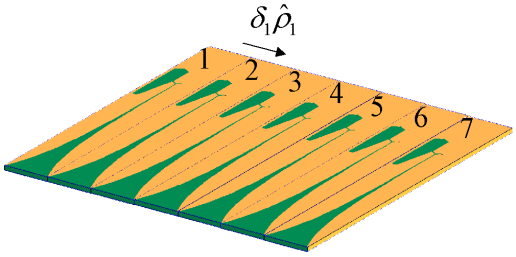
In Chapter 4, the array decomposition method (ADM) was presented for decomposing finite array analysis. In this chapter, the capabilities of ADM are expanded for multi-dimensional analysis. This chapter focuses strictly on dimensional analysis of the translation type. That is, the focus is on translational symmetry, where dimensions will be defined as vectors, having a direction (axis), and a spacing (magnitude). This distinction must be made to exclude consideration of rotational type dimensions - dimensions defined with an offset and angular spacing (for cylindrical surfaces).

This chapter is intended to give a brief introduction to multi-dimensional analysis, as the number of single-system configurations that can benefit from multi-dimensional decomposition is quite small. However, in Chapter 8 where simultaneous solution of multiple systems is considered, many benefits can be realized from the multi-dimensional approach.

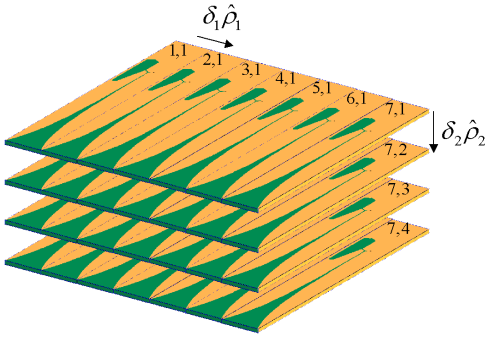
5.1 Introduction to Multi-Dimensional Analysis

Perhaps the most common type of finite array is the linear array, an example of which is shown in Figure 5.1 (a). Linear arrays are typically one-dimensional arrays, in

which the elements are placed on a regularly spaced lattice for a given dimension, typically along the *x*-axis, *y*-axis, or *z*-axis. However, in the general case, it is not necessary that linear arrays are one-dimensional. Likewise, the most common two-dimensional arrays are planar arrays, of the type shown in Figure 5.1 (b). Planar arrays are typically characterized by an array element with regular spacing in two orthogonal directions, say $\hat{x}$ and $\hat{y}$. Usually, the element spacing in each dimension is the same, though occasionally the spacing in each dimension may differ, or one dimension may be shifted to minimize co-polarized coupling (triangular or diamond-type lattices). However, generally speaking, planar arrays are not necessarily strictly two-dimensional, and two-dimensional arrays are not always planar. This is a subtle rule of general multi-dimensional array analysis. An example to explain these statements is given below.



(a) one-dimensional array



(b) two-dimensional array

Figure 5.1. Common array types.

Consider the array structure shown in Figure 5.2. This particular structure represents a common measurements setup in which the coupling may be studied between two arrays of the same antenna element type. Notice that this structure qualifies as a two-dimensional array, but happens to be linear as well. This example introduces the topic of this chapter, namely a generalized method for analyzing arrays using a single multi-dimensional specification. The class of problems appropriate for multi-dimensional analysis is characterized by a strict set of conditions. The first condition is that in any given dimension, elements must be numbered consecutively in a positive sequence starting with the same index for all dimensions and all systems. That is to say, in later chapters, where multiple systems are considered simultaneously, the element indices must all start with the number 1 and increase at the same rate for all systems (1, 2, 3, etc.). The second condition is that elements in each dimension must be spaced at regular intervals, i.e. non-uniform spacing cannot be accommodated. These two principles are fundamental to multi-dimensional decomposition.

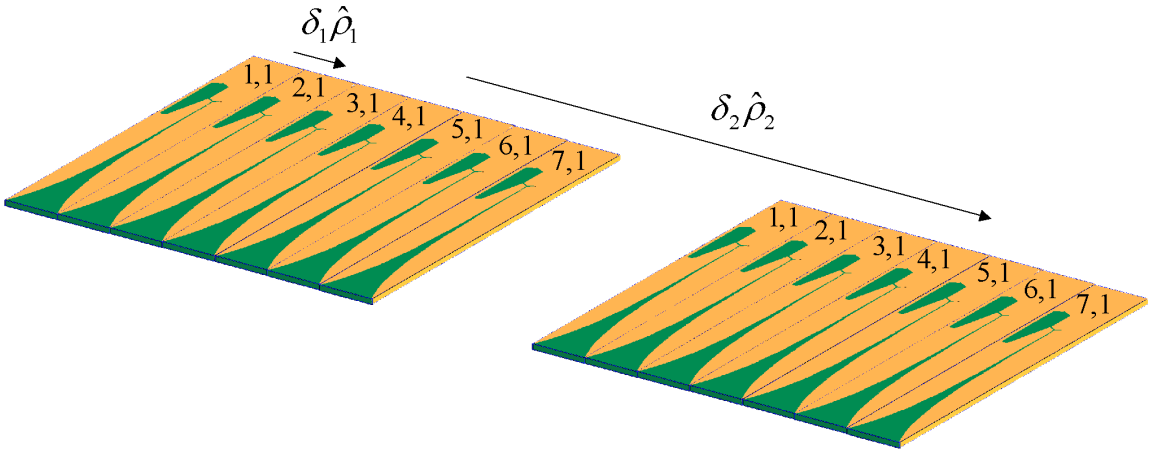


Figure 5.2. Two-dimensional linear array.

With these conditions in place, a general definition for array *dimension* can be given. For the class of problems considered in this thesis, a *dimension* is defined with a spacing parameter δ_d and a unit direction vector $\hat{\rho}_d$. The importance of these rigid conditions and definitions is that all array systems following these conventions will have a block-Toeplitz property for all dimensions, thereby allowing decomposition of array interactions across all dimensions simultaneously. Further, as strict as the conditions may be, they allow any number of directions and element spacings, with any possible combination thereof.

Similar to the kernel (4.3.2) given for the standard ADM, the multi-dimensional ADM kernel can be stated simply as

$$g(R) = \frac{e^{-jk_0\left|\bar{d}_m - \bar{d}_{m'} + (i_1 - i_1')\delta_1\hat{\rho}_1 + (i_2 - i_2')\delta_2\hat{\rho}_2 + \cdots + (i_{ndims} - i_{ndims}')\delta_{ndims}\hat{\rho}_{ndims}\right|}}{4\pi\left|\bar{d}_m - \bar{d}_{m'} + (i_1 - i_1')\delta_1\hat{\rho}_1 + (i_2 - i_2')\delta_2\hat{\rho}_2 + \cdots + (i_{ndims} - i_{ndims}')\delta_{ndims}\hat{\rho}_{ndims}\right|} \quad (5.1.1)$$

In this expression, the spacing of each dimension is given by δ_d, where d is the dimension number, and the direction is specified by $\hat{\rho}_d$. This kernel is used to form impedance matrix entries of the form

$$[a_{mm'}]_{nn'} = jk_0 \int_S \int_{S'} \overline{w}(\bar{d}_m) \bullet \overline{w}(\bar{d}_{m'}) g(R) dS' dS \,. \quad (5.1.2)$$

Note that the form of the basis function $\overline{w}(\bar{d}_m)$ and its evaluation are dependent only on the array element characteristics, and are completely independent of the kernel used in the coupling calculation. In general, multi-dimensional analysis conforms to simple rules of vector addition. The path of the source element influence to the testing location undergoes a transformation through each dimension, multiplying the spacing δ_d by the

difference in element indices $(i_d - i'_d)$ for dimension *d*, imposing an additional vector sum for each dimension of the array. Strict adherence to the rules outlined above results in a cascading block-Toeplitz matrix structure that can be maximally exploited to avoid calculation/storage of redundant coupling terms.

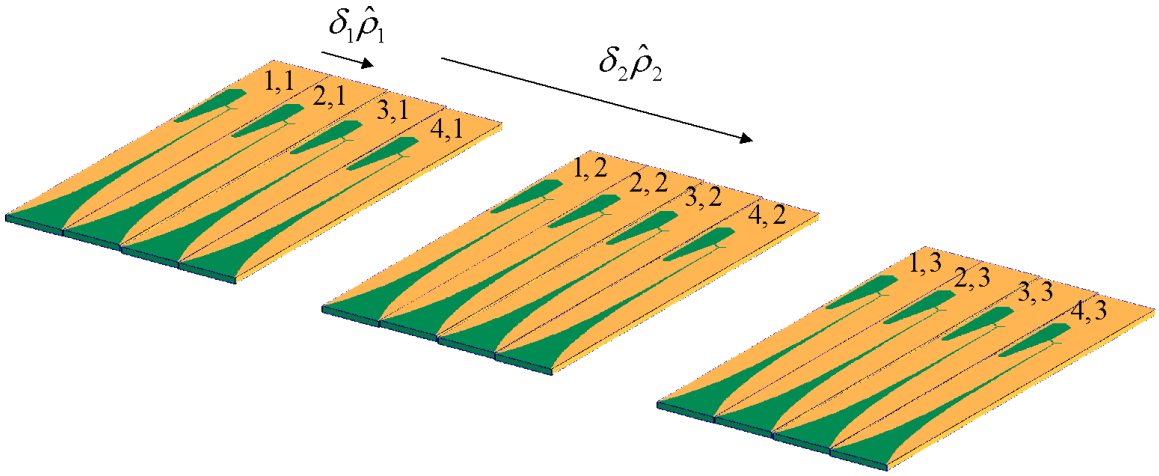
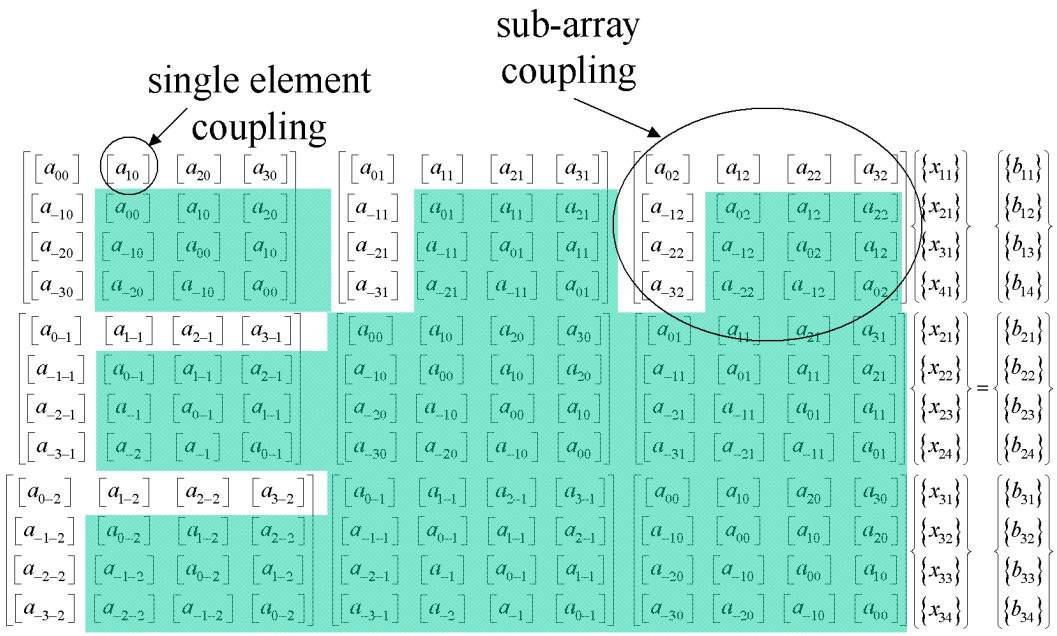


Figure 5.3. Decomposition illustration for two-dimensional array.

As an example, consider the 4×3 array shown in Figure 5.3. Essentially, the array is constructed from three four-element sub-arrays with some fixed separation. For this array, the form of the system is given as



(5.1.3)

In this expression, the matrix entries have been first arranged into coupling sub-matrices for individual element interactions, which in turn are arranged into sub-matrices for sub-array interactions. In (5.1.3), all redundant terms have been shaded, leaving only the unique interactions within the matrix system. It can be seen that only the first row and first column of each sub-matrix assembly is unique. That is, the first row and first column of the sub-array coupling blocks are kept, and at the lower level, the first row and first column of the individual element interactions (within the sub-array coupling systems) are kept.

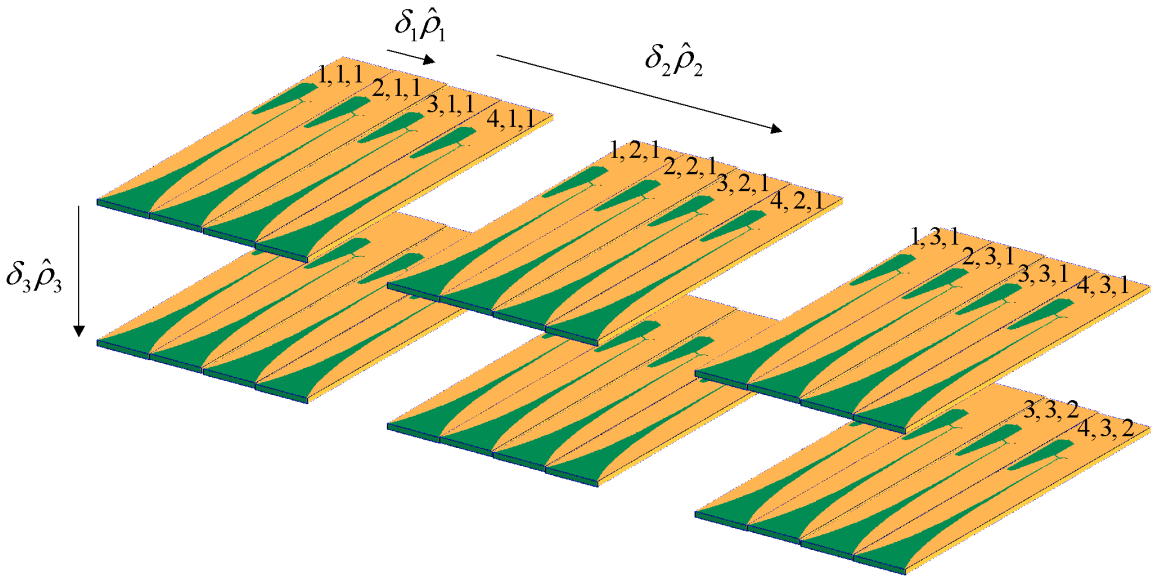
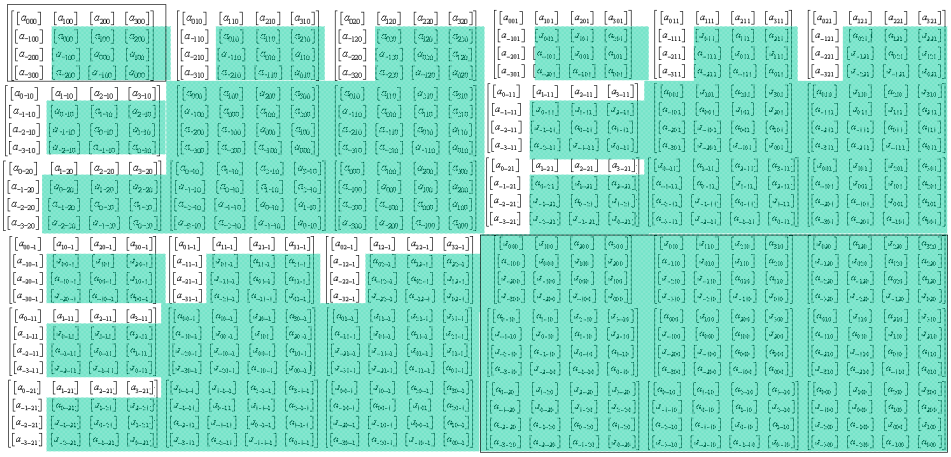


Figure 5.4. Decomposition illustration for three-dimensional array.

As a second example, consider an array structure of one more dimension than the one in Figure 5.3, shown here in Figure 5.4. This structure has been constructed simply by adding an additional dimension d=3 to the array in Figure 5.3. The coupling equation for this array structure will have the form



.(5.1.4)

Again, the redundant terms have been shaded out. The matrix system can be seen to consist of three levels of sub-array organization, for which only the first row and first

column at each level are preserved. For dimension 3, the lowest quadrant is shaded out. At dimension 2, the four lower right sub-matrix systems of each 3×3 assembly are shaded out. At level 1, where single element interactions are considered, only the first row and first column of element interactions need to be preserved. Though the form of this structure is complex, the kernel in (5.1.1) will only map to one of the unshaded areas of (5.1.3) or (5.1.4).

The complexity of this structure makes it much simpler to use a block-Toeplitz storage scheme. The matrix elements for a multi-dimensional block-Toeplitz system can be stored most efficiently and conveniently as simply

$$\left[a_{mm'}\right]_{(i_1-i_1')(i_2-i_2')\cdots(i_{ndims}-i_{ndims}')} = jk_0\int_S\int_{S'} \overline{w}(\overline{d}_m)\bullet\overline{w}(\overline{d}_{m'})g(R)dS'dS \qquad (5.1.5)$$

In other words, for the multi-dimensional system, the matrix entries, which consist of all sub-element (basis function) interactions between the source and testing array elements, are simply stored and accessed through the difference in their element indices for each dimension. This is entirely analogous to the case of simple arrays in the previous chapter. In this expression, there are no redundant terms.

The integral equation storage cost for a single system using multi-dimensional analysis can be precisely calculated as

$$\prod_{d=1}^{ndims}(2n_d-1)m_{BI}\,, \qquad (5.1.6)$$

where m_{BI} are the number of basis coefficients in a single array element for the integral equations, and n is the number of array elements in dimension d, for d=1..*ndims*, where *ndims* is the total number of dimensions. As in the previous chapter, the FEM storage

cost is simply $O(m_{FEM})$ for arrays of any number of elements/dimensions, where m_{FEM} is the number of FEM unknowns in a single array element.

5.2 Multi-Dimensional Analysis Examples

This chapter closes with a few examples demonstrating practical uses for multi-dimensional analysis. In later chapters, additional techniques will be developed that can benefit more from multi-dimensional analysis as well. Specifically, this topic will be touched upon again in Chapter 7, when elements within an array lattice are further decomposed to achieve additional acceleration, and also again in Chapter 8, when dealing with multiple systems.

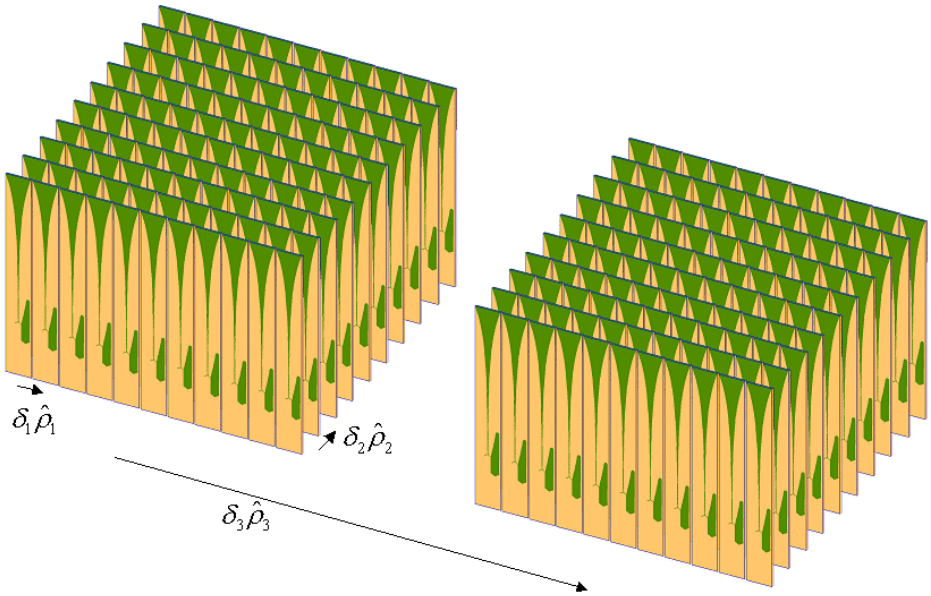


Figure 5.5. Two-array verses three-dimensional array problem.

A practical example of multi-dimensional analysis comes from considering a problem of the type shown in Figure 5.5. In this example, the coupling between two 10×11 arrays is considered. The element of the array is modeled with 857 FEM unknowns and 1500 BI unknowns. The configuration shown in Figure 5.5 has a total of 259,270 unknowns. At the system level, the impedance matrix for this problem will have the form

$$\begin{bmatrix} [Z_{11}] & [Z_{12}] \\ [Z_{21}] & [Z_{22}] \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_2 \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_2 \end{Bmatrix}. \tag{5.2.1}$$

In this expression, each $[Z_{ij}]$ represents the coupling from array j to array i. At first glance, this appears to be a straightforward analysis of two identical arrays. Per the arguments detailed in this chapter, this coupling system can also be formulated as a single three-dimensional array using the principles of multi-dimensional analysis. The primary advantage of formulating the problem as a three-dimensional array is that the required matrix storage is 3/4 of that required when formulated as a two-array configuration using ADM. This configuration would take 1,363GB of storage with conventional FE-BI. Treating this problem as a two-array configuration would require 30GB with ADM. Though the details of a two-system decomposition have not yet been presented, this topic will be addressed in a later chapter on multiple system analysis. However, treating this system as a single three-dimensional array would require only 22GB, exactly 3/4 that required for a two-system decomposition. These results are summarized in Table 5.1.

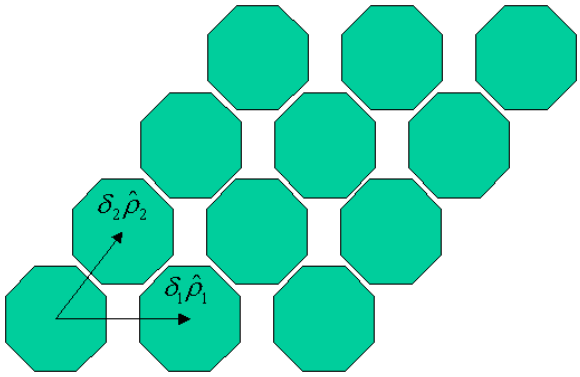
It is conceivable that there are circumstances when having a 22GB problem would be beneficial compared to the 30GB problem. However, even greater benefits from multi-dimensional analysis will be realized after the methods of the next few chapters are

developed. For the multi-dimensional analysis approach to work, it is only necessary that the second array not be rotated – it can be shifted or translated in any direction. In a later chapter, a similar but more complex array of this class will be considered. This later example will explicitly take advantage of the three-dimensional array formulation for this type of problem.

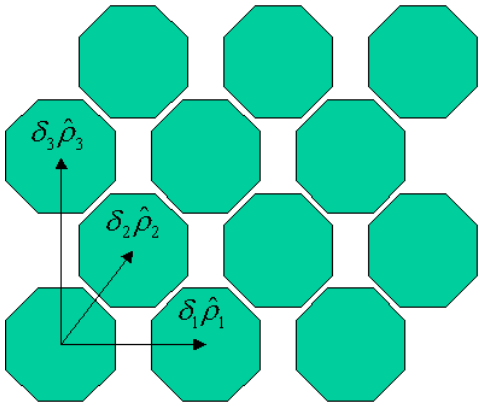
Table 5.1. Comparison of Storage Requirements for System in Figure 5.5.

Solution Method	Matrix Storage
Conventional FE-BI	1,363GB
Two-Array ADM	30GB
Multi-Dim ADM	22GB

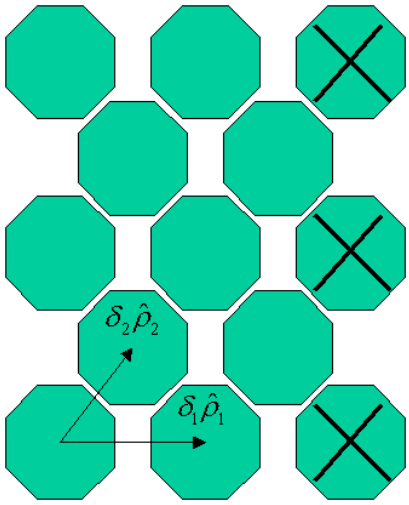
As a second practical example, consider the case of arrays with a triangular lattice, as shown in Figure 5.6. The array in Figure 5.6 (a) can easily be realized using a two-dimensional, planar array. However, the array in Figure 5.6 (b) is more challenging. In order to model this array and maintain the Toeplitz property of array element interactions, it is necessary to use a three-dimensional array description. In other words, the array in Figure 5.6 (b) can be analyzed as a $3 \times 2 \times 2$ array. This is a case of a planar array that is not simply two-dimensional. The array in Figure 5.6 (c) cannot be constructed as a single, multi-dimensional array. Hence, while a number of real-world array structures can be realized in a single, multi-dimensional description, it is often the case that a composite structure will require a multi-system approach for proper analysis – a topic addressed in a later chapter.



(a)



(b)



(c)

Figure 5.6. Array configurations with a triangular lattice.

5.3 Conclusion

In conclusion, a brief introduction to multi-dimensional analysis has been given. As mentioned, the utility of multi-dimensional analysis is limited for single array systems. However, when the topic of multiple system interaction is addressed, the benefits of multi-dimensional analysis will become more apparent.

It was demonstrated that multi-dimensional analysis makes is possible to analyze a two-array coupling problem as a single array problem of three dimensions. This capability was shown to reduce the cost of analyzing the system by 25% over a two-system interaction approach. In the next chapter, a method is presented that will further reduce the cost of array analysis to the point that finite array analysis will require the same fixed amount of storage for any number of elements in the same array class.

CHAPTER 6

ARRAY DECOMPOSITION-FAST MULTIPOLE METHOD

In previous chapters of this thesis, methods for rigorously analyzing large structures were examined. In Chapter 3, the fast multipole method (FMM) was presented and shown to be highly successful at reducing the cost of arbitrary structure analysis [1]. For arbitrary problems, the $O(N^{1.5})$ to $O(N \log N)$ memory requirements of FMM can make large problems solvable, compared to a conventional approach with $O(N^2)=O(n^2m^2)$ storage limitations (N=nm, m=array element unknowns, n=number of array elements, N = total number of unknowns). FMM reduces the near-zone matrix storage requirements by transferring the burden of distant interactions to the iterative solution process, thereby increasing the cost of each iteration. Because of this, an excessive number of iterations would lead to a negative impact on solution speed, a situation that can easily occur with a lack of effective preconditioning. Moreover, even though the $O(N^{1.5})$ to $O(N \log N)$ storage requirements of FMM are better than the $O(N^2)$ limitations of conventional methods, for increasingly large problems one is still faced with a worse than linear increase in matrix storage. Further, it was demonstrated at the end of Chapter 3 that FMM does not improve the condition of a matrix system. In other words, when large systems become ill-conditioned due to their size, FMM will not

be able to solve the problem, even if the sparse near-zone matrix system fits in available storage, or can be distributed across multiple systems. Hence, there are practical limits on the size of the problem that can be analyzed with FMM alone.

In this chapter, the focus is again on the analysis of a particular type of problem – standalone, finite array-type structures. In Chapter 4, ADM was shown to significantly reduce storage when used in combination with the finite element-boundary integral (FE-BI) method to model array elements. ADM is a near-zone decomposition method, featuring $O(nm^2)$ memory requirements, which exhibits a linear increase in required memory with problem size and therefore slightly better storage than FMM (for large problems and small m). ADM significantly increases solution speed by applying the fast Fourier transform (FFT) to the integral equation matrix operations, and further, demonstrates superior convergence rates (over conventional FE-BI) due in part to the block-symmetric matrix layout having a better disposition towards effective preconditioning schemes [49]. Though successful, this method has two main shortcomings. First, for large m (array element size), the storage may already be prohibitively large (i.e. the $O(m^2)$ individual element cost may be high). Secondly, for large arrays, the matrix storage continues to grow at a linear rate (per dimension), and thus is still a bottleneck for large array analysis, making the method nearly as impractical as FMM for very large array analysis.

The shortcomings of both FMM and ADM prevent them from being ideal array analysis tools. For array analysis, it would be ideal to combine the independent methods, ADM and FMM, into a single approach that maintains the good features of both methods while rejecting their shortcomings. This task is the theme of this chapter.

In this chapter, a hybrid approach to finite array-type problems is presented that combines the benefits of near-zone decomposition (ADM) [50] with the benefits of a far-zone decomposition (FMM) [51] for analyzing array-type problems. The chapter focuses specifically on applying the fast multipole method in a way that takes advantage of structural redundancies, much like the ADM from Chapter 4. The most obvious shortcoming of applying a generalized solution method such as FMM to an array-type problem is that this approach fails to exploit known redundancies in the problem geometry. Finite array-type problems consisting of regularly spaced and identical elements exhibit translational symmetry that can typically benefit from some form of decomposition approach. The analysis method presented here is innovative for several reasons. First, the combination of near-zone and far-zone decomposition for array-type problems has the effect of truncating the near-zone matrix storage. That is, the near-zone matrix size is fixed *for any sized array* of regularly spaced and arbitrary elements in the same class. This results in trivial matrix fill times, and for all practical purposes, matrix storage requirements that are on the same order as the storage of a single array element, viz. $O(m^2)$ compared with $O(n^2m^2)$ for conventional FE-BI. For large problems, overall storage is dominated by the solution vector, rather than the system matrix. That is, for a problem having an overall solution vector of length N, the *total* storage is limited to $O(N)$ or $O(nm)$. This represents a significant improvement over both ADM ($O(nm^2)$) and FMM ($O(N^{1.5}) = O(n^{1.5}m^{1.5})$). It is important to note that a distinction between *matrix* storage and *total* storage must now be made, since the matrix storage does not account for the majority of the storage as with previously considered methods. In addition, since each element of the array can be modeled with an identical unit cell,

this method only computes a single set of Fourier coefficients (signature functions) for the unit cell basis functions, as compared to the conventional FMM which arbitrarily computes coefficients for every array element at n times higher cost. Moreover, the far-zone element interactions (translations) have an inherent Toeplitz property corresponding to the array lattice that results in reduced far-zone storage and allows the FFT to be explicitly applied in accelerating the solution process, similar to what is done for the FMM-FFT of Chapter 3. Furthermore, this hybrid approach, to be referred to as the array decomposition-fast multipole method (AD-FMM), benefits from the same disposition towards effective preconditioning as ADM [50], resulting in solution convergence in relatively few iterations. Most importantly, AD-FMM is completely rigorous, as tuned by the FMM parameters, providing accurate solutions to large coupling problems with negligible resource requirements. It will be shown that AD-FMM elegantly combines the benefits of both ADM and FMM, avoids the main shortcomings of the methods, and features additional benefits not found in either method alone.

In this chapter, the details of how AD-FMM works are presented, with the storage and CPU requirements compared to the methods of previous chapters.

6.1 Near-Zone Decomposition

One of the main components of AD-FMM is the near-zone decomposition (ADM). For the most part, in AD-FMM the near-zone decomposition procedure is performed exactly as described in chapters 4 and 5. To summarize, the important benefit of the ADM is that the expanded system of equations has a block-Toeplitz structure,

resulting in only the first row and first column of the overall system matrix being stored, depicted here as

$$\begin{bmatrix} [a]_{11'} & [a]_{12'} & \cdots & [a]_{1n'} \\ [a]_{21'} & [a]_{22'} & \cdots & [a]_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ [a]_{n1'} & [a]_{n2'} & \cdots & [a]_{nn'} \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_2 \\ \vdots \\ \{x\}_n \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_2 \\ \vdots \\ \{b\}_n \end{Bmatrix}. \tag{6.1.1}$$

The form of (6.1.1) assumes a linear array of n elements, but can easily be expanded for the case of multi-dimensional arrays as detailed in Chapter 5. The shaded region represents the redundant coupling sub-matrices that are not stored. Unlike the ADM, for AD-FMM, there will be no FFT acceleration of the near-zone interactions. This is no longer necessary, as the majority of the array interactions, which are carried out entirely in the near-zone with ADM, will now be performed in the far-zone with AD-FMM. The remaining near-zone terms still have a block-Toeplitz property that is exploited for storage savings, but using the FFT on the small number of remaining near-zone terms in AD-FMM has no benefit. Once the far-zone decomposition has been applied, the required near-zone storage will shrink to nothing more than a few terms in the upper left-hand corner of (6.1.1) (self-element interactions and nearby element interactions).

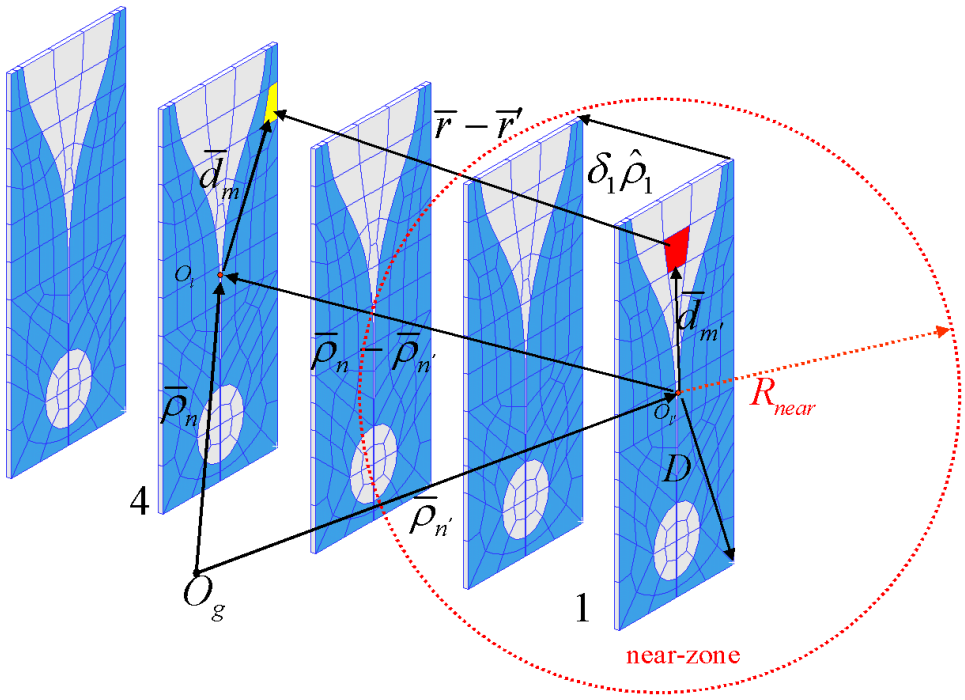


Figure 6.1. Depiction of element clustering grid for AD-FMM.

6.2 Far-zone decomposition

After introducing the same near-zone decomposition on the array structure, a far-zone decomposition on the element interactions within the array is applied. For convenience and efficiency, the array is partitioned into an FMM-type clustering grid using the element lattice of the finite array, as shown in Figure 6.1. The chosen FMM cluster spacing is identical to the array element spacing, with the same element numbering and spacing used for the far-zone clusters as well. Using a similar procedure as the conventional FMM for arbitrary cluster locations, all elements of the array are treated as identical clusters of basis functions. In the AD-FMM model, the basis functions of each array element are given a local vector position $\overline{d}_m$ referenced to the

center of the element's lattice position O_l (see Figure 6.1). Unlike the general case of FMM for arbitrary structures, in an array of identical elements, each element and hence each basis function cluster is of the same geometric extent, and can be treated simply as a unit cluster. The unit cluster has a maximum radius of extent D that encompasses every basis function of the array element. This is related to a near-zone of influence R_{near}. Recall that for conventional FMM, cluster sizes and spacing are typically constructed in a way so as to avoid overlapping. In AD-FMM, there is neither degree of freedom – both the array element size and lattice spacing are pre-specified. The degree to which cluster radii overlap has a significant effect on the overall cost of the near-zone storage for AD-FMM, and has an important impact on solution accuracy as well.

Consider first the simple case of a linear array. The elements of the array (for each dimension) are given an index *n*. The basis functions of each array element are given a local number *m*, that result in identical local basis numbering for each element of the array. That is, the same local numbering is used for every basis of every element, just as in ADM. In this notation, the global vector pointing to the m^{th} local basis location of array element *n* is given the notation $\overline{r}_{m,n}$. Here, an effort has been made to use the same notation of Chapter 4 for the array element numbering (*n*) and local basis numbering within the elements (*m*), as was used in Chapter 3 for cluster numbering (*n*) and local basis numbering within the clusters (*m*). In Chapter 4, the impedance sub-matrix entry describing the coupling between testing and source array elements n, n' (4.2.1) was given as

$$[a_{mm'}]_{nn'} = jk_0 \int_S \int_{S'} \overline{w}(\overline{r}_{m,n}) \bullet \overline{w}(\overline{r}_{m',n'}) \frac{e^{-jk_0 \left| \overline{r}_{m,n} - \overline{r}_{m',n'} \right|}}{4\pi \left| \overline{r}_{m,n} - \overline{r}_{m',n'} \right|} dS' dS \,. \tag{6.2.1}$$

This choice of notation was intentional, as the same notation was used in (3.1.1) to describe the coupling between FMM clusters n, n'. Hence, in the simplest implementation of AD-FMM, the concepts of *array element* and *cluster* are combined, and it is possible to use the same notation for treating array elements as clusters without loss of generality.

It is instructive to draw a few parallels between the FMM and ADM analyses, to see how they are alike, and exactly how they differ. To initiate the near-zone decomposition, the coupling interaction between array elements is first recast in terms of local element vectors and position vectors relating elements of the array, restated here as

$$[a_{mm'}]_{nn'} = jk_0 \int_S \int_{S'} \overline{w}(\overline{d}_m) \bullet \overline{w}(\overline{d}_{m'}) \frac{e^{-jk_0 \left| \overline{d}_m - \overline{d}_{m'} + \overline{\rho}_n - \overline{\rho}_{n'} \right|}}{4\pi \left| \overline{d}_m - \overline{d}_{m'} + \overline{\rho}_n - \overline{\rho}_{n'} \right|} dS' dS \,. \tag{6.2.2}$$

Again, notice that this same procedure was initiated for the FMM cluster interactions, in order that (3.1.3) could be applied. The intention is, of course, that distant array element interactions can be treated using the approximate expansion to the Green's function

$$\frac{e^{-jk_0 \left| \overline{\rho}_{nn'} + (\overline{d}_{m,n} - \overline{d}_{m',n'}) \right|}}{\left| \overline{\rho}_{nn'} + (\overline{d}_{m,n} - \overline{d}_{m',n'}) \right|} \approx \frac{-jk_0}{4\pi} \int e^{i\overline{k} \bullet (\overline{d}_{m,n} - \overline{d}_{m',n'})} \mathrm{T}_L [k_0 \left| \overline{\rho}_{nn'} \right|, \hat{k} \bullet \hat{\rho}_{nn'}] dk\Omega \,. \tag{6.2.3}$$

In other words, element interactions in the far-zone are treated via the FMM. The impedance matrix entries of (6.1.1) corresponding to the coupling between far-separated elements will now be equivalently performed as

$$[a_{mm'}]_{nn'} \approx \frac{k_0}{(4\pi)^2} \int \left[\int_S \overline{w}(\overline{r}_{m,n}) e^{j\overline{k} \bullet \overline{d}_{m,n}} dS \{\tau\}_{nn'} \int_{S'} \overline{w}(\overline{r}_{m',n'}) e^{-j\overline{k} \bullet \overline{d}_{m',n'}} dS' \right] dk\Omega \,. \tag{6.2.4}$$

This equation has the same form as (3.1.10), used to interact distant clusters in the far-zone. Similarly, here the array elements are being treated as clusters and interacted in the far-zone.

To couple distant elements via the FMM, it is required to pre-compute translation operators for mapping the fields of any given element of the array onto any other element. Since conditions of regular spacing of the element clusters and sequential numbering in each dimension have been imposed, the unique translation operators are defined by the difference in their array numbering, and are given by

$$\mathrm{T}_L[k_0\left|\overline{\rho}_{nn'}\right|, n-n'] = \sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k_0\left|\overline{\rho}_{nn'}\right|)P_l(\hat{k}\bullet\hat{\rho}_{nn'}))\,. \qquad (6.2.5)$$

The quantities in the translation operator expression are the same as in Chapter 3, though the variables have a different context for AD-FMM. In this expression, $\overline{\rho}_{nn'} = \overline{\rho}_n - \overline{\rho}_{n'}$ is the vector separating the source and testing *array* elements. For AD-FMM, the matrix structure $[\mathrm{T}_L]$ will have a dimension for each dimension of the array, plus one additional dimension for the *k*-space data $\hat{k} = 1..K$. The number of *k*-space directions, as before, is roughly $K \simeq 2L^2$, which will likely be higher than that required in Chapter 3, assuming the array elements are larger than the typical FMM cluster size.

Recall from Chapter 5 that the form of the multi-dimensional kernel used for near-zone decomposition was given as

$$g(R) = \frac{e^{-jk_0\left|\overline{d}_m - \overline{d}_{m'} + (i_1 - i_{1'})\delta_1\hat{\rho}_1 + (i_2 - i_{2'})\delta_2\hat{\rho}_2 + \cdots + (i_{ndims} - i_{ndims'})\delta_{ndims}\hat{\rho}_{ndims}\right|}}{4\pi\left|\overline{d}_m - \overline{d}_{m'} + (i_1 - i_{1'})\delta_1\hat{\rho}_1 + (i_2 - i_{2'})\delta_2\hat{\rho}_2 + \cdots + (i_{ndims} - i_{ndims'})\delta_{ndims}\hat{\rho}_{ndims}\right|}\,. \qquad (6.2.6)$$

In this expression, element indices in each dimension are denoted with *i*=1..*n*, up to a maximum number of dimensions *ndims*. The direction of each dimension is denoted $\hat{\rho}_d$,

and the cluster spacing is given as δ_d. This kernel allows a multi-dimensional decomposition of the near-zone interactions, leading to systematic reductions in the near-zone storage. Similarly, to perform a far-zone decomposition on an array with *ndims* dimensions, the multi-dimensional translation matrix takes the form

$$\mathrm{T}_L[k_0\,|\overline{\rho}|,\hat{k}\bullet\hat{\rho}] = \sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k_0\,|\overline{\rho}|)P_l(\hat{k}\bullet\hat{\rho}))\,, \tag{6.2.7}$$

where

$$|\rho| = \left|(i_1 - i_1')\delta_1\,\hat{\rho}_1 + (i_2 - i_2')\delta_2\,\hat{\rho}_2 + \cdots + (i_{ndims} - i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\right| \tag{6.2.8}$$

and

$$\hat{\rho} = \frac{(i_1 - i_1')\delta_1\,\hat{\rho}_1 + (i_2 - i_2')\delta_2\,\hat{\rho}_2 + \cdots + (i_{ndims} - i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}}{|\rho|} \tag{6.2.9}$$

Notice that (6.2.7) has the same form as (3.2.1), written here for an arbitrary number of array dimensions.

It is assumed here that the translation matrix is non-symmetric block-Toeplitz, verses symmetric block-Toeplitz. That is, only the first column and first row of the translation matrix (for each dimension) have been stored, as was done for the near-zone terms in the previous chapter. The number of entries for each dimension of the array will be $2n_\mathrm{d}$-1, n_d being the number of elements in dimension d. Note that it is possible to reduce the far-zone storage by an additional factor of two, simply by exploiting the similarities in the mapping of $\overline{\rho}_{nn'} = \overline{\rho}_n - \overline{\rho}_{n'}$ and $\overline{\rho}_{n'n} = \overline{\rho}_{n'} - \overline{\rho}_n$. However, this option is not pursued in this implementation, such that the Fourier transform of the translation data can be pre-computed and stored for later acceleration of the translation/interaction of

array elements via the FFT. Specifically, it is possible to pre-compute the FFT of the translation operators as

$$[\tilde{\mathrm{T}}_L] = FFT\{\mathrm{T}_L[k_0|\overline{\rho}|, \hat{k}\bullet\hat{\rho}]\}. \tag{6.2.10}$$

The FFT will be performed on each dimension of the array and be of a size reflecting the number of terms in the $[\mathrm{T}_L]$ matrix for that dimension. In other words, an array of size $n_1 \times n_2$ will require *K* FFTs of size $(2n_1-1)\times(2n_2-1)$ – one for each *k*-space direction. In the implementation used for this thesis, a split-radix FFT was employed that allows FFTs of combined multiples of 2, 3, 4, and 5 [30]. As a note, if 2*n*-1 is not a combination of these multiples, the size of the FFT can be increased to the next highest product of these multiples, or else a discrete Fourier transform (DFT) will be used instead. The cost trade-off between using a larger FFT size vs. a less efficient but smaller DFT has to be considered. In most cases, the Fourier transforms are not the bottleneck in the solution process, and thus typically this cost consideration can be overlooked, choosing instead the simplest implementation.

The consequence of the far-zone decomposition on near-zone matrix storage is now illustrated. With ADM alone, the near-zone matrix storage is restricted to the first column and first row of the original system matrix (in each dimension), as depicted in (6.1.1). The unique sub-matrices of the first column and first row of (6.1.1) represent all the rigorous array element interactions, for both near- and far-separated elements, the nearest array interactions being closest to the upper left-hand corner. When the FMM is used to interact the far-separated element interactions, only the near-separated element interactions will remain in (6.1.1), with the distant element interactions in the first row and first column of the system matrix removed for treatment via the FMM. Hence, with

the FMM expansion applied on top ADM, the near-zone matrix reduces to roughly the unshaded area of the matrix system

$$\begin{bmatrix} [a]_{11'} & [a]_{12'} & \cdots & [a]_{1n'} \\ [a]_{21'} & [a]_{22'} & \cdots & [a]_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ [a]_{n1'} & [a]_{n2'} & \cdots & [a]_{nn'} \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_2 \\ \vdots \\ \{x\}_n \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_2 \\ \vdots \\ \{b\}_n \end{Bmatrix}. \tag{6.2.11}$$

It should be immediately apparent that the number of near-zone terms remaining will be exactly the same for any number of elements *n* in a given array type. That is, a linear array of 10 elements has the same near-zone storage as an array with 1000 elements (same element, same spacing). The near-zone matrix storage is now on par with the storage of a single element, $[a_{11'}]$. Typically, for $\lambda_0/2$ element spacing, and elements with a near-zone radius or roughly λ_0, one might expect near terms from at most one or two neighboring elements in each dimension. However, the required near-zone storage depends on both the geometry of the array element and the array lattice. Any linear array will have the same storage for any number of elements (in the limit), and any planar array will have the same storage for any number of elements as well. However, a planar array will likely have more storage than a linear array. This is an important distinction, though the amount of storage will still be fixed in either case (in the limit).

As alluded to earlier, because of the truncated near-zone storage, it is not necessary to apply the FFT in accelerating the near-zone interactions, as is done in standard ADM. Of course, the near-zone portion of the matrix-vector product operation will still have a convolutional form. However, since the number or near-zone terms is now considerably smaller, there is no advantage in using the FFT. Pre-computing the FFT for the near-zone terms would require a vector size proportional to the data vectors,

which is typically (for large arrays) much larger in extent than the near-zone storage vector. Pre-computing FFT data of this size would nullify the storage advantage achieved by combining the near-zone decomposition with the far-zone decomposition, and hence should not be done.

6.3 AD-FMM Implementation Notes

In the conventional FMM, it is necessary to compute the signature functions of each cluster with the Fourier transform procedure

$$W_n[\hat{k},m] = \int_S \overline{w}(\overline{r}_{m,n}) e^{j\overline{k}\cdot\overline{r}_{m,n}} dS\,, \tag{6.3.1}$$

where $\overline{r}_{m,n}$ points from the global origin to basis function m of cluster n. This same procedure can be carried over to AD-FMM, but is not necessary. The signature functions can be equivalently written in terms of local cluster vectors $\overline{d}_m$. In AD-FMM, each element cluster is identical, and hence $\overline{d}_{m,n} = \overline{d}_m$. This allows the computation of a single signature function, based on the unit cluster of AD-FMM, given simply by

$$W[\hat{k},m] = \int_S \overline{w}(\overline{d}_m) e^{j\overline{k}\cdot\overline{d}_m} dS\,. \tag{6.3.2}$$

In AD-FMM, this common signature function is used for all array elements to compute their far-zone patterns. A general FMM approach to the same array would result in n times as much storage for signature functions, which can be considerable for very large array structures.

The number of k-space directions for each array element, $K \simeq 2L^2$, is again found from

$$L = k_0 2D + \alpha_L \ln(k_0 2D + \pi)\,. \tag{6.3.3}$$

As in Chapter 3, α_L is chosen to generate the desired accuracy for the application type and machine architecture. In AD-FMM, D is a fixed parameter. That is, unlike the case for conventional FMM, the cluster radius D is determined by the array element size – a fixed parameter.

As with conventional FMM, to use (6.2.5) it is necessary that L is not larger than $k_0 |\overline{\rho}_{nn'}|$, otherwise the Hankel function will oscillate, causing inaccuracies in the translation operators. This imposes the primary criterion that determines which elements can be treated as being in the far-zone. Specifically, the far-zone condition is

$$R_{near} > L / k_0\,. \tag{6.3.4}$$

Any two array elements separated by less than R_{near} will be treated as being in the near-zone, and thus contribute to matrix storage. While a larger L gives greater accuracy, it also increases R_{near}, the necessary distance between array elements for which the far-zone expansion is applicable. For a typical array element separation of around λ/2, one or two adjacent elements in each dimension will require rigorous near-zone treatment. Next, the solution procedure for AD-FMM is reviewed.

6.4 AD-FMM Solution Procedure

The solution procedure begins by evaluating the near-zone portion of the iterative solution procedure. The near-zone portion of the iterative solution procedure can be represented with the generalized expression

$$[A]_{near}\{x\}^{est} = \{b\}^{near}. \tag{6.4.1}$$

Because of the block-Toeplitz nature of the near-zone storage, this is equivalently carried out as a convolution of sub-matrix-vector product operations given by

$$\{[a]_{-near} \quad \cdots \quad [a]_0 \quad \cdots \quad [a]_{near}\} * \{x\}^{est} = \{b\}^{near}. \tag{6.4.2}$$

In this expression, $[a]_{near}$ signifies the furthest array element interaction that is carried out in the near-zone for a given dimension. As stated in the previous section, the near-zone storage is proportional to $[a]_0$. In fact, under certain conditions, $[a]_0$ will represent the majority of the near-zone interactions. Hence, if the procedure given in APPENDIX B is employed, where the self-cell interactions are preconditioned out from the iterative solution procedure, a large portion and potentially the majority of the near-zone interactions can be pre-solved. In general, the near-zone interactions take up a significant portion of the iterative solution time. Hence, a significant portion of the iterative solution cost can be completely avoided. This statement comes with the caveat that the preconditioning procedure must be performed at every iteration of the solution procedure as usual, a cost that cannot be avoided.

The far-zone portion of the iterative solution procedure is represented with the general expression

$$[A]_{far}\{x\}^{est} = \{b\}^{far}. \tag{6.4.3}$$

This portion of the matrix-vector product operation is performed in the following way. First, the far-zone fields of each array element are generated from the pre-computed far-zone basis function expansions and stored in a vector grid via the step

$$\left\{s_{\hat{k}}\right\}_n = \sum_m W^*(\hat{k}, m)\{x_m\}_n^{est},, \tag{6.4.4}$$

where $\{x\}_n^{est}$ is the estimated unknown current vector of array element n, and the summation determines the combined effect of all basis functions within the element in the far-zone. For this implementation, $\left[s_{\hat{k},n}\right]$ will be a matrix with the same dimensions as the array, plus an additional dimension for $\hat{k}$. The coupling between source and testing elements can then performed via the arbitrary-dimensional convolution

$$\{g_n\}_{\hat{k}} = \mathrm{T}_L[k_0|\bar{\rho}|, \hat{k}\bullet\hat{\rho}] * \{s_n\}_{\hat{k}}. \tag{6.4.5}$$

The dimension of the convolution will be the same as that of the array. For large systems, this is an expensive operation, but can easily be accelerated with FFT operations of the same dimension as the finite array. To achieve $\{g_n\}_{\hat{k}}$ via the FFT, it is necessary to perform the equivalent operation

$$\{g_n\}_{\hat{k}} = FFT^{-1}\{FFT\{\{s_n\}_{\hat{k}}\}[\tilde{\mathrm{T}}_L]_{\hat{k}}\}. \tag{6.4.6}$$

Hence, the convolution in (6.4.5) is replaced with $2K$ FFT operations on $\{s_n\}_{\hat{k}}$ (forward and inverse) of a size and dimension matching $[\tilde{T}_L]_{\hat{k}}$ as computed in (6.2.10), plus $K\prod_{d=1}^{ndims}(2n_d-1)$ point-by-point multiplications between $[\tilde{T}_L]_{\hat{k}}$ and $\{\tilde{s}_n\}_{\hat{k}}$. The overall savings of this alternative solution approach is proportional to the size of the array, and the exact cost will be explored later. The coupling for each basis *m* of array element *n* is then extracted via the inverse transformation

$$\{b_m\}_n^{far} = \int W[\hat{k}, m]\{g_n\}_{\hat{k}} dk\Omega. \tag{6.4.7}$$

Thus, $\{b\}^{far}$ represents the portion of the matrix-vector product carried out in the far-zone. This result is simply added to the near-zone contribution of the matrix-vector product operation to give the combined results, or

$$[[A]_{near} + [A]_{far}]\{x\}^{est} = \{b\}^{near} + \{b\}^{far} = \{b\}. \quad (6.4.8)$$

Below, a detailed discussion of the storage and CPU requirements of AD-FMM is given.

6.5 Storage Cost and Performance Analysis

In this section, the storage and computational cost of AD-FMM are explicitly evaluated, as compared with other methods. The two costs (storage and computation) are evaluated separately, since it is insightful to highlight the differences between the various approaches. Consider a volumetric antenna array element of arbitrary materials and shape, as shown in Figure 6.2.

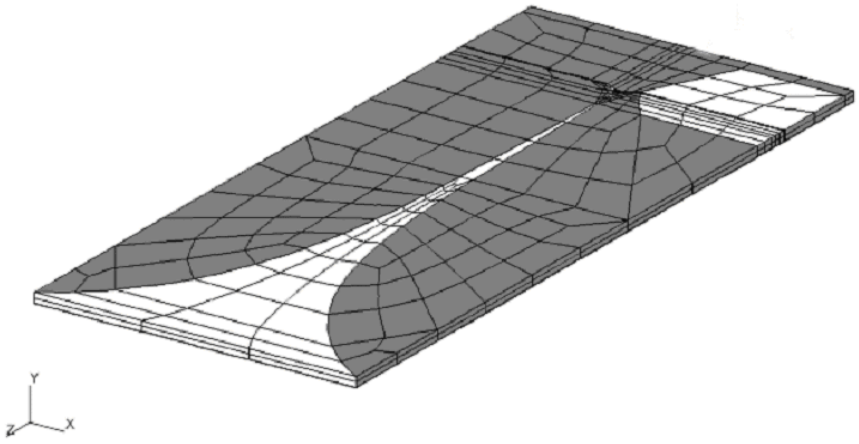


Figure 6.2. Tessellation of a tapered-slot antenna element.

Suppose that upon tessellation and assignment of boundary conditions there are m_{FEM} FEM edges or unknowns and m_{BI} boundary integral unknowns for each antenna element. Roughly speaking, there will be $O(m_{FEM})$ FEM storage and $O(m_{BI}^2)$ boundary integral storage in a conventional FE-BI formulation for this single element. Consider

now an arbitrary linear array of n antennas, with the total number of FEM unknowns equal to $N_{FEM} = nm_{FEM}$, and the total BI unknowns equal to $N_{BI} = nm_{BI}$. For a conventional FE-BI approach, there will be $O(nm_{FEM})$ FEM storage, and $O(n^2 m_{BI}^2)$ boundary integral matrix storage.

Next, consider a generalized FMM expansion of the same array problem, again implemented for FE-BI. Like the conventional FE-BI, FMM will have $O(nm_{FEM})$ FEM storage for the finite array. Regarding the integral equation terms, FMM reduces the problem storage down to a best case $O(nm_{BI} \log(nm_{BI}))$ for a multi-level implementation.

For ADM, since the method exploits the repeatability of the array elements, the FEM storage is only $O(m_{FEM})$, the same as for a single element. Thus, the FEM storage is n times less than that of conventional FE-BI and FE-BI with FMM. For the boundary integral equation terms, the storage is $O(nm_{BI}^2)$, which is typically larger than the corresponding FMM storage for large array elements. Though it may require slightly more BI storage, ADM has the advantage that matrix fill times are considerably faster than FMM because of much lower overhead. As will be shown later, the same is true for solution times. In the limit, ADM has a linear increase in BI matrix storage.

For AD-FMM, like standard ADM, the FEM storage is merely $O(m_{FEM})$. However, unlike FMM or ADM, both of which experience the greater than linear increase in the BI matrix storage, AD-FMM has only $O(m_{BI}^2)$ integral equation coefficient storage, i.e. the same order of magnitude as that of a single element. This is remarkable because it will allow solution of very large finite arrays using rigorous means. However, this cost analysis comes with some caveats, because additional overhead exists

for the AD-FMM approach to finite arrays verses the simpler analysis of a single array element. For example, it is necessary to store the far-zone signatures of the array element for AD-FMM, along with the translation operators for interacting the elements in the far-zone. However, as the clusters are aggregated to a regular grid, the translation operators have a Toeplitz property with the required storage of $O(nK)$, K being the number of points or *k*-space directions used to represent the element signatures in the far-zone. Upon implementation, for very large problems the memory requirements to storing solution vectors, excitation vectors, and solver work vectors will exceed the storage of the translation operators, assuming that the number of far-zone directions K will be less than the number of surface unknowns per cluster, or $K < m_{BI}$. Hence, the *total* storage requirements for larger problems will be proportional to the length of the overall solution vector, or $O(n(m_{FEM} + m_{BI})) = O(N)$.

A performance analysis on the FEM portion of the matrix-vector product reveals that both conventional FE-BI and FE-BI with FMM have a solution cost of $O(nm_{FEM})$. However, both ADM and AD-FMM pre-solve or precondition the FEM portions of the overall matrix system prior to the iterative solution process [50], thereby avoiding this cost entirely. This is critical, as iterations over FEM matrix systems are typically costly due to the conditions of these systems.

The computational cost of the BI portion of the matrix-vector product for conventional FE-BI is proportional to its storage cost of $O(n^2 m_{BI}^2)$. The same principle applies to MLFMM, which has a cost of $O(nm_{BI} \log(nm_{BI}))$. However, it is instructive to remark that this estimate does not reflect the cost of excessive iterations. The

computational cost of ADM is based on m_{BI} FFT operations of cost $n\log n$, plus n matrix-vector products with m_{BI}^2 operations. This gives a combined total cost of $O(nm_{BI}^2 + m_{BI} n\log(n))$. Though it is hard to make a direct comparison, in practice there is very little overhead for ADM, resulting in significantly faster solution times as compared to FMM or FE-BI for finite arrays. This can be directly observed in the results of the next section.

By comparison, AD-FMM has a near-zone computational cost of $O(nm_{BI}^2)$, plus a far-zone cost of roughly $O(Kn\log(n))$. For large problems, the far-zone cost is largely dominated by the translation operations, which can be reduced to $O(Kn\log n)$ operations using the FFT. In the same way that the FEM operations can be pre-computed, it is also possible to avoid a large fraction of the BI matrix-vector product cost for the near-zone interactions using block-diagonal preconditioning. Again, the fractional savings depends on the array element size and spacing. For example, in a case where the element spacing is larger than the array element diameter, the near-zone CPU cost is reduced to zero, as the near-zone interactions can be entirely pre-computed. A cursory glance shows that the computational cost of ADM and AD-FMM are on par. Assuming that $K < m_{BI}$, one may expect the solution times for AD-FMM to be faster than ADM. However, due to the additional overhead cost associated with AD-FMM, whether or not AD-FMM is faster will be implementation dependent. The storage cost and performance analysis of the various methods discussed here are summarized in Table 6.1. This table does not reflect the total storage cost ($O(n(m_{FEM} + m_{BI}))$ for AD-FMM).

Table 6.1. Storage Cost and Performance Analysis of a Finite Array Problem.

	Storage Cost FEM	Storage Cost BI	Computational Cost - FEM	Computational Cost - BI
Conventional FE-BI	$O(nm_{FEM})$	$O(n^2 m_{BI}^2)$	$O(nm_{FEM})$	$O(n^2 m_{BI}^2)$
Conventional MLFMM	$O(nm_{FEM})$	$O(nm_{BI}\log(nm_{BI}))$	$O(nm_{FEM})$	$O(nm_{BI}\log(nm_{BI}))$
ADM	$O(m_{FEM})$	$O(nm_{BI}^2)$	-	$O(nm_{BI}^2 + m_{BI}n\log(n))$
AD-FMM	$O(m_{FEM})$	$O(m_{BI}^2)$	-	$O(nm_{BI}^2 + Kn\log(n))$

6.6 Multi-Level AD-FMM

In theory, it would be possible to use a multi-level approach to AD-FMM. The multi-level approach has not been explored in this thesis for several reasons. First, in the way the FMM is applied to array-type problems in later chapters, especially in multi-system analysis, there tends to be a very low occurrence of empty clusters, reducing the weight behind the argument for the multi-level approach. Rather, the approach presented in this thesis features a large block-Toeplitz translation matrix using a single level algorithm with multiple dimensions. Secondly, AD-FMM benefits highly from the fact that the cluster signature functions are all the same, and hence only one set it computed for an entire array system (which may be very huge). In a multi-level implementation, this advantage would be lost, although some re-use of information would likely be available at various levels in the multi-level approach as well. Moreover, by sticking with the single level implementation, the additional overhead of interpolations between interaction levels be avoided. Additionally, the single-level implementation has the advantage of explicit acceleration with a FFT for automatic $O(n\log n)$ solution

complexity. Further, parallel implementations of the single-level AD-FMM are far easier to achieve.

While explicit evaluation of the far-zone coupling with the FFT is expedient, in reality a multi-level approach might have some benefits. For example, the FFT must explicitly treat the near-zone translations as well, though the translation operators are null for these matrix entries. Likewise, if the element decomposition leads to empty clusters, this creates further waste in the translation matrix. In any case, the near-zone interactions will likely continue to be the factor of highest cost, and hence the need for a multi-level AD-FMM algorithm is not a priority.

6.7 AD-FMM Results

In this section, the storage cost and performance of the various methods discussed up to this point are evaluated using a specific array example. The analysis is carried out on two-dimensional arrays of various sizes ranging from 3×3 to 64×64. To form the array, the antenna element (see Figure 6.2) is placed on a regularly spaced grid of 6.25cm in x and y -- exactly $\lambda_0/2$ at 2.4GHz, the frequency used in this analysis. The element is a double-sided exponentially tapered-slot antenna, fed via stripline and matched with a double-Y balun (not depicted here). The element dimensions are $11.45\times 5.0\times 0.1524\,\text{cm}$. For the evaluation cases, this element is modeled with m_{FEM} = 805 and m_{BI} = 892 unknowns, based on the tessellation depicted in Figure 6.2. Modeling this type of structure is challenging due to the detailed feed characteristics and the electrically large

element size. This combination of conditions can easily lead to edge ratios of 50 to 1 at lower frequencies. For reference, the storage cost of this single element is about 14MB.

To begin, the E-plane and H-plane patterns for the 5×5 array of Figure 6.3 are compared. The corresponding patterns for all methods are shown in Figure 6.4, and it can be seen that the results from all methods are nearly identical. The MLFMM plots deviate the most from the other methods, as a low cost $\alpha_L = 1.0$ was used for maximum storage savings at the cost of some accuracy. While a 5×5 array may seem small for a test case, for this electrically large element, a conventional FE-BI solution requires roughly 8.5GB of matrix storage (apart from the challenges associated with the poor conditioning of the matrix). Hence, this 5×5 array problem is near the practical analysis limit for conventional FE-BI. For equivalent comparisons, the analysis was performed on the Itanium server of previous chapters.

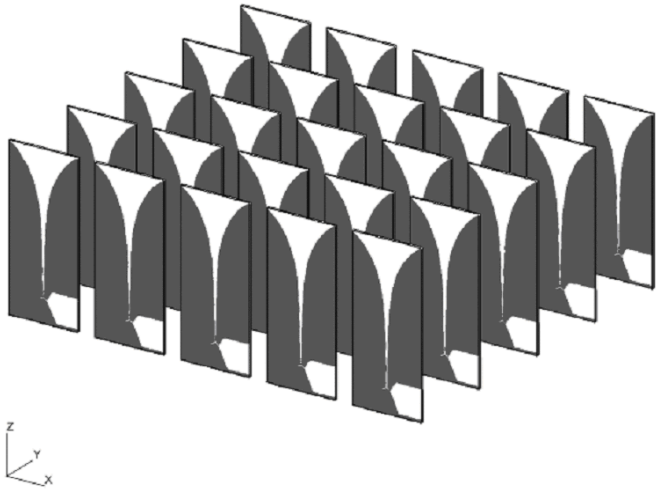
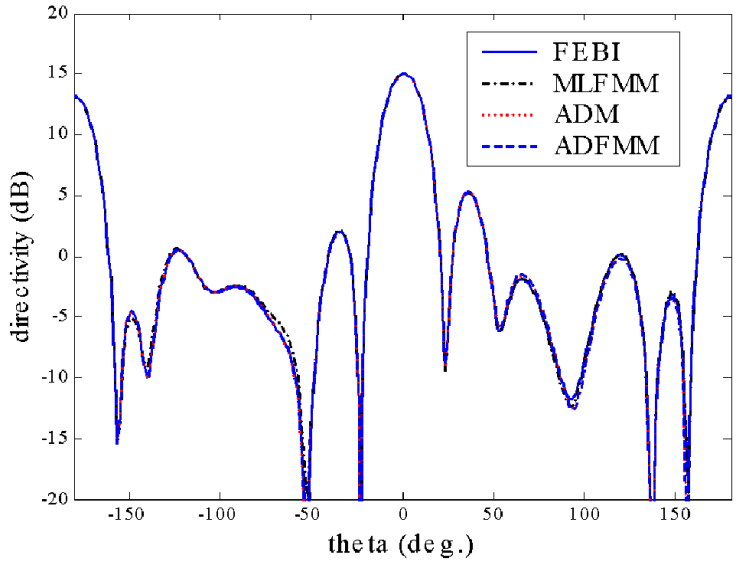
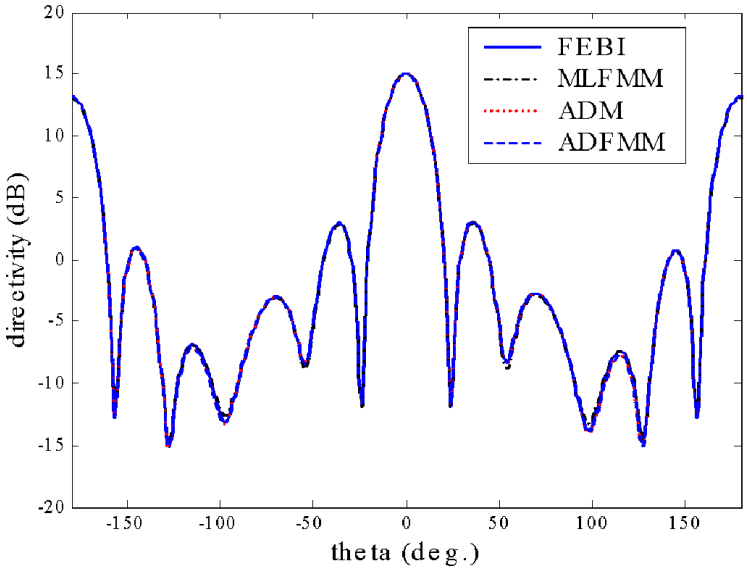


Figure 6.3. 5×5 array used for consistency comparisons.



(a)



(b)

Figure 6.4. Pattern comparisons of each method for the 5×5 array, a) E-plane, b) H-plane.

To demonstrate the effectiveness of AD-FMM, the aforementioned methods are now compared with AD-FMM for arrays of several sizes. Of particular interest is the required matrix storage for each problem, the matrix fill time, the number of required iterations for solution, and also the full iterative solution time for each of the methods. Each figure of merit has its own virtues. The storage requirements are a measure of how well the method can be implemented on various platforms, in the sense that a user with no memory constraints may opt to choose the method with the fastest solution times. The matrix fill time is significant as well, as this affects the overall solution time, and serves to further distinguish AD-FMM from other implementations of FMM. The required number of iterations reflects on the matrix conditioning and the effect of the employed matrix preconditioning method, whereas the CPU time per iteration reflects directly on the CPU speed of the method. Table 6.2 summarizes the results for the various figures of merit on several example arrays of different sizes.

For many of the test cases, the more expensive methods (FE-BI, MLFMM) cannot solve the problems with the allotted resources (or time constraints), and the required resources were projected as needed. For FE-BI and MLFMM, it is difficult to accurately project the solution times. Preconditioning options differ for these large systems, and the number of iterations for convergence can be very large (if the system converged at all). For these test cases, extra memory was allotted for the conventional FE-BI and MLFMM, such that an LU factorization could be used to pre-solve the FEM portions of the matrix system.

The results for AD-FMM given in Table 6.2 speak for themselves. AD-FMM is superior to all other methods compared here for finite array analysis. In all AD-FMM

test cases, a block-diagonal preconditioner was employed with the BICGSTAB(L) iterative solver, having the equivalent of eight matrix-vector product operations per iteration (L=4) [24]. The overall storage cost for AD-FMM consists of the matrix storage, the unknown and excitation vector storage, the far-zone basis representations and the pre-computed translation operators. While the 64x64 array required only 193MB for matrix storage, a *total* storage of 432MB was required for carrying out the solution (not counting solver work vectors). This particular problem has crossed the size threshold beyond which unknown and excitation storage exceeded matrix storage. As a conclusion, a rigorous analysis of a 7-million unknown problem has been performed using less than half a gigabyte of total storage. Being able to rigorously solve such large systems in a reasonable amount of time is significant in itself. What is not shown in the table is the overhead time associated with the FMM based methods necessary to calculate and fill the far-zone operators. For the conventional MLFMM, this cost can be quite high. However, for AD-FMM the overhead cost is quite low, due to the efficiency of the hybrid decomposition approach.

Finally, it is worth noting that the matrix preconditioning for the array decomposition methods is quite good, as evidenced by the low number of iterations (to achieve 1% solution error). As mentioned, good preconditioning is critical for iterative solution methods, especially for FMM methods that shift the computational burden to the iterative procedure. Better preconditioning methods for the FEM matrix systems were employed in this chapter compared with previous chapters, resulting in a much smaller number of iterations for the conventional FE-BI and MLFMM. However, due to the high

cost of this effective preconditioning method, it was not possible to analyze an array larger than 5×5 using either conventional FE-BI or MLFMM.

Table 6.2. Evaluation of AD-FMM for Various Finite Array Problems.

Array Size		3×3	5×5	16×16	32×32	64×64
Unknowns		15,273	42,425	434,432	1,737,728	6,950,912
Matrix Storage Requirements	FE-BI	1.1GB	8.5GB	868TB	13PB	217PB
	MLFMM	162MB	502MB	5.3GB	21GB	88GB
	ADM	193MB	612MB	7.2GB	29GB	121GB
	AD-	193MB	193MB	193MB	193MB	193MB
Matrix Fill Time	FE-BI	24m	3h	*14d	*218d	*9.5y
	MLFMM	5m	18m	6h	*28h	*5d
	ADM	8m	25m	5h	*20h	*3d
	AD-	7m	7m	7m	7m	7m
Iterations	FE-BI	6	108	-	-	-
	MLFMM	6	69	-	-	-
	ADM	2	4	19	*62	*100
	AD-	2	4	19	62	100
Iterative Solution Time	FE-BI	2m	1h	-	-	-
	MLFMM	1m	51m	-	-	-
	ADM	6s	27s	42m	*7h	*25h
	AD-	10s	1m	1h	17h	2d
Total Storage Cost	FE-BI	1.1GB	8.5GB	868TB	13PB	217PB
	MLFMM	175MB	536MB	5.6GB	23GB	94GB
	ADM	194MB	613MB	7.2GB	29GB	121GB
	AD-	200MB	201MB	214MB	258MB	432MB
Total Solution Time	FE-BI	26m	4h	-	-	-
	MLFMM	6m	1h	-	-	-
	ADM	9m	26m	6h	*1d	*4d
	AD-	8m	9m	1h	17h	2d

* estimated results

6.8 Concluding Remarks

The main theme of chapters 4-6 was the exploitation of known geometrical redundancies and the decomposition of these redundancies into re-usable cells where the Toeplitz property of finite array interactions results in reduced storage and accelerated solution. The concept applies to many simpler problems, such as large flat or smoothly curving surfaces (viz. a cylindrical airplane fuselage), and is in no way limited to antenna array analysis. To reiterate, one of the main themes of this thesis is that explicit decomposition of a problem into identical cells (when possible) is extremely advantageous in terms of storage reductions and solution speeds, as compared to generalized solution approaches that are insensitive to geometric redundancies. The near-zone (ADM) and far-zone (FMM) decomposition combination leads to remarkable speed-ups for finite array-type problems.

The details of the method presented in this chapter present a major advance in rigorous analysis of large finite arrays, unhindered by the restrictions of matrix storage limitations. However, it is understood that the AD-FMM presented in this chapter is not without its weaknesses. For very large array elements, or closely packed arrays of elongated elements, the near-zone storage can potentially become prohibitively large. This drawback can be somewhat alleviated by partitioning the array elements into smaller clusters using intra-element decomposition, as explained in the next chapter.

Further, as a stand-alone method, AD-FMM has limited utility. The analysis of a freestanding array does not match well with the way arrays are found in the real world. The full advantage of AD-FMM is realized when implemented for multiple systems. In

other words, finite arrays can be modeled in their actual environment, which may be on the surface of a vehicle or in the presence of other structures.

As a side note, it is important to consider the ramifications of using methods that are not limited by matrix storage. Typically, problems with large matrix storage are split up using distributed memory and analyzed in pieces on multiple machines, with the full solution vector is present on all nodes. From the analysis presented here, one can conclude that AD-FMM has the potential of analyzing problems for which it is not possible to store the entire solution vector on a single machine, assuming 32-bit memory addressing, making the popular 32-bit PC-based distributed cluster model obsolete. That is, using AD-FMM, it is easily possible to analyze problems with 100 million or more unknowns, requiring over two gigabytes to store the solution and excitation vector alone, (assuming COMPLEX(8) data types), thereby exceeding the limit of 32-bit addressing. The advent of this type of problem may bring about the necessity of a new paradigm in distributed processing methods, a challenge to be explored in future work.

CHAPTER 7

AD-FMM WITH INTRA-ELEMENT DECOMPOSITION

When the near-zone of the array element is too large, the simplest implementation of AD-FMM, which uses the entire array element as a single FMM cluster, may fail to provide adequate storage savings. The worst-case example of this condition is an array element that is less than a wavelength in width, but several wavelengths or more long. An end-fire tapered-slot antenna is good example of an antenna element of this type. In the limit, even for large array elements of this type, AD-FMM will continue to exhibit $O(N)$ storage. That is, even for elements with a large near-zone influence, the near-zone storage will still be fixed for arrays with any number of elements. However, it may be that this fixed amount of storage is a prohibitively large quantity in itself.

In this chapter, an approach is introduced that can alleviate the potential storage burden caused by array elements with prohibitively large near-zone influences. For simplicity, this additional level of complexity was avoided in Chapter 6, as it requires advanced concepts of multi-dimensional analysis to treat properly. In this chapter, the AD-FMM approach of using the entire array element as a single FMM cluster will be complicated by decomposing the element into sub-dimensions of smaller clusters. This is

done with the intention of creating clusters of smaller near-zone influences, thereby potentially reducing the overall near-zone storage cost. This procedure complicates the far-zone decomposition, typically resulting in grids of more than three dimensions for most array type problems. As a note, the introduction of this piece of the puzzle to the AD-FMM makes it possible to reduce the cost of analyzing array and *non*-array-type systems simultaneously. That is, *non*-array-type systems can be decomposed into regular clustering grids and interacted with the clustering grids of the array type systems for cross-system decomposition. This is a topic addressed in the next chapter on multiple system analysis.

In this chapter, the benefits of intra-element AD-FMM will be examined for several practical problems involving electrically long tapered-slot antenna arrays that greatly benefit from this additional level of decomposition. It will be demonstrated that intra-element decomposition makes it practical to analyze real-world arrays of electrically long tapered-slot antennas using limited resources.

7.1 Intra-Element Decomposition

Under many conditions, the simple and elegant option of using an entire array element as the cluster for AD-FMM will suffice to produce excellent results. However, under some conditions, this model may not be adequate to generate a small enough matrix system to work with on machines with limited resources. The general class of problem for which entire-element AD-FMM fails to provide adequate results is exemplified by elements with an extensive near-zone influence. This can result in

excessive near-zone matrix storage when too many adjacent array elements fall within the near-zone radii of their neighboring elements. Though AD-FMM will still produce *O*(N) matrix storage in the limit (for larger arrays), as a practical consideration, the near-zone storage may be a fixed, but still prohibitively large amount. The logical solution to the problem is to reduce the near-zone influence of the element, which can be accomplished by decomposing it into smaller sub-sections as depicted in Figure 7.1. The efficient implementation of this capability is the topic of this chapter.

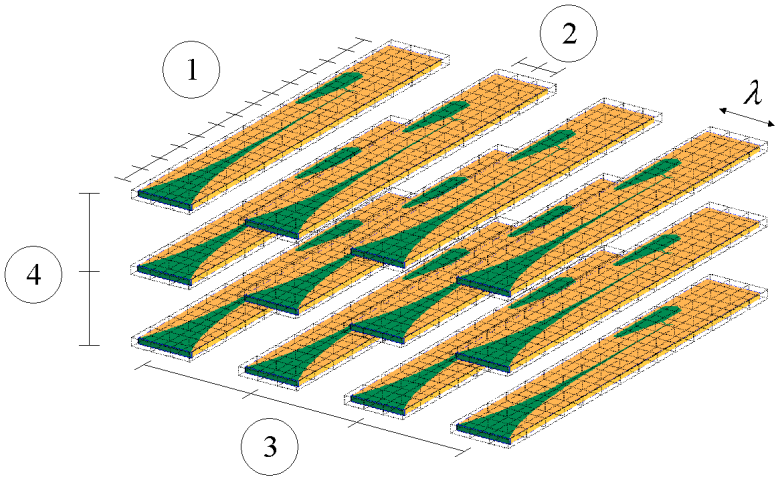


Figure 7.1. Illustration of intra-element decomposition on antenna array.

At the element level, intra-element decomposition is no different than applying FMM-FFT to a generalized problem. In other words, the array element is simply being decomposed into additional dimensions of sub-clusters, in a sense treating the array element with a decomposition approach for arbitrary structures. The difficulty comes when this decomposition is then combined with the array decomposition already in place on the array lattice. For example, in the general case for conventional planar arrays, there will be two dimensions of clusters conforming to the array lattice, and then potentially

three more sub-dimensions, used to decompose the element into smaller clusters. It is important to realize that all dimensions (now five in total) maintain their Toeplitz relations, and it is hence possible to implement a higher-dimensional FFT to accelerate the solution of the array and cluster interactions simultaneously. Algorithms for multi-dimensional FFTs exist precisely for uses such as this, and are no more difficult to implement than 2D FFTs when done correctly.

Consider the array element of the structure shown in Figure 7.1, assuming a frequency for which the width of the element is proportional to a wavelength. Because of its electrical size, it is beneficial to decompose the element into several clusters along both the length and width of the array element as illustrated by the wire grid. This creates a two-dimensional lattice of basis clusters, denoted as the first and second dimensions in Figure 7.1 (dimensions noted in circled numbers). Assuming the model of the FMM for far-zone coupling, this results in a two-dimensional translation operator table for intra-element cluster interactions. The elements then must be considered in the finite array environment. For the example case of a planar array, this allows two more dimensions of decomposition. In the example of Figure 7.1, there are 14 clusters in dimension 1, 3 clusters in dimension 2, four elements in dimension 3, and three elements in dimension 4. Using the multi-dimensional model for the FMM analysis, the result will be a modified translation operator of the form

$$\mathrm{T}_L[k_0|\overline{\rho}|,\hat{k}\bullet\hat{\rho}] = \sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k_0|\overline{\rho}|)P_l(\hat{k}\bullet\hat{\rho}))\,, \tag{7.1.1}$$

where now

$$|\rho| = \left|\begin{matrix}(i_{c1}-i'_{c1})\delta_{c1}\hat{\rho}_{c1}+(i_{c2}-i'_{c2})\delta_{c2}\hat{\rho}_{c2}+(i_{c3}-i'_{c3})\delta_{c3}\hat{\rho}_{c3}+ \\ (i_1-i'_1)\delta_1\hat{\rho}_1+(i_2-i'_2)\delta_2\hat{\rho}_2+\cdots+(i_{ndims}-i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\end{matrix}\right| \tag{7.1.2}$$

and

$$\hat{\rho} = \frac{\begin{array}{c}(i_{c1} - i'_{c1})\delta_{c1}\hat{\rho}_{c1} + (i_{c2} - i'_{c2})\delta_{c2}\hat{\rho}_{c2} + (i_{c2} - i'_{c2})\delta_{c3}\hat{\rho}_{c3} + \\ (i_1 - i'_1)\delta_1\,\hat{\rho}_1 + (i_2 - i'_2)\delta_2\,\hat{\rho}_2 + \cdots + (i_{ndims} - i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\end{array}}{|\rho|}. \qquad (7.1.3)$$

This translation operator accommodates for three potential dimensions of sub-clustering of the element itself, in combination with an arbitrary number of array dimensions *ndims*. The operator definition is a modified version of the model suggested for multi-dimensional analysis of arrays in Chapter 5. Here it has been applied to the FMM clustering procedure, expecting the decomposition to adhere to the same conditions outlined for multi-dimensional analysis of arrays. It is required (arbitrarily) that the clusters in each dimension be numbered consecutively in increasing order, starting with the index 1. In the expression (7.1.1), distinction for the clustering dimensions have been made with the subscripts $c1, c2, c3$, etc. The cluster interactions, translating from source cluster to testing cluster, will follow the same vector sum rules of the multi-dimensional array analysis, transforming first across each dimension of the cluster grid and then across each dimension of the array. Though the assumption of at most three clustering dimensions has been made, in theory a multi-dimensional decomposition could be applied to the clustering as well (to reduce the occurrence of empty clusters).

One main difference in the intra-element decomposition model for whole-element AD-FMM is that the near-zone and far-zone decomposition models are now disjoint, dimension wise. That is, there are more dimensions in the far-zone decomposition than in the near-zone decomposition. In intra-element AD-FMM, the near-zone kernel is given by

$$g(R)=\frac{e^{-jk_0\left|\bar{d}_m-\bar{d}_{m'}+(i_1-i_1')\delta_1\hat{\rho}_1+(i_2-i_2')\delta_2\hat{\rho}_2+\cdots+(i_{ndims}-i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\right|}}{4\pi\left|\bar{d}_m-\bar{d}_{m'}+(i_1-i_1')\delta_1\hat{\rho}_1+(i_2-i_2')\delta_2\hat{\rho}_2+\cdots+(i_{ndims}-i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\right|},\quad(7.1.4)$$

which is three dimensions less in extent than the far-zone kernel

$$\frac{e^{-jk_0\left|\bar{\rho}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}}{\left|\bar{\rho}+(\bar{d}_{m,n}-\bar{d}_{m',n'})\right|}\approx\frac{-jk_0}{4\pi}\int e^{i\bar{k}\bullet(\bar{d}_{m,n}-\bar{d}_{m',n'})}\mathrm{T}_L[k_0\left|\bar{\rho}\right|,\hat{k}\bullet\hat{\rho}]dk\Omega\,,\qquad(7.1.5)$$

where

$$\bar{\rho}=(i_1-i_1')\delta_1\,\hat{\rho}_1+(i_2-i_2')\delta_2\,\hat{\rho}_2+\cdots+(i_{ndims}-i'_{ndims})\delta_{ndims}\hat{\rho}_{ndims}\,.\qquad(7.1.6)$$

This reduces the elegance of the AD-FMM model, but it is strictly an implementation issue and does not detract from the usefulness of the method.

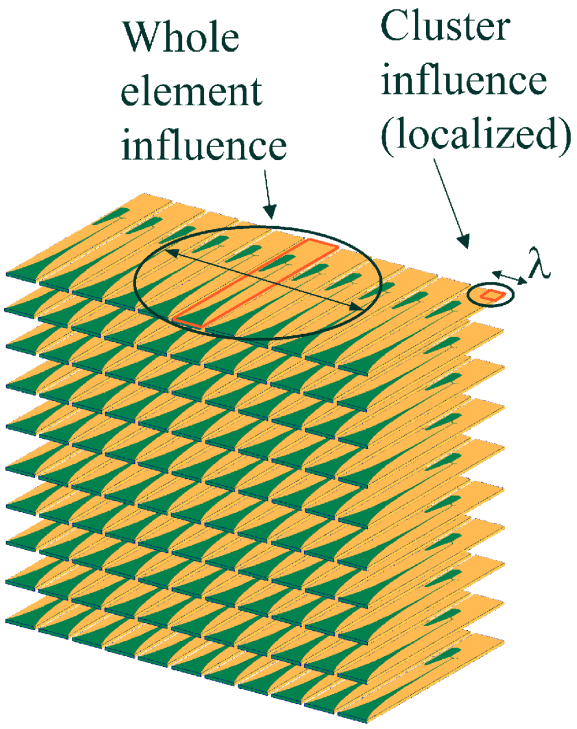


Figure 7.2. Depiction of reduced near-zone influence with intra-element decomposition.

As a visual aid, to help comprehend a common situation where intra-element decomposition is important, observe Figure 7.2. The radius of near-zone influence for the entire element encompasses many of the neighboring elements. All the interactions within the radius must be treated in the near-zone at high computational and storage cost. However, if the element is decomposed into sub-clusters, the region of influence will be potentially reduced to at most the nearest adjacent element, but no further. Again, this decomposition is intended to reduce the near-zone storage cost of AD-FMM to a small and manageable number.

7.2 Implementation considerations

The introduction of the intra-element decomposition significantly increases the choices facing how the method is implemented. For instance, there is a choice as to how much information to leave in the near-zone for preconditioning purposes. In entire-element AD-FMM, as well as standard ADM, the entire element is used as a near-zone preconditioner, in the sense that the isolated element matrix is pre-solved and only the coupling between array elements is iterated over in the solution procedure. With intra-element decomposition, there is now the option of placing some of the self-cell element interactions in the far-zone, thereby reducing the overall cost of the near-zone storage. However, when it comes time to precondition the system solution, this choice will result in poorer solution performance, as significant portions of the intra-element interaction are no longer pre-solved. Hence, it may (or may not) be beneficial to treat some of the intra-element interactions in the near-zone, even when they are technically in the far-zone.

Another implementation issue occurs in deciding when the single-cell AD-FMM model is justified, verses when the problem should be split up into multiple systems. An important consideration to make is that the models examined thus far have typically created well-populated clusters on the FMM grid. An array-type problem that had a distributed element, such that it could not be partitioned well into a continuous grid, would probably be a better candidate for multi-system analysis presented in the next chapter. The array elements examined thus far have been typically planar in nature, and efficient to partition with a regular grid. A larger array element object with a more distributed nature (think large dielectric shell), could result in a less effective clustering model.

7.3 Results

The benefits of using an intra-element decomposition approach are now evaluated. As a first example, consider the benefit in analyzing the structure depicted in Figure 7.2. At the analysis frequency, the elements of the array are 3.3λ in length. For various analysis methods, the required near-zone matrix storage is listed in Table 7.1. Due to the problem complexity, a multi-level FMM solution was not attempted.

Table 7.1. Near-zone Storage Requirements for Structure in Figure 7.2.

FE-BI (Conventional)	340.8GB
ADM	7.5GB
AD-FMM	1.1GB
AD-FMM (Intra-Element)	0.1GB

The benefits of intra-element AD-FMM are very clearly demonstrated in these results for this type of structure. This particular structure is analyzable via any of the array decomposition methods, as compared with the conventional FE-BI approach with much higher storage cost. In particular, intra-element AD-FMM exhibits over three orders magnitude in storage savings for a problem this size. Again, for either of the AD-FMM implementations (whole-element or intra-element), an array will have the same storage listed in Table 7.1 for any number of elements (100×100, 2000×500, etc) in the same two-dimensional configuration.

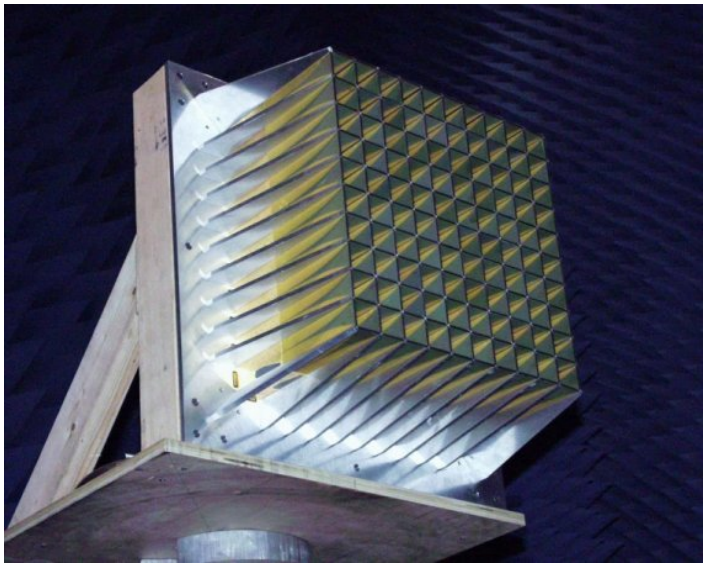


Figure 7.3. NRL array measurement setup.

A more difficult problem is now considered. In this section, validations of the intra-array coupling for the structure pictured in Figure 7.3 are presented. The array consists of a horizontal and vertical egg crate arrangement of the element from Figure 2.11. The horizontally polarized array is 10×11 in size, and the vertically polarized array

is 11×10 in size. In this configuration, only the central 8×8 grid of elements in each polarization are excitable, the surrounding elements being parasitic only. The elements of the array are rigidly connected via a metallic post running the full length of the elements. This post serves to create a current bridge between the elements, allowing the necessary modes required for the array to function at lower frequencies. Additionally, the array is surrounded with supporting fins that have been neglected in the analyses of this thesis for simplicity.

Because the necessary tools have not yet been developed to model this array exactly, in this section an accurate approximate model based on the techniques developed thus far will be used. Though there is not a storage problem for this analysis, there are several difficulties that must be overcome. As a first difficulty, it is not possible to model the structure of Figure 7.3 correctly using a single unit cell. With the single-cell model, the closest approximation to the actual structure is using a 10×10 array of the element shown in Figure 7.4. The resulting approximate structure is shown in Figure 7.5, where it can be seen that the dummy elements along two of the outer edges of the array are unaccounted for. Still, this model produces reasonably close results, and will suffice for the analysis at hand. As a further approximation, for this analysis it is necessary to neglect the connecting post between the elements, as the formulation is not yet equipped to model this current junction. That is, the array analysis assumes that there is a crack between adjacent elements. This is necessary in a single array analysis approach, or the Toeplitz property of the array system will be lost. The nature of this difficulty will be addressed in Chapter 9, where the problem is dealt with efficiently. For this particular array system, the designers went to great lengths to ensure that good electrical contacts

were made between all elements of the array. Hence, in this case, the assumption of bad electrical contacts between adjacent array elements is not a good one.

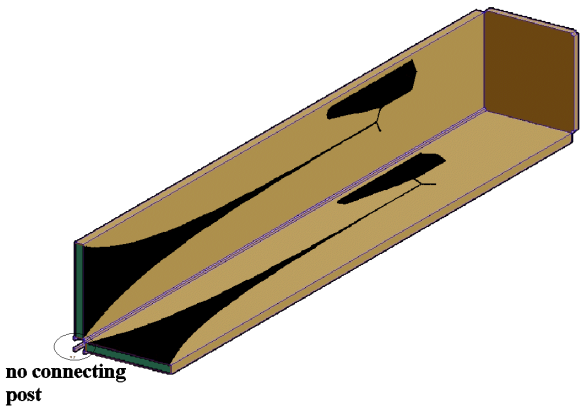


Figure 7.4. Unit cell for the dual polarized array element.

The unit cell element consists of 1,714 FEM unknowns and 3,764 BI unknowns. Using this approximate model, the array in Figure 7.5 consists of 171,400 FEM unknowns and 376,400 BI unknowns, for a total of 547,800 unknowns. In this example, the structure is analyzed at the single center frequency of 3.02GHz. Though this number of unknowns is not significantly high, it is a rather high density of unknowns for an array that is only 10×10 ($4\lambda\times4\lambda$) in extent. The high detail of this array element, combined with its electrical size, make it a particularly difficult element (and array) to model.

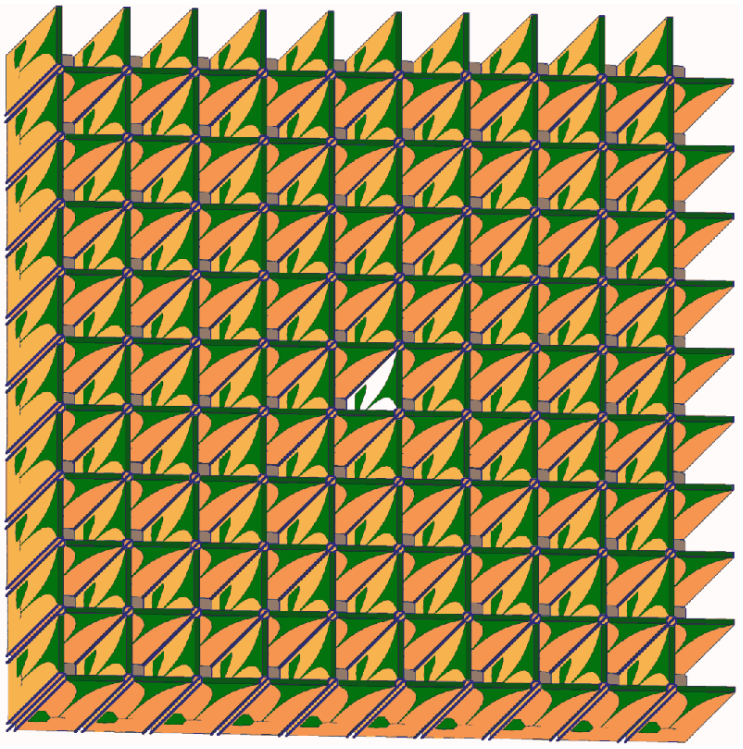


Figure 7.5. Approximate analysis model for intra-array measurements.

First, the required resources for analyzing this structure are evaluated, and the results are listed in Table 7.2. In order to analyze this structure with a conventional FE-BI approach, a total of 1,735GB of storage would be required. This is a completely unreasonable number, suggesting that advanced fast methods are necessary. However, the complexity of this problem is also too high for the conventional MLFMM algorithm used in previous chapters. As a result, it was not possible to even calculate the required resources for analyzing this difficult problem. Had the analysis computation been successful, the result would most likely have shown that the required resource to be far higher than the ideal $O(N \log N)$ storage predicted for MLFMM. This is likely the case, as the dense basis representation required for the array model, is much higher than the typical $\lambda/10$ meshing required to achieve $O(N \log N)$ storage. Using ADM, which uses

Toeplitz storage of the near-zone interactions only, the storage cost reduces to 42GB, almost two orders of magnitude improvement over conventional FE-BI. While this would now be an analyzable problem on a distributed memory system with at least 48MB of memory, a more manageable solution would be preferable. The standard AD-FMM, which uses the entire element as a cluster, reduces the cost down to 21GB. This is only a marginal improvement over the ADM model, due entirely to the fact that the near-zone influence of this element is so high that it encompasses nearly half the elements in the array. The problem of the large near-zone influence is clearly solved by using AD-FMM with intra-element decomposition. With the intra-element decomposition employed, the storage requirements reduce to a manageable 0.4GB – more than three orders of magnitude improvement over the conventional FE-BI approach. For larger arrays of this type, the matrix requirements for conventional methods would increase exponentially, whereas the storage requirements for AD-FMM would remain constant at 0.4GB. Further, this small amount of storage is truly phenomenal, considering the density of field representation required to model this antenna structure.

Table 7.2. Performance and Storage Requirements for the array in Figure 7.5.

	Matrix Storage
Conventional FE-BI	1,735GB
MLFMM	-
ADM	42GB
Standard AD-FMM	21GB
AD-FMM with Intra-Element Decomposition	0.4GB

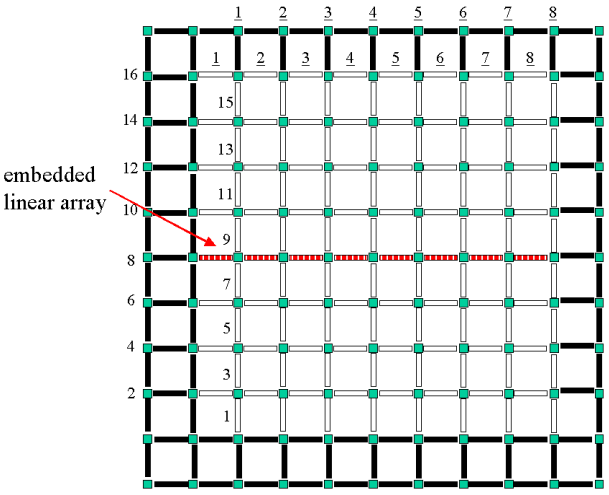
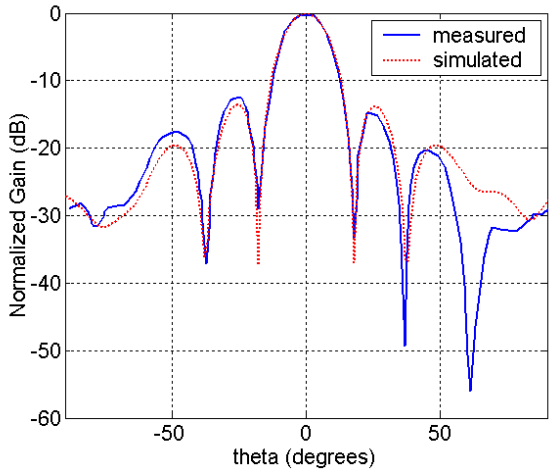
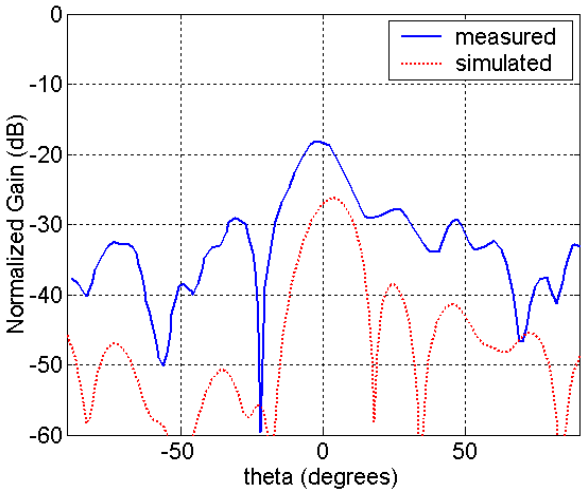


Figure 7.6. Embedded linear array layout.



(a) co-polarized



(b) cross-polarized

Figure 7.7. Patterns for linear array embedded within structure of Figure 7.5.

Consider the use of AD-FMM in computing field patterns of the array structure. As indicated in Figure 7.6, row 8 at the center of the array structure is excited uniformly. The pattern comparisons are shown in Figure 7.7, where both the co-polarized and cross-polarized measurements are compared. The patterns are measured in a compact range, exactly as shown in the setup of Figure 7.3. As indicated by the plots, AD-FMM achieves adequate agreement, even with the approximate model of Figure 7.5. The cross-pol. levels are calculated to be about 8dB lower than the measurements. However, it is expected for calculations of cross-pol. to be lower with ideal calculations.

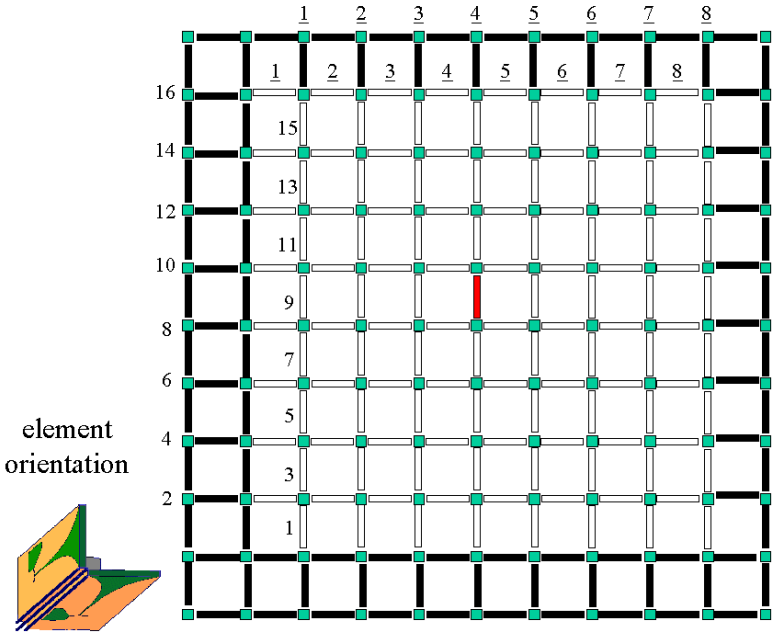


Figure 7.8. Array element indexing guide.

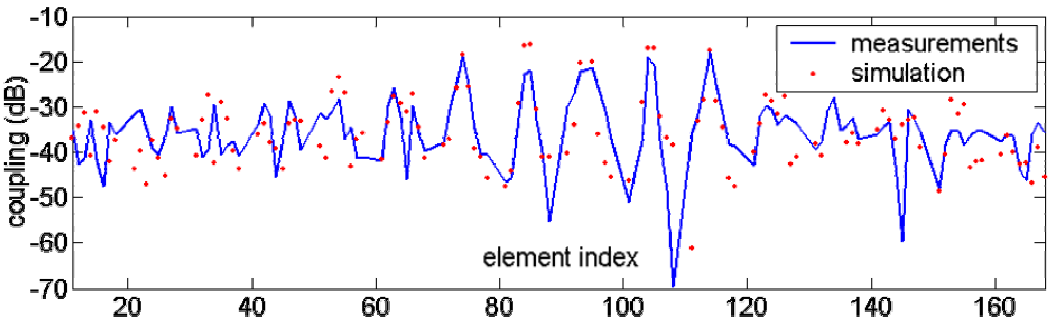


Figure 7.9. Intra-element AD-FMM results comparison.

Consider next a more difficult measurement setup in which an element near the center of the array is excited and then the voltage ratios are measured at other ports of the array (S_{12} = Vout/Vin). The excited element is indicated by the central discolored element in Figure 7.5, and is marked as the shaded element (index 9,4 or 94) in the central region of Figure 7.8. The element indexing can be interpreted as follows. The element positions are indexed by row and column, referring only to the central 8×8 portion of the array. The row numbers are on the range [1 16], counting alternating rows of vertical *and* horizontal elements, and the column numbers are on the range [1 8], with one index per each of the eight elements in a row. The comparisons (at center frequency 3.02GHz) for the S_{12} values are given in Figure 7.9, plotted against the element index given as [row,col]. In other words, an index of 21 is row 2, column 1, whereas an index of 145 is row 14, column 5. As shown, the quality of the agreement between measurements and simulation is relatively close. Most of the results are within 5dB of the expected values, with significant differences of 10dB or more in some cases. The results show better agreement near the excited element, and larger discrepancies at further distances from the fed element. This is due in part because of the multi-path

effects that plague these lower power measurements. In other words, the inaccuracies occur furthest from the radiating source element. However, most importantly, it is critical to recall that the model used here is not exact and neglects the critical supporting posts between elements of the array. This connection is necessary to allow currents to flow directly between the array elements. The importance of these connections is evidenced by the fact that initial designs of the manufactured array operated poorly, until electrically sound connections were made between the elements. It is expected that better agreement can be achieved once the direct current paths between elements are allowed in the simulations. This critical coupling component is developed in Chapter 9.

7.4 Conclusion

It must be commented that AD-FMM with intra-element decomposition achieves the same accuracy as the original FE-BI formulation upon which it is based. In other words, the results of this section would be the same if conventional FE-BI were used to analyze the array. Of course, as addressed above, such an analysis is prohibited by storage and solution time considerations. Likewise, these results would be the same for ADM, or FE-BI with MLFMM. The prime difference here is that using any of the other methods to solve problems of this type is not practical. AD-FMM with intra-element decomposition is the only viable way to approach dense finite array problems from both the standpoint of storage and solution time, when rigor is required. Further, it must be noted that this type of measurement comparison cannot be made with infinite array approximations, semi-infinite array approximations, or even approximations where the

central array elements are modeled with a given field distribution, with only the edge and corner elements are modeled as having different excitations. These methods assume that array elements in the central portion of the array have the same field distributions. Clearly, in a case where only a single array element is excited, this is not a viable assumption.

In summary, a very effective analysis method for finite arrays has been proposed. With the addition of intra-element decomposition to the AD-FMM, the single system array tools have matured to a point where accurate, fast, and low-cost analysis of large finite arrays is possible for both large and small array elements. In the next chapter, these tools will be extended for simultaneous solution of multiple systems, thereby achieving potentially accurate modeling capability for a wide range of real world structures.

CHAPTER 8

RIGOROUS INTERACTION OF MULTIPLE SYSTEMS

Many real-world systems of the finite array type can be decomposed into identical repeating cells, and in some cases, where more than a single array is involved, multiple sets of like cells. In previous chapters, it was shown that array decomposition methods can be used to greatly simplify and accelerate the storage and solution process of finite array-type structures. In this chapter, these array techniques (ADM, AD-FMM) are extended to simultaneously consider multiple finite arrays-type systems, each consisting of a different set (and number) of like elements. An example problem that benefits highly from this type of decomposition would be a finite antenna array (system 1) with an extended ground plane backing it (system 2), and a radome of arbitrary periodicity forming a cover over the array (system 3). The method detailed in this chapter is specifically tailored to address this type of problem. For brevity, the combined techniques of this chapter are referred to as the multi-cell array decomposition method (multi-cell ADM or multi-cell AD-FMM), or simply multi-system ADM/AD-FMM.

The multi-cell array decomposition method is applicable to complex structures with repeating features, which are treated as array structures, and also arbitrary systems, which can be treated simply as single element arrays. It is true that a completely arbitrary

structure devoid of repeating features will not benefit greatly from the array decomposition methods presented in previous chapters. However, in some cases, at least a portion of a structure could benefit from array decomposition of repeating features. One approach might be to neglect including the non-repeating portion of the structure in the analysis, an approximation that may not be very accurate. Alternatively, using the approach presented here, the arbitrary portion of any system that cannot be decomposed into repeating cells can be treated as a separate, single-element array, and interacted simultaneously with other systems without array-type decomposition. As an alternative, it is also possible to use the FMM-FFT technique of Chapter 3 to decompose completely arbitrary structures into regular clustering grids, thereby allowing cluster-level decomposition, and then use the methods of this chapter to achieve cross-system decomposition between the arbitrary structures. This capability is inherent in the intra-element decomposition method presented in the previous chapter.

This chapter is intended as an introduction to multi-system analysis. Though the concepts of this chapter can be used for fairly accurate solutions to composite array-type structures, the algorithms still lack the ability to model current continuity between array elements, and therefore will not be accurate for all problem types. This critical piece of the coupling equation was purposely left out up to this point, as it requires concepts of all the preceding chapters, including concepts presented in this one, to correctly model the mechanism of currents flowing directly between elements of the array.

8.1 Introduction to Multi-Cell Decomposition Approach

Much like the conventional ADM, the multi-cell ADM decomposition is carried out across *common* dimensions. Here, the focus is again strictly on spatial or translational-type dimensions having translational symmetry. A translational dimension is defined solely by a spacing parameter δ_d and a direction $\hat{\rho}_d$, where *d* is the dimension. By this definition, dimensions can be differentiated by a unique spacing between elements, or by aligning an array lattice in a different spatial direction, or both. In this way, complex systems can be constructed using multiple dimensions of unique specification. As in previous chapters, it is mandatory that element indices in each dimension be consecutively numbered. To enable cross-system decomposition, an arbitrary number of systems are constructed from a table of common dimensions *d*, ranging from *d*=1..*ndims*. For each dimension, a self-system Toeplitz property will exist that can be used to decompose *intra*-system interactions, as described in previous chapters. Similarly, in this chapter, for systems sharing a common dimension, a cross-system Toeplitz property will exist for *inter*-system element interactions that can be exploited to decompose the system matrix even further.

Because of the complexity of the multi-system analysis scheme, this chapter begins with a simple example in which three systems of a single common dimension are simultaneously interacted. These systems are shown in Figure 8.1. Each system consists of an independent number of elements, denoted $n_{d,s}$, where *d* denotes the dimension, and *s* denotes the system. However, all three systems share a common element spacing of δ_1 and the axis of the array $\hat{\rho}_1$ is the same for all systems. Systems 1, 2, and 3 are defined

as having (BI) unknown counts per array element of m_1, m_2, and m_3, respectively. In a conventional approach to this problem, the total matrix storage requirements would be $O(n_{1,1}^2 m_1^2 n_{1,2}^2 m_2^2 n_{1,3}^2 m_3^2)$. However, using a combination of self-system and cross-system decomposition, this cost can be reduced to

$$O(\sum_{t=1}^{3}\sum_{s=1}^{3}(n_{1,s}+n_{1,t}-1)m_s m_t), \tag{8.1.1}$$

simply by exploiting the block-Toeplitz features of the system. This is the maximum possible near-zone storage cost, assuming far-zone decomposition (AD-FMM) has not been implemented. The computational and storage costs are explained below.

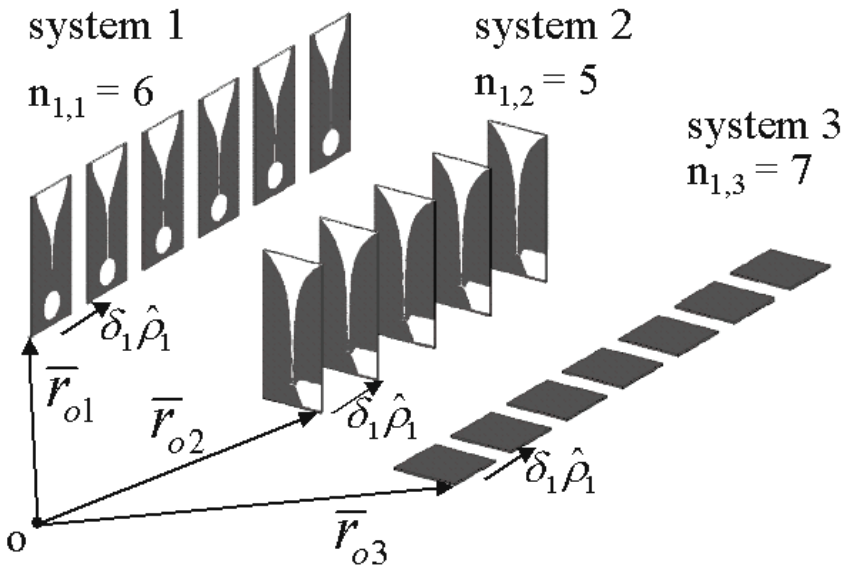
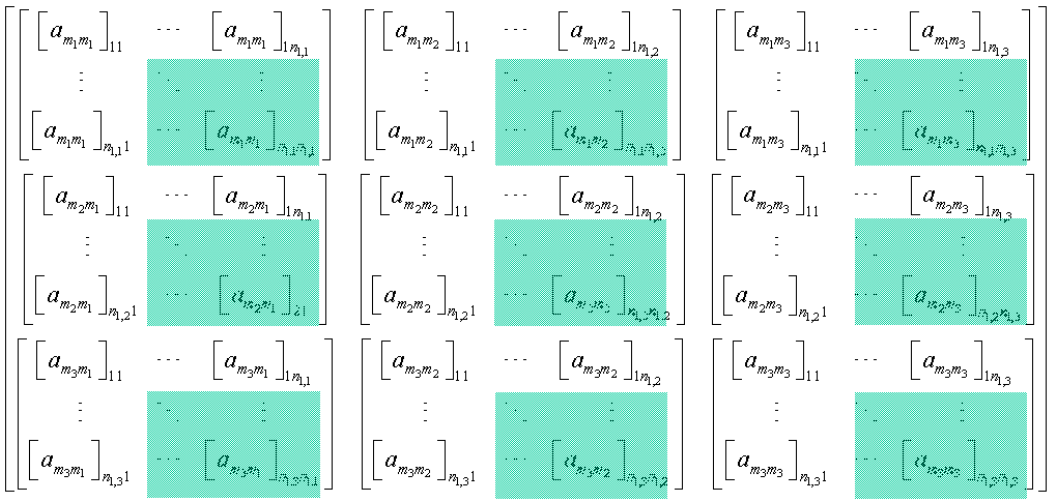


Figure 8.1. Illustration of dimensional decomposition for multiple structures.

For the three-system configuration shown in Figure 8.1, the near-zone matrix system (or far-zone translation matrix) will be of the form


.(8.1.2)

In (8.1.2), the redundant terms have been shaded in, accounting for the cost reduction given above. The shading gives a general idea that Toeplitz-based reductions are taking place across multiple systems. The larger sub-arrays along the diagonal of (8.1.2) represent the coupling within the independent systems (intra-array coupling). The larger sub-arrays at off diagonal locations represent the coupling between systems (inter-array coupling). It can be seen that for all cases, only the first column and first rows of the element interaction sub-matrices $\left[a_{m_t m_s}\right]$ are being stored in both the self-system and cross-system coupling sub-matrices. Each sub-matrix $\left[a_{m_t m_s}\right]_{i_{1,t} i_{1,s}}$ of (8.1.2) is of size $m_t m_s$, and represents the interaction of source basis elements m_s with testing basis elements m_t, for source array element number $i_{1,s}$ onto testing array element number $i_{1,t}$, between the source system *s* and the testing system *t*. Again, both systems share a single common dimension *d*=1.

In (8.1.2), there are nine sub-systems, one each for the intra-system coupling, and one each for cross-coupling between systems. Because these three systems share a common dimension, the sub-systems are all non-symmetric block-Toeplitz, and hence

each individual sub-system can be reduced from a conventional storage cost of $O(n_{1,t}m_t n_{1,s}m_s)$ down to $O((n_{1,s}+n_{1,t}-1)m_s m_t)$, where $n_{1,s}$ and $n_{1,t}$ are the number of source and testing array elements in the source and testing systems, respectively. This reduction corresponds to the unshaded portion of each sub-system in (8.1.2). Further, the contribution of each sub-system to the matrix-vector product operation of the iterative solution process can be accelerated with the equivalent of $m_s m_t$ fast Fourier transforms (FFT) of length $n_{1,s}+n_{1,t}-1$. This assumes that only ADM is applied, and not AD-FMM. If AD-FMM is employed, the amount of near-zone storage will be dependent both upon the array element spacing within the individual systems, as well as the separation between the systems. As an example, if the systems are far-enough separated such that they are completely in the far-zone of each other, then there will be no near-zone storage of the cross terms for AD-FMM, and hence all off-diagonal terms in (8.1.2) would be shaded. For AD-FMM, the far-zone is determined by the cluster size for each system, and not the system size.

As mentioned, for AD-FMM, the translation matrix will have the same general form as (8.1.2), except that each element-interaction sub-matrix $\left[a_{m_t m_s}\right]$ will be replaced with a translation operator $\{\tau_L\}_{m_t m_s}$, containing a number of *k*-space entries dependent on the radii of the source and testing array elements (or sub-clusters if intra-element decomposition is employed). Specifically, the number of *k*-space directions $K \approx L^2$, where $L > k_0(D_s + D_t)/2$. In this expression, k_0 is the wave number of free-space, whereas D_s and D_t are the radii of the source and testing clusters, respectively. In multi-

system decomposition, it is possible that these cluster diameters will be different, particularly when whole-element AD-FMM is employed.

This simple example has been used to introduce the concepts of multi-cell analysis. In the next section, these relations will be generalized to multiple systems of multiple dimensions and extended to generate a most robust implementation.

8.2 Generalized Multi-System Multi-Dimensional Decomposition

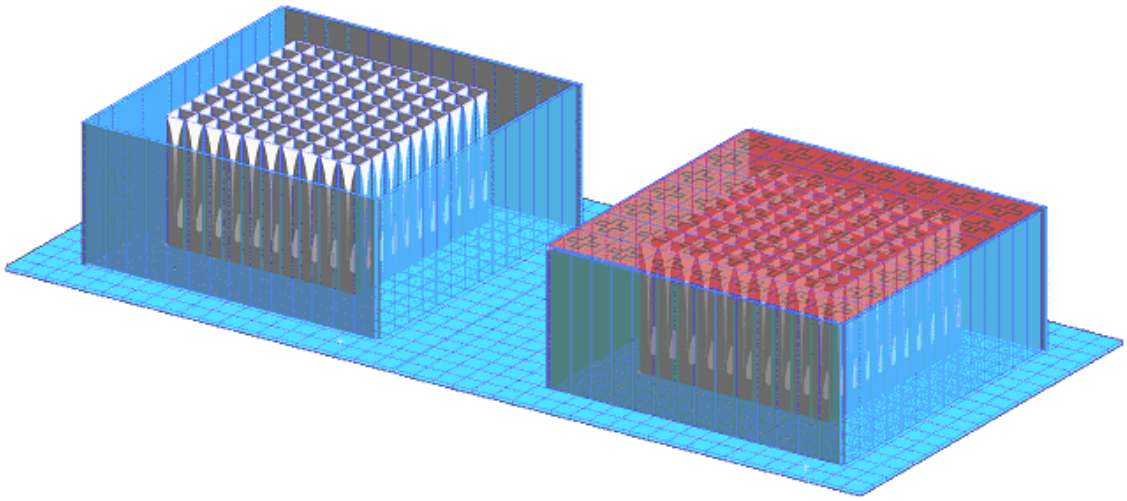


Figure 8.2. Arbitrary system interaction in multiple dimensions.

Having introduced the concept of multi-cell decomposition for a simple example, a general formulation for decomposing multiple systems with arbitrary numbers of dimensions is now presented. A more complicated structure that may be considered is shown in Figure 8.2. The structure consists of two dual-polarized tapered-slot antenna arrays in an egg crate configuration. The horizontal elements form 10×11 arrays, while the vertical elements form 11×10 element arrays. Within the arrays there are other

details that will be overlooked here, such as connecting posts between elements. The arrays are backed by a common ground plane of finite extent, and are surrounded by metallic shielding. One array is shown as covered by a frequency selective surface of a different periodicity. A potential solution approach would be to analyze this configuration as an arbitrary structure. However, it is obvious that some amount of decomposition on redundant features might be beneficial. This particular structure has an infinite number of decomposition options available. The approach taken in this chapter would be to decompose the independent systems, and then partition the continuous ground plane and shielding structures using the same lattice as the array dimensions. Using this approach, some portions of the ground plane or shielding structures may be left over after the decomposition, since the array lattice may not exactly partition these structures into like cells. The excess portions at the ends or corners could be discarded from the analysis, or potentially treated as another system. The benefits of such an approach are now examined.

The necessary tools for the multi-dimensional, multi-system near-zone decomposition are considered first. The generalized form of the near-zone kernel for interacting multiple systems with arbitrary dimensions is of the form

$$g(R) = \frac{e^{-jk_0 R}}{4\pi R}, \tag{8.1.3}$$

where

$$R = \left| \begin{array}{l} \overline{d}_m - \overline{d}_{m'} + (i_{1,t} - i'_{1,s})\delta_1 \hat{\rho}_1 + (i_{2,t} - i'_{2,s})\delta_2 \hat{\rho}_2 + \\ \cdots + (i_{ndims,t} - i_{ndims'})\delta_{ndims} \hat{\rho}_{ndims} + \overline{r}_{ot} - \overline{r}'_{os} \end{array} \right|. \tag{8.1.4}$$

This expression is appropriate for an arbitrary number of dimensions given by *ndims*, as well as an arbitrary number of systems, *nsys*. For multi-system analysis, the systems can

be numbered in any sequence, though it is recommended for simplicity that the source and testing systems *s* and *t*, be numbered consecutively from $s,t = 1..nsys$. An analysis of the near-zone kernel (8.1.3) is performed by walking through the decomposed coupling path from the source basis function m_s to the testing basis function m_t. First, it is necessary that the offset between systems be accounted for, where the source and testing systems are referenced to the global origin via the global vectors $\overline{r}_{os}$ and $\overline{r}_{ot}$, respectively. This is in contrast to previous decomposition models, in which the global offset played no role. As previously, the elements of the array systems are required to be numbered consecutively as $i_{d,s} = 1..n_{d,s}$, where $i_{d,s}$ is the element index in dimension *d* for system *s*, and $n_{d,s}$ is the total number of elements. The coupling influence must be transformed through each dimension of the array systems as $(i_{d,t} - i'_{d,s})\delta_d \hat{\rho}_d$, where δ_d is the element spacing in dimension *d*, and $\hat{\rho}_d$ is the direction of the transform dimension. The source and testing array element indices are given by $i'_{d,s}$ and $i_{d,t}$, respectively, for the given dimension *d*. As stated in the previous section, all systems are constructed from the same set of common dimensions. In the most general formulation, the minimum number of elements per dimension for any system is 1, and hence it is necessary to transform through *all* the dimensions for both systems. That is to say, if *ndims*=2, and system *s* is a linear array with *n* elements in dimension 1, it still has 1 element in dimension 2. This is an important concept to master in order to fully appreciate multi-dimensional, multi-system analysis, and is required for general implementation.

For the near-zone storage then, the unique array element interaction sub-matrices in Toeplitz storage are indexed as

$$\Pi[i_{1,t}-i'_{1,s}, i_{2,t}-i'_{2,s}, \ldots, i_{ndims,t}-i'_{ndims,s}, m_t, m_s] = \left[a_{m_t m_s}\right]_{(i_{1,t}-i'_{1,s})(i_{2,t}-i'_{2,s})\cdots(i_{ndims,t}-i'_{ndims,s})}, \tag{8.1.5}$$

where these matrix entries are determined by inserting the multi-system, multi-dimensional kernel (8.1.3) into the EFIE and MFIE equations (2.2.10) and (2.2.11). The Toeplitz matrix structure for the near-zone decomposition has the same number of dimensions as the array systems, with sub-matrix sizes dependent on the number basis interactions for the elements or clusters. For standard ADM, this structure can be transformed via FFT for fast matrix-vector multiplication, though this step is avoided for AD-FMM. The matrix system will be of the form (8.1.2), though it can be more generally expressed as

$$\begin{bmatrix} \begin{bmatrix}\text{intra-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{11} & \begin{bmatrix}\text{inter-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{12} & \cdots & \begin{bmatrix}\text{inter-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{1nsys} \\ \begin{bmatrix}\text{inter-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{21} & \begin{bmatrix}\text{intra-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{22} & & \vdots \\ \vdots & & \ddots & \\ \begin{bmatrix}\text{inter-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{nsys1} & & \cdots & \begin{bmatrix}\text{intra-}\\ \text{system}\\ \text{coupling}\end{bmatrix}_{nsys,nsys} \end{bmatrix} \tag{8.1.6}$$

The intra-system coupling matrices occur along the system diagonal, whereas the inter-system coupling matrices appear off the main diagonal. The intra-system sub-matrices will be $N \times N$. However, it is important to realize that the inter-system sub-matrices are not $N \times N$ – they will be $N_1 \times N_2$, where in most cases $N_1 \neq N_2$, and hence the sub-systems are not square. This has to be considered, particularly if the cross-system interactions are to be accelerated via the FFT.

For Toeplitz storage of multi-system ADM, in which AD-FMM has not been implemented, the number of terms will be

$$O(\sum_{s=1}^{nsys}\sum_{t=1}^{nsys}\prod_{d=1}^{ndims}(n_{d,s}+n_{d,t}-1)m_s m_t)\,. \tag{8.1.7}$$

Similarly, for ADM the interaction between source system *s* and testing system *t* can be accelerated using $m_t m_s$ simultaneous FFTs of size/dimension

$$(n_{1,s}+n_{1,t}-1)\times(n_{2,s}+n_{2,t}-1)\times\cdots\times(n_{ndims,s}+n_{ndims,t}-1)\,. \tag{8.1.8}$$

Efficient consideration of cross-system interactions implies a need for multi-dimensional FFT algorithms. This will be applied to the far-zone interactions of AD-FMM, or the near-zone interactions of ADM.

Now the far-zone components of the multi-system analysis approach are examined. Here the additional complexity of intra-element decomposition is also addressed. The generalized far-zone translation operator for multiple systems and arbitrary numbers of dimensions will be of the form

$$\mathrm{T}_L[k_0|\overline{\rho}|,\hat{k}\bullet\hat{\rho}]=\sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(k_0|\overline{\rho}|)P_l(\hat{k}\bullet\hat{\rho}))\,, \tag{8.1.9}$$

where

$$|\rho|=\left|\begin{array}{l}(i_{c1,t}-i'_{c1,s})\delta_{c1}\hat{\rho}_{c1}+(i_{c2,t}-i'_{c2,s})\delta_{c2}\hat{\rho}_{c2}+(i_{c2,t}-i'_{c2,s})\delta_{c3}\hat{\rho}_{c3}+\\(i_{1,t}-i'_{1,s})\delta_1\hat{\rho}_1+(i_{2,t}-i'_{2,s})\delta_2\hat{\rho}_2+\cdots+(i_{ndims,t}-i'_{ndims,s})\delta_{ndims}\hat{\rho}_{ndims}+(\overline{r}_{o,t}-\overline{r}'_{o,s})\end{array}\right|, \tag{8.1.10}$$

and

$$\hat{\rho}=\frac{\begin{array}{l}(i_{c1,t}-i'_{c1,s})\delta_{c1}\hat{\rho}_{c1}+(i_{c2,t}-i'_{c2,s})\delta_{c2}\hat{\rho}_{c2}+(i_{c2,t}-i'_{c2,s})\delta_{c3}\hat{\rho}_{c3}+\\(i_{1,t}-i'_{1,s})\delta_1\hat{\rho}_1+(i_{2,t}-i'_{2,s})\delta_2\hat{\rho}_2+\cdots+(i_{ndims,t}-i'_{ndims,s})\delta_{ndims}\hat{\rho}_{ndims}+(\overline{r}_{o,t}-\overline{r}'_{o,s})\end{array}}{|\rho|}\,. \tag{8.1.11}$$

The system offsets $\overline{r}_{o,t}, \overline{r}'_{o,s}$ and the dimensional transformations for the far-zone interactions are treated exactly the same as for the near-zone interactions. Namely, the global system offsets must be referenced to relate separation of the individual systems, and it is necessary to transform the source influence through all *ndims* dimensions. Though the assumption of three clustering dimensions has been made, in theory a multi-dimensional decomposition could be applied to the clustering as well. To use intra-element decomposition most effectively in multi-system analysis, it should be that the cluster dimensions are the same for every system in the simultaneous analysis. That is, the cluster size and lattice should be the same for all systems, thereby allowing cross-system decomposition across the cluster dimensions as well. It is implicit in the definition of (8.1.9) that this be the case, otherwise it would not be correct to decompose the system based on the difference in cluster indices $i_{cd,t} - i'_{cd,s}$. Hence, even for completely arbitrary systems that are not of the array-type, so long as the same cluster grid layout is used for each of the systems, the inter-system interactions can be decomposed for storage savings and solution acceleration as well.

The intention is that the translation operators are stored in Toeplitz format for fast FFT solution procedures. The multi-dimensional, multi-system form of the Toeplitz translation operator for each source and testing *system* interaction is of the form

$$[\mathrm{T}_L]_{t,s} = \begin{bmatrix} \ddots & \vdots & \iddots \\ \cdots & \{\tau_L\}_{(i_{c1,t}-i'_{c1,s})(i_{c2,t}-i'_{c2,s})(i_{c3,t}-i'_{c3,s})(i_{1,t}-i'_{1,s})(i_{2,t}-i'_{2,s})\cdots(i_{ndims,t}-i'_{ndims,s})} & \cdots \\ \iddots & \vdots & \ddots \end{bmatrix}. \quad (8.1.12)$$

As usual, the number of *k*-space directions *K* in $\{\tau_L\}$ will be proportional to the cluster sizes for the source and testing system clusters. As a note, the same rules for cluster

interactions apply for multiple systems as apply for single systems. Namely, clusters must be separated a distance proportional to the sum of the interacting cluster radii as defined in previous chapters to be considered in the far-zone. The far-zone interactions can be accelerated with K simultaneous FFTs of size and dimension

$$\begin{aligned}&(n_{c1,s}+n_{c1,t}-1)\times(n_{c2,s}+n_{c2,t}-1)\times(n_{c3,s}+n_{c3,t}-1)\times\\&(n_{1,s}+n_{1,t}-1)\times(n_{2,s}+n_{2,t}-1)\times\cdots\times(n_{ndims,s}+n_{ndims,t}-1)\end{aligned}. \tag{8.1.13}$$

The cluster dimensions and array dimensions can be ordered as desired without loss of generalization, with the forethought that the dimensions with the most entries should be transformed first for highest efficiency.

The storage cost of the translation operators will be proportional to

$$O\left(\sum_{s=1}^{nsys}\sum_{t=1}^{nsys}\left[\prod_{d=1}^{ndims}(n_{d,s}+n_{d,t}-1)\prod_{i=1}^{3}(n_{ci,s}+n_{ci,t}-1)K_{s,t}\right]\right). \tag{8.1.14}$$

It is not typically the case that the translation operator storage is a significant portion of the storage requirements. However, if the number of clusters in each dimension are excessively large compared to the basis density of the clusters, it is possible for the translation storage to become more significant.

8.3 Multi-System Results

As a first example, the array in Figure 7.5 will be reconsidered from a multi-system approach. The composite structure is constructed from four distinct array systems, as depicted in Figure 8.3. The four systems include a 10×10 array of horizontally polarized elements, a 10×10 array of vertically polarized elements, a 10×10 array of connecting posts, and a 10×10 array of back plane material. Of course, with the multi-system approach it is possible to construct near exactly the structure in Figure 7.3, using the 10×11 array of horizontally polarized elements, the 11×10 array of vertically polarized elements, and a corresponding 11×11 array of connecting posts. However, the point of this example is to compare the single-system and multi-system approach to the same problem. As a note, the connecting posts employed here do not provide electrical contact between the elements – they are only designed to correctly simulate the scattering conditions of the finite array environment.

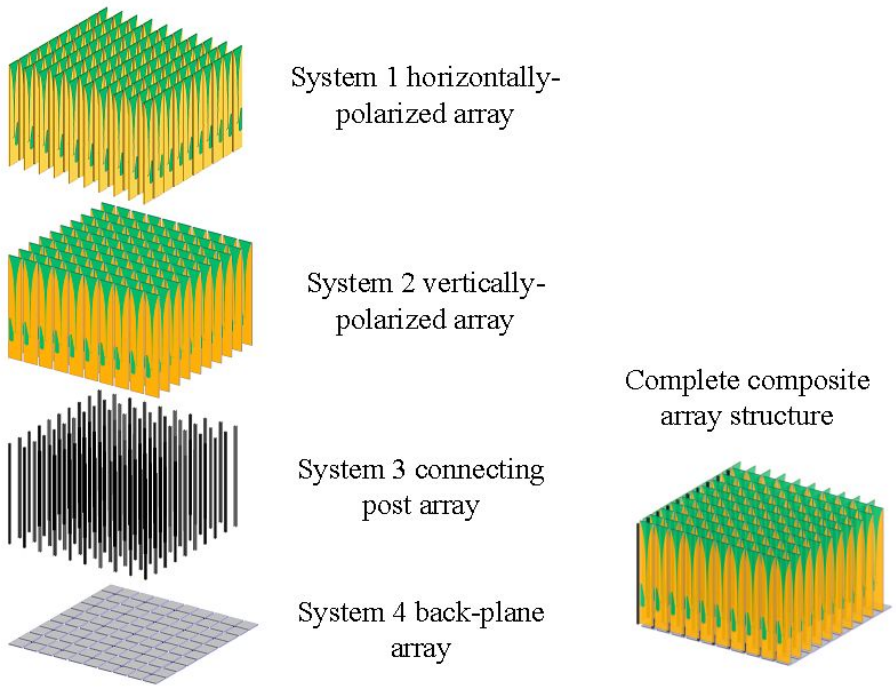


Figure 8.3. Construction of composite finite array structure using multi-system approach.

Table 8.1 compares the storage requirements for a multi-system approach to this structure with the single system approach of Chapter 7. These results show that for standard ADM, the single-system approach has identical storage requirements as the multi-system approach. This is to be expected, as the matrix system has the same exact entries in each case, simply in a different arrangement. However, for ADM there will be a difference in the matrix system preconditioning approach. This difference will result in larger storage requirements for the single-system method, and consequently a smaller number of iterations to achieve convergence. That is, in the single-system approach, the self-cell matrix block contains more information, corresponding to the four self-cells of the multi-system approach, plus the strong mutual-coupling interaction of these cells. In the single-system approach, these interactions are pre-solved and used to accelerate the system

solution, whereas in the multi-system approach, the strong mutual-coupling interactions between the four self-cells must be iteratively solved for, at the expense of a larger number of iterations for convergence.

Unlike the case for the ADM, the two AD-FMM comparisons show a slight improvement for the multi-system approach verses the single-system approach to the array problem. This reduction is a consequence of the multi-system approach achieving a more efficient clustering scheme. That is, the multi-system clustering grid conforms more closely to the array structure, resulting in a lack of empty clusters, and hence more efficient storage. Thus, in the case of the intra-element AD-FMM approach, the storage reduces from the already low 0.4GB down to 0.25GB, nearly four orders of magnitude improvement over conventional FE-BI at a storage cost of 1,735GB.

Table 8.1. Comparison of Near-zone Storage for Single- vs. Multi-System Approaches to Array Problem.

Near-zone matrix Storage	Single-system approach	Multi-system approach
ADM	42GB	42GB
Standard AD-FMM	21GB	18GB
AD-FMM with Intra-Element Decomposition	0.4GB	0.25GB

Note that when a multi-cell approach to this problem is utilized, there are nearly an infinite number of combinations that could be chosen for analysis. For example, using the multi-system approach with only the element previously used in the single-system analysis (see Figure 7.4), it is possible to model the composite structure as two 5×10

arrays, or perhaps four 5×5 arrays, to name just a few alternatives. However, the greatest matrix savings is realized with the full 10×10 element approach, as this results in the removal of more redundant coupling terms, and allows larger, more efficient FFT operations. As a further note, care must be exercised in using multi-cell models in which the systems have strong coupling. An example of strong coupling occurs when the elements of the two systems are interleaved, a condition that leads to a significant increase in iterations for convergence.

One advantage the multi-system approach has over the single-system approach is that the array geometry of Figure 7.5 can be modeled exactly, minus the good electrical contacts between adjacent elements. That is, the full 10×11 horizontally polarized and 11×10 vertically polarized arrays can be modeled, whereas the previous approaches neglect the outer rows of dummy elements on both sides. The cost of the two approaches are compared in Table 8.2. The exact model requires 606,028 unknowns, 10.6% more than the approximate model. For ADM, this added cost results in an 11.9% increase in near-zone storage cost. However, as expected, the near-zone cost of the AD-FMM methods do not increase at all, since the near-zone influence remains the same.

Table 8.2. Comparison of Storage Cost for Exact and Approximate Representations of Array in Figure 7.5.

Near-Zone Matrix Storage	Approximate Model	Exact Model
Standard FE-BI	1,735GB	2,134GB
ADM	42GB	47GB
Standard AD-FMM	18GB	18GB
AD-FMM with Intra-Element Decomposition	0.25GB	0.25GB

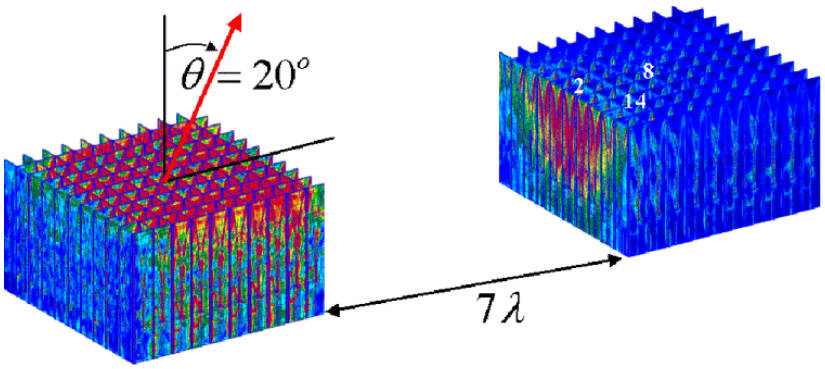


Figure 8.4 Simulation geometry for two-array coupling problem.

In this example, two arrays of the type depicted in Figure 7.5 are considered simultaneously. Again, the unit cell of Figure 7.4 is used. The arrays are placed seven wavelengths apart at the analysis frequency of 3.02GHz, as shown in Figure 8.4. For various solution approaches, the results are tabulated in Table 8.3. The conventional FE-BI approach considers the entire problem (both arrays) as a single arbitrary structure. The two-array approach treats each array as an independent system, where as the multi-dimensional, single-system approach treats the entire structure as a single multi-dimensional array. Recall that for the ADM, the multi-dimensional approach resulted in exactly ¾ the matrix storage of the two-array approach. On a similar note, using AD-FMM with a multi-dimensional, single-system approach results in exactly ½ the matrix storage of the two-array, multi-system approach. That is, the multi-system approach requires two self-coupling systems (one for each 10×10 array), whereas the multi-dimensional, single-system approach has only a single self-coupling matrix used for both arrays, as allowed with Toeplitz storage. This accounts for the storage difference between the single- and multi-system approaches. For the AD-FMM cases, the cross-coupling terms are handled in the far-zone.

Table 8.3. Comparison of Advanced Solution Methods for Array in Figure 8.4.

Solution Method	Matrix Storage
Conventional FE-BI	6,940GB
Two-Array Multi-Sys ADM	168GB
Multi-Dim Single-Sys ADM	126GB
Two-Array Multi-Sys Whole-Element AD-FMM	36GB
Multi-Dim Single-Wys Whole-Element AD-FMM	18GB
Two-Array Multi-Sys Intra-Element AD-FMM	0.5GB
Multi-Dim Single-Sys Intra-Element AD-FMM	0.25GB

The array layout of Figure 8.4 is now analyzed for inter-array coupling. The actual measurement setup is shown in Figure 8.5. In the measurement configuration, the arrays are placed on a metallic ground plane approximately $24\lambda \times 12\lambda$ in size. For the measurement, the 8×8 array of horizontal elements in the first array is excited uniformly and scanned electronically at increments of 10 degrees, towards the second (receiving) array. At each scan increment, the coupling is measured as various ports of the receive array. The locations of the receive elements are depicted in Figure 8.4.

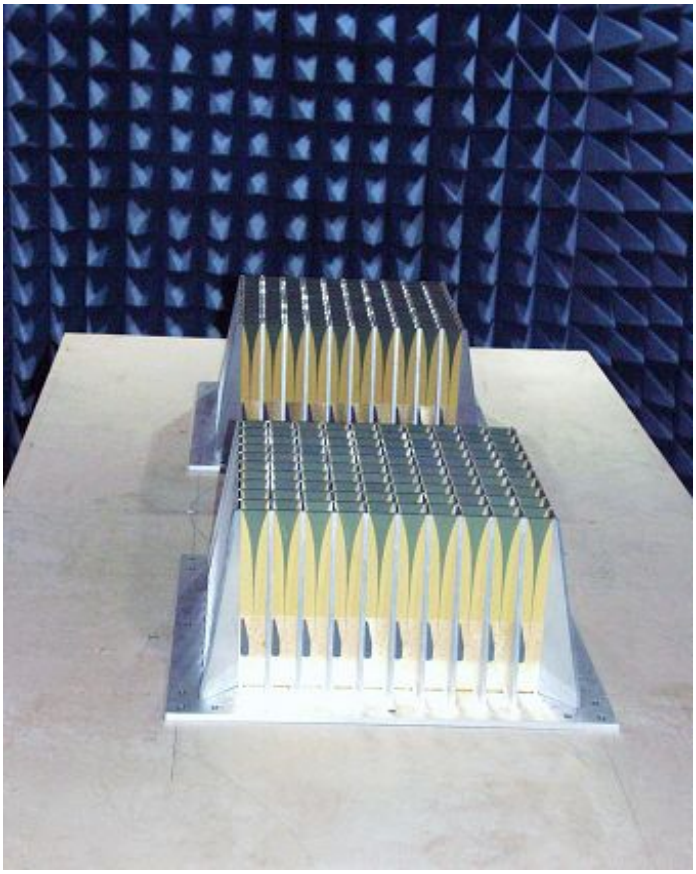


Figure 8.5 Measurement setup for intra-array coupling.

Figure 8.6 shows the inter-array coupling results for the measurement setup, comparing measurements with the simulation geometry of Figure 8.4. It is observed that the agreement between measurements and simulations is reasonably close, though there are marked differences. The simulation geometry does not have the ground plane of the measurement setup, and recall from Chapter 7 that the array model is not exactly the same as the measured structure, as it is missing two outer rows of dummy elements, and some critical structural supports (connection posts between elements and supporting fins on the outside of the structure. Though a more exact simulation model is

possible with the tools developed in this thesis, this example serves to demonstrate the inter-array coupling capabilities of AD-FMM.

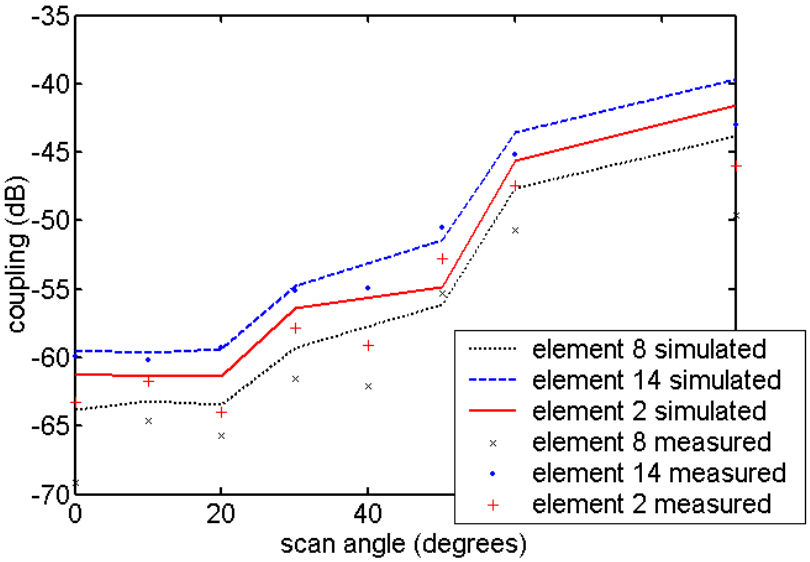


Figure 8.6 Inter-array coupling vs. scan angle.

8.4 Conclusion

The developments presented through this chapter represent a collection of very robust analysis methods for the rigorous analysis of nearly any type of structure, from entirely arbitrary structures to large finite arrays, as well as any combination thereof. Of course, these developments have stopped short of being able to analysis an entire navy ship at microwave frequencies, an interesting problem and a subject of considerable interest in industry. However, the objective of this thesis is to push rigorous analysis methods for electromagnetic structures as far as possible, without resorting to the approximate methods that would be necessary to handle a problem the size of an entire

ship. Indeed, the developments of this thesis have pushed the analysis capabilities of rigorous analysis methods to new limits. However, a critical component of inter-system coupling remains to be dealt with. The developments of past chapters model arbitrary volumetric structures as closed domains. In the case of array elements, each element is treated as an independent closed structure. When two closed volumetric structures are placed near to each other, such as in a finite array where adjacent array elements touch, the formulation developed thus far treats this condition as an electrical disconnect, as if an infinitesimal gap exists between the element structures. The subject of the next chapter deals with how to bridge this gap, such that electrical currents can flow freely between element domains.

Regarding implementation issues, the methods in this thesis have made considerable progress towards allowing large, complex structure analysis on conventional personal computers and smaller workstations. However, it is worth commenting that the methods presented in this thesis can very easily be implemented on distributed memory networks with reasonably scalable results. For the AD-FMM methods, the near-zone interaction matrices between array elements and even between multiple systems can be used to create simple matrix division schemes for distributed storage and processing. This is most advantageous, as the near-zone interactions represent the most expensive portion of the solution process. Likewise, inter-system and intra-system far-zone coupling can be distributed using the same pre-existing sub-matrix divisions. This approach is very effective and far simpler than an arbitrary scheme of decomposing the matrix interactions for distributed processing.

CHAPTER 9

INTRA-CELL DOMAIN CONNECTIVITY

The thesis ends with a chapter on domain connectivity. This issue is addressed last, owing to the fact that large, interconnected array problems require concepts from all the preceding chapters to correctly and thoroughly address the inherent difficulties. In previous chapters, the analyses have been limited to applications in which the domains of individual systems or array elements are closed and coupled via the free-space Green's function only. As a consequence of this choice in formulation, if the closed volumetric domains of any two systems (such as adjacent array elements) touch, the condition is modeled as though a bad electrical contact or gap exists, and no current is allowed to flow directly between domains. In some situations, this modeling choice may be more correct than to assume current flows freely between the systems. However, in situations that require direct current flow between element domains, such as for broadband arrays supporting low frequency modes across multiple elements of the array, it will be necessary to model the electrical contact between adjacent systems for correct results.

For the gap condition, where the coupling between adjacent systems is carried out through the radiation mechanism of the free-space Green's function, system interactions can be conveniently treated with the integration testing procedure given in previous chapters. However, when the domains of closed volumetric structures are joined

electrically, it is necessary to enforce current continuity by defining a current that flows directly between adjacent domains. To do this, one must introduce field or current constructs in the form of additional basis functions to handle the resulting junction condition. While this additional complexity may seem trivial at first glance, in the context of array systems, where symmetry must be maintained, junction treatment is far from trivial. Further, under many conditions it will be necessary to open the closed domain structures, a situation that must be correctly remedied for formulations relying on a closed structure (MFIE).

For array-type problems, the required decomposition is quite cumbersome, with many components to consider, each of which deserves careful attention. Before approaching the array problem, connectivity between arbitrary non-array-type systems will be examined. Under certain conditions, such as when entire faces of adjacent domains overlap, the complexity of the systems may be decreased through a condensation process, resulting in fewer unknowns of lower complexity than prior to joining the systems. Conversely, when only the edges of adjacent systems are collocated, it is necessary to *increase* the complexity of the overall system, introducing additional elements to bridge the gap between adjacent systems, where no elements existed previously. For completeness, these conditions will be addressed separately.

In previous chapters, an attempt was made to validate the intra-array coupling of the structure in Figure 7.3. Using the disjoint domain approaches in those chapters, agreement with measurements was only relatively close. Using the domain interconnectivity developments of this chapter to augment advanced array methods such

as AD-FMM, it should be possible to achieve better agreement with the intra-array measurements.

9.1 Connecting System Domains

Consider the general case of two arbitrary, closed-domain systems with a portion of each domain overlapping, as suggested in Figure 9.1. In the figure, the systems have been separated to show the features of the inner structure, but it is clear which faces of the separate domains overlap. In this illustration, the systems or cells are drawn as cubic structures with a regular mesh grid. Though the cells have been illustrated in this way, there is no requirement for cubic cells or regular grids, and certainly no guarantee of this condition for arbitrary structures. The only condition for proper enforcement of boundary conditions is that the overlapping faces have a matching mesh where individual basis elements meet. This condition is necessary to enforce current and field continuity between cells, using the same basis functions as the underlying formulation for constructing fields and currents. If faces touch, or even just portions of faces, the parts that do touch *must* have matching meshes, or the decomposition will be invalid.

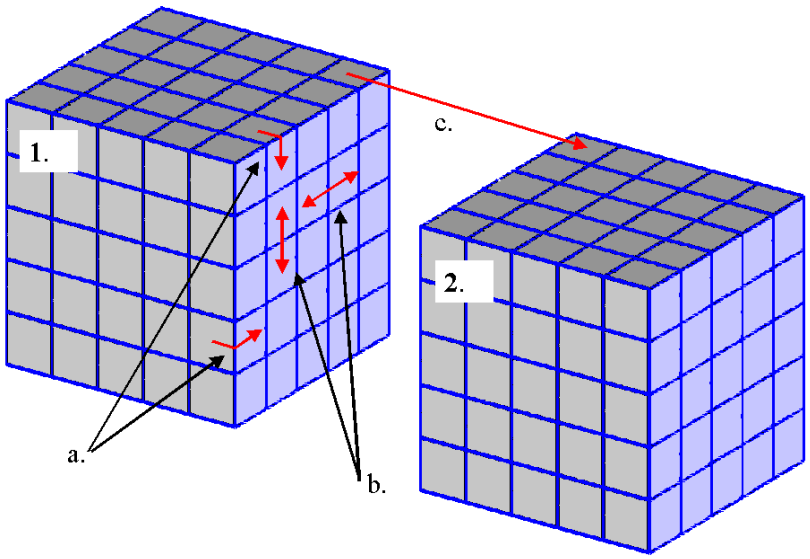


Figure 9.1. Two closed systems with overlapping domains.

When modeled individually, the systems are modeled as *closed* systems, meaning that surface currents are defined over all external surfaces of the structure. Therefore, these surface currents are defined both directly on and flowing over the faces that will be collocated when the systems touch. Consider first the currents labeled b. in the above figure. If these surface currents on system 1 are allowed to persist when the systems are directly overlapping, they will be radiating directly onto the opposing surface from system 2 at a distance of zero. Hence, testing on the opposing surface will create a potential singularity condition. Further, there will be currents on the opposing surface at exactly the same physical location, also radiating. Additionally, the collocated surfaces of both systems must be used for testing of the combined system, and the paths to the collocated surfaces are identical. This implies a possible condition of identical row entries in the system of equations, a condition leading to a singular matrix system. In the end, it will be necessary to remove these surface currents, thereby opening the closed

domains. The open domains can potentially cause problems with formulations requiring closed domains (MFIE), and must be treated accordingly.

There are additional issues to consider. First, when the systems are touching, there should be field continuity between the systems at coincident faces. Based on the developments of previous chapters, an appropriate mechanism for this energy transfer has not been implemented, but can be through the FEM. Further, though there are surface currents defined that flow from adjacent faces of the system to the collocated face, denoted by a. in Figure 9.1, there are no currents defined that flow from system to system, or adjacent face to adjacent face, as depicted with c. Allowing both currents to persist, one flowing over the sharp corner at adjacent faces of system 1, one flowing between systems, would necessitate a current divider rule for proper enforcing of Kirchhoff's current law. Needless to say, it should be apparent that additional boundary conditions must be enforced for the case where separate systems overlap, as it is not sufficient to use the methods developed in previous chapters without modification for overlapping domains. In the following sections, efficient solutions to these problems are addressed in detail.

9.2 Surface Continuity

Before considering volumetric structures, it is instructive to look at the simpler case of thin-surface systems. Take the simple example of two separate surface structures placed exactly next to each other such that their edges overlap, as depicted in Figure 9.2. In this case, both structures are perfect electric conductors (PEC) supporting only electric

currents. This choice makes it possible to avoid the additional complexity of FEM volumetric field continuity issues and focus directly on the specific problem of currents flowing across the outer surfaces of joined systems. This analysis uses the same surface currents developed in Chapter 2. The current constructs for curvilinear, bi-quadratic patches are defined in (2.3.10), and depicted in Figure 2.4. For open surface structures such as the two shown in Figure 9.2, the currents are constructed from basis elements that define the flow of current from surface element to surface element only, and hence there are no basis elements defined at the open edges, as shown in Figure 9.2. That is, there is no basis function defined at open edge of the quad patch surface structures, since there is no outward flowing current at the truncated edge. Thus, if the surface structures were allowed to touch, a current would not be able to flow across the gap, since no mode exists to do so.

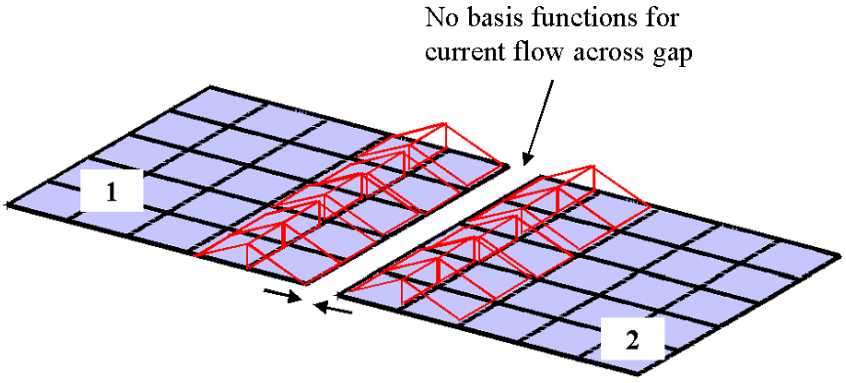


Figure 9.2. Depiction of overlapping surface structures.

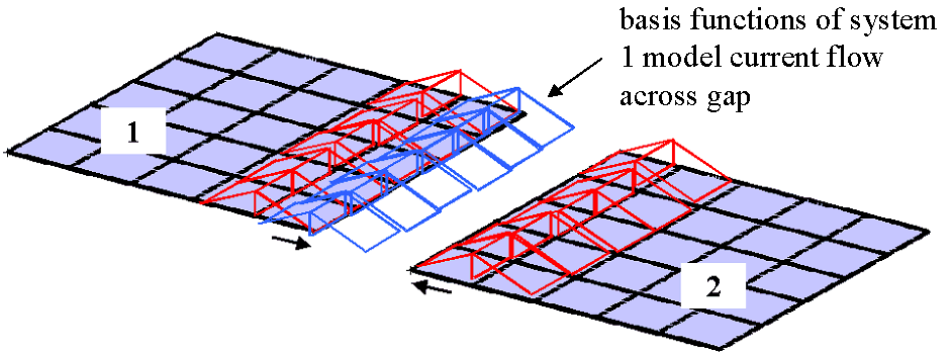


Figure 9.3. Enabling current continuity through additional unknowns.

To facilitate current flowing between the structures, it is necessary to define rooftops across the gap, as depicted in Figure 9.3. To support this current, additional unknowns (new basis functions) have been defined along the outermost edge of system 1, where the gap between system 1 and system 2 exists. These unknowns did not previously exist in the two-structure system, and hence constitute an additional analysis cost. In this case, no condensation process occurs. Though the unknowns have been associated with system 1 in this case, in reality, a portion of the testing and radiation associated with these unknowns requires integration on the adjacent quads of system 2 as well. That is, although the unknown coefficients have been associated with system 1 in this case, these coefficients correspond to rooftops defined on both systems. The unknowns along the outermost edge of system 1 define the surface current flowing from the outermost quad of system 1 to the adjacent and corresponding quad of system 2. From a record keeping perspective, this choice of enforcing domain connectivity can be undesirable.

Alternatively, it is possible to facilitate current flow between adjacent systems without disturbing the structure of the original systems. This is done by creating a *secondary* bridge system that overlaps the original systems, as shown in Figure 9.4. To

distinguish between primary and secondary systems, secondary systems have no physical structure of their own. They are mathematical and not physical constructs. While the secondary systems support unknowns, the coefficients are associated with basis functions (rooftops) on other physical systems. For bridge systems, the source and testing locations are on separate systems, and hence the coefficients in the matrix of the auxiliary secondary system will be generated from testing and acting as source currents on these other systems. In this process, care must be taken to observe the self-testing requirements of equations (2.2.10) and (2.2.11). That is, where bridge systems overlap with primary systems, self-testing procedures must be enforced.

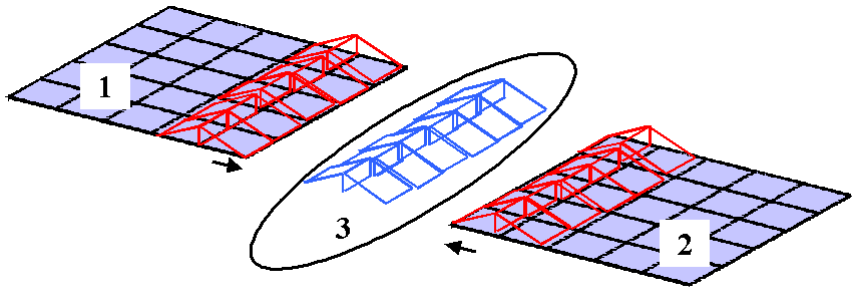


Figure 9.4. Surface structure connectivity via bridge systems.

The system of equations for this open-surface system would be of the form

$$\begin{bmatrix} [Z]_{11} & [Z]_{13} & [Z]_{12} \\ [Z]_{31} & [Z]_{33} & [Z]_{32} \\ [Z]_{21} & [Z]_{23} & [Z]_{22} \end{bmatrix} \begin{Bmatrix} \{x\}_1 \\ \{x\}_3 \\ \{x\}_2 \end{Bmatrix} = \begin{Bmatrix} \{b\}_1 \\ \{b\}_3 \\ \{b\}_2 \end{Bmatrix}. \tag{9.2.1}$$

This system consists of only electric field integral equations, as the PEC surfaces only support electric currents, and the system is open. This system of equations can be enforced using the formulation of (2.2.10), where the magnetic currents have been set to

zero. In general, the same concepts apply to situations where there are magnetic currents as well.

This section serves as an introduction to surface current bridge systems for domain connectivity. In a later section, this concept will be extended to the more complicated case of array analysis. For arbitrary surface system analysis, the bridge system approach results in an increase in the number of systems proportional to the number of systems with collocated edges, and an increase in unknowns equal to the total number of collocated edges.

9.3 Volumetric Continuity

The task becomes more difficult when considering volumetric FEM regions as well. Again consider the joining of two systems as depicted in Figure 9.1. It is assumed, as should be the general case for a volumetric-type problem, that the solution is being generated for fields both inside and outside the systems. The assumption is that surface currents have been defined over the entire outer surfaces of both systems, including on the surface regions that will touch once they are joined, as depicted in Figure 9.1. As mentioned, allowing these currents to persist as the systems are joined is problematic, and hence the surface currents must be removed. However, upon removing these currents, both systems will no longer be closed systems, and a decoupled consideration of the individual systems will not be valid. That is, it will be necessary to solve the systems of equations simultaneously, as the coupling between the systems will be necessary in creating a complete and closed combined system. This differs from previous chapters,

where the isolated domains of individual systems made it possible to use a decoupled system solution approach. Here, the two systems under analysis have been opened to facilitate their joining, and are no longer valid systems when isolated.

Consider two separate solutions to this domain connectivity problem. As in the case for surface system continuity, the first approach does not increase the number of systems under consideration, and in this sense, does not increase the complexity of the combined problem. The first solution approach is depicted in Figure 9.5. In this figure, the joined systems have been separated to give an inside view as to how they are connected. As previously stated, the surface currents defined on the quads in the overlapping region have been removed. For system 2, the unknowns across the entire overlapping surface have been removed, including around the outermost edge of the overlapping region. Necessarily, for system 2, this means that the surface currents still associated with the system on the remaining closed portion of its boundary do not flow all the way to the edge of the now-opened area. Previously, an unknown coefficient existed at this outermost edge. In the treatment of system 1, the unknowns along the common surfaces are not removed, but it is necessary to change the way in which they are used. The unknowns around the outermost edge of the overlapping region of system 1 are used to form a current bridge to the adjacent (when joined) quads of system 2. Necessarily, this implies that in the source and testing procedure for surface currents of system 2, references will be made to unknowns in system 1. That is, the quads on the non-overlapping surface at the outermost edge of the overlapping region for system 2 will be testing and radiating using the unknown current coefficients associated with system 1.

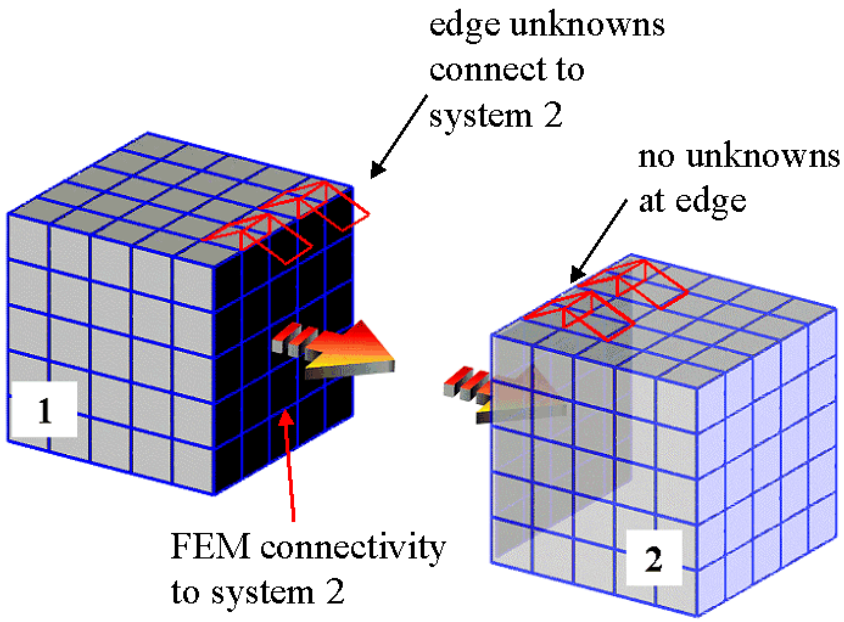


Figure 9.5. Volumetric system connection without bridge system.

In this connectivity model, the unknowns over the entire overlapping face of system 1 will be used to 'complete' the hexahedral FEM elements of system 2 in the overlapping region. This allows for seamless enforcement of field continuity in the FEM across the joined face. The method considered above is most likely the least complex solution to facilitate joining the domain of separate systems. It does not result in an increase in the number of systems under analysis, and results in an overall reduction in unknowns. However, this approach is not well suited for dealing with array-type structures, and instead a modified approach is recommended.

In the second approach, the unknowns on the overlapping surfaces of both systems 1 and 2 are removed and associated with a secondary bridge system. This bridge system, denoted as system 3 in Figure 9.6, is used to create a current bridge between

systems 1 and 2, as well as to create a path for enforcing field continuity in the FEM region. The current bridge, denoted as a. in Figure 9.6, consists of basis elements associated with the edges surrounding the open regions of systems 1 and 2. In this case, the edges corresponding to the surface current coefficients of the bridge system completely encircle the open regions of systems 1 and 2, enabling a current to flow between the systems at the point where the open perimeters of the systems meet. Again, it is vital that the meshes align where the systems touch, or the decomposition will not work.

In addition to providing field coefficients along the open perimeter where the systems meet, the bridge system also provides field coefficients within the overlapping face region. These unknowns are used to 'close' or 'complete' the volumetric hexahedral elements on the open portions of both systems 1 and 2. Both the field coefficients on the perimeter of the bridge system, as well as those within the perimeter, are used to enforce field continuity between systems via the FEM. Similar to the case for surface integral equations, the basis elements of the bridge system use the volumetric elements of the primary systems 1 and 2 for testing procedures. For the two-system case shown in Figure 9.6, the system of equations will take the form

$$\begin{bmatrix} \begin{bmatrix} A^{II}_{11'} & A^{IS}_{11'} & 0 \\ A^{SI}_{11'} & A^{SS}_{11'} & B_{11'} \\ 0 & P_{11'} & Q_{11'} \end{bmatrix} & \begin{bmatrix} A^{II}_{13'} & A^{IS}_{13'} & 0 \\ A^{SI}_{13'} & A^{SS}_{13'} & B_{13'} \\ 0 & P_{13'} & Q_{13'} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{12'} & Q_{12'} \end{bmatrix} \\ \begin{bmatrix} A^{II}_{31'} & A^{IS}_{31'} & 0 \\ A^{SI}_{31'} & A^{SS}_{31'} & B_{31'} \\ 0 & P_{31'} & Q_{31'} \end{bmatrix} & \begin{bmatrix} A^{II}_{33'} & A^{IS}_{33'} & 0 \\ A^{SI}_{33'} & A^{SS}_{33'} & B_{33'} \\ 0 & P_{33'} & Q_{33'} \end{bmatrix} & \begin{bmatrix} A^{II}_{32'} & A^{IS}_{32'} & 0 \\ A^{SI}_{32'} & A^{SS}_{32'} & B_{32'} \\ 0 & P_{32'} & Q_{32'} \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{21'} & Q_{21'} \end{bmatrix} & \begin{bmatrix} A^{II}_{23'} & A^{IS}_{23'} & 0 \\ A^{SI}_{23'} & A^{SS}_{23'} & B_{23'} \\ 0 & P_{23'} & Q_{23'} \end{bmatrix} & \begin{bmatrix} A^{II}_{22'} & A^{IS}_{22'} & 0 \\ A^{SI}_{22'} & A^{SS}_{22'} & B_{22'} \\ 0 & P_{22'} & Q_{22'} \end{bmatrix} \end{bmatrix} \begin{Bmatrix} E^i_1 \\ E^s_1 \\ H^s_1 \end{Bmatrix} \quad \begin{Bmatrix} b^i_1 \\ b^s_1 \\ b^e_1 \end{Bmatrix}$$
$$\begin{bmatrix} A^{II}_{31'} & A^{IS}_{31'} & 0 \\ A^{SI}_{31'} & A^{SS}_{31'} & B_{31'} \\ 0 & P_{31'} & Q_{31'} \end{bmatrix} \begin{bmatrix} A^{II}_{33'} & A^{IS}_{33'} & 0 \\ A^{SI}_{33'} & A^{SS}_{33'} & B_{33'} \\ 0 & P_{33'} & Q_{33'} \end{bmatrix} \begin{bmatrix} A^{II}_{32'} & A^{IS}_{32'} & 0 \\ A^{SI}_{32'} & A^{SS}_{32'} & B_{32'} \\ 0 & P_{32'} & Q_{32'} \end{bmatrix} \begin{Bmatrix} E^i_3 \\ E^s_3 \\ H^s_3 \end{Bmatrix} = \begin{Bmatrix} b^i_3 \\ b^s_3 \\ b^e_3 \end{Bmatrix}. \quad (9.3.1)$$
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{21'} & Q_{21'} \end{bmatrix} \begin{bmatrix} A^{II}_{23'} & A^{IS}_{23'} & 0 \\ A^{SI}_{23'} & A^{SS}_{23'} & B_{23'} \\ 0 & P_{23'} & Q_{23'} \end{bmatrix} \begin{bmatrix} A^{II}_{22'} & A^{IS}_{22'} & 0 \\ A^{SI}_{22'} & A^{SS}_{22'} & B_{22'} \\ 0 & P_{22'} & Q_{22'} \end{bmatrix} \begin{Bmatrix} E^i_2 \\ E^s_2 \\ H^s_2 \end{Bmatrix} \quad \begin{Bmatrix} b^i_2 \\ b^s_2 \\ b^e_2 \end{Bmatrix}$$

The details of this expansion are now explained. The self-coupling sub-matrices of systems 1 and systems 2, in the upper left and lower right of the matrix system (9.3.1), are created using the same procedure that created (2.4.9), and are the same as the upper left and lower right sub-matrices of (4.2.2), *except* that the open regions of these systems have now been excluded from consideration. Also, as in (4.2.2), systems 1 and 2 couple only via integral equations, represented by the sub-matrices in the lower left and upper right of (9.3.1). The sub-matrix directly in the center of (9.3.1) represents the interactions within the bridge system itself. Geometrically, these interactions actually happen between physical portions of systems 1 and 2, with the understanding that mathematically, the coupling coefficients are stored in a separate, non-physical bridge system 3. The terms in this sub-matrix represent interactions associated with the field coefficients of the bridge system only. That is, the surface integral terms $\left[P\right]_{33'}$ and $\left[Q\right]_{33'}$ represent the external coupling associated with edges along the domain connectivity perimeter. The terms $[A]^{II}_{33'},[A]^{IS}_{33'},[A]^{SI}_{33'},[A]^{SS}_{33'}$ represent FEM interactions between adjacent edges on the system faces of the internal bridge region (given as b. in Figure 9.6), i.e. they are evaluated only at the limit of the FEM element faces they connect. The $[B]_{33'}$ sub-matrix handles the relation between E^s_3 and H^s_3 on the surface regions of systems 1 and 2 associated with the edges of system 3 at the connecting perimeter.

The remaining sub-matrices of (9.3.1) handle the coupling between the bridge system and the primary systems 1 and 2. Though these are cross-coupling matrices, they contain self-testing information as well. That is, the unknown coefficients of the bridge

system correspond to the edges of surface quads belonging to systems 1 and 2. When an edge from the bridge system is tested on a quad in system 1 or 2 connected to that edge, self-testing procedures must be carried out. This accounts for the presence of the $[B]_{\#3'}$ and $[B]_{3\#'}$ terms in these cross-coupling sub-matrices. Likewise, the $[P]$ and $[Q]$ operators in these sub-matrices will have potentially strong terms.

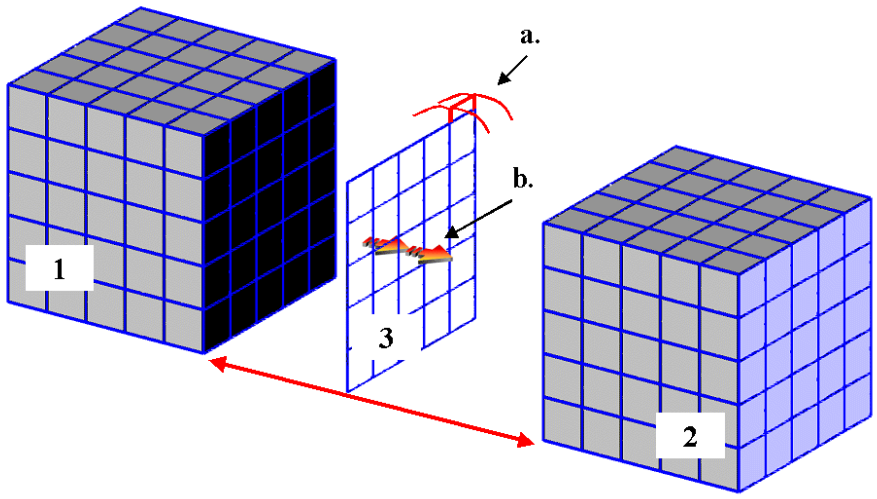


Figure 9.6. Volumetric domain connectivity via bridge system.

Again, it seems like an additional complexity to create a third system, where only two were necessary previously. As a third option, it would be possible to condense the systems into a single matrix of the form (2.4.9). However, In the case of array systems, this would defeat the purpose of removing redundant interactions via the Toeplitz-based expansion. In the next section, it will become apparent that the bridge system approach results in maximum reuse of system redundancies for array-type problems.

9.4 Surface Array Element Connectivity

The domain connectivity approaches discussed thus far are now extended to the case of array-type problems, starting first with a simple surface-based analysis. Consider a linear array of identical PEC surface elements as shown in Figure 9.7. As required for ADM and AD-FMM, each element has exactly the same structure, and the same number of unknown field coefficients. As discussed previously, there is not a mode for current to flow between the surface elements, as there is no basis function defined at the edge of the element. Hence, one solution is to simply add unknowns to the leading edge of each element, such that current can flow between elements of the array, as shown in Figure 9.8. Additional basis functions have been added to each of the elements in the array, except for the last array element, which has no adjacent element.

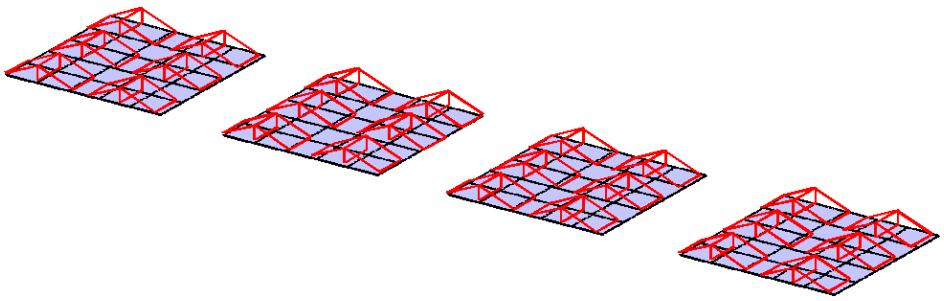


Figure 9.7. Linear array of surface-based structures prior to connectivity.

This is one correct solution to the problem. However, if this approach is taken, the elements of the array will no longer be identical. Hence, it is not possible to preserve the Toeplitz interaction property for all elements of the array. This is not entirely an unacceptable approach. In essence, this approach creates two new systems – an array

structure of one less element than the original system with Toeplitz properties preserved, plus an additional system consisting of a single element from the end of the original array.

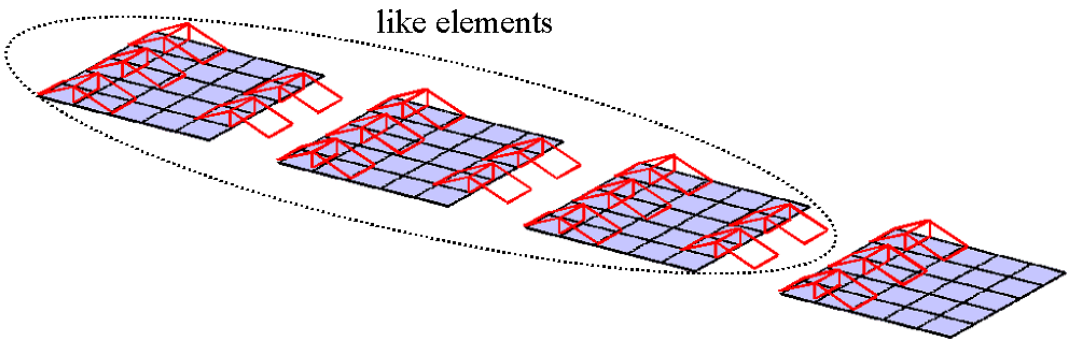
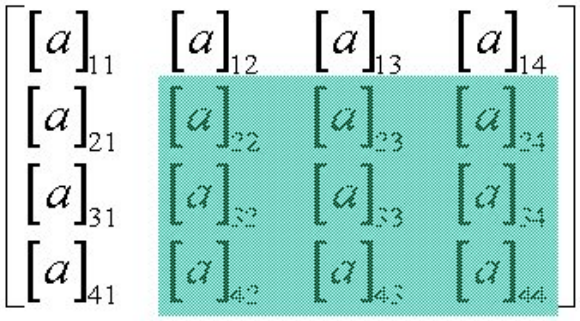
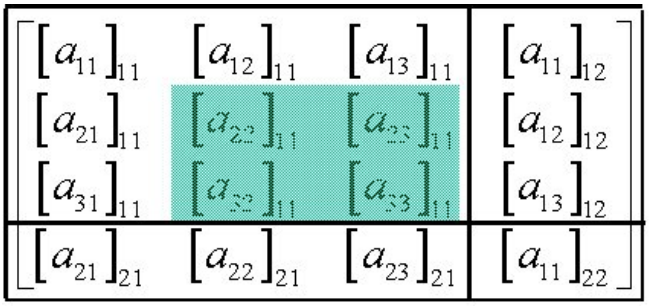


Figure 9.8. Linear surface array with domain connectivity.

This is one solution to the array problem, and is not the preferred one. The main problem with this approach is that there are now two unit cells of approximately the same cost as the single unit cell of the previous systems. This increases the preconditioning cost two fold. Moreover, this approach reduces the length of the array systems, creating less efficiency in matrix storage as well as solution procedures. Further, it introduces cross-coupling systems that cannot be decomposed. The effect of this decomposition approach on the matrix structure is illustrated in Figure 9.9, where as before, the redundant terms have been grayed out.



(a) before connectivity



(b) after connectivity

Figure 9.9. Cost analysis of domain connectivity approach.

In order to avoid this cost, it is preferable to use the bridge system approach, depicted in Figure 9.10. In this approach, the structure of the original array system is left intact, and bridge systems are introduced between the elements of the array. When placed between identical array elements, the bridge systems will be identical as well, thus creating an array of bridge system elements. The size of this array will be one element less than the original linear array. With this domain connectivity model, the cost of analyzing the original array has not increased – the system remains unchanged. To facility current flow between elements of the array, a second array system has been introduced at additional cost. The matrix structure corresponding to this approach is shown in Figure 9.11. The original linear array matrix system appears in the upper left

corner of the new matrix system, and a new system for the bridge array appears in the lower right corner. The two systems are interacted through cross-coupling sub-matrices. The bridge system array has the same spacing and direction as the original system, and hence the cross-coupling systems can be decomposed via Toeplitz interactions as well. It is clear that this approach is more efficient than the latter approach.

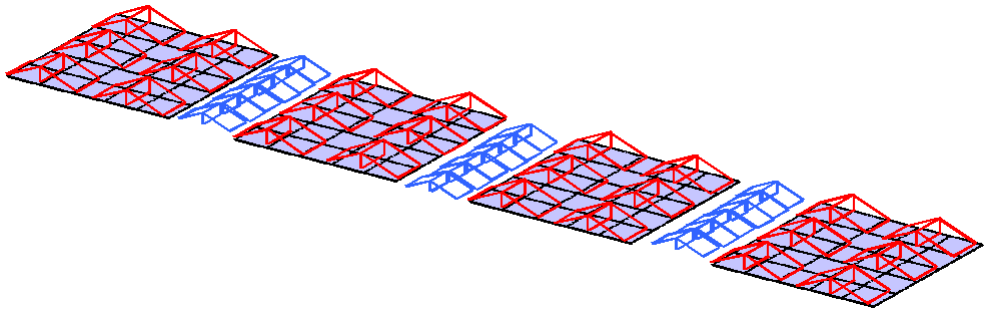


Figure 9.10. Linear surface array connected with bridge elements.

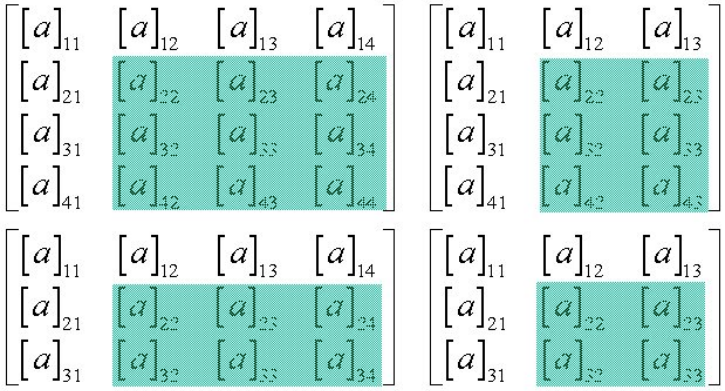


Figure 9.11. Cost representation of domain connectivity for bridge system approach.

Both approaches to the surface-based array element problem increase the cost and complexity of the overall system. It is not possible to reduce the cost of analyzing

surface-based systems, since unknowns are being added, not taken away. The same two approaches can be taken to analyzing planar arrays of surface-based elements. In the first approach to a planar array of surface elements, where now there are touching element edges in two dimensions, there will be four distinct systems of dimensions $(n_1-1)\times(n_2-1)$, $(n_1-1)\times 1$, $1\times(n_2-1)$, and 1×1, as depicted in Figure 9.12.

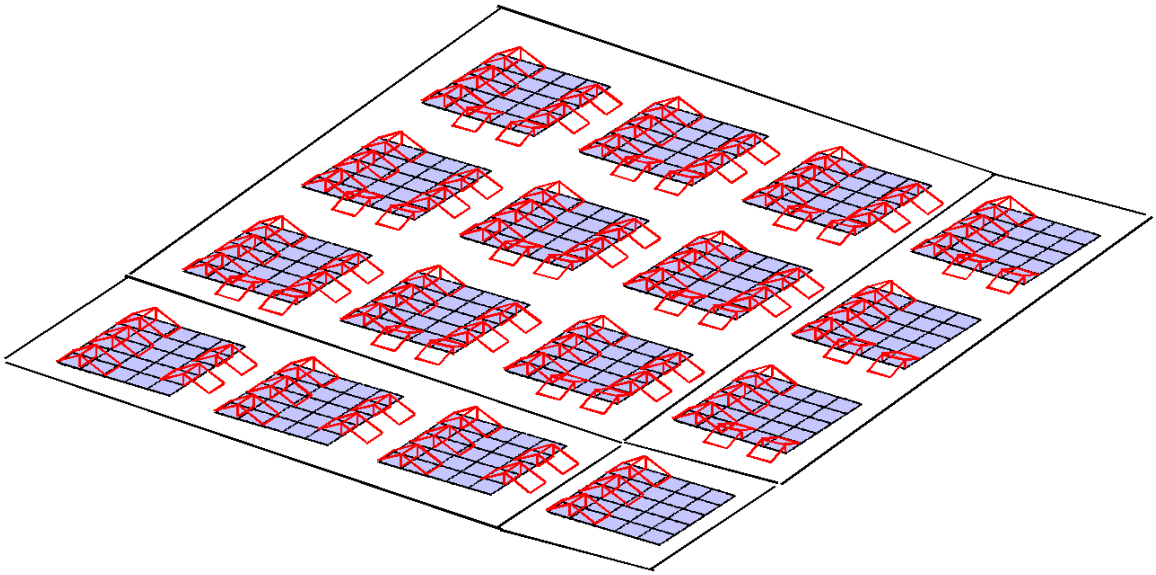


Figure 9.12. Planar surface array without current bridges.

Alternatively, the approach using bridge elements between the array elements does not disturb the structure of the original system, introducing bridge element arrays of one less element in each dimension, as depicted in Figure 9.13. This results in three distinct systems with dimensions $n_1\times n_2$, $(n_1-1)\times n_2$, and $n_1\times(n_2-1)$. All three arrays share the same element spacing in each dimension, thereby allowing multi-dimensional

cross-system decompositions. It can easily be seen that this approach is more efficient that the previous approach. With the superior efficiency of the bridge array approach demonstrated, the other approach will now be abandoned. The concept of bridge systems will now be used to analyze arrays of volumetric elements.

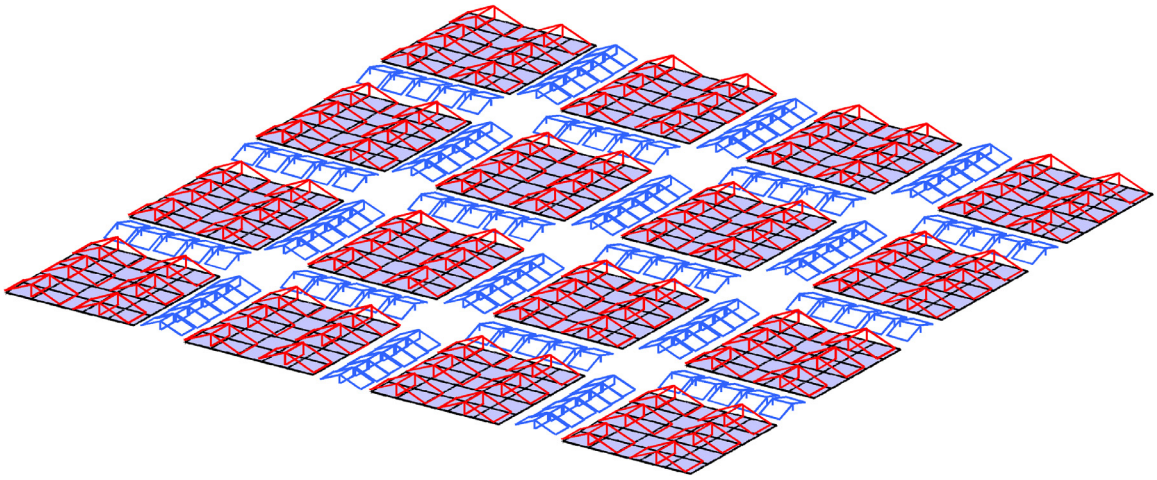


Figure 9.13. Planar surface array with current bridge arrays for domain connectivity.

9.5 Volumetric Array Element Connectivity

Now that surface-structure array systems has been discussed, it is time to grapple with the more difficult problem of volumetric array elements requiring continuity of FEM fields as well as surface currents. There is a marked increase in complexity associated with each additional dimension of volumetric element arrays, and hence it is instructive to first consider the case of a linear array of unit cells, as depicted in Figure 9.14. Again, the unit cells are drawn as cubic structures, with the understanding that the analysis presented here applies to any arbitrarily shaped array elements.

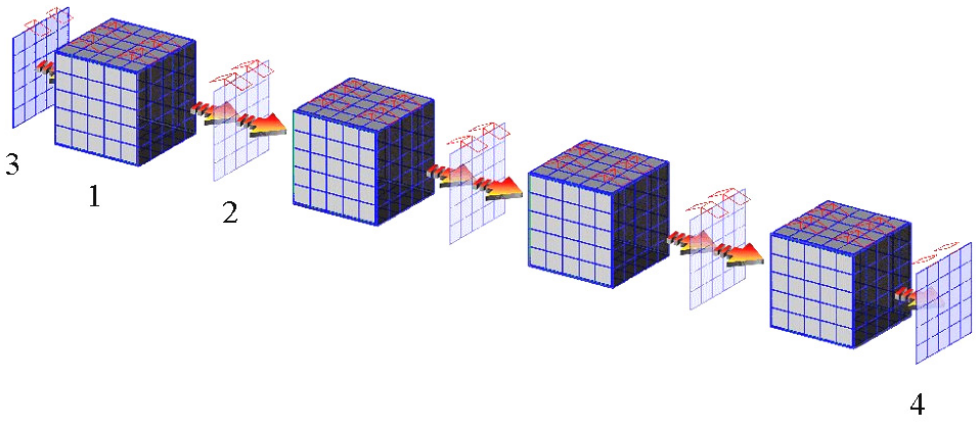


Figure 9.14. Linear array of volumetric cells with bridge elements.

It is clear from the diagram that as the single four-element linear array undergoes the domain interconnectivity process, it is transformed into four distinct systems. These systems are now described. Between the elements of the original array, bridge systems have been introduced (system 2). The form of these bridge systems is the same as those described in Figure 9.6. Since the original array elements are all identical, the bridge systems between each of the array elements will be identical as well, and have the same element spacing as the original array. The exact extent of the bridge system element can be determined by testing for overlapping surfaces between adjacent array elements in each dimension. Further, the size of the bridge element array is one element smaller than the original array in the dimension of overlap. In terms of unknown count, the cost of this array is exactly the same as the reduction in cost achieved through condensation of collocated element edges between the original array elements. In other words, while the bridge element array has been introduced, two times the number of unknowns from the opposing faces of the adjoining array elements of the original system have been removed, thus reducing the overall unknown count.

Consider again the testing procedure that determines which portions of adjacent elements overlap in each dimension. The corresponding surfaces of the lagging edge of the first array element and the leading edge of the last array element have been partitioned into separate systems 3 and 4 as shown in Figure 9.14. By doing this, it is possible to preserve the majority of the original array at its original size, with the same elements, minus the portions of the elements that overlap with each other. In other words, the complexity of the elements in the primary array are now reduced, as they do not have a leading or lagging surface corresponding to the overlapping region. Unlike the bridge systems, systems 3 and 4 support surface currents around their perimeter and across their face as well. In essence, these systems correspond to the surface field coefficients that have been stripped away from the elements they came from. Because they are still associated with a volumetric FEM element, albeit in another system, it is important to realize that they maintain a well-defined surface normal.

Using this model, the cost of the overall system has not increased. Rather, the storage cost has been reduced, at the complexity tradeoff of additional systems that require simultaneous consideration with a multi-system analysis approach. Where before there was only a single linear array, there are now four total systems. For a bridge system decomposition of linear arrays, there will always be a resulting increase of three systems (to four total).

The complexity increases significantly as the focus advances to planar arrays of volumetric elements. The domain connectivity model for an example planar array of volumetric elements is shown in Figure 9.15. The first step is to simply apply the same procedure used for the linear array to the planar array. That is, for each dimension of the

array, a testing procedure is applied to determine if adjacent elements in that dimension overlap. In each dimension that has overlapping surface area, three new systems will be created. As before, there will be a bridge array between the elements of the array in each dimension where the overlap occurs, as well as new surface systems at the lagging edge of the first array element and the leading edge of the last array element in each dimension. It is instructive to divide the domain connectivity procedure into two phases. Let this domain connectivity procedure be referred to as phase 1. The phase 1 domain connectivity procedure results in a number of new systems equal to $3\times$ the number of overlapping dimensions. For the two-dimensional array shown in Figure 9.15, the phase 1 domain connectivity procedure results in 6 new systems (3×2). The size of the bridge systems will be the same as the original array system, minus one element in the dimension that the overlap occurred. The size of the end systems will be the same as the original array, with only a single element in the dimension of the overlapping regions.

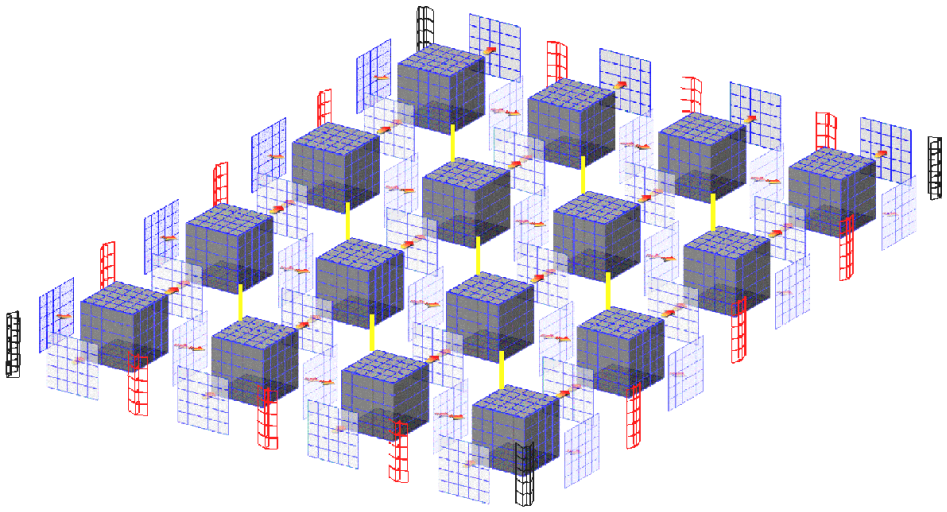


Figure 9.15. Planar array of volumetric elements with bridge elements.

The phase 1 domain connectivity procedure is not adequate to ensure proper system modeling for arrays of more than a single dimension. To make the domain connectivity procedure general, it is necessary to undergo a second phase of testing for overlaps, this time on the additional systems created in the first phase of overlapping domain testing. Again, this will create three additional systems in each dimension that an overlap is detected, for each of the array systems. There will be a bridge array system between each of the adjacent elements, plus additional new systems at the lagging edge of the first element of the array, and the leading edge of the last element in the array. For the array shown in Figure 9.15, this will ultimately result in a total of 16 systems (from one originally). It is important to realize that this domain connectivity procedure results in a number of open systems. Therefore, it is critically important to stress the simultaneous solution of systems that undergo the domain connectivity process. It is not possible to solve the systems independently, and then link them in a decoupled, iterative process. The systems must be solved simultaneously via the multi-system analysis approach of the previous chapter.

Because the decomposition is performed on volumetric array elements, in the end there will be less unknowns total than if domain connectivity is not enforced. Again, the size of these systems will be the same as the original array system, minus one element in the dimension that the overlap occurred. This system of expansion is valid for any number of array dimensions, with the concept in mind that the expansion is only applied on dimensions that feature overlap between adjacent elements.

9.6 Results and Comparisons

This concludes the introduction to domain connectivity for finite array systems. The inclusion of this domain connectivity model should suffice to properly model the necessary modes across adjacent array elements. For validation, the results of intra-element domain connectivity are now observed in the intra-array coupling measurements. Figure 9.16 shows the new comparisons with the added current paths between adjacent elements. For these results a CFIE parameter of 0.9 (90% EFIE, 10% MFIE) was used. It is observed that while the simulation results are reasonably close to the measurements, the agreement is not significantly better than the results obtained in Chapter 7. Mainly, this discrepancy is due to a problem in using the MFIE for thin geometries. Unfortunately for the simulated geometry, it was not possible to use EFIE alone, as the system of equations would not converge in a reasonable amount of time, as explained below. However, this example serves the purpose of demonstrating domain connectivity.

Improved results could likely be obtained by either developing a geometric model that works well with the MFIE, for example, by introducing air layers to offset the problem of thin geometries, or by researching improvements to the system of equations generated by the EFIE formulation. The domain decomposition approach presented in this chapter results in a highly coupled system of equations. This is because individual systems are physically linked by direct current flow between adjacent domains. For improved solution convergence, it may be necessary to explore more effective preconditioning methods that pre-solve the strong system connections, rather than attempting an iterative solution over this strong coupling between systems.

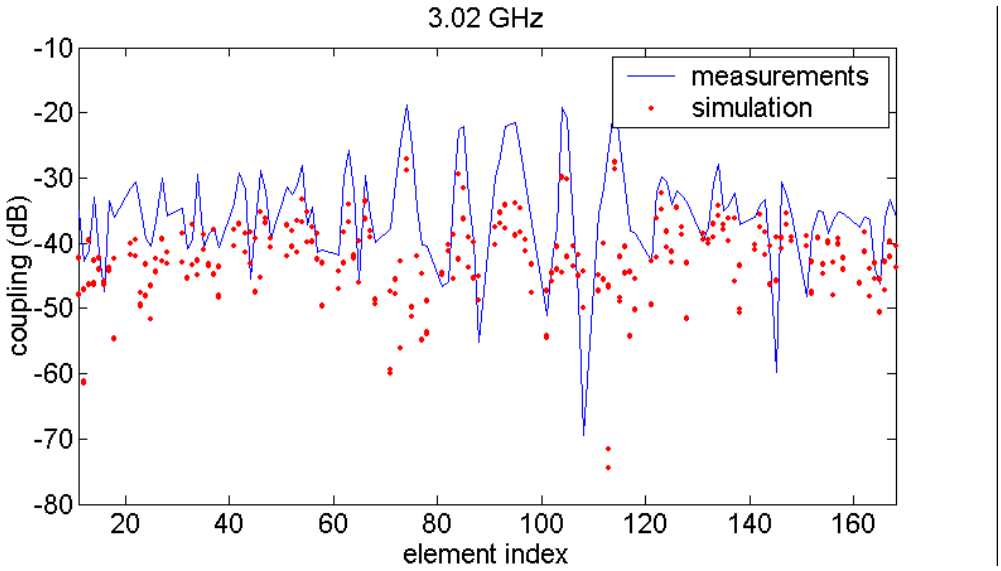


Figure 9.16. Intra-array coupling with domain connectivity.

9.7 Concluding Issues

This concludes the final chapter of this thesis. The purpose of this thesis was to develop rigorous analysis methods for finite arrays and their supporting structures. In summary, it has been demonstrated that the analysis of complex finite array systems can be realized with limited resources and a reasonable degree of accuracy. The main contribution of this work is the development of an analysis method for finite arrays that results in a fixed and manageable amount of matrix storage for any sized array in the same class. That is, 10×10 element arrays and 1000×1000 element arrays have the same (matrix) storage requirements for the same element and lattice spacing. For rigorous analysis methods, this is a new development.

The next level of enhancements to the developments presented here might include a decoupled solution approach for weakly coupled systems. This would allow the analysis of much larger systems, where portions of the problem could be solved independently, using the pre-solved incident fields from the other systems as external excitation. An example of this might be a classroom propagation model, in which the walls, floors, source excitation and scattering targets (chairs, desks, students, etc.) could all be interacted as separate systems with a decoupled, iterative solution approach. This propagation model works well when the systems involved are not strongly coupled, but may fail to converge otherwise. In general, the decoupled solution approach could be applied to any structures that are in the far-zone of each other with reasonable results.

This approach could be used for in part for analyzing large navy ships at microwave frequencies. Using the hybrid, decoupled solution approach, it would be possible to interface with additional methods, such as high frequency codes. For problems as large as an entire ship, distinct portions of the geometry could be analyzed with the most appropriate analysis tool. The pre-solved ship sections can then be interacted with each other using a decoupled iterative approach until the desired level of convergence is reached.

On another front, the advanced array methods presented in this chapter are only as accurate as the underlying formulation upon which they are based. Therefore, one important future development would be to improve upon the underlying FE-BI formulation. Specifically, the EFIE formulation is not well suited to iterative solution procedures. This problem is magnified as the size of the system increases, placing potential limits on the maximum system size that can be analyzed with the array methods.

Additionally, the MFIE formulation, which has excellent convergence properties, does not provide accurate solutions for thin geometries, such as might commonly be found in structures made from dielectric substrates. By improving these issues in the underlying formulation, the performance of the advanced array-based methods can be improved as well.

Additionally, implementing higher order basis functions for the field representations could make it possible to analyze even larger structures. Higher order field expansions would allow the use of fewer total basis elements, which translates into less unknowns and therefore lower necessary system resources. Again, saving unknowns in the array element translates into compounded savings for large arrays. This is particularly of issue when the potential to analyze arrays with millions of unknowns can be realized with these advanced array codes.

This concludes the thesis.

APPENDICES

APPENDIX A.

ILLUSTRATION OF NON-SYMMETRIC TOEPLITZ PROPERTY

In the case where a linear array is modeled with a single expansion function per array element, it is indeed true that $[a]_{mn'} = [a]_{nm'}$ as illustrated in Figure A.1.

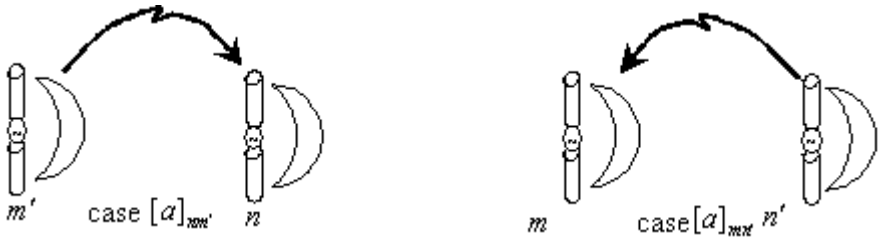


Figure A.1. Illustration of equivalent coupling $[a]_{mn'} = [a]_{nm'}$ for single expansion function.

However, in the general case where multiple basis functions are used to model each array element, and the numbering scheme is not specified, there is no guarantee of reciprocal matrices. To illustrate this, consider the example of two tapered slot antennas with a simple linear numbering scheme for the basis functions, as depicted in Figure A.2. As noted earlier, the indices within the brackets refer to interaction between basis functions, whereas the indices outside the brackets refer to the element interactions. From Figure A.2, it is clear that the coupling paths $[a_{15}]_{mn'}$ and $[a_{15}]_{nm'}$ are not the same. Consequently, it must be that $[a]_{mn'} \neq [a]_{nm'}$. This is an important result to recognize since it directly affects storage and the implementation of the FFT.

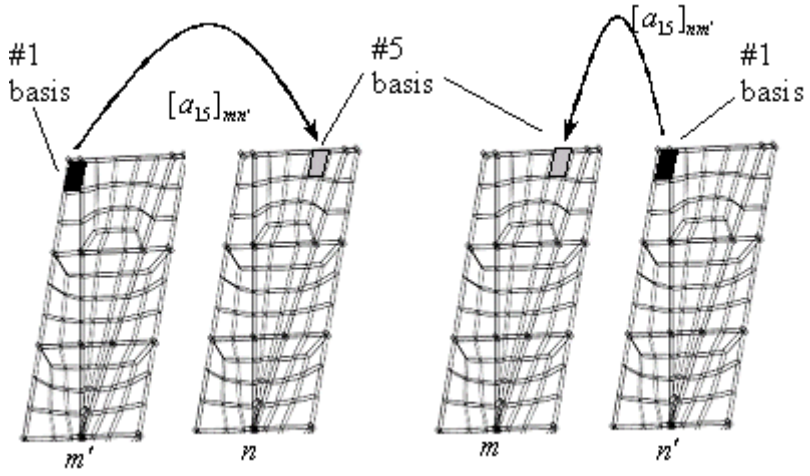


Figure A.2. Illustration of property $[a]_{mn'} \neq [a]_{nm'}$ for general case.

Moreover, in the case where CFIE is used in the formulation, the $[P]$ and $[Q]$ operators will not be symmetric.

APPENDIX B.

USE OF THE BLOCK-DIAGONAL-LU PRECONDITIONER WITH ARRAY DECOMPOSITION

It is possible to save valuable cycles of the matrix-vector multiplication operation when using the block-diagonal LU preconditioner in combination with the array decomposition. In the associated matrix system $[A]$, the blocks along the diagonal are all self-cells, which can be represented as $[M]$. Since the block-diagonal preconditioner is equivalently the inversion of this block (functionally so), it can be represented as $[M]^{-1}$.

Now divide the matrix system $[A]$ into three equivalent matrices $[L]$, $[U]$, and $[D]$, representing the lower, upper, and block-diagonal regions of $[A]$, respectively, such that $[A] = [L]+[U]+[D]$. Thus, the operation

$$[A]\{x\} = \{b\} \tag{B.1}$$

is equivalent to

$$[L]\{x\}+[D]\{x\}+[U]\{x\} = ([L]+[D]+[U])\{x\} = \{b\}. \tag{B.2}$$

The preconditioned matrix-vector operation is given as

$$[M]^{-1}[A]\{x\} = [M]^{-1}\{b\}, \tag{B.3}$$

or equivalently

$$[M]^{-1}[L]\{x\}+[M]^{-1}[D]\{x\}+[M]^{-1}[U]\{x\} = [M]^{-1}\{b\}. \tag{B.4}$$

However, in (B.4) the operation

$$[M]^{-1}[D] = [I], \tag{B.5}$$

where $[D] = [M]$ and

$$[M]^{-1}[M] = [I]. \tag{B.6}$$

Consequently, since $[I]\{x\} = \{x\}$, it is possible to replace the matrix-vector operation (B.4) with the equivalent operation

$$[M]^{-1}[L]\{x\} + \{x\} + [M]^{-1}[U]\{x\} = [M]^{-1}\{b\}. \tag{B.7}$$

Since the diagonal block $[D]$ contains all the FEM operations of the matrix-vector product as well as all the self-cell coupling terms, it is possible to skip this step and save some potentially expensive operations during the iterative solution.

REFERENCES

REFERENCES

[1] R. Coifman, V. Rokhlin, and S. Wandzura, "The fast multipole method for the wave equation: a pedestrian prescription," in *IEEE Antennas Propagat. Mag.*, vol. 35, 1993, pp. 7-12.

[2] T. F. Eibert, J. L. Volakis, D. Wilton, and D. Jackson, "Hybrid FE/BI Modeling of 3D Doubly Periodic Structures Using Triangular Prismatic Elements and a MFIE Accelerated by the Ewald Transformation," *IEEE Trans. Antennas Propagat.*, vol. 47, pp. 843-850, May 1999.

[3] T. F. Eibert and J. L. Volakis, "Fast spectral domain algorithm for hybrid finite element/boundary integral modeling of doubly periodic structures," *IEE Proc.-Microw. Antennas Propag.*, vol. 147, pp. 329-334, Oct. 2000.

[4] A. D. Khzmalyan, "Finite Array Antenna, Scattering, and Transmission-Line Analysis Using the FFT," in *IEEE Antennas Propagat. Mag.*, vol. 42, 2000, pp. 41-53.

[5] D. T. McGrath, Pyati, V. P., "Phased Array Antenna Analysis with Hybrid Finite Element Method," *IEEE Trans. Antennas Propagat.*, vol. 42, pp. 1625-1630, Dec. 1994.

[6] E. W. Lucas and T. P. Fontana, "A 3-D Hybrid Finite Element/Boundary Element Method for the Unified Radiation and Scattering Analysis of General Infinite Periodic Arrays," *IEEE Trans. Antennas Propagat.*, vol. 43, pp. 145-153, Feb. 1995.

[7] H.-T. H. Chou, H.-K.; Pathak, P.H.; Nepa, P.; Civi, O.A., "Efficient hybrid discrete Fourier transform-moment method for fast analysis of large rectangular arrays," in *Microwaves, Antennas and Propagation, IEE Proceedings* -, vol. 149, 2002, pp. 1-6.

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

[9] J. L. Volakis, K. Sertel, and T. F. Eibert, "Hybrid Finite Element Modeling of Conformal Antenna and Array Structures Utilizing Fast Integral Methods," presented at 4th International Workshop on Finite Elements for Microwave Engineering, Futuroscope-Poitiers, France, 1998.

[10] E. Topsakal, R. Kindt, K. Sertel, and J. L. Volakis, "Input Impedance Characteristics of Tapered Slot Antennas," presented at IEEE AP-S International Symposium and URSI Radio Science Meeting, Boston, MA, 2001.

[11] G. E. Antilla and N. G. Alexopoulos, "Scattering from complex three-dimensional geometries by a curvilinear hybrid finite element-integral equation approach," *J. Opt. Soc. Am. A*, vol. 11, pp. 1445-1457, 1994.

[12] X. Q. Sheng, J. M. Jin, J. M. Song, C. C. Lu, and W. C. Chew, "On the Formulation of Hybrid Finite-Element and Boundary-Integral Methods for 3D Scattering," *IEEE Trans. Antennas Propagat.*, vol. 46, pp. 303-311, March 1998.

[13] G. E. Antilla, "Radiation and scattering from complex three-dimensional geometries using a curvilinear hybrid finite element-integral equation approach," in *Ph.D. dissertation*. Los Angeles: University of California, 1993.

[14] K. Sertel, "Multilevel Fast Multipole Method for Modeling Permeable Structures Using Conformal Finite Elements," in *Ph.D. dissertation*. Ann Arbor: University of Michigan, 2003.
[15] J. M. Jin, J. L. Volakis, and J. D. Collins, "A Finite Element-Boundary Integral Method for Scattering by Two and Three Dimensional Structures," in *IEEE Antennas and Propagat. Mag.*, vol. 33, 1991, pp. 22-32.
[16] J. M. Jin, *The Finite Element Method in Electromagnetics*. New York: J. Wiley & Sons, 1993.
[17] J. M. Song, "Moment Method Solutions Using Parametric Geometry," *J. of Electromagnetic Waves and Appl.*, vol. 9, pp. 71-83, Jan-Feb. 1995.
[18] A. F. Peterson, "The interior resonance problem associated with surface integral equations of electromagnetics: numerical consequences and a survey of remedies.," *Electromagnetics*, vol. 10, pp. 293-312, July-September 1990.
[19] J. R. Mautz and R. F. Harrington, "H-Field, E-Field, and Combined Field Solutions for Conducting Bodies of Revolution," *Arch. Elektron. Übertragungstech (AEÜ)*, vol. 32, pp. 159-164, April 1978.
[20] L. V. Kantorovich and V. I. Krylov, *Approximate Methods of Higher Analysis*: Interscience Publishers, 1958.
[21] R. E. Collin, "Field Theory of Guided Waves," second ed. New Jersey: IEEE Press, 1991, pp. 142-143.
[22] E. S. Siah, J. L. Volakis, and D. Pavlidis, "Electromagnetic Analysis of Plane Wave Illumination Effects onto Passive and Active Curcuit Topologies," *IEEE Ant. and Wireless Prop. Letters, accepted for publication*, 2003.
[23] Y. Saad, *Iterative methods for sparse linear systems*. New York, NY: PWS Publishing, 1996.
[24] G. L. G. Sleijpen and D. R. Fokkema, "BICGSTAB(L) for linear equations involving unsymmetric matrices with complex spectrum," *Electron. Trans. Numer. Anal.*, vol. 1, pp. 11-32, 1993.
[25] B. X. W. E. Schoenlinner, J.P.; Eleftheriades, G.V.; Rebeiz, G.M., "Wide-scan spherical-lens antennas for automotive radars," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 50, pp. 2166-2175, 2002.
[26] D. P. Forrai and E. H. Newman, "Radiation and scattering from loaded microstrip antennas over a wide bandwidth," The Ohio State University, Columbus, OH, Tech. Rep. 719493-1, Sept. 1988.
[27] C. A. Balanis, *Antenna Theory, Analysis, and Design*, 2nd. ed, 1997.
[28] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. Cambridge, MA, 1972.
[29] K. Sertel and J. L. Volakis, "Effects of the Fast Multipole Method(FMM) Parameters on RCS Computations," presented at IEEE AP-S International Symposium and URSI Radio Science Meeting, Orlando, FL, 1999.
[30] R. C. Singleton, "An algorithm for computing the mixed radix fast Fourier transform," *IEEE Trans. on Audio and Electroacoustics*, vol. AU-17, pp. 93-103, 1969.
[31] E. Bleszynski, M. Bleszynski, and T. Jaroszewicz, "AIM: Adaptive integral method for solving large scale electromagnetic scattering and radiation problems," *Radio Science*, vol. 31, pp. 1225-1251, 1996.

[32] S. Bindiganavali and J. L. Volakis, "Scattering from plates containing small features using the adaptive integral method (AIM)," *IEEE Trans. Antennas Propagat.*, pp. 1867-1878, December 1998.
[33] W. C. Chew, J. M. Jin, C. C. Lu, E. Michielssen, and J. M. Song, "Fast solution methods in electromagnetics," *IEEE Trans. Antennas Propagat.*, vol. 45, pp. 533-543, March 1997.
[34] A. L. Van Koughnett, "Mutual Coupling Effects in Linear Antenna Arrays," *Canadian Journal of Physics*, vol. 48, pp. 659-674, 1970.
[35] N. N. Bojarski, "k-space formulation of the electromagnetic scattering problem," *Air Force Avionics Lab. Tech. Rep.*, vol. AFAL-TR-71-75, March 1971.
[36] W. H. Preis, "The Toeplitz Matrix: Its Occurrence in Antenna Problems and a Rapid Inversion Algirithm," *IEEE Trans. Antennas Propagat.*, vol. 20, pp. 204-206, Feb. 1972.
[37] R. Mittra and Y. Rahmat-Samii, "Spectral Theory of Diffraction," *Applied Physics*, vol. 10, pp. 1-13, Jan. 1976.
[38] W. L. Ko and R. Mittra, "A New Approach Based on a Combination of Integral Equation and Asymptotic Techniques for Solving Electromagnetic Scattering Problems," *IEEE Trans. Antennas Propagat.*, vol. 25, pp. 187-197, March 1977.
[39] H. L. Nyo, A. T. Adams, and R. F. Harrington, "The Discrete Convolution Method for Electromagnetic Problems," *Electromag.*, vol. 5, pp. 191-208, 1985.
[40] P. M. van den Berg, "Iterative Computational Techniques in scattering based upon the integrated square error criterion," *IEEE Trans. Antennas Propagat.*, vol. 32, pp. 1063-1071, Oct. 1984.
[41] P. M. van den Berg and R. E. Kleinman, "The Conjugate Gradient Spectral Iterative Technique for Planar Structures," *IEEE Trans. Antennas Propagat.*, vol. 36, pp. 1418-1423, Oct. 1988.
[42] T. K. Sarkar, E. Arvas, and S. M. Rao, "Application of FFT and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies.," *IEEE Trans. Antennas Propagat.*, vol. 34, pp. 635-640, May 1986.
[43] H. Gan and W. C. C. Chew, "A Discrete BCG-FFT Algorithm for Solving 3D Inhomogeneous Scatter Problems," *J. Electromag. Waves*, vol. 9, pp. 1339-1357, 1995.
[44] J. M. Jin and J. L. Volakis, "A Biconjugate gradient FFT solution for scattering by planar plates," *Electromag.*, vol. 12, pp. 105-119, 1992.
[45] T. J. Peters and J. L. Volakis, "Application of a conjugate gradient FFT method to scattering from thin planar material plates," *IEEE Trans. Antennas Propagat.*, vol. 36, pp. 518-526, Apr. 1988.
[46] Y. Zhuang, K. L. Wu, and J. Litva, "A Combined Full-Wave CG-FFT Method for Rigorous Analysis of Large Microstrip Antenna Arrays," *IEEE Trans. Antennas Propagat.*, vol. 44, pp. 102-109, 1996.
[47] L. Andersen, Y. Erdemli, and J. L. Volakis, "Finite printed antenna array modeling using an adaptive multiresolution approach," presented at 2000 Applied Computational Electromagnetics Society, Monterey, CA, 2000.

[48] R. Kindt, K. Sertel, E. Topsakal, and J. L. Volakis, "An extension of the array decomposition method for large finite-array analysis," *Microwave and Optical Technology Letters*, vol. 38, pp. 323-328, August 20 2003.
[49] R. Kindt, K. Sertel, E. Topsakal, and J. L. Volakis, "A Domain Decomposition of the Finite Element-Boundary Integral Method for Finite Array Analysis," presented at Applied Computational Electromagnetics Society Annual Review, Monterey, CA, 2002.
[50] R. W. Kindt, K. Sertel, E. Topsakal, and J. L. Volakis, "Array Decomposition Method for the Accurate Analysis of Finite Arrays," *IEEE Trans. Antennas Propagat.,*, vol. 51, pp. 1364-1372, June 2003.
[51] R. Kindt and J. L. Volakis, "The Array Decomposition-Fast Multipole Method," presented at IEEE AP-S International Symposium, Columbus, OH, 2003.