Nested Reduction Algorithms for Generating a Rank-Minimized $\mathcal{H}^2$-Matrix From FMM for Electrically Large Analysis

Chang Yang, *Graduate Student Member, IEEE*, and Dan Jiao, *Fellow, IEEE*

***Abstract*— In this work, we develop efficient algorithms to generate a rank-minimized $\mathcal{H}^2$-matrix to represent electrically large surface integral operators for a prescribed accuracy. We first generate an $\mathcal{H}^2$-matrix using the fast multipole method (FMM), and hence, the complexity for $\mathcal{H}^2$-construction is as low as $O(NlogN)$ for solving electrically large surface integral equations. We then develop fast algorithms to convert the FMM-based $\mathcal{H}^2$-matrix whose rank is full asymptotically to a new $\mathcal{H}^2$-representation, whose rank is minimized based on accuracy. The proposed algorithms cost $O(k^3)$ in time for each cluster in cluster basis generation and $O(k^2)$ in memory, where k is the minimal rank of the cluster basis required by accuracy. When the rank of the $\mathcal{H}^2$-matrix is a constant, the complexity of the proposed algorithms is $O(N)$ in both time and memory consumption. When the rank is a variable dependent on electrical size, the total complexity can be evaluated based on the rank's behavior. The resultant rank-minimized $\mathcal{H}^2$-matrix has been employed to accelerate both iterative and direct solutions. Numerical experiments on large-scale surface integral equation-based scattering analysis have demonstrated its accuracy and efficiency.**

***Index Terms*— $\mathcal{H}^2$-matrix, electric field integral equations, electrically large analysis, fast multipole method (FMM), surface integral equations.**

I. Introduction

THE $\mathcal{H}^2$-matrix [1] is a general mathematical framework for compact storage and efficient computation of dense matrices, using which fast solvers can be developed to speed up a computational method. There exist methods for generating an $\mathcal{H}^2$-matrix to represent electrically large surface integral operators (IEs). However, they are generally computationally expensive. For example, in an interpolation-based method [1], [2], the Green's function is directly interpolated to obtain a nested $\mathcal{H}^2$-representation. The resultant rank is of full rank for electrically large kernels, and the transfer matrix and coupling matrix are both dense. As a result, the complexity of using an interpolation-based method to generate the $\mathcal{H}^2$-matrix of an electrically large IE can be as high as $O(N^3)$ in computation time. The adaptive cross approximation (ACA) [3], [4], in conjunction with reduced SVD, has also been employed to generate an $\mathcal{H}^2$-representation of IE operators [5], [6]. However, the complexity remains high for electrically large analysis, scaling as $O(N^2)$ for electrically large SIEs considering rank's growth with electrical size. The methods reported in [7]–[10] show a good complexity for electrically small or moderate problems. Nevertheless, their computational complexity increases for electrically large analysis and remains to be studied in detail.

In this work, we propose to start from a fast multipole method (FMM)-based [11]–[13] $\mathcal{H}^2$-representation of an electrically large IE operator and convert it to a new $\mathcal{H}^2$-representation, whose rank is minimized based on accuracy. In this way, the initial $\mathcal{H}^2$-matrix obtained from the FMM has a low complexity of $O(NlogN)$ for electrically large surface IEs. The FMM and multilevel fast multipole algorithm (MLFMA) have been widely used for solving electrically large electromagnetic problems [14]. There exists research to compress MLFMA to further reduce the matrix-vector multiplication time and storage [15], the cost of which can be high due to the initial high rank of the MLFMA. The matrix structure resulting from an FMM is an $\mathcal{H}^2$-matrix. Furthermore, the FMM-based $\mathcal{H}^2$-matrix is special in the sense that its coupling matrix is diagonal, and its transfer matrix is sparse [16]. However, the resultant rank is a full rank asymptotically for electrically large analysis since it increases quadratically with the electrical size of a cluster in a surface IE. Sometimes, the rank of a cluster in an FMM-based representation can even be larger than the size of the cluster. As a consequence, there is room to further improve the efficiency of the FMM-based representation. In [17], a linear-complexity algorithm is developed to convert a constant-rank $\mathcal{H}^2$-matrix, whose rank is not minimized for accuracy, into a new rank-minimized $\mathcal{H}^2$-matrix. This algorithm is used in [18] to generate an $\mathcal{H}^2$-matrix from the FMM. Even so, the resultant complexity is high due to the large rank of FMM in the electrically large analysis. In [19], a more efficient conversion algorithm is developed, the complexity of which is $O(k^4)$ for each cluster, where k is the minimal rank of the cluster basis required by accuracy.

In this work, we develop fast nested reduction algorithms (NRAs) of further reduced complexity of $O(k^3)$ for each cluster to convert the FMM-based representation into a

Manuscript received May 25, 2020; revised October 12, 2020; accepted November 12, 2020. Date of publication December 21, 2020; date of current version July 7, 2021. This work was supported by a grant from DARPA under Award FA8650-18-2-7847. *(Corresponding author: Dan Jiao.)*

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: djiao@purdue.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAP.2020.3044584.

Digital Object Identifier 10.1109/TAP.2020.3044584

new $\mathcal{H}^2$-matrix whose rank is minimized based on accuracy. Different from the method reported in [19], the proposed new algorithms are much simplified and take much less CPU run time while retaining the same accuracy. Considering the rank's growth with electrical size, the overall time complexity of the proposed algorithms is $O(N^{1.5})$ for electrically large SIEs. The resultant rank-minimized $\mathcal{H}^2$-matrix can then be used to accelerate both iterative and direct solutions. Comparisons with analytical Mie series solutions and reference solutions from a commercial tool have validated the accuracy and efficiency of the proposed method for solving electrically large surface IEs.

The rest of this article is organized as follows. In Section II, we review the background of this article. In Section III, we present the approach to generating an $\mathcal{H}^2$-matrix from the FMM. In Section IV, we detail the proposed NRAs, which converts an FMM-based $\mathcal{H}^2$-matrix into a rank-minimized $\mathcal{H}^2$-matrix efficiently. In Section V, we show another algorithm to further reduce the complexity of the NRA. In Section VI, accuracy and complexity are analyzed. In Section VII, a number of numerical results are presented to validate the accuracy and complexity of the proposed algorithms. In Section VIII, conclusions are drawn.

II. Background

A. Surface Electric Field Integral Equations (S-EFIEs)

The Method of Moments based discretization of a surface electric field integral equation results in a dense system matrix $\mathbf{Z}$, the mnth entry of which can be written as

$$\mathbf{Z}_{m,n} = \int_{S_m}\int_{S_n}\left(\vec{f}_m \cdot \vec{f}_n - \frac{1}{k_0^2}\nabla_s \cdot \vec{f}_m \nabla_s \cdot \vec{f}_n\right) G dS' dS \quad (1)$$

where $G = e^{-jk_0R}/(4\pi R)$ is the Green's function, k_0 is the wavenumber, $R = |\mathbf{r} - \mathbf{r}'|$ is the distance between a source point $\mathbf{r}'$ and an observer point $\mathbf{r}$, and $\vec{f}_m$ and $\vec{f}_n$ are the vector bases on triangular patches S_m and S_n, respectively. Here, triangular elements are used to discretize a surface, and RWG bases [20] are employed to expand unknown currents.

$\mathbf{Z}$ can be expressed as the sum of the electric scalar and magnetic vector potential based components as follows:

$$\mathbf{Z} = -\mathbf{Z}_\phi + \mathbf{Z}_{Ax} + \mathbf{Z}_{Ay} + \mathbf{Z}_{Az} \quad (2)$$

where

$$\mathbf{Z}_{\phi,mn} = \frac{1}{k_0^2}\int_{S_m}\int_{S_n}(\nabla_s \cdot \vec{f}_m \nabla_s \cdot \vec{f}_n) G dS' dS \quad (3)$$

and

$$\mathbf{Z}_{A\xi,mn} = \int_{S_m}\int_{S_n}(f_{m\xi} f_{n\xi}) G dS' dS, \quad \xi = \{x, y, z\}. \quad (4)$$

B. $\mathcal{H}^2$-Matrix

An $\mathcal{H}^2$-matrix [1] represents the interaction between two binary trees, an example of which is shown in Fig. 1. We call the binary tree a cluster tree since each node in the tree corresponds to a cluster of unknowns. All the source basis functions form a column tree, whereas all the observer bases form a row tree. When the testing function is chosen to be the same as the basis function, the resultant matrix is symmetric, whose row and column trees are identical. In an $\mathcal{H}^2$-matrix, by checking the admissibility condition level by level between a row cluster tree and a column cluster tree, the original matrix is partitioned into multilevel admissible and inadmissible blocks. The admissible block is represented by a green submatrix, and the inadmissible one is shown in red in Fig. 1. Physically, an admissible block represents the interaction between separated sources (column cluster) and observers (row cluster), which satisfies the following admissibility condition:

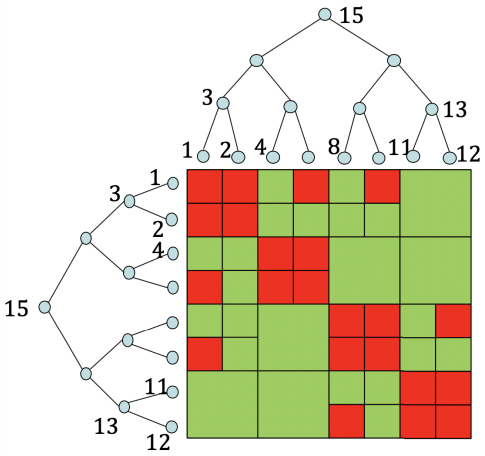


Fig. 1. Illustration of an $\mathcal{H}^2$-matrix.

$$d < \eta D \quad (5)$$

where d denotes the maximal diameter of the row and column cluster, D is the distance between two clusters, and η is a positive parameter, which can be used to control the matrix partition. An inadmissible block is stored in its original full matrix format. An admissible block has compact storage. Take an admissible block formed between a row cluster t and a column cluster s as an example. It is stored as $\mathbf{V}^t\mathbf{S}^{t,s}(\mathbf{V}^s)^H$, where $\mathbf{S}$ is called a coupling matrix and $\mathbf{V}$ is called a cluster basis. $\mathbf{V}$ is nested in the sense that for a cluster t whose children clusters are t_1 and t_2, their cluster bases have the following relationship:

$$\mathbf{V}^t = \begin{bmatrix}\mathbf{V}^{t_1} & \\ & \mathbf{V}^{t_2}\end{bmatrix}\begin{bmatrix}\mathbf{T}^{t_1} \\ \mathbf{T}^{t_2}\end{bmatrix} \quad (6)$$

in which $\mathbf{T}$ matrix is called a transfer matrix.

III. Method for Generating an Initial $\mathcal{H}^2$-Matrix From FMM

According to the addition theorem, Green's function G can be written as

$$\frac{e^{-jk_0R}}{4\pi R} \approx \sum_p \omega_p e^{-jk_0\vec{S}_p\cdot\vec{d}} T_{L,\vec{x}}(\vec{S}_p) \quad (7)$$

where p refers to the index of the sampling point defined on a unit spherical surface, $\vec{S}_p$ and ω_p are the position vector and the quadrature weight of each sampling point, respectively, L

is the truncation parameter used in the addition theorem, and $\vec{d} = \vec{R} - \vec{X} = \mathbf{r} - \mathbf{r}' - \vec{X}$, $\vec{X} = \vec{O}_1 - \vec{O}_2$, with $\vec{O}_1$ being the center of an observer group whose points are denoted by $\mathbf{r}$ and $\vec{O}_2$ the center of a source group whose points are $\mathbf{r}'$ and

$$T_{L,\vec{X}}(\vec{S}_p) = \frac{-jk_0}{4\pi} \sum_{l=0}^{L} \frac{(-j)^l(2l+1)}{4\pi} h_l^{(1)*}(k_0 X) P_l(\vec{S}_p \cdot \hat{X}) \tag{8}$$

where $h_l^{(1)}$ stands for a spherical Hankel function of the first kind, superscript $*$ denotes a complex conjugate, P_l are Legendre polynomials, and $\hat{X}$ denotes a unit vector along $\vec{X}$. Substituting (7) into (3), we obtain

$$\mathbf{Z}_{\phi,ij} = \mathbf{V}_{i,p} S_{p,p} \mathbf{V}_{j,p}{}^H \tag{9}$$

in which

$$\mathbf{V}_{i,p} = \sum_q \frac{w_{i,q}}{k_0} e^{-jk_0 \vec{S}_p \cdot (\vec{r}_i - \vec{O}_1)} \nabla_s \cdot \vec{f}_i(\vec{r}_{i,q}) \tag{10}$$

$$\mathbf{S}_{p,p} = diag(\omega_p T_{l,\vec{O}_1 - \vec{O}_2}(\vec{S}_p)) \tag{11}$$

$$\mathbf{V}_{j,p} = \sum_q \frac{w_{j,q}}{k_0} e^{-jk_0 \vec{S}_p \cdot (\vec{r}_j - \vec{O}_2)} \nabla_s \cdot \vec{f}_j(\vec{r}_{j,q}) \tag{12}$$

where w_q are weighting coefficients used for a numerical surface integration on the source and field triangular patch. It is clear that the number of the sampling points, i.e., the number of p indexes, is the rank of cluster basis $\mathbf{V}$. In the θ-direction, the sampling points are chosen as Gauss–Legendre points, whereas in the ϕ-direction, a uniform sampling is used.

Consider a cluster i, whose parent cluster is i', the $\mathbf{V}_{i'}$ is related to $\mathbf{V}_i$ by $\mathbf{V}_{i'} = \mathbf{V}_i \mathbf{T}$, where $\mathbf{T}$ is called a transfer matrix shown as the following:

$$\mathbf{T}_{pp'} = \sum_{m,l \le K} Y_{ml}(\theta'_{p'}, \phi'_{p'}) Y^*_{ml}(\theta_p, \phi_p) \omega_p e^{-jk_0 \vec{S}'_p (\vec{O}_1 - \vec{O}'_1)}$$

where p' is the index of the sampling points for $\mathbf{V}_{i'}$, K here stands for the number of quadrature points in the θ-direction of $\mathbf{V}_i$, and Y_{ml} are spherical harmonics. $\mathbf{T}$ is sparse for a prescribed accuracy, whose number of entries per column (row) is a constant regardless of matrix size. This number can be further reduced using a filter [13]. The $\mathcal{H}^2$ representation for the vector magnetic potential term (4) can be generated in a similar way as (9).

IV. Proposed NRA

For an arbitrary cluster t in the cluster tree, let its original cluster basis obtained from the FMM be $\mathbf{V}^t$. Such $\mathbf{V}^t$ is of size $\#t$ by k_F, where $\#t$ denotes the number of unknowns contained in cluster t and k_F is the rank resulting from the FMM. Since k_F is large, the algorithm presented here is to generate a new cluster basis $\tilde{\mathbf{V}}^t$ such that its rank k is the minimal one required by accuracy and hence being much smaller than k_F. Certainly, such a rank reduction algorithm must retain the original nested property of the $\mathbf{V}$ across the cluster tree while keeping the computational complexity low. In addition, for an electrically large analysis, both k and k_F are electrical size-dependent and, hence, tree-level dependent. For the simplicity of the notation, we would not add a tree-level dependence for k and k_F. However, it should be noted that they are different at different tree levels. The k_F scales quadratically with the electrical size, whereas k scales linearly with the electrical size [21].

We start from the leaf level $l = \mathcal{L}$ (root level is at $l = 0$) of the cluster tree. For a leaf cluster t, we perform an SVD on $\mathbf{V}^t$. Based on prescribed accuracy ϵ, we keep the singular vectors whose singular values normalized by the maximum one are no less than ϵ. These singular vectors make the new leaf cluster basis $\tilde{\mathbf{V}}^t$. Thus, we obtain

$$\mathbf{V}^t_{\#t \times k_F} \overset{\epsilon}{\approx} \tilde{\mathbf{V}}^t_{\#t \times k} (\tilde{\mathbf{U}}^t)^H_{k \times k_F} \tag{13}$$

where the subscripts denote the matrix dimension and $(\tilde{\mathbf{U}}^t)^H$ is the other factor resulting from the SVD. Since $\#t$ is bounded by $leafsize$ which is a constant, the cost of (13) is constant, which is small.

We then proceed to the nonleaf level. For each nonleaf cluster t, its cluster basis is related to its children clusters' bases as shown in (6). Now, the children clusters' bases have been changed. Hence, the transfer matrices must be updated for nonleaf cluster t. To see how to update them, we can substitute (13) into (6) to obtain

$$\mathbf{V}^t = \begin{bmatrix} (\tilde{\mathbf{V}}^{t_1}) & \\ & (\tilde{\mathbf{V}}^{t_2}) \end{bmatrix} \begin{bmatrix} (\tilde{\mathbf{U}}^{t_1})^H \mathbf{T}^{t_1} \\ (\tilde{\mathbf{U}}^{t_2})^H \mathbf{T}^{t_2} \end{bmatrix} \tag{14}$$

from which it can be seen that

$$\hat{\mathbf{T}}^t = \begin{bmatrix} (\tilde{\mathbf{U}}^{t_1})^H \mathbf{T}^{t_1} \\ (\tilde{\mathbf{U}}^{t_2})^H \mathbf{T}^{t_2} \end{bmatrix} \tag{15}$$

makes the new transfer matrix of cluster t. However, its rank may not be the minimal one required by accuracy. Therefore, we perform another SVD on (15) and truncate singular vectors based on accuracy ϵ to obtain the new transfer matrix $\tilde{\mathbf{T}}^t$. As a result, we have

$$\hat{\mathbf{T}}^t_{(k_1+k_2) \times k_F} \overset{\epsilon}{\approx} \tilde{\mathbf{T}}^t_{(k_1+k_2) \times k} (\tilde{\mathbf{U}}^t)^H_{k \times k_F} \tag{16}$$

where the rank $k_1 + k_2$, which is the sum of the rank of the two children's new cluster bases, is further reduced to k based on prescribed accuracy.

The aforementioned procedure at a nonleaf level is then repeated level by level up, until we reach the minimal level that has admissible blocks. At that level, we finish generating the new cluster bases. After that, we update the original FMM-based coupling matrix, $\mathbf{S}^{t,s}$, as follows for each admissible block:

$$\tilde{\mathbf{S}}^{t,s} = (\tilde{\mathbf{U}}^t)^H \mathbf{S}^{t,s} \tilde{\mathbf{U}}^s. \tag{17}$$

The overall procedure is a bottom-up tree traversal procedure summarized as follows, which is termed the NRA.

At each tree level of the cluster tree, we do the following computation.

1) For a leaf cluster t, do (13) to obtain new cluster basis $\tilde{\mathbf{V}}^t$ based on required accuracy ϵ.
2) For a nonleaf cluster t, compute (15) to obtain $\hat{\mathbf{T}}^t$, then factorize it to obtain (16), and hence new transfer matrix $\tilde{\mathbf{T}}^t$ as well as $(\tilde{\mathbf{U}}^t)^H$, based on accuracy ϵ.

If column cluster bases are different from row cluster bases such as those in an unsymmetrical matrix, Steps 1 and 2 are repeated for column cluster bases. After cluster basis update, for each admissible block, we update coupling matrix based on (17).

At a nonleaf level, the cost of computing (15) is only of $O(kk_F)$, which is $O(k^3)$, since transfer matrix $\mathbf{T}$ is sparse. However, the subsequent step of computing the SVD of $\hat{\mathbf{T}}^t$ can be costly, which is $O(k^2k_F)$, and hence scaling as $O(k^4)$. To circumvent this cost, we propose a fast algorithm as the following. For a nonleaf cluster t, in (15), we randomly choose $O(k_1+k_2)$ columns to compute its low-rank factorization (16) instead of operating on the full k_F columns. This can be done because the rank of $\hat{\mathbf{T}}^t$ is no greater than its row rank, which is bounded by k_1+k_2. We then compute a full cross approximation (FCA) [1] with prescribed accuracy on the randomly selected $O(k_1+k_2)$ columns of $\hat{\mathbf{T}}^t$, obtaining row pivots τ and column pivots σ. As a result, $\hat{\mathbf{T}}^t$ is factorized to

$$\hat{\mathbf{T}}^t \overset{\epsilon}{\approx} \hat{\mathbf{T}}^t_{:,\sigma}\left(\hat{\mathbf{T}}^t_{\tau,\sigma}\right)^{-1}\hat{\mathbf{T}}^t_{\tau,:} \tag{18}$$

whose new rank is k. Since the FCA is performed on the $O(k_1+k_2)$ columns of $\hat{\mathbf{T}}^t$, the entire computational cost of obtaining (18) is reduced to $O(k^3)$ compared to a brute-force SVD-based low-rank compression of $\hat{\mathbf{T}}^t$. In (18), $\hat{\mathbf{T}}^t_{:,\sigma}$ denotes the selected k columns of $\hat{\mathbf{T}}^t$, whose indexes are contained in σ. $\hat{\mathbf{T}}^t_{:,\sigma}$ multiplied by the following small k by k matrix, $(\hat{\mathbf{T}}^t_{\tau,\sigma})^{-1}$, is nothing, but the new transfer matrix of t, thus

$$\tilde{\mathbf{T}}^t = \hat{\mathbf{T}}^t_{:,\sigma}\left(\hat{\mathbf{T}}^t_{\tau,\sigma}\right)^{-1} \tag{19}$$

whereas the

$$(\tilde{\mathbf{U}}^t)^H = \hat{\mathbf{T}}^t_{\tau,:} \tag{20}$$

is the k rows of $\hat{\mathbf{T}}^t$, whose row indexes are contained in τ. In this way, $(\tilde{\mathbf{U}}^t)^H$ can be obtained in $O(k^3)$ cost because it involves k-rows of $(\tilde{\mathbf{U}})^H$ multiplied by a sparse $\mathbf{T}$, as can be seen from (15).

It is worth mentioning the FCA instead of ACA is employed here because the accuracy of the former is better than the latter. Furthermore, for a low-rank matrix, the accuracy of an FCA is guaranteed. Certainly, the SVD can be directly used on the selected $O(k)$ columns to obtain new bases, which has a reduced cost of $O(k^3)$ as well. However, if we do that, the subsequent step of obtaining $(\tilde{\mathbf{U}}^t)^H$ would cost more than $O(k^3)$.

The pseudocode of the aforementioned fast NRA is shown in **Algorithm** 1. In line 2 of this algorithm, Algorithm 2 is called starting from the leaf level of the cluster tree, which factorizes $\mathbf{V} = \tilde{\mathbf{V}}(\tilde{\mathbf{U}})^H$ for each leaf cluster and then factorizes $\hat{\mathbf{T}} = \tilde{\mathbf{T}}(\tilde{\mathbf{U}})^H$ for each nonleaf cluster. Algorithm 2 yields new rank-minimized cluster bases based on accuracy. In line 3, we apply Algorithm 3 to the root of the block cluster tree of the $\mathcal{H}^2$-matrix, which updates the coupling matrix of each admissible block.

From the aforementioned cost analysis given along with the description of the algorithm, it can be seen that the time cost of obtaining new cluster bases $\tilde{\mathbf{V}}^t$ is low, which is $O(k^3)$ for each cluster. Here, note that k is the minimal rank required by accuracy instead of the original FMM's rank. However, the storage of $\tilde{\mathbf{U}}^t$ would cost $O(kk_F)$ units for each cluster, thus being $O(k^3)$. In Section V, we propose another NRA to reduce the memory cost.

Algorithm 1 Nested_Reduction_Algorithm

1: **procedure** NESTED_REDUCTION_ALGORITHM(t, b)
2: update_cluster_basis(t)
3: update_coupling_matrix(b)
4: **end procedure**

Algorithm 2 Update_Cluster_Basis

This algorithm efficiently obtains new nested cluster bases with rank minimized based on accuracy

1: **procedure** RANDOMIZED_UPDATE_CLUSTER_BASIS(t)
2: **if** t is a non-leaf cluster **then**
3: Compute $\hat{\mathbf{T}} = \begin{bmatrix}(\tilde{\mathbf{U}}^{t1})^H\mathbf{T}^{t1}\\(\tilde{\mathbf{U}}^{t2})^H\mathbf{T}^{t2}\end{bmatrix}$
4: Randomly select $|c^t|$ columns from $\hat{\mathbf{T}}$ to form $\hat{\mathbf{T}}_w$.
5: Do FCA based on ϵ on $\hat{\mathbf{T}}_w$ to get τ and σ
6: Obtain $(\tilde{\mathbf{U}}^t)^H = \hat{\mathbf{T}}^t_{\tau,:}$, $\tilde{\mathbf{T}} = \hat{\mathbf{T}}^t_{:,\sigma}(\hat{\mathbf{T}}^t_{\tau,\sigma})^{-1}$
7: **else**
8: Do SVD on $\mathbf{V}^t$ such that $\mathbf{V}^t = \tilde{\mathbf{V}}^t(\tilde{\mathbf{U}}^t)^H$
9: **end if**
10: **end procedure**

V. Nest Reduction Algorithm With Reduced Memory Complexity

In this new algorithm, there are also two steps. One is to generate new cluster basis whose rank is minimized, and the other is to update coupling matrices, similar to the algorithm presented in Section IV. However, we do not explicitly compute or store $\tilde{\mathbf{U}}$ matrix for each cluster. As can be seen from (13) and (16), if the new cluster basis $\tilde{\mathbf{V}}^t$ is made unitary, then $(\tilde{\mathbf{U}}^t)^H$ is nothing, but the projection of the original basis onto the new basis, thus

$$(\tilde{\mathbf{U}}^t)^H = (\tilde{\mathbf{V}}^t)^H\mathbf{V}^t. \tag{21}$$

Hence, whenever $(\tilde{\mathbf{U}}^t)^H$ is involved in computation, we can utilize the nested property of both the original basis $\mathbf{V}^t$ and the new basis $(\tilde{\mathbf{V}}^t)$ to compute it efficiently. Storage wise, we only need to store the new cluster basis $(\tilde{\mathbf{V}}^t)$ whose rank is minimized and the original cluster basis $\mathbf{V}^t$ which is sparse,

Algorithm 3 Update_Coupling_Matrix

This algorithm updates coupling matrices

1: **procedure** UPDATE_COUPLING_MATRIX(b)
2: **if** b is an admissible block **then**
3: $\tilde{\mathbf{S}}^{t,s} = (\tilde{\mathbf{U}}^t)^H\mathbf{S}^{t,s}\tilde{\mathbf{U}}^s$
4: **else if** i is b's child **then**
5: update_coupling_matrix(b^i)
6: **end if**
7: **end procedure**

thus bypassing the storage of $(\tilde{\mathbf{U}}^t)^H$. Next, we elaborate this algorithm.

At the leaf level, the computation is the same as that in the previous algorithm, where SVD is used to obtain $\tilde{\mathbf{V}}^t$ so that each leaf cluster basis is unitary. At a nonleaf level, for each cluster t, we randomly select $O(k_1+k_2)$ columns of $\mathbf{T}^t$ to compute (15). Let this set be c^t. Computing c^t-columns of $\hat{\mathbf{T}}^t$ is the same as computing c^t-columns of $(\tilde{\mathbf{V}}^t)_{ch}^H\mathbf{V}^t$, where $(\tilde{\mathbf{V}}^t)_{ch}$ is a block diagonal matrix containing t's two children's new cluster bases. To see this more clearly, we can rewrite (15) as

$$\hat{\mathbf{T}}^t = \begin{bmatrix} (\tilde{\mathbf{V}}^{t_1})^H & \\ & (\tilde{\mathbf{V}}^{t_2})^H \end{bmatrix} \begin{bmatrix} \mathbf{V}^{t_1}\mathbf{T}^{t_1} \\ \mathbf{V}^{t_2}\mathbf{T}^{t_2} \end{bmatrix} \tag{22}$$

which is nothing but

$$\hat{\mathbf{T}}^t = (\tilde{\mathbf{V}}^t)_{ch}^H\mathbf{V}^t. \tag{23}$$

For each index c in the set c^t, we form a cardinal vector $\boldsymbol{e}^{\boldsymbol{c}}$, which has only one nonzero element at the cth entry. Multiplying $(\tilde{\mathbf{V}}^t)_{ch}^H\mathbf{V}^t$ by $\boldsymbol{e}^{\boldsymbol{c}}$ is the same as computing $\mathbf{V}^t\boldsymbol{e}^{\boldsymbol{c}}$ first and then multiplying the resultant vector by $(\tilde{\mathbf{V}}^t)_{ch}^H$, each of which costs $O(nlogn)$ complexity using the nested property of both bases, where n is the size of cluster t. After obtaining the c^t columns of $\hat{\mathbf{T}}^t$, we perform an SVD on it based on prescribed accuracy ϵ to obtain new transfer matrix $\tilde{\mathbf{T}}^t$ whose rank is reduced to k. The cost of this step is $O(k^3)$. In addition, such a new transfer matrix is unitary. The aforementioned procedure of computing new transfer matrix $\tilde{\mathbf{T}}$ at a nonleaf level continues level by level up until the minimum level having admissible block is reached.

The pseudocode of the new algorithm is shown in Algorithm 4, in which the fast matrix-vector multiplication algorithm for $\mathbf{V}^t$ and $(\tilde{\mathbf{V}}^t)^H$ is shown in Algorithms 5 and 6, respectively.

Algorithm 4 New_Update_Cluster_Basis

This algorithm is for new cluster basis generation.

```
1: procedure NEW UPDATE CLUSTER BASIS(t)
2:   if t is a non-leaf cluster then
3:     Randomly select c^t pivots from k_F columns of
           t's original transfer matrix T^t
4:     for c ∈ c^t do
5:       ω^t = e^c
6:       recursively_multi_old_trans(t, ω_t)
7:       recursively_multi_new_trans(t, ω_t)
8:     end for
9:     Use the resultant c^t columns of T̂^t to form T̂^t_w
10:    Do SVD on T̂^t_w based on ε to get T̃^t
11:  else
12:    Do SVD on V^t to obtain Ṽ^t
13:  end if
14: end procedure
```

After generating new cluster bases, next, we update the coupling matrix of each admissible block. Similarly, using (21), (17) becomes the computation of

$$\tilde{\mathbf{S}}^{t,s} = (\tilde{\mathbf{V}}^t)^H\mathbf{V}^t\mathbf{S}^{t,s}(\mathbf{V}^s)^H\tilde{\mathbf{V}}^s. \tag{24}$$

Algorithm 5 recursively_Multi_Old_Trans

This algorithm performs a top-down tree travserval to compute $\mathbf{V}\omega$

```
1: procedure RECURSIVELY_MULTI_OLD_TRANS(t, ω)
2:   if t is a non-leaf cluster then
3:     for i is t's child do
4:       ω^i = T^i ω
5:       recursively_multi_old_trans(t_i, ω^i)
6:     end for
7:   else
8:     ω^t = V^t ω^t
9:   end if
10: end procedure
```

Algorithm 6 Recursively_Multi_New_Trans

This Algorithm performs a bottom-up tree travserval to compute $\tilde{\mathbf{V}}^H\omega$

```
1: procedure RECURSIVELY_MULTI_NEW_TRANS(t, ω)
2:   if t is a non-leaf cluster then
3:     for i is t's child do
4:       recursively_multi_new_trans(t_i, ω^i)
5:     end for
6:     ω^t ← [(T̃^{t_1})^H ω^1 ; (T̃^{t_2})^H ω^2]
7:   else
8:     ω^t ← (Ṽ^t)^H ω^t
9:   end if
10: end procedure
```

Since $\tilde{\mathbf{V}}$ basis is of rank k, $\tilde{\mathbf{V}}^s$ has only k columns. The computation of $(\mathbf{V}^s)^H\tilde{\mathbf{V}}^s$ is the computation of k matrix-vector multiplications of $(\mathbf{V}^s)^H$ multiplying the k columns in $\tilde{\mathbf{V}}^s$. This can be done using Algorithm 6 where $\tilde{}$ on top of the symbols is removed. This step costs $O(nlogn)$ for one vector. Hence, the total cost of $(\mathbf{V}^s)^H\tilde{\mathbf{V}}^s$ is $O(knlogn)$, which is $O(k^3logk)$ since n scales as k^2. Next, we multiply $\mathbf{S}^{t,s}$ by the computed $(\mathbf{V}^s)^H\tilde{\mathbf{V}}^s$. Since $\mathbf{S}^{t,s}$ is diagonal, the cost of this step is $O(k_Fk)$, which is $O(k^3)$. After that, we multiply $(\tilde{\mathbf{V}}^t)^H\mathbf{V}^t$ by the k vectors resulting from $\mathbf{S}^{t,s}(\mathbf{V}^s)^H\tilde{\mathbf{V}}^s$, which again can be computed by k matrix-vector multiplications using $\mathbf{V}^t$, followed by the other k matrix-vector multiplications of $(\tilde{\mathbf{V}}^t)^H$. The cost of this step is also $O(knlogn)$ utilizing the nest property of the cluster bases.

The memory requirement of the new algorithm for each cluster is $O(k^2)$. This is because we do not store $\tilde{\mathbf{U}}$ for each cluster. Instead, we store $\tilde{\mathbf{V}}$ and $\mathbf{V}$. At the leaf level, the storage is a constant for each cluster. At a nonleaf level, the storage is a new transfer matrix of size $k\times k$ for each cluster. As for the original cluster basis $\mathbf{V}$, the storage is a transfer matrix of size $k_F\times k_F$ for each cluster. However, this transfer matrix is sparse, thus costing $O(k_F)\approx O(k^2)$ units to store also.

VI. Accuracy and Complexity

A. Accuracy

In the algorithms proposed in this work, only the steps of SVD and FCA involve approximations. However, they are

performed subject to a prescribed accuracy. Hence, the overall procedure is error controlled. In the FCA part, we randomly select $O(k_1+k_2) = c(k_1+k_2)$ columns to perform FCA, where c is a constant coefficient greater than 1. Since the matrix upon which FCA is performed is known to be bounded by $k_1 + k_2$ in rank and c is chosen to be greater than 1, the resultant cross approximation yields an accurate rank-k representation. As can be seen from [10], the choice of k columns is not unique in an ACA or FCA algorithm for approximating a rank-k matrix. As long as the k columns are linearly independent for a prescribed accuracy, they produce an accurate rank-k model. Different from ACA, in an FCA, at every step, the maximum entry in the residual matrix is identified, whose row and column pivots are chosen to generate a rank-1 model. Hence, the accuracy of FCA is guaranteed [1]. If it happens that the randomly selected $c(k_1+k_2)$ columns do not contain k linearly independent columns, it can be identified in the FCA process, and more columns can then be selected.

B. Time and Memory Complexity

The complexity analysis is dependent on the rank's behavior. Consider a rank that grows linearly with the electrical size. This is also shown to be the minimal rank required by accuracy in an electrically large IE analysis [21]. Let n be the size of a cluster, and then, in an SIE, the rank k scales as

$$k = O(\sqrt{n}). \tag{25}$$

As for the rank of FMM, it scales quadratically with the electrical size, thus

$$k_F = O(n). \tag{26}$$

In the proposed fast NRA algorithm, every cluster basis needs $O(k^3+k_Fk) \approx O(k^3)$ operations to be computed. Based on (25) and (26), the total time complexity for new cluster basis generation can be computed as

$$\begin{aligned} C_t &= \sum_{l=0}^{L} 2^l O(kk_F + k^3) \\ &= \sum_{l=0}^{L} 2^l O\left(\sqrt{\frac{N}{2^l}}\frac{N}{2^l} + \left(\sqrt{\frac{N}{2^l}}\right)^3\right) = O(N^{1.5}). \end{aligned} \tag{27}$$

The time complexity for coupling matrix updates can be computed as

$$\begin{aligned} C_{t,S} &= \sum_{l=0}^{L} 2^l C_{sp} O(k^2 k_F) \\ &= \sum_{l=0}^{L} 2^l C_{sp} O\left(\sqrt{\frac{N}{2^l}}^2 \frac{N}{2^l}\right) = O(N^2) \end{aligned} \tag{28}$$

where C_{sp} denotes the maximal number of blocks formed by a single cluster, which is a constant [1]. The total memory complexity can be computed by adding the memory cost of each cluster basis with that of each admissible block as follows:

$$\begin{aligned} C_m &= \sum_{l=0}^{L} (2^l O(k^3) + C_{sp} 2^l O(k^2)) \\ &\approx \sum_{l=0}^{L} 2^l O\left(\frac{N}{2^l}\right)^{1.5} = O(N^{1.5}). \end{aligned} \tag{29}$$

Hence, $O(N^{1.5})$ memory is required during the rank reduction. After the rank reduction is finished, the memory of storing the new rank-minimized $\mathcal{H}^2$-matrix would be just $O(NlogN)$ for SIE since only $O(k^2)$ is required for storing each cluster basis and each admissible block.

As for the new algorithm shown in Section V, the complexity for every cluster is $O(knlogn)$ in time and $O(k^2)$ in memory. Adding the cost of every cluster across all tree levels, we obtain the total time cost as

$$\begin{aligned} C_t &= \sum_{l=0}^{L} 2^l O(knlogn) \\ &= \sum_{l=0}^{L} 2^l O\left(\sqrt{\frac{N}{2^l}}\frac{N}{2^l} l\right) = O(N^{1.5}). \end{aligned} \tag{30}$$

Similarly, updating all the admissible blocks has also $O(N^{1.5})$ complexity since, at each tree level, there are $2^l O(C_{sp})$ admissible blocks, each of which costs $O(knlogn)$ operations to update. The total memory consumption, including the memory required for both cluster basis generation and coupling matrix updates, scales as

$$\begin{aligned} C_m &= \sum_{l=0}^{L} (2^l O(k^2) + C_{sp} 2^l O(k^2)) \\ &= \sum_{l=0}^{L} 2^l O\left(\frac{N}{2^l}\right) = O(NlogN) \end{aligned} \tag{31}$$

for electrically large SIE analyses.

It is worth mentioning that although the new algorithm mentioned in Section V has the same time complexity in cluster basis generation compared to that in Section IV, the constant in front of the $N^{1.5}$ is larger. As for the coupling matrix update, the new algorithm in Section V has a reduced complexity of $O(N^{1.5})$, but the constant is also larger. We note that the absolute run time of the algorithm in Section V can exceed that of the fast NRA algorithm in Section IV when simulating medium-sized problems. However, memory and its scaling rate are reduced. In addition, from the aforementioned complexity analysis, it can be seen that for applications where the rank is a constant, the total complexity of the proposed algorithms is $O(N)$.

C. Further Rank Reduction

In the proposed NRA algorithms, the original FMM-based cluster bases are reduced in rank based on accuracy. To further explore the redundancy in the matrix content, we employ the algorithm in [16] on top of the new $\mathcal{H}^2$-matrix generated from the proposed NRA algorithms to further reduce its rank. The algorithm in [16] can be used to convert an $\mathcal{H}^2$-matrix whose

rank is not minimized into a new one that is minimized based on accuracy. However, if the algorithm in [16] is directly applied to an FMM-based matrix, the conversion cost would be too high since the starting rank is high. In contrast, when using the $\mathcal{H}^2$-matrix generated from the proposed NRA algorithms to do further compression using [16], the cost is as low as $O(k^3)$ for each cluster and admissible block in time and $O(k^2)$ in memory, thus not increasing the complexity of the proposed NRA algorithms.

VII. Numerical Results

A common set of simulation parameters are used for all examples simulated in this section. Specifically, in the FMM, we set the truncation criterion of the addition theorem as $L = k_0 d + 1.8 d_0^{2/3} (k_0 d)^{1/3}$, where $d_0 = log_{10}(1/\epsilon_F)$ and $\epsilon_F = 10^{-2}$, k_0 is the wavenumber, and d is the diameter of the targeted cluster. When generating an $\mathcal{H}^2$-representation of the FMM, we use six points along each of the θ- and ϕ-directions in the Lagrange polynomial-based interpolation to obtain transfer matrices. In addition, we choose $\eta = 0.8$ in the admissibility condition and $leafsize$ to be 40. The choice of η and $leafsize$ follows the same rule discussed in [2]. The only simulation parameter in the proposed NRA algorithm is accuracy parameter ϵ, which is a user-defined parameter. When randomly choosing $\#c^t = c(k_1 + k_2)$ columns from $\hat{\mathbf{T}}^t$ in fast NRA, the c can be chosen as an arbitrary constant, but a larger choice of c would increase CPU run time. In the case that the direct solver of [22] is used to solve the $\mathcal{H}^2$-matrix generated from the proposed algorithm, the accuracy of the direct solver is set to be 10^{-3}.

A. Accuracy

We first validate the accuracy of the proposed algorithms before examining their complexity in time and memory.

1) Accuracy Comparison Between the Proposed Fast NRA Algorithm and the Proposed NRA: In Algorithm 2, a fast algorithm is developed to bypass the cost of the SVD in the proposed NRA algorithm. In this algorithm, $O(k)$ columns are randomly selected to perform FCA. Here, we examine the accuracy of this approach compared to the brute-force NRA. We take a random vector x and perform a matrix-vector multiplication using the new $\mathcal{H}^2$-matrix generated by the brute-force NRA to obtain $\mathbf{Z}_{\mathcal{H}^2} x$. We also use the fast NRA to generate an $\mathcal{H}^2$-matrix $\mathbf{Z}_{fast\mathcal{H}^2}$ and compute $\mathbf{Z}_{fast\mathcal{H}^2} x$. The error is then assessed by comparing the result with the original FMM-based matrix-vector multiplication, $\mathbf{Z}_{FMM} x$. The results for a sphere example whose diameter ranges from 2.21 wavelengths to 17.68 wavelengths are shown in Table I for $\epsilon = 10^{-2}$ and $\epsilon = 10^{-3}$, respectively. In this table, $err_0 = \|\mathbf{Z}_{\mathcal{H}^2} x - \mathbf{Z}_{FMM} x\| / \|\mathbf{Z}_{FMM} x\|$ represents the relative error of the new $\mathcal{H}^2$ generated by the brute-force NRA, and $err_1 = \|\mathbf{Z}_{Fast\mathcal{H}^2} x - \mathbf{Z}_{FMM} x\| / \|\mathbf{Z}_{FMM} x\|$ represents that of the fast NRA. As can be seen, the fast NRA is accurate, and its accuracy is also controllable like the NRA. In this example, we also simulate a larger sphere whose diameter is 35.36 m, and the number of unknowns is 1 179 648. With $\epsilon = 10^{-2}$, we find that the accuracies of the proposed NRA algorithm and fast NRA are 2.42×10^{-2} and 4.51×10^{-2}, respectively.

TABLE I
Accuracy Comparison Between NRA and Fast NRA

N	4608	18432	73728	294912
$err_0(\epsilon = 10^{-2})$	7.44e-3	1.07e-2	1.66e-2	2.07e-2
$err_1(\epsilon = 10^{-2})$	6.92e-3	1.18e-2	1.83e-2	2.41e-2
$err_0(\epsilon = 10^{-3})$	1.03e-3	1.13e-3	1.79e-3	2.28e-3
$err_1(\epsilon = 10^{-3})$	9.42e-4	1.17e-3	1.95e-3	2.71e-3

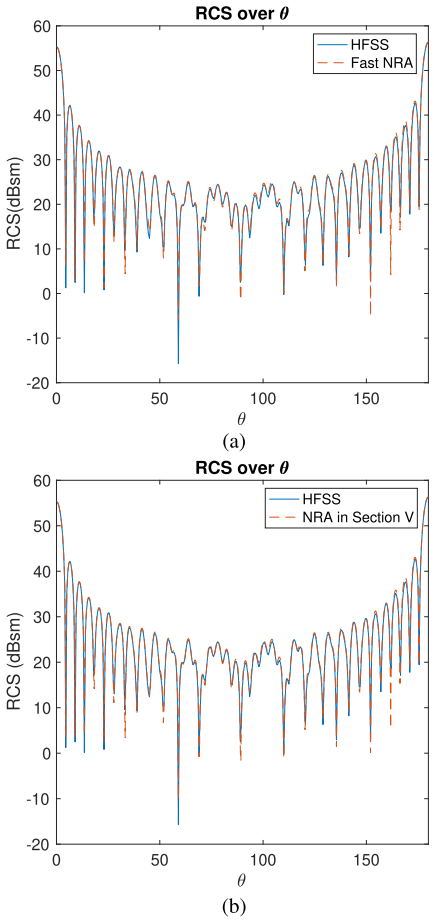


Fig. 2. RCS of a conducting cube simulated using two different algorithms. (a) Fast NRA. (b) NRA in Section V.

2) Scattering From a Conducting Cube: In this example, we compute the bistatic RCS of a conducting cube of size $12.8\lambda \times 12.8\lambda \times 12.8\lambda$ at 300 MHz, which has 294 912 unknowns. The new $\mathcal{H}^2$-matrix generated from the proposed method is solved using the direct solver of [22]. The resultant bistatic RCS is compared with that from HFSS, which shows very good agreement, as can be seen from Fig. 2(a). In this example, the accuracy criterion used in the fast NRA is set to be $\epsilon = 10^{-3}$. We also use the NRA with reduced memory complexity, shown in Section V, to simulate the same example. As can be seen from Fig. 2(b), the algorithm shows good accuracy as well.

3) Scattering From a Conducting Plate: In this example, we compute the bistatic RCS of a conducting plate of size $60.8\lambda \times 60.8\lambda$ at 300 MHz, which has 1 107 776 unknowns. A BiCGStab iterative solver with a diagonal preconditioner is employed to solve the new $\mathcal{H}^2$-matrix generated from

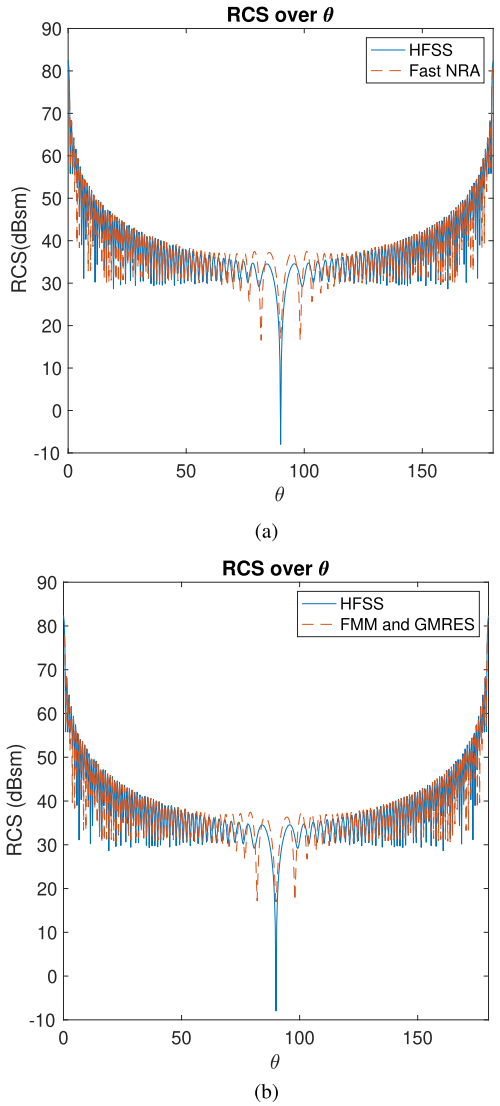


Fig. 3. (a) RCS of a conducting plate simulated using the new $\mathcal{H}^2$-matrix generated from fast NRA. (b) RCS simulated using FMM.

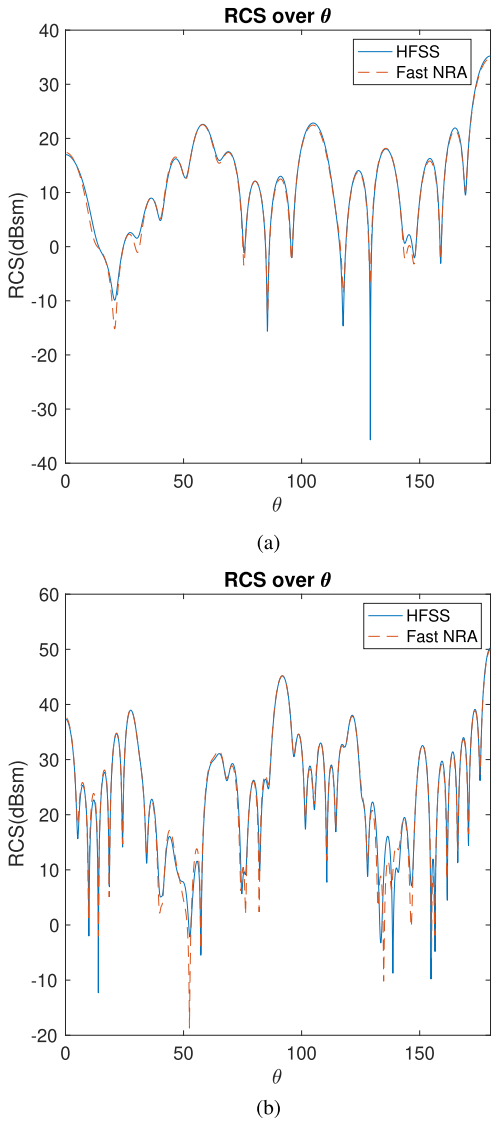


Fig. 4. Simulated RCS of an array of conducting spheres using the new $\mathcal{H}^2$-matrix generated from fast NRA, solved by a direct solver. (a) $4 \times 4 \times 4$ array. (b) $6 \times 6 \times 6$ array.

the proposed fast NRA. The result is then compared with HFSS and shown in Fig. 3(a). Again, very good agreement is observed, which validates the accuracy of the proposed algorithm. The simulation parameters are chosen the same as in the previous cube example. For comparison, we also plot the RCS obtained by using FMM with GMRES and $\epsilon = 5 \times 10^{-3}$ in Fig. 3(b). As can be seen, the slight discrepancy between the proposed method and HFSS is not due to the proposed method.

4) Scattering From an Array of Spheres: In this example, we simulate an array of spheres, having $4 \times 4 \times 4$ spheres, each of which has a diameter of 0.5525 m and is discretized with 288 unknowns at 300 MHz. The distance between the two adjacent spheres is 1.3820 m. The bistatic RCS is computed with a direct solution of the new $\mathcal{H}^2$-matrix by the direct solver in [22] and compared with HFSS's result. Good agreement is observed, as can be seen from Fig. 4(a). The accuracy criterion used in the fast NRA is $\epsilon = 10^{-3}$.

We also simulate another array having $6 \times 6 \times 6$ spheres, each of which is of diameter 0.8288 m and is discretized with 648 unknowns, yielding 139 968 unknowns in total at 300 MHz. The distance between two adjacent spheres is 2.0730 m. Again, the new $\mathcal{H}^2$-matrix generated from the Fast NRA is directly solved using the solver in [22], and the bistatic RCS is extracted and compared with HFSS. The comparison is shown in Fig. 4(b), which further validates the accuracy of the proposed algorithm. The simulation parameters are the same as those in the previous example.

For comparison, we use an FMM with GMRES to iteratively solve the $6 \times 6 \times 6$ sphere array, the result of which is shown in Fig. 5(a). Similarly, we use the new $\mathcal{H}^2$-matrix generated by fast NRA and GMRES to iteratively solve the same example. As can be seen from Fig. 5(b), a similar agreement is observed in the RCS result.

5) Scattering From More Complicated Structures: We also simulate a coil, whose shape is shown in Fig. 6(a). This coil has a diameter of 14.156 m and illuminated by an incident field at 300 MHz. After discretization, the number of unknowns is

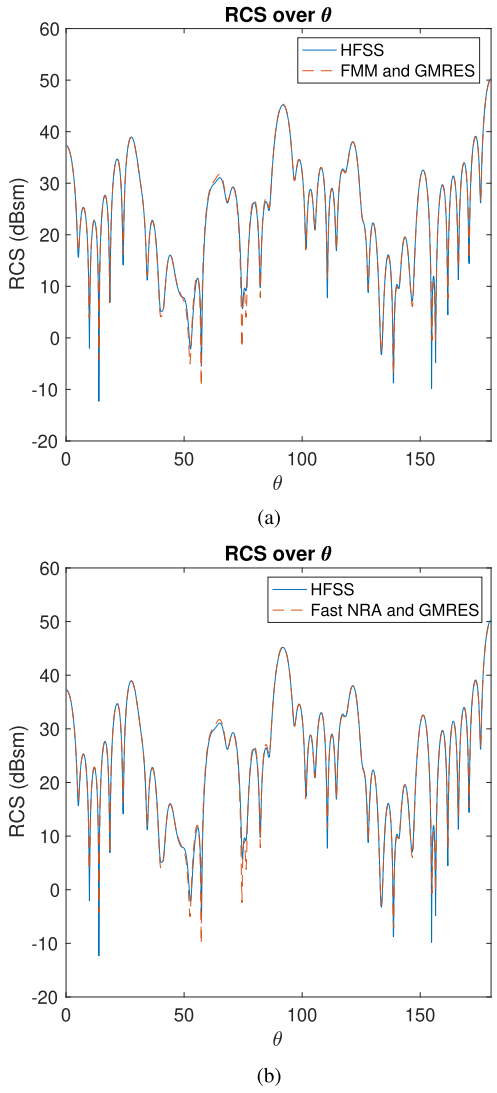


Fig. 5. RCS of the $6 \times 6 \times 6$ sphere array simulated using (a) FMM with a GMRES iterative solver. (b) New $\mathcal{H}^2$-matrix generated from the Fast NRA with a GMRES solver.

121 914. The new $\mathcal{H}^2$-matrix generated from the Fast NRA is then solved using the fast direct solver mentioned in [22]. The resultant bistatic RCS is compared with HFSS's result in Fig. 6(b). As can be seen, the two agree well with each other.

Another example we simulated is shown in Fig. 7(a). The structure has a diameter of 17.302 m and discretized into 172 077 unknowns at 300 MHz. The new $\mathcal{H}^2$-matrix generated from the Fast NRA is again solved using the fast direct solver mentioned in [22]. The resultant bistatic RCS is compared with HFSS's result in Fig. 7(b). Good agreement can be observed.

B. Time and Memory Complexity

With the accuracy of the proposed algorithms validated, next, we examine the complexity of the proposed algorithms for generating a rank-minimized $\mathcal{H}^2$-matrix for electrically large analysis.

1) Growth Rate of the Rank: First, we examine the growth rate of the rank with electrical size since it is one of the key parameters in the complexity analysis. We use a conducting sphere as an example and find its rank level by level using the proposed NRA algorithm with $\epsilon = 10^{-3}$. The results are listed in Table II. In this table, e_s denotes the largest electrical size of all clusters at a tree level, n is the largest unknown number of all clusters, k_F is the FMM's rank (note that in FMM, all cluster bases at the same tree level share the same rank in common), k is the rank of the new $\mathcal{H}^2$-matrix of the entire SIE obtained from the proposed reduction algorithm, and k_ϕ is the rank of the new $\mathcal{H}^2$-matrix for $\mathbf{Z}_\phi$ part only. We also plot the new rank as a function of electrical size in

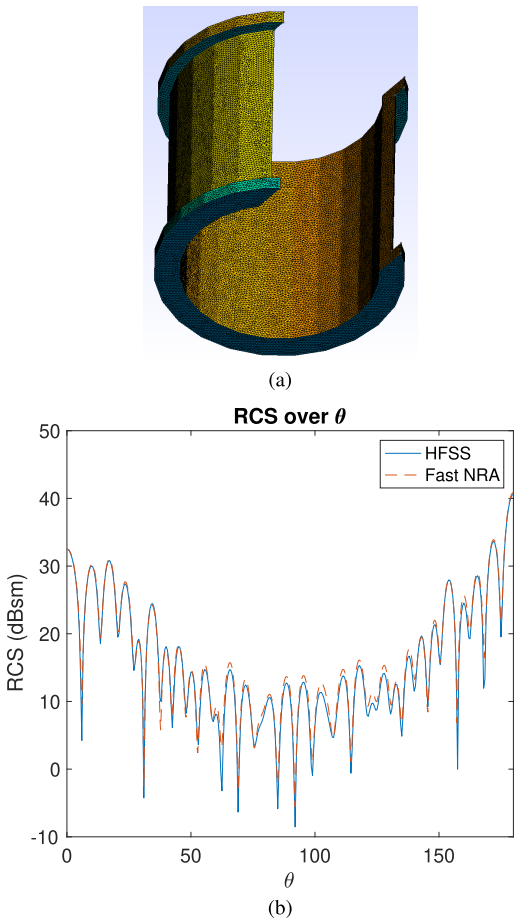
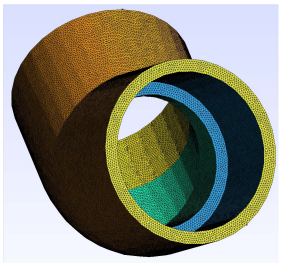


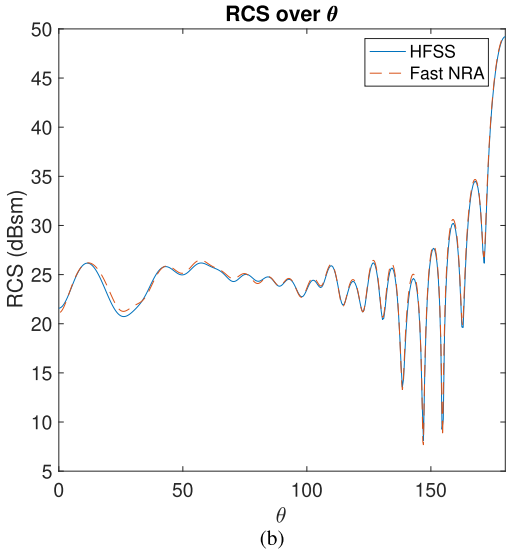
Fig. 6. (a) Geometry and mesh of a coil. (b) Bistatic RCS.

TABLE II
RANK VERSUS TREE LEVEL USING NRA WITH $\epsilon = 10^{-3}$

tree level	e_s	n	k_F	k	k_ϕ
3	20.61	49152.00	42050	967	954
4	15.74	24576.00	25538	721	552
5	10.26	12288.00	11552	509	323
6	8.29	6144.00	7938	324	204
7	5.26	3072.00	3698	207	125
8	4.48	1536.00	2738	134	86
9	2.79	768.00	1250	90	57
10	2.46	384.00	1058	63	43
11	1.50	192.00	512	44	31
12	1.39	96.00	450	32	25
13	0.88	48.00	242	25	18
14	0.84	24.00	242	18	15



(a)



(b)

Fig. 7. (a) Geometry and mesh of a joint. (b) Bistatic RCS.

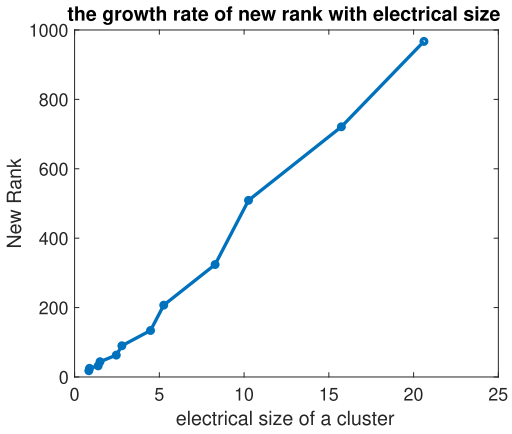


Fig. 8. New rank's growth rate with electrical size in a sphere example.

Fig. 8. From this figure, it can also be seen clearly that the rank scales linearly with electrical size, which agrees with the one used in our complexity analysis.

2) Complexity Analysis: A suite of conducting spheres of various diameters at 300 MHz is then simulated to examine the time and memory complexity of the proposed fast NRA algorithm. First, as a sanity check of the accuracy, we simulate one case, which is a sphere of 17.68 m diameter, whose number of unknowns is 294 912. The accuracy criterion is chosen to be $\epsilon = 10^{-3}$ in the fast NRA algorithm. We then use the direct solver in [22] to solve the $\mathcal{H}^2$ matrix generated from the proposed algorithm. In Fig. 9, the simulated bistatic RCS is plotted as a function of θ, which reveals an excellent agreement with MIE series solution [23].

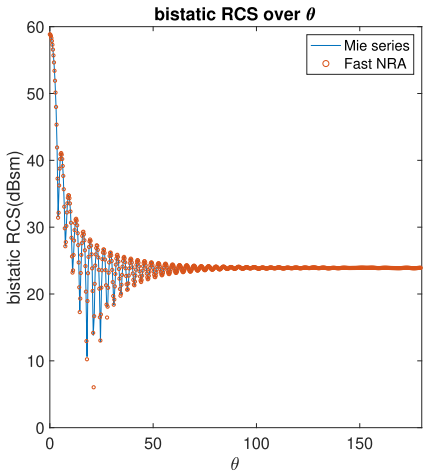
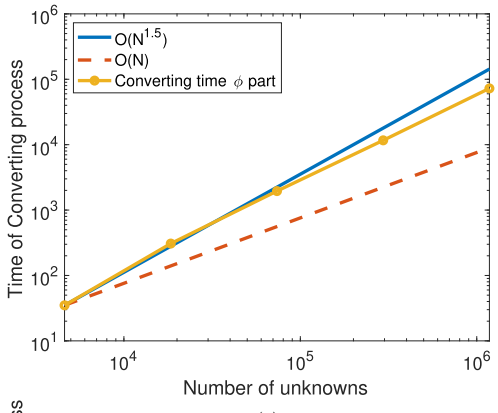
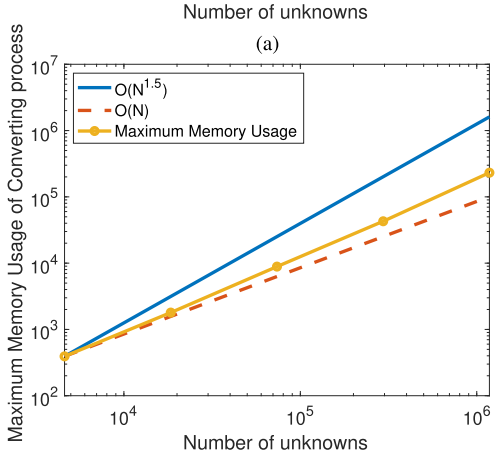


Fig. 9. Simulated RCS of a conducting sphere using the new $\mathcal{H}^2$-matrix generated from Fast NRA in comparison with Mie series solution.



(a)



(b)

Fig. 10. Complexity of the proposed fast NRA. (a) Time. (b) Memory.

We then vary the size of the sphere and examine the time and memory scaling of the proposed fast NRA algorithm. The scaling data are listed in Table III. In this table, N is the number of unknowns, D is the diameter of the conducting sphere, e_{MV} denotes the relative error between the result of an FMM-based matrix multiplied by a vector and the new $\mathcal{H}^2$-matrix multiplied by the same vector, $t_{C,\phi}$ is the time for converting the ϕ part of $\mathbf{Z}$ matrix, t_C is the time for converting the whole $\mathbf{Z}$ matrix, and $t_{FMM}(s)$ is the assembly time of FMM. $k_{F,\phi}$ is the rank of the FMM $\mathcal{H}^2$-matrix's ϕ part, k_ϕ is the rank of the new $\mathcal{H}^2$-matrix's ϕ part, $t_{F,MV}$

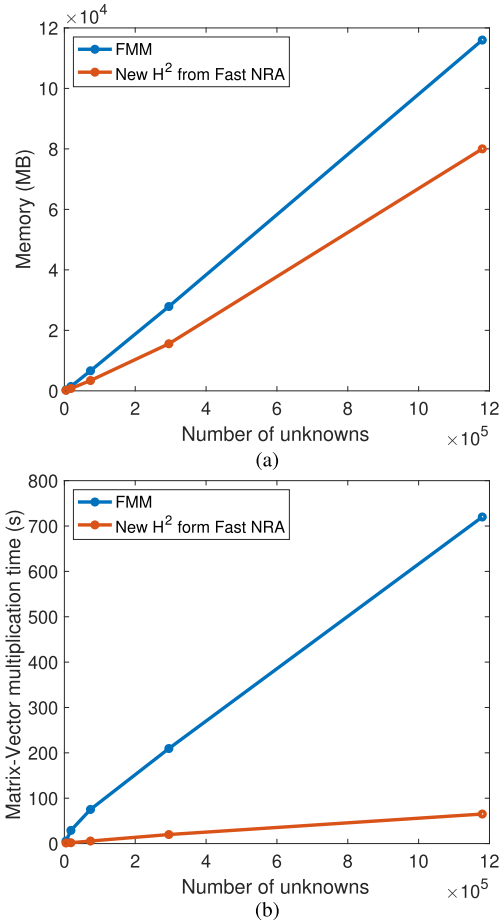


Fig. 11. Comparison between the FMM and the new $\mathcal{H}^2$-matrix obtained from the proposed fast NRA. (a) Memory. (b) Time.

TABLE III

TIME, RANK, AND MEMORY SCALING OF THE PROPOSED FAST NRA WITH $\epsilon = 10^{-2}$ AND COMPARISON WITH THE FMM-BASED REPRESENTATION

N	4608	18432	73728	294912	1179648
D	2.21λ	4.42λ	8.84λ	17.68λ	35.38λ
e_{MV}	6.92e-3	1.18e-2	1.83e-2	2.41e-2	4.51e-2
$t_C(s)$	114.75	1180.56	7570.54	46893.61	295287.38
$t_{C,\phi}(s)$	30.29	287.69	1874.19	10269.44	70441.52
$t_{FMM}(s)$	72.96	283.85	1188.37	4798.54	18731.23
$k_{F,\phi}$	288	1152	3698	11858	42050
k_ϕ	21	50	115	302	898
$t_{F,MV}(s)$	5.53	29.00	75.26	209.48	719.94
$t_{MV}(s)$	1.27	1.89	5.61	19.90	65.12
$m_F(Mb)$	278.36	1429.50	6633.92	27883.57	115991.14
$m_{gF}(Mb)$	318.25	1622.83	7481.95	31392.16	130411.31
$m(Mb)$	176.05	776.78	3416.73	15623.0	80965.49
$m_C(Mb)$	312.78	1793.66	8844.36	42692.75	231295.43

is the time of the FMM $\mathcal{H}^2$-matrix multiplied by a vector, t_{MV} is the time of the new $\mathcal{H}^2$-matrix multiplied by the same vector, m_F is the memory to store the FMM-based $\mathcal{H}^2$-matrix, m_{gF} is the memory used to assemble the FMM-based matrix, m is the memory to store the new $\mathcal{H}^2$-matrix, and m_C is the maximal memory used in the converting process. It can be seen clearly that the new $\mathcal{H}^2$-matrix generated from the proposed work has a much reduced rank, memory, and matrix-vector multiplication time compared with the FMM-based representation. We also plot the time and memory usage in log scale in Fig. 10 for the minimal-rank $\mathcal{H}^2$-generation time. They are shown to agree very well with our theoretical complexity analysis.

In addition, we plot the memory required to store an FMM matrix and the new $\mathcal{H}^2$-matrix generated by Fast NRA in Fig. 11(a) and the CPU time of one matrix-vector multiplication of the two methods in Fig. 11(b). It is obvious that due to its minimal-rank representation, the new $\mathcal{H}^2$-matrix generated by the proposed algorithms has a much reduced storage and CPU run time while scaling with N in the same way.

VIII. CONCLUSION

We present new algorithms to generate a rank-minimized $\mathcal{H}^2$-matrix to represent electrically large surface IE operators. First, the FMM is leveraged to obtain an initial $\mathcal{H}^2$-matrix in low complexity. Fast NRAs are then developed to convert the FMM-based $\mathcal{H}^2$-representation into a new $\mathcal{H}^2$-matrix whose rank is minimized based on accuracy. The resultant new $\mathcal{H}^2$-matrix is found to have a much reduced rank without sacrificing prescribed accuracy, which accelerates both iterative and direct solutions. The proposed work has been applied to solve electrically large SIE equations for scattering analysis. Its accuracy and efficiency are demonstrated by numerical experiments. In addition to surface IEs, it is also applicable to volume IEs and other IE operators.

REFERENCES

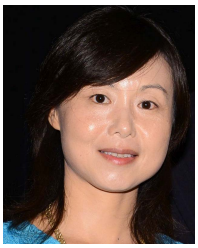
[1] S. Börm, L. Grasedyck, and W. Hackbusch, "Hierarchical matrices," in *Lecture Notes*, vol. 21. 2003.
[2] W. Chai and D. Jiao, "An H^2-matrix-based integral-equation solver of reduced complexity and controlled accuracy for solving electrodynamic problems," *IEEE Trans. Antennas Propag.*, vol. 57, no. 10, pp. 3147–3159, Oct. 2009.
[3] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Math.*, vol. 86, no. 4, pp. 565–589, Oct. 2000.
[4] K. Zhao, M. N. Vouvakis, and J.-F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Trans. Electromagn. Compat.*, vol. 47, no. 4, pp. 763–773, Nov. 2005.
[5] W. Chai and D. Jiao, "A complexity-reduced H-matrix based direct integral equation solver with prescribed accuracy for large-scale electrodynamic analysis," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, Jul. 2010, pp. 1–4.
[6] S. Omar, M. Ma, and D. Jiao, "Low-complexity direct and iterative volume integral equation solvers with a minimal-rank $\mathcal{H}^2$-representation for large-scale three-dimensional electrodynamic analysis," *IEEE J. Multiscale Multiphys. Comput. Techn.*, vol. 2, pp. 210–223, 2017.
[7] M. Bebendorf and R. Venn, "Constructing nested bases approximations from the entries of non-local operators," *Numerische Math.*, vol. 121, no. 4, pp. 609–635, Aug. 2012.
[8] M. Li, M. A. Francavilla, F. Vipiana, G. Vecchi, and R. Chen, "Nested equivalent source approximation for the modeling of multiscale structures," *IEEE Trans. Antennas Propag.*, vol. 62, no. 7, pp. 3664–3678, Jul. 2014.
[9] M. A. E. Bautista, M. A. Francavilla, P. G. Martinsson, and F. Vipiana, "$O(N)$ nested skeletonization scheme for the analysis of multiscale structures using the method of moments," *IEEE J. Multiscale Multiphys. Comput. Techn.*, vol. 1, pp. 139–150, 2016.
[10] Y. Zhao, D. Jiao, and J. Mao, "Fast nested cross approximation algorithm for solving large-scale electromagnetic problems," *IEEE Trans. Microw. Theory Techn.*, vol. 67, no. 8, pp. 3271–3283, Aug. 2019.
[11] J. M. Song and W. C. Chew, "Multilevel fast-multipole algorithm for solving combined field integral equations of electromagnetic scattering," *Microw. Opt. Technol. Lett.*, vol. 10, no. 1, pp. 14–19, Sep. 1995.

[12] R. Coifman, V. Rokhlin, and S. Wandzura, "The fast multipole method for the wave equation: A pedestrian prescription," *IEEE Antennas Propag. Mag.*, vol. 35, no. 3, pp. 7–12, Jun. 1993.
[13] E. Darve, "The fast multipole method: Numerical implementation," *J. Comput. Phys.*, vol. 160, no. 1, pp. 195–240, May 2000.
[14] W. C. Chew, E. Michielssen, J. Song, and J.-M. Jin, *Fast and Efficient Algorithms in Computational Electromagnetics*. Norwood, MA, USA: Artech House, 2001.
[15] Z. Jiang, Y. Xu, R.-S. Chen, Z. Fan, and D. Ding, "Efficient matrix filling of multilevel simply sparse method via multilevel fast multipole algorithm," *Radio Sci.*, vol. 46, no. 5, pp. 1–7, Oct. 2011.
[16] D. Jiao and S. Omar, "Minimal-rank $\mathcal{H}^2$-matrix-based iterative and direct volume integral equation solvers for large-scale scattering analysis," in *Proc. IEEE Int. Symp. Antennas Propag. USNC/URSI Nat. Radio Sci. Meeting*, Jul. 2015, pp. 740–741.
[17] W. Chai and D. Jiao, "Linear-complexity direct and iterative integral equation solvers accelerated by a new rank-minimized $\mathcal{H}^2$-representation for large-scale 3-D interconnect extraction," *IEEE Trans. Microw. Theory Techn.*, vol. 61, no. 8, pp. 2792–2805, Aug. 2013.
[18] C. Yang and D. Jiao, "Method for generating a minimal-rank $\mathcal{H}^2$-matrix from FMM for electrically large analysis," in *Proc. IEEE Int. Symp. Antennas Propag. USNC/URSI Nat. Radio Sci. Meeting*, Jul. 2018, pp. 2503–2504.
[19] C. Yang, M. Ma, and D. Jiao, "Fast algorithms for converting an FMM-based representation of electrically large integral operators to a minimal-rank $\mathcal{H}^2$-matrix," in *Proc. IEEE Int. Symp. Antennas Propag. USNC-URSI Radio Sci. Meeting*, Jul. 2019, pp. 1441–1442.
[20] S. Rao, D. Wilton, and A. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Trans. Antennas Propag.*, vol. 30, no. 3, pp. 409–418, May 1982.
[21] W. Chai and D. Jiao, "Theoretical study on the rank of integral operators for broadband electromagnetic modeling from static to electrodynamic frequencies," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 3, no. 12, pp. 2113–2126, Dec. 2013.
[22] M. Ma and D. Jiao, "Accuracy directly controlled fast direct solution of general $\mathcal{H}^2$-matrices and its application to solving electrodynamic volume integral equations," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 1, pp. 35–48, Jan. 2017.
[23] C. Balanis, *Advanced Engineering Electromagnetics* (CourseSmart Series). Hoboken, NJ, USA: Wiley, 2012. [Online]. Available: https://books.google.com/books?id=cRkTuQAACAAJ

Chang Yang (Graduate Student Member, IEEE) received the B.S. degree in electronic science and technology from Xi'an Jiaotong University, Xi'an, China, in 2016. He is currently pursuing the Ph.D. degree with Purdue University, West Lafayette, IN, USA.

He is also a member of the On-Chip Electromagnetics Group, School of Electrical and Computer Engineering, Purdue University. His current research interests include computational electromagnetics, fast and high-capacity numerical methods, and scattering analysis.

Mr. Yang was a recipient of the Honorable Mention Award in the 2020 IEEE Antennas and Propagation Symposium (AP-S) Student Paper Competition.



Dan Jiao (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 2001.

She then worked at the Technology Computer-Aided Design (CAD) Division, Intel Corporation, Santa Clara, CA, USA, until September 2005, as a Senior CAD Engineer, a Staff Engineer, and a Senior Staff Engineer. In September 2005, she joined Purdue University, West Lafayette, IN, USA, as an Assistant Professor with the School of Electrical and Computer Engineering, where she is currently a Professor. She has authored three book chapters and over 300 papers in refereed journals and international conferences. Her current research interests include computational electromagnetics, high-frequency digital, analog, mixed-signal, and RF integrated circuit (IC) design and analysis, high-performance VLSI CAD, modeling of microscale and nanoscale circuits, applied electromagnetics, fast and high-capacity numerical methods, fast time-domain analysis, scattering and antenna analysis, RF, microwave, and millimeter-wave circuits, wireless communication, and bioelectromagnetics.

Dr. Jiao was a recipient of the 2010 Ruth and Joel Spira Outstanding Teaching Award, the 2008 National Science Foundation (NSF) CAREER Award, the 2006 Jack and Cathie Kozik Faculty Startup Award (which recognizes an outstanding new faculty member of the School of Electrical and Computer Engineering, Purdue University), the 2006 Office of Naval Research (ONR) Award under the Young Investigator Program, the 2004 Best Paper Award presented at the Intel Corporation's annual corporate-wide technology conference (Design and Test Technology Conference) for her work on generic broadband model of high-speed circuits, the 2003 Intel Corporation's Logic Technology Development (LTD) Divisional Achievement Award, the Intel Corporation's Technology CAD Divisional Achievement Award, the 2002 Intel Corporation's Components Research the Intel Hero Award (Intel-wide she was the tenth recipient), the Intel Corporation's LTD Team Quality Award, and the 2000 Raj Mittra Outstanding Research Award presented by the University of Illinois at Urbana–Champaign. She was a recipient of Intel's 2019 Outstanding Researcher Award. She received the 2013 S. A. Schelkunoff Prize Paper Award of the IEEE Antennas and Propagation Society, which recognizes the Best Paper published in the IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION during the previous year. She was among the 21 women faculty selected across the country as the 2014–2015 Fellow of Executive Leadership in Academic Technology and Engineering (ELATE) at Drexel, a national leadership program for women in the academic STEM fields. She has been named a University Faculty Scholar by Purdue University since 2013. She was among the 85 engineers selected throughout the nation for the National Academy of Engineering's 2011 U.S. Frontiers of Engineering Symposium. She has served as the reviewer for many IEEE journals and conferences. She is also an Associate Editor of the IEEE TRANSACTIONS ON COMPONENTS, PACKAGING, AND MANUFACTURING TECHNOLOGY and the IEEE JOURNAL ON MULTISCALE AND MULTIPHYSICS COMPUTATIONAL TECHNIQUES. She served as the General Chair for the 2019 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO), Boston, MA, USA. She was selected as an IEEE MTT-Society Distinguished Microwave Lecturer in 2020.