2-1-2010

H-Matrix-Based Fast Direct Finite Element Solver for Large-Scale Electromagnetic Analysis

Haixin Liu
Purdue University - Main Campus, haixin@purdue.edu

Dan Jiao
Purdue University - Main Campus, djiao@purdue.edu

$\mathcal{H}$-Matrix-Based Fast Direct Finite Element Solver for Large-Scale Electromagnetic Analysis

Haixin Liu and Dan Jiao

School of Electrical and Computer Engineering
465 Northwestern Ave.
Purdue University
West Lafayette, IN 47907-2035

– This work was supported by NSF under award No. 0747578 and No. 0702567.

Abstract

In this work, we prove that the sparse matrix resulting from a finite-element-based analysis of electrodynamic problems can be represented by an $\mathcal{H}$-matrix without any approximation, and the inverse of this sparse matrix has a data-sparse $\mathcal{H}$-matrix approximation with error well controlled. Based on this proof, we develop an $\mathcal{H}$-matrix-based direct finite-element solver of $O(kN\log N)$ memory complexity and $O(k^2N\log^2 N)$ time complexity for solving electromagnetic problems, where k is a small variable that is adaptively determined based on accuracy requirements, and N is the number of unknowns. Both inverse-based and LU-based direct solutions are developed. The LU-based solution is further accelerated by nested dissection. Both theoretical analysis and numerical experiments have demonstrated the accuracy and almost linear complexity of the proposed solver in large-scale electrostatic and electrodynamic applications involving over 1 million unknowns. A comparison with the state-of-the-art direct finite element solver that employs the most advanced sparse matrix solution has shown clear advantages of the proposed solver. In addition, the proposed solver is applicable to arbitrarily-shaped three-dimensional structures and arbitrary inhomogeneity.

$\mathcal{H}$-Matrix-Based Fast Direct Finite Element Solver for Large-Scale Electromagnetic Analysis

Haixin Liu and Dan Jiao

School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906, USA

haixin@purdue.edu, djiao@purdue.edu

Abstract

In this work, we prove that the sparse matrix resulting from a finite-element-based analysis of electrodynamic problems can be represented by an $\mathcal{H}$-matrix without any approximation, and the inverse of this sparse matrix has a data-sparse $\mathcal{H}$-matrix approximation with error well controlled. Based on this proof, we develop an $\mathcal{H}$-matrix-based direct finite-element solver of $O(kN\log N)$ memory complexity and $O(k^2N\log^2 N)$ time complexity for solving electromagnetic problems, where k is a small variable that is adaptively determined based on accuracy requirements, and N is the number of unknowns. Both inverse-based and LU-based direct solutions are developed. The LU-based solution is further accelerated by nested dissection. Both theoretical analysis and numerical experiments have demonstrated the accuracy and almost linear complexity of the proposed solver in large-scale electrostatic and electrodynamic applications involving over 1 million unknowns. A comparison with the state-of-the-art direct finite element solver that employs the most advanced sparse matrix solution has shown clear advantages of the proposed solver. In addition, the proposed solver is applicable to arbitrarily-shaped three-dimensional structures and arbitrary inhomogeneity.

Keywords: Finite Element Methods; Electromagnetic Analysis; Fast Solvers; Direct Solution; $\mathcal{H}$-Matrix; Nested Dissection**.**

1 Introduction

Compared to other computational electromagnetic methods such as finite-difference-based methods and integral-equation-based methods, finite-element methods (FEM) have demonstrated an increased capability in handling both irregular geometries and arbitrary inhomogeneity. A finite-element-based analysis of a complex electromagnetic problem generally results in a large-scale system matrix. Although the matrix is

sparse, solving it can be a computational challenge when the problem size is large. A direct solution can be computationally intensive. As yet, no linear complexity has been reported for FEM-based direct solutions for general electromagnetic problems. In [1], the optimal operation count of a direct solution of an FEM matrix was shown to be $O(N^{1.5})$, where N is matrix dimension. Recent exploration of fast direct solutions for FEM-based electromagnetic analysis can be seen in [2, 3], where two-dimensional problems were studied. State-of-the-art finite-element-based solvers rely on iterative approaches to solve large-scale matrices. The resultant computational complexity is $O(N_{it}N_{rhs}N)$, where N_{it} is the number of iterations, and N_{rhs} is the number of right hand sides. When N_{it} and N_{rhs} are large, iterative solutions become inefficient. In addition, the complexity is problem dependent since the iteration number N_{it} is, in general, problem dependent.

In this work, we consider the fast direct solution of FEM based matrices for solving electromagnetic problems. Our solution is built upon the observation that although the inverse of an FEM-based matrix generally leads to a dense matrix, this matrix can be thought of as "data-sparse," i.e., it can be specified by a few parameters. There exists a general mathematical framework called the "Hierarchical ($\mathcal{H}$) Matrix" framework [4-7], which enables a highly compact representation and efficient numerical computation of the dense matrices. To be specific, if matrix $\mathbf{C}$ is an $m \times n$ off-diagonal block in an $\mathcal{H}$ matrix which describes interactions on upper levels in the hierarchy, it can be written as $\mathbf{C} = \mathbf{A}\mathbf{B}^{\mathrm{T}}$ where $\mathbf{A}$ is of dimension $m \times r$, $\mathbf{B}$ is of dimension $n \times r$, and r denotes the rank of $\mathbf{C}$ with $r < m$ and $r < n$. Storage requirements and matrix-vector multiplications using $\mathcal{H}$-matrices have been shown to be of complexity $O(N\log N)$. Moreover, the inverse of an $\mathcal{H}$-matrix can be obtained in $O(N\log^2 N)$ complexity. In [17, 14], such an $\mathcal{H}$-matrix based form of the system matrix was used in the integral equation based methods to solve large-scale electrodynamic problems involving over 1 million unknowns. In [14-15], the error bound of the $\mathcal{H}$- and $\mathcal{H}^2$-matrix-based representation of an electrodynamic problem was derived for integral equation based analysis. It was shown that exponential convergence of the error with respect to the number of interpolation points can be achieved irrespective of the electric size. In addition, different from static cases in which a constant rank can maintain the same order of accuracy regardless of problem size, the rank required by an electrodynamic system for a given accuracy is a variable with respect to tree level, electric size, admissible block, and admissibility condition. It is worth mentioning that the matrices underlying generic Fast Multiple Algorithms [18-21] are $\mathcal{H}^2$-matrices, as noted in [22].

In the mathematical literature, the existence of an $\mathcal{H}$-matrix approximation was proved for elliptic partial differential equations (PDE) [8]. The use of $\mathcal{H}$-matrix-based techniques in an FEM-based framework has been mainly for solving elliptic PDEs such as a Poisson equation. No work has been reported for the finite-element-based analysis of vector wave equations. The research challenges are three-fold. First, one has to

prove that there exists an $\mathcal{H}$-matrix-based representation of the inverse of the FEM-based matrix for electrodynamic problems so that the accuracy of the $\mathcal{H}$-based approach can be controlled; Second, one has to develop a direct solver that is faster than state-of-the-art direct sparse solvers so that it is worthwhile to explore an $\mathcal{H}$-based fast solution. Third, one has to back up the accuracy and complexity of this fast solver by a theoretical analysis in addition to numerical experiments since the conclusions drawn from numerical experiments are often problem dependent. In [9-11], we have published preliminary results on a fast $\mathcal{H}$-inverse-based direct solver for the FEM-based analysis of electromagnetic problems.

The main contributions of this paper are as follows. *First*, we theoretically proved the existence of an $\mathcal{H}$-matrix-based representation of the FEM matrix and its inverse for electrodynamic problems. We realize the fact that it is difficult to develop such a proof solely from a mathematical point of view. However, by combining an appreciation of the electromagnetic physics with elegant results from mathematics, such a proof becomes obvious. *Second*, we developed an $\mathcal{H}$-matrix-based direct FEM solver of $O(kN\log N)$ memory complexity and $O(k^2N\log^2 N)$ time complexity for solving vector wave equations, where k is a variable that is adaptively determined based on an accuracy requirement, which is small compared to N. In proposed direct solver, we developed an inverse-based direct solution as well as an LU-decomposition-based direct solution with accuracy well controlled. In addition, we incorporated nested dissection [1] to further expedite the $\mathcal{H}$-LU-based solution of vector wave equations. *Third*, we performed a theoretical analysis of the computational complexity of the proposed fast direct solver. In addition, we analyzed the accuracy of the proposed direct solver and showed that it is error controllable. *Last but not the least*, we compared the proposed direct solver with the state-of-the-art direct FEM solver that employs the most advanced sparse matrix solution such as UMFPACK 5.0 [12]. UMFPACK has been adopted by Matlab for fast sparse matrix solutions. It has incorporated almost all the advanced sparse matrix techniques such as the multifrontal method and the approximate minimum degree (AMD) ordering for solving large-scale sparse matrices. The proposed solver is shown to outperform the UMFPACK 5.0 in both matrix decomposition and matrix solution time without sacrificing accuracy.

The remainder of this paper is organized as follows. In Section 2, the vector FEM-based analysis of general electromagnetic problems is outlined. In Section 3, the existence of the $\mathcal{H}$- matrix representation of the FEM matrix and its inverse is proved for electrodynamic problems. In Section 4, the detailed numerical procedure of the proposed direct solver is given. In Section 5, the complexity and accuracy of the proposed solver are analyzed. In Section 6, the choice of simulation parameters is discussed. In Section 7, numerical results are shown to demonstrate the accuracy and almost linear complexity of the proposed direct FEM solver. Section 8 relates to our conclusions.

2 Vector FEM-based Analysis of General Electromagnetic Problems

Consider the second-order vector wave equation

$$\nabla\times\left(\frac{1}{\mu_r}\nabla\times\mathbf{E}\right)-k_0^2\varepsilon_r\mathbf{E}=-jk_0Z_0\mathbf{J} \tag{1}$$

subject to the following boundary conditions:

$$\hat{n}\times\mathbf{E}=\mathbf{P}\quad \text{on } S_1 \tag{2}$$

$$\frac{1}{\mu_r}\hat{n}\times\left(\nabla\times\mathbf{E}\right)+\gamma_e\hat{n}\times\left(\hat{n}\times\mathbf{E}\right)=\mathbf{U}\ \text{ on } S_2\quad , \tag{3}$$

where (3) can be used to truncate the computational domain for an FEM-based analysis, in which γ_e and **U** can be frequency and position dependent.

An FEM-based solution to the above boundary value problem results in a linear system of equations [13]

$$\mathbf{Y}\{E\}=\{I\}\,, \tag{4}$$

where **Y** can be written as

$$\mathbf{Y}=-k_0^2\mathbf{T}+\mathbf{S}+\mathbf{B}\,, \tag{5}$$

in which

$$\begin{aligned}\mathbf{T}&=\iiint_V\left[\varepsilon_r\mathbf{N}_i\cdot\mathbf{N}_j\right]dV\\ \mathbf{S}&=\iiint_V\left[\frac{1}{\mu_r}\left(\nabla\times\mathbf{N}_i\right)\cdot\left(\nabla\times\mathbf{N}_j\right)\right]dV\cdot\\ \mathbf{B}&=\iint_{s_2}\left[\gamma_e\left(\hat{n}\times\mathbf{N}_i\right)\cdot\left(\hat{n}\times\mathbf{N}_j\right)\right]dS\end{aligned} \tag{6}$$

where V denotes the computational domain, and **N** is the vector basis used to expand unknown **E**. In (6), **T** is known to be a mass matrix, and **S** is known to be a stiffness matrix. **T** is positive definite, **S** is semi-positive definite, and the combined system **Y** is, in general, indefinite.

When the problem size is large, solving $\mathbf{Y}$ is a computational challenge even $\mathbf{Y}$ is sparse. In the following section, we show that $\mathbf{Y}$ and its inverse $\mathbf{Y}^{-1}$ both can be represented by an $\mathcal{H}$-matrix, from which a significant reduction in computational complexity can be achieved.

3 On the Existence of $\mathcal{H}$-Matrix Representation of the FEM Matrix and Its Inverse for Electrodynamic Analysis

It has been proven in the mathematical literature that the FEM matrix resulting from the analysis of elliptic partial differential equations such as a Poisson equation has an $\mathcal{H}$- matrix representation. Moreover, its inverse also allows for a data-sparse $\mathcal{H}$-matrix approximation [8]. However, the full Maxwell's equations are hyperbolic partial differential equations in nature. Therefore, the proof developed for elliptic PDE-based equations does not apply to the wave equation, which governs all the electrodynamic phenomena. In the following, we give a rigorous proof on the existence of the $\mathcal{H}$-matrix representation of the FEM matrix and its inverse for electrodynamic problems. We developed this proof by using electromagnetic physics. In our opinion, one cannot solely rely on mathematics to prove the existence of the $\mathcal{H}$- matrix representation for electrodynamic problems. This is possibly one of the reasons why so far such a proof has not been seen from the mathematical literature. It is, in fact, electromagnetic physics that dictates the nature of the system matrix arising from the analysis of electrodynamic problems.

An $\mathcal{H}$-matrix is generally associated with an admissibility condition [6]. To define an admissibility condition, we denote the whole index set containing the indices of the basis functions in the computational domain by $\mathcal{I} = \{1, 2, \ldots, N\}$, where N is the total number of unknowns. Considering two subsets t and s of the $\mathcal{I}$, the admissibility condition is defined as

$$\min\{diam(\Omega_t), diam(\Omega_s)\} \leq \eta\, dist(\Omega_t, \Omega_s) \tag{7}$$

where Ω_t is the minimal subset of the space containing the supports of all basis functions belonging to t, $diam(\,.\,)$ is the Euclidean diameter of a set, $dist(.\,,.)$ is the Euclidean distance between two sets, and η is a positive constant. If subsets t and s satisfy (7), they are admissible; otherwise, they are inadmissible.

Denoting the matrix block formed by t and s by $\mathbf{Y}_{t\times s,}$ if all the blocks $\mathbf{Y}_{t\times s}$ formed by the admissible block cluster (t, s) in $\mathbf{Y}$ can be represented by a low-rank matrix, $\mathbf{Y}$ is an $\mathcal{H}$ matrix. In other words, if $\mathbf{Y}$ possesses the following property

$$\mathbf{Y} \in \mathbb{C}^{\mathcal{I}\times\mathcal{I}}:\ \mathbf{Y}_{t\times s} \text{ is low rank for all admissible } (t, s) \tag{8}$$

it is an $\mathcal{H}$-matrix.

From the above definition of an $\mathcal{H}$-matrix, it is clear that the FEM system matrix **Y** formulated for an electrodynamic problem as shown in (5) is exactly an $\mathcal{H}$-matrix. This is because when the admissibility condition (7) is satisfied, the subsets *t* and *s* are geometrically disconnected, and hence the basis functions in these two sets cannot belong to the same element, and therefore the matrix entries in $\mathbf{Y}_{t\times s,}$ are all zero. Hence, the FEM matrix resulting from the analysis of a general electromagnetic problem always has an exact $\mathcal{H}$-matrix representation without involving any approximation.

Next, we prove the inverse of **Y** also allows for an $\mathcal{H}$-matrix representation. We will first use free space as an example, then generalize the proof to inhomogeneous cases.

Consider the electric field **E** due to an arbitrary current distribution **J** in free space**.** The current distribution **J** can always be decomposed into a group of electric dipoles $\tilde{I}_i l_i$, where $\tilde{I}_i$ is the current of the *i*-th element and l_i is the length of the *i*-th current element. Using the FEM-based method, we solve a system equation (4) to obtain **E,** where the right-hand-side vector $\{I\}$ has the following entries for a normalized **N**

$$I_i = -j\omega\mu_0 \tilde{I}_i l_i \tag{9}$$

On the other hand, **E** due to any current distribution **J** can be evaluated from the following integral:

$$\mathbf{E}(\mathbf{r}) = -j\omega\mu_0 \iiint_V \left(\mathbf{J}(\mathbf{r}')G_0 + \frac{1}{k_0^2}\nabla' \cdot \mathbf{J}(\mathbf{r}')\nabla G_0 \right) dV' \tag{10}$$

where G_0 is free-space Green's function.

For a group of electric dipoles $\tilde{I}_n l_n$ $(n = 1, 2, \cdots N)$, the **E** at any space point $\mathbf{r}$ can be obtained from (10) as

$$\mathbf{E}(\mathbf{r}) = -j\omega\mu_0 \sum_{n=1}^{N} \left[I_n l_n \hat{l}_n(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}') + \frac{1}{k_0^2}\nabla\left[I_n l_n \nabla \cdot (\hat{l}_n(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}')) \right] \right], \tag{11}$$

where $\hat{l}_n$ is the unit vector tangential to the *n*-th current element. The above simply means that **E** is the summation of each dipole's contribution.

By sampling (11) at the center point of each edge in the 3-D finite-element based discretization, and testing (11) by the unit vector tangential to the edge, we obtain

$$\{E\} = \mathbf{Z}\{I\}, \tag{12}$$

where $\{I\}$ is the same as that in (4), the entries of which are given in (9), and **Z** is a dense matrix having the following matrix elements:

$$\mathbf{Z}_{mn} = \frac{1}{-j\omega\mu_0}\left\{-j\omega\mu_0 \hat{t}_m(\mathbf{r}_m)\cdot\hat{l}_n(\mathbf{r}_n')G_0(\mathbf{r}_m,\mathbf{r}_n') - \frac{j}{\omega\varepsilon}\hat{t}_m(\mathbf{r}_m)\cdot\nabla\left[\nabla\cdot(\hat{l}_n(\mathbf{r}')G_0(\mathbf{r},\mathbf{r}'))\right]\right\}, \tag{13}$$

where $\hat{t}_m$ is the unit vector tangential to the *m*-th edge, $\mathbf{r}_m$ denotes the center point of the *m*-th edge, $\mathbf{r}_n'$ denotes the point where the *n*-th current element is located. In (12), $\{E\}$ vector has the following entries

$$E_m = \hat{t}_m(\mathbf{r}_m)\cdot\mathbf{E}(\mathbf{r}_m) \ ,$$

which is the same as the $\{E\}$ vector in (4).

Comparing (12) to (4), it is clear that the inverse of the FEM matrix **Y** is **Z,** the elements of which are given in (13). If we can prove **Z** has an $\mathcal{H}$-matrix representation with error well controlled, $\mathbf{Y}^{-1}$ also has an $\mathcal{H}$-matrix approximation. Such a proof in fact has already been given in [14-15], in which we show that the dense system matrix resulting from the analysis of an electrodynamic problem can be represented by an $\mathcal{H}$-matrix or an $\mathcal{H}^2$-matrix (a special class of $\mathcal{H}$-matrix) with error bounded irrespective of the electric size. Different from static cases in which a constant rank can maintain the same order of accuracy regardless of problem size, the rank required by an electrodynamic system for a given accuracy is a variable with respect to electric size, tree level, admissible block, and admissibility condition.

In an inhomogeneous problem, the **E** field due to a group of electric dipoles $\{\tilde{I}_i l_i\}$ can be written as

$$\mathbf{E} = \mathbf{E}^{inc} + \mathbf{E}^{sca} \ , \tag{14}$$

where $\{E^{inc}\} = \mathbf{Z}\{I\}$. Thus, (14) can be written as

$$\mathbf{Z}_1\{\mathbf{E}\} = -\mathbf{Z}\{I\} \tag{15}$$

where $\mathbf{Z}_1$ is a matrix. Comparing (15) to (4), it can be seen that

$$\mathbf{Y}^{-1} = -\mathbf{Z}_1^{-1}\mathbf{Z} \tag{16}$$

Since **Z** is an $\mathcal{H}$-matrix, even if $\mathbf{Z}_1^{-1}$ is a full matrix, $\mathbf{Y}^{-1}$ is still an $\mathcal{H}$-matrix. This can be readily proved as follows. Since **Z** is an $\mathcal{H}$-matrix, its admissible blocks can be represented by $\mathbf{Z} = \mathbf{A}\mathbf{B}^{\mathrm{T}}$ where **A** is of dimension $m \times k$, **B** is of dimension $m \times k$, where $k < m$. Multiplying a full matrix **C** by $\mathbf{A}\mathbf{B}^{\mathrm{T}}$ still yields an

$\mathcal{H}$-matrix $\mathbf{DB}^T$ with $\mathbf{D} = \mathbf{CA}$ that is of dimension $m \times k$. As a result, the existence of the $\mathcal{H}$-matrix representation for the inhomogeneous cases is also proved.

4 Fast Direct Solution of the FEM System Matrix

Once the existence of the $\mathcal{H}$-matrix representation is proved for $\mathbf{Y}$ and $\mathbf{Y}^{-1}$, the $\mathcal{H}$-matrix arithmetics can be used to significantly accelerate the solution of $\mathbf{Y}$. In our proposed fast direct solver, we first build a block cluster tree to efficiently store the $\mathcal{H}$-matrix-based representation of $\mathbf{Y}$, its inverse, as well as $\mathbf{Y}$'s $\mathbf{LU}$ factors. This tree structure is also used to efficiently capture the hierarchical dependence in the $\mathcal{H}$-matrix. We then perform fast inverse and LU factorization based on the $\mathcal{H}$-based representation of $\mathbf{Y}$. To further expedite the $\mathcal{H}$-based LU factorization, we incorporate nested dissection [1] to reduce the number of nonzero blocks to be computed. In addition, we develop an adaptive truncation scheme to systematically control the accuracy of the $\mathcal{H}$-based operations for accurate analysis of electrodynamic problems.

4.1 Cluster Tree and Block Cluster Tree Construction

We use a block cluster tree to efficiently store the $\mathcal{H}$-matrix-based representation of the FEM system matrix $\mathbf{Y}$, its inverse, as well as $\mathbf{Y}$'s $\mathbf{LU}$ factors. To construct a block cluster tree, a cluster tree needs to be built first. For the index set of the basis functions $\mathcal{I} = \{1, 2, \ldots, N\}$, we construct a cluster tree $T_{\mathcal{I}}$, which is a tree with vertex set V and edge set E as shown by the left (right) part of Fig. 1(a). Each vertex in the tree is called as a cluster. The set of children for a cluster $t \in T_{\mathcal{I}}$ is denoted by children(t). The root of the tree is the index set $\mathcal{I} = \{1, 2, \ldots, N\}$.

To construct a cluster tree, we start from the full index set of basis functions $\mathcal{I}$. We split the computational domain into two subdomains. We continue to split until the number of unknowns in each subdomain is less than or equal to the *leafsize* ($n_{\min}$) which is a parameter to control the tree depth. Clusters with indices no more than *leafsize* are leaves. The set of leaves of $\mathcal{I}$ is denoted by $\mathcal{L}_{\mathcal{I}}$. In Fig. 1(a), the left (right) part is a cluster tree $T_{\mathcal{I}}$ with $N = 8$ and tree depth $p = 3$. The total number of clusters in this tree is 15.

A block cluster tree $T_{\mathcal{I}\times\mathcal{J}}$ is built from two cluster trees $T_{\mathcal{I}}$ and $T_{\mathcal{J}}$, and a given admissibility condition. Each block cluster $b \in T_{\mathcal{I}\times\mathcal{J}}$ has the form $b = (t, s)$ with clusters $t \in T_{\mathcal{I}}$ and $s \in T_{\mathcal{J}}$, and b, t, s being in the same level. In an FEM procedure, the testing function is often chosen the same as the basis function. Therefore, the block cluster tree is constructed from the cluster tree $T_{\mathcal{I}}$ and itself. To build a block cluster

tree $T_{\mathcal{I}\times\mathcal{I}}$, we test blocks level by level starting with the root clusters of $T_{\mathcal{I}}$ and $T_{\mathcal{I}}$, and descending in the tree. Given two clusters $t \in T_{\mathcal{I}}$ and $s \in T_{\mathcal{I}}$, we check whether the admissibility condition is satisfied or not. If the two clusters are admissible, we stop at this level, draw a link between the two clusters as shown in Fig. 1(a), and do not check their children. If they are not admissible, we repeat the procedure for all combinations of the children of t and the children of s. The construction process stops when either at least one of t and s is a leaf or clusters t and s satisfy the admissibility condition. This procedure results in an $\mathcal{H}$-matrix structure as shown in Fig. 1(b). Each matrix block corresponds to a link drawn between $T_{\mathcal{I}}$ and $T_{\mathcal{I}}$ as shown in Fig. 1(a). Links drawn at the upper level of the tree correspond to admissible blocks denoted by $\mathcal{L}^{+}_{\mathcal{I}\times\mathcal{I}}$, while those drawn at the bottommost level represent inadmissible ones denoted by $\mathcal{L}^{-}_{\mathcal{I}\times\mathcal{I}}$. In Fig. 1(b), admissible blocks are represented by shaded blocks.

4.2 Representation of the FEM System Matrix, Its Inverse, and LU Factors by an $\mathcal{H}$ matrix

In an $\mathcal{H}$-matrix, inadmissible blocks are stored in a full matrix form, namely all the matrix entries are stored without any approximation. Admissible blocks $\mathbf{M}_{t\times s}$ are stored in a factorized form: $\mathbf{M}_{t\times s} = \mathbf{A}\mathbf{B}^{\mathrm{T}}$, where $\mathbf{A}$ is a $t\times k$ matrix and $\mathbf{B}$ is an $s\times k$ matrix, with k being the rank of the $\mathcal{H}$-matrix.

When constructing an $\mathcal{H}$-matrix-based representation of the FEM matrix $\mathbf{Y}$, all the non-zero matrix entries in $\mathbf{Y}$ are stored in inadmissible blocks and admissible blocks do not need to be filled because they are all zero. But we still have to form a block cluster tree to identify all the admissible blocks at each tree level because these blocks will be filled by the factorized $\mathbf{A}$ and $\mathbf{B}$ during the process of inverse or LU factorization. Recognizing the difference between an electrodynamic system and a static system, for frequency-dependent problems, the rank in each admissible block is adaptively determined based on a required level of accuracy, the detail of which is given in Section 4.5.

4.3 Fast Direct Inverse

The procedure of an $\mathcal{H}$-based inverse is given in [6]. Here, we outline the basic algorithm to facilitate the complexity and accuracy analysis to be developed in Section 5 for the proposed direct solver.

Rewriting the FEM matrix $\mathbf{Y}$ in the following form

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix} \tag{17}$$

The inverse of $\mathbf{Y}$ can be done recursively by using the following equation:

$$\mathbf{Y}^{-1}=\begin{pmatrix} \mathbf{Y}_{11}^{-1}\oplus\mathbf{Y}_{11}^{-1}\otimes\mathbf{Y}_{12}\otimes\mathbf{S}^{-1}\otimes\mathbf{Y}_{21}\otimes\mathbf{Y}_{11}^{-1} & -\mathbf{Y}_{11}^{-1}\otimes\mathbf{Y}_{12}\otimes\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\otimes\mathbf{Y}_{21}\otimes\mathbf{Y}_{11}^{-1} & \mathbf{S}^{-1} \end{pmatrix} \tag{18}$$

where $\mathbf{S}=\mathbf{Y}_{22}\oplus\left(-\mathbf{Y}_{21}\otimes\mathbf{Y}_{11}^{-1}\otimes\mathbf{Y}_{12}\right)$. All the additions $\oplus$ and multiplications $\otimes$ in (18) are performed by $\mathcal{H}$-based arithmetics defined in [6-7], which is much faster than conventional matrix additions and multiplications. For example, for dense matrices, a formatted addition in $\mathcal{H}$-based arithmetics has $O(NlogN)$ complexity, whereas a formatted multiplication has $O(Nlog^2N)$ complexity.

The pseudo-code for the $\mathcal{H}$-inverse is shown as the following:

Recursive inverse algorithm: (**X** is used for temporary storage)
Procedure H-inverse(**Y**, **X**)
If matrix **Y** is a non-leaf matrix block
. H-inverse ($\mathbf{Y}_{11}$, $\mathbf{X}_{11}$)
$\mathbf{Y}_{21}\otimes\mathbf{X}_{11}\rightarrow\mathbf{X}_{21}$, $\mathbf{X}_{11}\otimes\mathbf{Y}_{12}\rightarrow\mathbf{X}_{12}$, $-\mathbf{X}_{22}\oplus(\mathbf{X}_{21}\otimes\mathbf{Y}_{12})\rightarrow\mathbf{X}_{22}$ (19)
. H-inverse ($\mathbf{X}_{22}$, $(\mathbf{Y}^{-1})_{22}$)
$-(\mathbf{Y}^{-1})_{22}\otimes\mathbf{X}_{21}\rightarrow(\mathbf{Y}^{-1})_{21}$, $-\mathbf{X}_{12}\otimes(\mathbf{Y}^{-1})_{22}\rightarrow(\mathbf{Y}^{-1})_{12}$, $\mathbf{X}_{11}\oplus(-\mathbf{Y}_{12}\otimes\mathbf{X}_{21})\rightarrow(\mathbf{Y}^{-1})_{11}$
else
Inverse (**Y**) (normal full matrix inverse)

4.4 Fast LU Decomposition with Nested Dissection

Since what is to be solved in (4) is $\mathbf{Y}^{-1}\{I\}$ instead of $\mathbf{Y}^{-1}$ and the number of right hand sides is smaller than N in many applications, an LU-factorization-based direct solution is generally more efficient than an inverse-based direct solution. In addition, in an LU factorization process, the input matrix can be overwritten by **L** and **U** factors, thus the memory usage can be cut by half. In contrast, when computing inverse, a temporary $\mathcal{H}$-matrix **X** is needed as shown in (19), which increases memory usage.

The proposed LU-based direct solution has three components: (1) $\mathcal{H}$-based recursive LU factorization; (2) matrix solution by $\mathcal{H}$-based backward and forward substitution; and (3) acceleration by nested dissection. The first two components have been developed in the $\mathcal{H}$-matrix arithmetics [6-7]. We will brief the first two, and focus on the third component.

4.4.1 Recursive LU factorization

We use an $\mathcal{H}$-matrix block $\mathbf{Y}_{tt}$ to demonstrate the $\mathcal{H}$-LU factorization process, where t is a non-leaf cluster in the cluster tree T_I. Since t is a non-leaf, block $t\times t$ is not a leaf block. Hence, $\mathbf{Y}_{tt}$ can be subdivided into four sub blocks:

$$\mathbf{Y}_{tt} = \begin{pmatrix} \mathbf{Y}_{t_1t_1} & \mathbf{Y}_{t_1t_2} \\ \mathbf{Y}_{t_2t_1} & \mathbf{Y}_{t_2t_2} \end{pmatrix} \tag{20}$$

where t_1 and t_2 are the children of t in the cluster tree $T_{\mathcal{I}}$.

Assuming $\mathbf{Y}$ can be factorized into $\mathbf{L}$ and $\mathbf{U}$ matrices, $\mathbf{Y}$ can also be written as:

$$\begin{aligned} \mathbf{Y}_{tt} = \mathbf{L}_{tt}\mathbf{U}_{tt} &= \begin{pmatrix} \mathbf{L}_{t_1t_1} & 0 \\ \mathbf{L}_{t_2t_1} & \mathbf{L}_{t_2t_2} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{t_1t_1} & \mathbf{U}_{t_1t_2} \\ 0 & \mathbf{U}_{t_2t_2} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{L}_{t_1t_1}\mathbf{U}_{t_1t_1} & \mathbf{L}_{t_1t_1}\mathbf{U}_{t_1t_2} \\ \mathbf{L}_{t_2t_1}\mathbf{U}_{t_1t_1} & \mathbf{L}_{t_2t_1}\mathbf{U}_{t_1t_2} + \mathbf{L}_{t_2t_2}\mathbf{U}_{t_2t_2} \end{pmatrix} \end{aligned} \tag{21}$$

By comparing (20) and (21), it can be seen that the LU factorization can be computed recursively as follows:

1) Compute $\mathbf{L}_{t1t1}$ and $\mathbf{U}_{t1t1}$ by $\mathcal{H}$-LU factorization $\mathbf{Y}_{t1t1} = \mathbf{L}_{t1t1}\mathbf{U}_{t1t1}$;

2) Compute $\mathbf{U}_{t1t2}$ by solving $\mathbf{L}_{t1t1}\mathbf{U}_{t1t2} = \mathbf{Y}_{t1t2}$;

3) Compute $\mathbf{L}_{t2t1}$ by solving $\mathbf{L}_{t2t1}\mathbf{U}_{t1t1} = \mathbf{Y}_{t2t1}$; (22)

4) Compute $\mathbf{L}_{t2t2}$ and $\mathbf{U}_{t2t2}$ by $\mathcal{H}$-LU factorization $\mathbf{L}_{t2t2}\mathbf{U}_{t2t2} = \mathbf{Y}_{t2t2} - \mathbf{L}_{t2t1}\mathbf{U}_{t1t2}$.

If $t \times t$ is a leaf block, $\mathbf{Y}_{tt}$ is not subdivided. It is stored in full matrix format, and factorized by a conventional pivoted LU factorization.

In Step 2), a matrix equation $\mathbf{L}_{tt}\mathbf{X}_{ts} = \mathbf{Y}_{ts}$ needs to be solved, where $\mathbf{L}_{tt}$ is a lower triangular matrix. In Step 3), $\mathbf{X}_{ts}\mathbf{U}_{ss} = \mathbf{Y}_{ts}$ needs to be solved, where $\mathbf{U}_{tt}$ is an upper triangular matrix. These two are solved by recursive block forward and backward substitution based on $\mathcal{H}$ arithmetics.

4.4.2 Matrix Solution by Backward and Forward Substitution

After $\mathbf{Y}$ is factorized as $\mathbf{Y} = \mathbf{LU}$, FEM system $\mathbf{Y}\{E\} = \{I\}$ can be solved in two steps: 1) Solve the lower triangular system $\mathbf{L}\{x\} = \{I\}$; 2) Solve the upper triangular system $\mathbf{U}\{E\} = \{x\}$. In the first step, lower triangular system $\mathbf{L}_{tt}\{x_t\} = I_t$ is solved recursively by forward substitution as follows.

If $t \times t$ is not a leaf block, $\mathbf{L}_{tt}$ is subdivided and the lower triangular system can be written as:

$$\begin{pmatrix} \mathbf{L}_{t_1t_1} & 0 \\ \mathbf{L}_{t_2t_1} & \mathbf{L}_{t_2t_2} \end{pmatrix} \begin{pmatrix} x_{t_1} \\ x_{t_2} \end{pmatrix} = \begin{pmatrix} I_{t_1} \\ I_{t_2} \end{pmatrix} \tag{23}$$

where t_1 and t_2 are the children of t in the cluster tree $T_{\mathcal{I}}$. We can write (23) as

$$\begin{pmatrix} \mathbf{L}_{t_1 t_1} x_{t_1} \\ \mathbf{L}_{t_2 t_1} x_{t_1} + \mathbf{L}_{t_2 t_2} x_{t_2} \end{pmatrix} = \begin{pmatrix} I_{t_1} \\ I_{t_2} \end{pmatrix} \tag{24}$$

By comparing both sides of (24), we obtain $\{x\}$ by

1) Solving x_{t1} from $\mathbf{L}_{t1t1} x_{t1} = I_{t1}$;

2) Solving x_{t2} from $\mathbf{L}_{t2t2} x_{t2} = I_{t2} - \mathbf{L}_{t2t1} x_{t1}$.

If $t \times t$ is a leaf block, $\mathbf{L}_{tt}$ is not subdivided and x_t is solved by a conventional forward substitution. Note that different from the construction of $\mathcal{H}$-based $\mathbf{L}$, solving a lower triangular system $\mathbf{L}_{tt}\{x_t\} = I_t$ is exact without introducing any approximation. Solving the upper triangular system can be done in a similar way.

4.4.3 Acceleration by Nested Dissection

Our numerical experiments show that the advantage of the $\mathcal{H}$-based LU over the state-of-the-art sparse factorization such as UMFPACK is not that obvious since the latter incorporates the most advanced ordering technique, which almost minimizes the number of nonzeros to be processed. We hence further accelerate the $\mathcal{H}$-based LU factorization by nested dissection. It is known that the smaller the number of nonzeros to be processed in an LU process, the better the computational efficiency. Nested dissection [1] can be used as an ordering technique to reduce the number of non-zero blocks to be computed in the LU factorization. In addition, this scheme naturally fits the $\mathcal{H}$-based framework compared to many other ordering techniques. It serves an efficient approach to construct a block cluster tree.

We divide the computational domain into three parts: two domain clusters D_1 and D_2 which do not interact with each other and one interface cluster I which interacts with both domain clusters.

Since the domain clusters D_1 and D_2 do not have interaction, their crosstalk entries in the FEM matrix $\mathbf{Y}$ are all zero. If we order the unknowns in D_1 and D_2 first and the unknowns in I last, the resultant matrix will have large zero blocks as shown in the matrix K in Fig. 2. These zero blocks are preserved during the LU factorization as shown in the matrices L and U in Fig. 2, and hence the computation cost of LU factorization is reduced.

We further partition the domain clusters D_1 and D_2 into three parts. This process continues until the number of unknowns in each cluster is smaller than *leafsize* ($n_{\min}$), or no interface edges can be found to divide the domain. Since the matrices in the non-zero blocks are stored and processed by $\mathcal{H}$-matrix techniques in the proposed direct solver, the computational complexity is significantly reduced compared to a conventional nested dissection based LU factorization.

4.5 Adaptive Truncation for Accurate Electrodynamic Analysis

As proved in Section 3, the inverse of FEM matrix **Y** can be represented by an $\mathcal{H}$-matrix. However, which rank to use in the admissible blocks is unknown beforehand. In addition, the choice of rank for electrodynamic problems is more sophisticated compared to static problems. If a constant rank is used across the tree level of a block cluster tree, accuracy may not be guaranteed if the constant rank is too small. If the constant rank is chosen to be very large, the computational efficiency will be sacrificed since for many admissible blocks, the large rank may not be necessary. To address this issue, we developed an adaptive truncation scheme in the proposed direct solver, i.e., the rank for each admissible block is determined adaptively based on a required level of accuracy. The detail is given as follows.

In the original FEM matrix **Y**, all the admissible blocks are zero and hence do not need to be stored. An admissible block becomes non-zero during the inverse/LU process when adding the sum of several matrices to this block or adding the product of two matrices to this block. To give an example, consider $\mathbf{M}_{t\times s} = \mathbf{M}^1_{t\times s} \oplus \mathbf{M}^2_{t\times s}$, where $\mathbf{M}^1_{t\times s} = \mathbf{A}_1\mathbf{B}_1^{\mathrm{T}}$, $\mathbf{M}^2_{t\times s} = \mathbf{A}_2\mathbf{B}_2^{\mathrm{T}}$ and they have the rank *k1*, and *k2* respectively. The direct addition $\mathbf{M'}_{t\times s} = \mathbf{M}^1_{t\times s} + \mathbf{M}^2_{t\times s} = \mathbf{A}_1\mathbf{B}_1^{\mathrm{T}} + \mathbf{A}_2\mathbf{B}_2^{\mathrm{T}} = [\mathbf{A}_1\ \mathbf{A}_2][\ \mathbf{B}_1\ \mathbf{B}_2]^{\mathrm{T}}$ has rank *k1*+*k2*. To determine which rank is necessary, the singular value decomposition of $\mathbf{M'}_{t\times s}$ is first performed:

$$\mathbf{M'}_{t\times s} = \mathbf{U'}\sum\mathbf{'V'}^{\mathrm{T}} \tag{25}$$

where **U**' is a $|t|\times(k1+k2)$ matrix, **V**' is a $|s|\times(k1+k2)$ matrix, and $\sum$' is a $(k1+k2)\times(k1+k2)$ diagonal matrix with diagonal entries: $\sum'_{11} \geq \sum'_{22} \geq \ldots \geq \sum'_{(k1+k2)(k1+k2)} > 0$. We then truncate $\mathbf{M'}_{t\times s}$ as

$$\mathbf{M}_{t\times s} = \mathbf{U}\sum\mathbf{V}^{\mathrm{T}} \tag{26}$$

where $\mathbf{U} = \mathbf{U'}|_{|t|\times k}$, $\mathbf{V} = \mathbf{V'}|_{|s|\times k}$, $\sum = \mathrm{diag}(\sum_{11}, \ldots, \sum_{kk})$, and *k* satisfies

$$\sum_{kk} > \varepsilon \sum'_{11} \text{ and } \sum_{k+1,k+1} \leq \varepsilon \sum'_{11} \tag{27}$$

where ε is the relative truncation error chosen based on the required level of accuracy. The adaptive truncation for adding the product of two matrices to an admissible block can be conducted in a similar fashion.

Unlike the fixed truncation scheme, the rank *k* here is not a constant. It is determined by the truncation accuracy of each admissible block adaptively. In case that the new rank *k* is larger than the original rank, the storage of **A** and **B** matrices need to be expanded to accommodate the larger rank. In addition, the singular value decomposition is performed by using $\mathcal{H}$-based arithmetics, which has a linear complexity of $O(k^2\max(|t|, |s|))$ [6].

5 Complexity and Accuracy Analysis

5.1 Complexity Analysis

The storage complexity of an $\mathcal{H}$-matrix is shown to be $O(kN\log N)$ in the literature [4-6], which equally applies to FEM-based analysis of electrodynamic problems. In the following, we give a detailed complexity analysis for the inverse and LU factorization, which is different from what is reported in the literature for $\mathcal{H}$-based inverse and LU factorization [6]. The latter is based on analogy without accounting for the actual number of operations. In addition, the proposed complexity analysis takes electrodynamic problems into consideration.

Before proceeding to the detail, we introduce an important parameter, sparsity constant C_{sp}, which is used extensively in the complexity analysis. Defining the number of blocks $t \times s \in T_{\mathcal{I}\times\mathcal{J}}$ associated with a given cluster $t \in T_{\mathcal{I}}$ by:

$$c_{sp}^{r}(T_{\mathcal{I}\times\mathcal{J}}, t) := |\{s \subset \mathcal{J}:\ t \times s \in T_{\mathcal{I}\times\mathcal{J}}\}| \tag{28}$$

and that associated with $s \in T_{\mathcal{J}}$ by:

$$c_{sp}^{c}(T_{\mathcal{I}\times\mathcal{J}}, s) := |\{t \subset \mathcal{I}:\ t \times s \in T_{\mathcal{I}\times\mathcal{J}}\}| \tag{29}$$

the sparsity constant C_{sp} of $T_{\mathcal{I}\times\mathcal{J}}$ is defined as:

$$C_{sp}(T_{\mathcal{I}\times\mathcal{J}}) := \max\{\max_{t\in T_{\mathcal{I}}} c_{sp}^{r}(T_{\mathcal{I}\times\mathcal{J}}, t), \max_{s\in T_{\mathcal{J}}} c_{sp}^{c}(T_{\mathcal{I}\times\mathcal{J}}, s)\} \tag{30}$$

Despite a complicated mathematical definition, graphically, C_{sp} is the maximum number of links that can exist in each tree level in Fig. 1(a).

5.1.1 Inverse Complexity

The procedure $\mathcal{H}$-inverse shown in (19) can be divided into two sub procedures to analyze its complexity: 1) $\mathcal{H}$-inverse_M, which only performs $\mathcal{H}$-based multiplications; 2) $\mathcal{H}$-inverse_A, which only performs $\mathcal{H}$-based additions.

In sub-procedure $\mathcal{H}$-inverse_M, each leaf block cluster $r\times t$ is computed twice. Each computation is performed by

$$\mathbf{Y}_{r\times t} = \sum_{l=0}^{p} \sum_{s\in S(r\times t,l)} \mathbf{Y}_{r\times s} \cdot \mathbf{Y}_{s\times t} \tag{31}$$

where $S(r\times t,\ l) = \{\, s \in T_{\mathcal{I}} \,|\, \mathcal{F}^l(r)\times s \in T_{\mathcal{I}\times\mathcal{I}}, s\times\mathcal{F}^l(t) \in T_{\mathcal{I}\times\mathcal{I}}$ and at least one of the two is a leaf$\}$. $\mathcal{F}^l(r)$ represents the parent block clusters of cluster r in level l. If $r\times s$ is a leaf block, it is either an admissible leaf or an inadmissible leaf. If $r\times s$ is admissible, its corresponding matrix block is stored as $\mathbf{M}_{r\times s} = \mathbf{A}\mathbf{B}^T$, where $\mathbf{A}$ is a $|r|\times k$ matrix and $\mathbf{B}$ is a $|s|\times k$ matrix. The product of $\mathbf{M}_{r\times s}$ and block cluster $s\times t$ is an admissible matrix block $\mathbf{M}_{r\times t} = \mathbf{A}\mathbf{C}^T$, where $\mathbf{C}$ is computed by multiplying $\mathbf{B}^{\mathrm{T}}$ by block cluster $s\times t$, which involves k $\mathcal{H}$-matrix-vector multiplications and hence has the complexity of $kC_{sp}k_lO(\max\{|s|, |t|\}log(\max\{|s|, |t|\}))$, where,

$$k_l = \max\{k, n_{\min}\} \tag{32}$$

If $r\times s$ is inadmissible, its matrix size is at most $n_{\min}\times n_{\min}$. So the multiplication with block cluster $s\times t$ involves at most n_{min} $\mathcal{H}$-matrix-vector multiplications and has $n_{min}C_{sp}k_lO(\max\{|s|, |t|\}log(\max\{|s|, |t|\}))$ complexity. If the block cluster tree is balanced, in level l, $\max\{|s|, |t|\}$ can be approximated by $N/2^l$. Therefore, overall, the complexity of multiplying $r\times s$ by $s\times t$ is:

$$Complexity((r\times s)\otimes(s\times t)) \le k_1^2 C_{sp} O(N/2^l \log(N/2^l)) \tag{33}$$

The complexity of $\mathcal{H}$-inverse_M can then be obtained by summing the cost for multiplying $r\times s$ and $s\times t$ in each level:

$$\begin{aligned} Complexity(\text{H-inverse\_M}) &\le 2\sum_{l=0}^{p} \sum_{r\times s\in\mathcal{L}(T,l)} \sum_{s\times t\in T^{(l)}} N_{(r\times s)\otimes(s\times t)} \\ &+ 2\sum_{l=0}^{p} \sum_{s\times t\in\mathcal{L}(T,l)} \sum_{r\times s\in T^{(l)}} N_{(r\times s)\otimes(s\times t)} \\ &\le 4\sum_{l=0}^{p} \sum_{r\times s\in\mathcal{L}(T,l)} \sum_{s\times t\in T^{(l)}} k_1^2 C_{sp} O(N/2^l \log(N/2^l)) \end{aligned} \tag{34}$$

where $\mathcal{L}(T,l)$ denotes the set of leaves in block cluster tree T in level l. Since the number of blocks satisfying $r\times s\in\mathcal{L}(T,l)$ for certain cluster s is smaller than C_{sp}, and there are at most 2^lC_{sp} block clusters in level l, we have

$$\begin{aligned}
&\textit{Complexity}(\text{H-inverse_M}) \le 4\sum_{l=0}^{p}\sum_{s\times t\in T^{(l)}} C_{sp}k_1^{\,2}C_{sp}O(N/2^l\log(N/2^l))\\
&\le 4\sum_{l=0}^{p}2^l C_{sp}k_1^{\,2}C_{sp}^{\,2}O(N/2^l\log(N/2^l))\\
&\le 4k_1^{\,2}C_{sp}^{\,3}O(N\sum_{l=0}^{\log N}(\log N-l))\\
&= 2k_1^{\,2}C_{sp}^{\,3}O(N\log N(\log N+1)) \sim O(k_1^2N\log^2(N))
\end{aligned} \tag{35}$$

As for the complexity of $\mathcal{H}$-inverse_A, since the complexity of formatted addition is $C_{sp}k_1^{\,2}O(N\log N)$ [6], the complexity of $\mathcal{H}$-inverse_A can be obtained by adding the cost of formatted addition level by level as the following:

$$\begin{aligned}
&\textit{Complexity}(\text{H-inverse_A})\\
&\le 2\sum_{l=0}^{p}\sum_{r\times t\in T^{(l)}} C_{sp}k_1^{\,2}O(\max\{|r|,|t|\}\log(\max\{|s|,|t|\}))\\
&\le 2C_{sp}^{\,2}k_1^{\,2}\sum_{l=0}^{p}2^lO(\frac{N}{2^l}\log\frac{N}{2^l}) = 2C_{sp}^{\,2}k_1^{\,2}O(N\sum_{l=0}^{p}(\log N-l))\\
&\le 2C_{sp}^{\,2}k_1^{\,2}O(N\sum_{l=0}^{\log N}(\log N-l))\\
&= C_{sp}^{\,2}k_1^{\,2}O(N\log N(\log N+1)) \sim O(k_1^{\,2}N\log^2 N)
\end{aligned} \tag{36}$$

Therefore, the total complexity of inverse is:

$$\textit{Complexity}(\mathbf{Y}^{-1}) = \textit{Complexity}(\text{H-inverse\_M}) + \textit{Complexity}(\text{H-inverse\_A}) \sim O(k_1^2N\log^2 N) \tag{37}$$

5.1.2 LU Factorization and Solution Complexity

As can be seen from (22), the LU factorization of $\mathbf{Y}_{tt}$ is computed in four steps. In these four steps, $\mathbf{Y}_{t1t1}$, $\mathbf{Y}_{t1t2}$, and $\mathbf{Y}_{t2t1}$ are computed once, $\mathbf{Y}_{t2t2}$ is computed twice. Since in inverse, each block is computed twice, the complexity of $\mathcal{H}$-based LU factorization is bounded by $\mathcal{H}$-based-inverse, which is $O(k^2N\log^2N)$.

After obtaining the $\mathcal{H}$-LU factorization, the FEM system is solved by the algorithm outlined in Section 4.4.2. Since the matrix entries are stored in the leaf block clusters, matrix solving is done in the leaf block clusters similar to $\mathcal{H}$-matrix based matrix-vector multiplication. If the diagonal leaf block $t\times t$ is inadmissible, full matrix forward and backward substitutions are performed to solve $\mathbf{L}_{tt}x_t = b_t$, which

requires $O(|t|^2)$ operations. If the off-diagonal leaf block $t \times s$ is inadmissible, full matrix-vector multiplication is performed, which requires $O(|t||s|)$ operations. If the off-diagonal leaf block $t \times s$ is admissible, the matrix is stored in a factorized form: $\mathbf{M}_{t\times s} = \mathbf{AB}^{\mathrm{T}}$, which requires $kO(|t|+|s|)$ operations. The total complexity of matrix solving is hence

$$\begin{aligned} &Complexity(\text{LU\_Solve}) = \sum_{t\times s\in\mathcal{L}^-} O(|t||s|) + \sum_{t\times s\in\mathcal{L}^+} kO(|t|+|s|) \\ &\le Storage(\text{an } \mathcal{H} \text{ matrix of rank } k) \sim O(kN\log N) \end{aligned} \tag{37}$$

where $\mathcal{L}^-$ denotes all the inadmissible leaves, and $\mathcal{L}^+$ denotes all the admissible leaves.

5.2 Accuracy Analysis

From the proof developed in Section 3, there exists an $\mathcal{H}$-matrix-based representation of the inverse of the FEM matrix **Y**. In such a representation, which block is admissible and which block is inadmissible are determined by an admissibility condition. Rigorously speaking, this admissibility condition should be determined based on $\mathbf{Y}^{-1}$. However, since $\mathbf{Y}^{-1}$ is unknown, we decide it based on **Y**. Apparently, this will induce error. However, as analyzed in Section 3, the **Y**'s inverse can be mapped to the dense matrix formed for an integral operator. For this dense matrix, the admissibility condition used to construct an $\mathcal{H}$-matrix representation has the same form as (7) as shown in [14-16]. Thus, the $\mathcal{H}$-matrix structure, i.e., which block can have a potential low-rank approximation and which block is a full matrix, is formed correctly for $\mathbf{Y}^{-1}$. In addition, the accuracy of the admissibility condition (7) can be controlled by η.

In the inverse and LU factorization process, the rank of each admissible block is adaptively determined based on an accuracy requirement as shown in Section 4.5. If the rank is determined to be a full rank based on the adaptive truncation scheme, then a full rank will be used. Thus, the low-rank approximation for each admissible block is also error controllable through parameter ε used in the adaptive truncation scheme.

Based on the aforementioned two facts, the error of the proposed direct solver is controllable.

6 Choice of Simulation Parameters

There are only three parameters to choose in the proposed direct solver: η in (7), $n_{\min}$ (*leafsize*), and $\boldsymbol{\varepsilon}$ in (27) for adaptively determining the rank. The smaller η is, the better the accuracy. However, the computation will become inefficient if η is too small. For all the electrodynamic simulations conducted in this work, we choose η = 1. The parameter ε can be chosen based on a required level of accuracy. For example, ε can be set to 10^{-4} if 0.01% error is required. As for leafsize $n_{\min}$, if it is chosen to be too large, on one hand, the accuracy becomes better; on the other hand, more full matrix blocks will be formed, and hence computation becomes slow. Therefore, we determine the *leafsize* $n_{\min}$ by balancing CPU time and error. In the simulation conducted in this work, $n_{\min}$ is in the range of (10, 50).

7 Numerical Results

To demonstrate the accuracy and almost linear complexity of the proposed direct FEM solver, we simulated a number of static and electrodynamic examples from small unknowns to over one million unknowns, from small electric sizes to more than sixty wavelengths.

7.1 Shielded Bus Structure

A shielded microstrip line [pp. 115-116, 13] was first simulated to demonstrate the feasibility of the proposed solver in static electromagnetic applications. Node-based triangular basis functions were used. The proposed direct inverse was used to simulate this example. The simulation parameters were chosen as $n_{min} = 10$ and $\eta = 2$. A fixed rank $k = 4$ was used for such a static simulation. To test the large-scale modeling capability of the proposed direct solver, we increased the size of the original problem by adding more lines parallel to the original microstrip line, resulting in 23000 unknowns to 0.8 million unknowns. In Fig. 3(a) and (b), we plot the CPU time and storage of the proposed direct FEM solver with respect to the number of unknowns. The time complexity and storage complexity show an excellent agreement with our theoretical prediction represented by the dashed line, which shows a memory complexity of $O(N\log N)$, and a time complexity of $O(N\log^2 N)$. Meanwhile, good accuracy is achieved in the entire range as can be seen from Fig. 3(c). The relative error in Fig. 3(c) is measured by the inverse error $\left\| I - \mathbf{Y_H}^{-1}\mathbf{Y} \right\|_F / \left\| I \right\|_F$, which is less than 0.5% in the entire range.

7.2 Waveguide Discontinuity

The validity of the proposed solver in solving electrodynamic problems was first demonstrated by a dielectric-loaded waveguide problem shown in Fig. 4(a) (p. 202, [13]). The rectangular waveguide was loaded by a dielectric obstacle with $\varepsilon_r = 6$. The computational domain was discretized by prism elements. The vector prism basis functions [13] were used to expand the unknown **E** in each element. The mesh size was chosen to be 1/25 of the wavelength. The proposed direct inverse was used to simulate this example. The simulation parameters were chosen as $n_{min} = 50$ and $\eta = 1$. The rank k varied from 4 to 6. In Fig. 4(b), we plotted $|S_{11}|$ computed using the proposed direct solver with respect to electric size. An excellent agreement with the reference result [13] computed using a traditional FEM solver is observed.

To test the large-scale modeling capability of the proposed direct inverse, we increased the size of the original problem by increasing the length of the waveguide as well as the loaded dielectric rod. The length was increased from 4.8 b to 256.8 b, resulting in an electric size from ~1.2 wavelengths to ~64 wavelengths. The number of unknowns increased from 5,630 to 0.3 million. In Fig. 5, the CPU time and memory cost are plotted as a function of the number of unknowns. Once again, the time complexity and storage complexity of the proposed solver agree very well with the theoretical prediction which is plotted in dashed line. Moreover, a constant order of accuracy is achieved in the entire range. The relative inverse error $\left\| I - \mathbf{Y_H}^{-1}\mathbf{Y} \right\|_F / \left\| I \right\|_F$ is less than 1.5% in the entire range. Note that in our simulation, to test the

general capability of the proposed solver, we did not take advantage of the fact that the unknowns are increased only along one dimension in this typical example. Otherwise, the complexity can be further reduced to linear [23].

We also used UMFPACK 5.0 [12], a state-of-the-art sparse matrix solver that incorporates most advanced multi-frontal and ordering techniques, to simulate the 0.3 million unknown problem. It takes UMFPACK 4.8s to solve one column of the inverse of the FEM matrix, and the time to compute the entire inverse is approximately 4.8s × 0.3 million ≈ 1.4 million. If we store all the computed columns of the inverse matrix, UMFPACK soon fails due to the shortage of memory. In contrast, the proposed solver only takes 26,000s to compute the entire inverse with relative error no greater than 1.5%, and memory usage no greater than 15 GB.

7.3 Inductor Array

A large-scale package inductor array was simulated to demonstrate the accuracy and efficiency of the proposed $\mathcal{H}$-LU-based direct solver accelerated by nested dissection. The geometry and material data of each inductor is shown in Fig. 6(a), and a 7×7 inductor array is shown in Fig. 6(b). We simulated a series of inductor arrays from a 2×2 array to a 7×7 array, the number of unknowns of which ranged from 117,287 to 1,415,127. The simulation parameters were chosen as $n_{min} = 32$ and $\eta = 1$. The adaptive truncation with $\boldsymbol{\varepsilon}$ = 1e-4 was used to adaptively determine the rank for each admissible block. In Table 1, we gave the rank distribution with respect to tree level observed in the simulation of the 7×7 inductor example that involved more than 1 million unknowns. As can be seen from Table 1, the rank k fluctuates across all the tree levels. In Table 1, the smaller the tree level, the closer it is to the root cluster, which is at level 0. The minimum rank denotes the smallest rank present in the admissible block in a tree level; and the maximum rank denotes the largest rank present in the admissible block in the same level. It can be seen that even in the same tree level, the required rank for each admissible block is different to achieve the same level of accuracy. However, overall, the rank k is a small number compared to the number of unknowns. We also compared the rank distribution between different problem sizes. For example, for a 3×3 inductor array, with the same $\boldsymbol{\varepsilon}$, the maximum rank was 83, which appeared at level 11. The minimum rank was 1.

In Fig. 7(a), we plot LU factorization time cost by the proposed direct solver, and that cost by UMFPACK 5.0 with respect to the number of unknowns. The proposed solver demonstrates a complexity of $O(N\log^2 N)$, which agrees very well with theoretical analysis, whereas UMFPACK has a much higher complexity. In Fig. 7(b), we plot the matrix solution time of the proposed direct solver, and that of UMFPACK for one right hand side. Once again, the proposed direct solver outperforms UMFPACK. In addition, the proposed direct solver is shown to have an $O(N\log N)$ complexity in matrix solution (backward and forward substitution). In Fig. 7(c), we plot the storage requirement of the proposed direct solver and that of UMFPACK in simulating this example. Even though the storage of the proposed solver is shown to be a little bit higher than that of UMFPACK, the complexity of the proposed solver is lower, and hence for larger number of unknowns, the proposed solver will outperform UMFPACK in storage. In

Fig.7 (d), we plot the relative error of the proposed $\mathcal{H}$-LU-based direct solver accelerated by nested dissection. Good accuracy is observed in the entire range.

8 Conclusions

In this work we introduced the $\mathcal{H}$ matrix as a mathematical framework to develop fast solvers for direct FEM-based analysis of electromagnetic problems. We proved the existence of the $\mathcal{H}$-matrix-based representation of the FEM matrix and its inverse for electrodynamic problems, thus laid a theoretical foundation for developing error-controlled $\mathcal{H}$-based solutions for fast FEM-based analysis of electrodynamic problems.

Both inverse- and LU-based direct solutions were developed. Accuracy was controlled by adaptively determining the rank k for each admissible block based on required accuracy. The computation and storage complexity were shown to be $O(k^2N\log^2N)$, and $O(kN\log N)$ respectively by both theoretical analysis and numerical experiments. Since k is a small number that is adaptively determined by the accuracy requirement, we have observed $O(N\log^2N)$ time complexity and $O(N\log N)$ memory complexity with a constant order of accuracy across a wide range of unknowns and electric sizes. The LU-based solution was further accelerated by nested dissection based ordering. A comparison with the state-of-the-art direct FEM solution that employs the most advanced sparse matrix solver such as UMFPACK has shown a clear advantage of the proposed solver. Moreover, existing sparse solvers such as UMFPACK cannot afford to computing a direct inverse because storing each column of the inverse is not feasible for large matrices, whereas the proposed solver can store the dense inverse in $O(N\log N)$ units.

The proposed direct FEM solver of almost linear complexity and controlled accuracy is applicable to general problems involving arbitrarily-shaped geometries and non-uniform materials. It has been successfully applied to both electrostatic and electrodynamic problems involving millions of unknowns. More electrodynamic applications will be explored in the future.

Acknowledgements

This work was supported by NSF under award No. 0747578 and No. 0702567.

Rreferences

[1] A. George, "Nested dissection of a regular finite element mesh," SIAM J. on Numerical Analysis, 10(2):345–363, April 1973.

[2] J.-S. Choi1, T. C. Kramer1, R. J. Adams, F. X. Canning "Factorization of Finite Element Matrices Using Overlapped Localizing LOGOS Modes," 4 pages, IEEE International Symposium on Antennas and Propagation, 2008.

[3] J. Choi, R. J. Adams, and F. X. Canning, "Sparse factorization of finite element matrices using overlapped localizing solution modes," Microwave and Optical Technology Letters, pp. 1050 – 1054, vol. 50, no. 4, 2008.

[4] W. Hackbusch and B.Khoromaskij, "A Sparse Matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-Matrices," Computing, 62:89-108, 1999.

[5] W. Hackbusch and B. N. Khoromskij, "A sparse $\mathcal{H}$-matrix arithmetic. Part II: Application to multi-dimensional problems," Computing, 64: 21-47, 2000.

[6] S. Borm, L. Grasedyck, and W. Hackbusch, "Hierarchical matrices," Lecture note 21 of the Max Planck Institute for Mathematics in the Sciences, 2003.

[7] L. Grasedyck and W. Hackbusch, "Construction and Arithmetics of $\mathcal{H}$-Matrices," Computing, vol. 70, no.4, 295-344, August, 2003.

[8] M. Bebendorf and W. Hackbusch, "Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L∞-coefficients," Numerische Mathematik, 95: 1-28, 2003.

[9] H. Liu, W. Chai, and D. Jiao, "An $\mathcal{H}$-Matrix-Based Fast Direct Solver for Finite-Element-Based Analysis of Electromagnetic Problems," 5 pages, the 2009 International Annual Review of Progress in Applied Computational Electromagnetics (ACES), March, 2009.

[10] H. Liu and D. Jiao, "A Direct Finite-Element-Based Solver of Significantly Reduced Complexity for Solving Large-Scale Electromagnetic Problems," 4 pages, International Microwave Symposium (IMS), June 2009.

[11] H. Liu and D. Jiao, "Performance Analysis of the H-Matrix-Based Fast Direct Solver for Finite-Element-Based Analysis of Electromagnetic Problems," IEEE International Symposium on Antennas and Propagation, 4 pages, June 2009.

[12] UMFPACK5.0, http://www.cise.ufl.edu/research/sparse/umfpack/.

[13] J. M. Jin, The Finite Element Method in Electromagnetics, New York: John Wiley & Sons, 2nd edition (442 p.), 2002.

[14] W. Chai and D. Jiao, "$\mathcal{H}$- and $\mathcal{H}^2$-Matrix-Based Fast Integral-Equation Solvers for Large-Scale Electromagnetic Analysis," accepted for publication, IET Microwaves, Antennas & Propagation, 2009.

[15] W. Chai and D. Jiao, "An $\mathcal{H}^2$-Matrix-Based Integral-Equation Solver of Reduced Complexity and Controlled Accuracy for Solving Electrodynamic Problems," vol. 57, no. 10, pp. 3147-3159, IEEE Trans. Antennas Propagat., Oct. 2009.

[16] W. Chai and D. Jiao, "An $\mathcal{H}^2$-Matrix-Based Integral-Equation Solver of Linear-Complexity for Large-Scale Full-Wave Modeling of 3D Circuits," IEEE 17th conference on electrical performance of electronic packaging (EPEP), pp. 283-286, Oct. 2008.

[17] J. Shaeffer, "Direct Solve of Electrically Large Integral Equations for Problem Sizes to 1 M unknowns," IEEE Trans. Antennas Propag., vol. 56, no. 8, pp. 2306–2313, Aug. 2008.

[18] V. Rokhlin, "Rapid solution of integral equations of classic potential theory," J. Comput. Phys., vol. 60, pp. 187-207, Sep. 1985.

[19] W. Chew, J. Jin, C.Lu, E. Michiielssen, and J. Song, "Fast and Efficient Algorithms in Computational Electromagnetics," edited by W. C. Chew, J. M. Jin, E. Michielssen, and J. M. Song. Norwood, MA: Artech House, 2001.

[20] K. Nabors and J. White, "FastCap: A multipole accelerated 3-D capacitance extraction program," IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 10, pp. 1447-1459, Nov. 1991.

[21] W. Shi, J. Liu, N. Kakani, and T. Yu, "A fast hierarchical algorithm for three-dimensional capacitance extraction," IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 21, no. 3, pp. 330-336, Mar. 2002.

[22] W. Hackbusch, B. Khoromskij, and S. Sauter, "On $\mathcal{H}^2$-matrices," Lecture on Applied Mathematics, H. Bungartz, R. Hoppe, and C. Zenger, eds., pages 9–29, 2000.

[23] H. Liu and D. Jiao, "Layered $\mathcal{H}$-Matrix Based Direct Matrix Inversion of Significantly Reduced Complexity for Finite-Element-Based Large-Scale Electromagnetic Analysis," submitted to the 2010 IEEE International Symposium on Antennas and Propagation, Jan. 2010.

FIGURE CAPTIONS

Fig. 1. (a) A block cluster tree. (b) An $\mathcal{H}$-matrix structure.

Fig. 2. A nested dissection based partition and matrix patterns in LU factors.

Fig. 3. Performance of the proposed direct inverse in simulating a shielded bus structure. (a) CPU time for computing $\mathbf{Y}^{-1}$. (b) Storage of $\mathbf{Y}^{-1}$. (c) Relative error of the inverse.

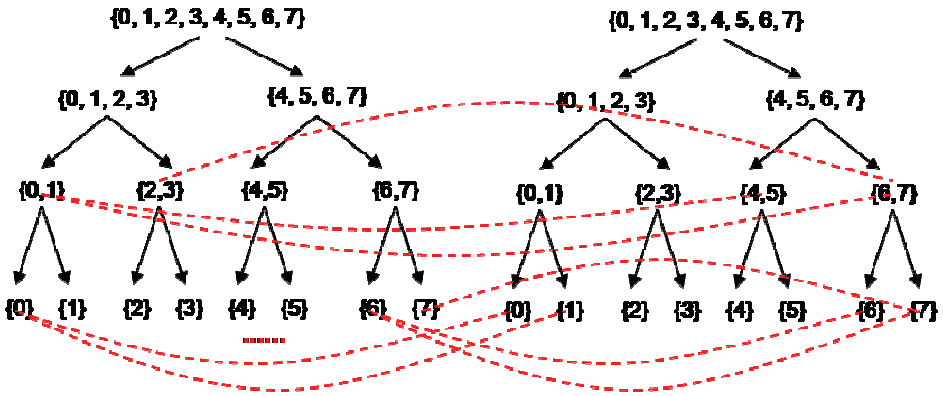
Fig. 4. (a) Illustration of the dielectric-loaded waveguide. (b) $|S_{11}|$ simulated by traditional and proposed solvers.

Fig. 5. Performance of the proposed direct inverse in simulating a dielectric-loaded waveguide from 1.2 wavelengths to 64 wavelengths. (a) CPU time for computing $\mathbf{Y}^{-1}$. (b) Storage of $\mathbf{Y}^{-1}$. (c) Inverse Error. inverse.

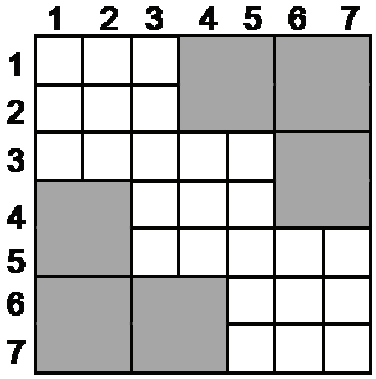
Fig. 6. Illustration of an inductor array. (a) Geometrical and material detail of one inductor. (b) A 7×7 inductor array.

Fig. 7. Performance of the proposed LU-based direct solver for simulating an inductor array. (a) CPU time for LU factorization. (b) CPU time for solving one right hand side. (c) Storage. (d) Accuracy.

FIGURE 1


{0, 1, 2, 3, 4, 5, 6, 7}
{0, 1, 2, 3}
{4, 5, 6, 7}
{0,1}
{2,3}
{4,5}
{6,7}
{0}
{1}
{2}
{3}
{4}
{5}
{6}
{7}
{0, 1, 2, 3, 4, 5, 6, 7}
{0, 1, 2, 3}
{4, 5, 6, 7}
{0,1}
{2,3}
{4,5}
{6,7}
{0}
{1}
{2}
{3}
{4}
{5}
{6}
{7}

(a)


1 2 3 4 5 6 7
1
2
3
4
5
6
7

(b)

FIGURE 2

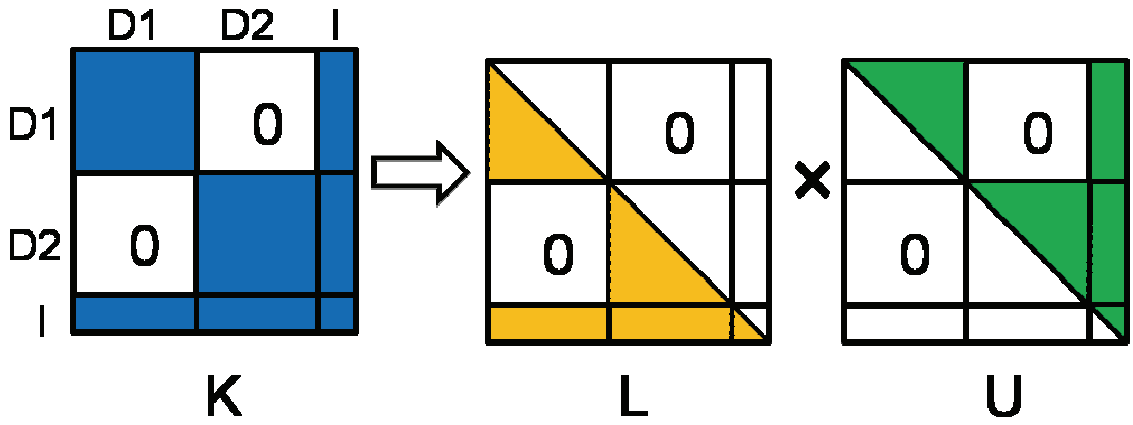
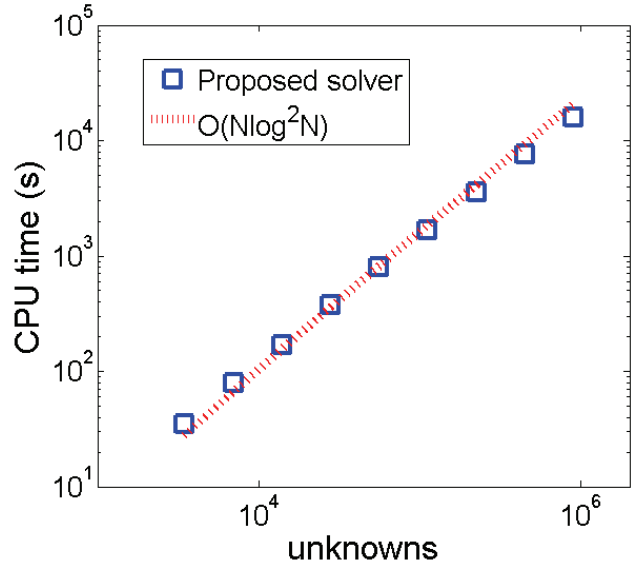
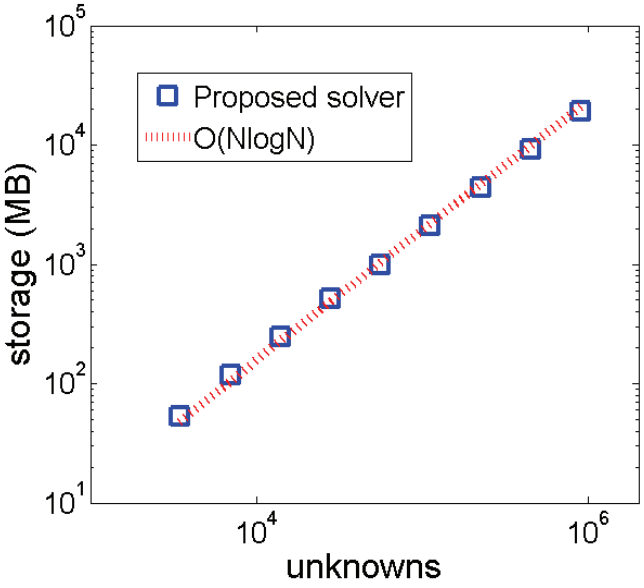


FIGURE 3


Proposed solver
O(Nlog2N)
CPU time (s)
unknowns
10^5
10^4
10^3
10^2
10^1
10^4
10^6

(a)


Proposed solver
O(NlogN)
storage (MB)
unknowns
10^5
10^4
10^3
10^2
10^1
10^4
10^6

(b)

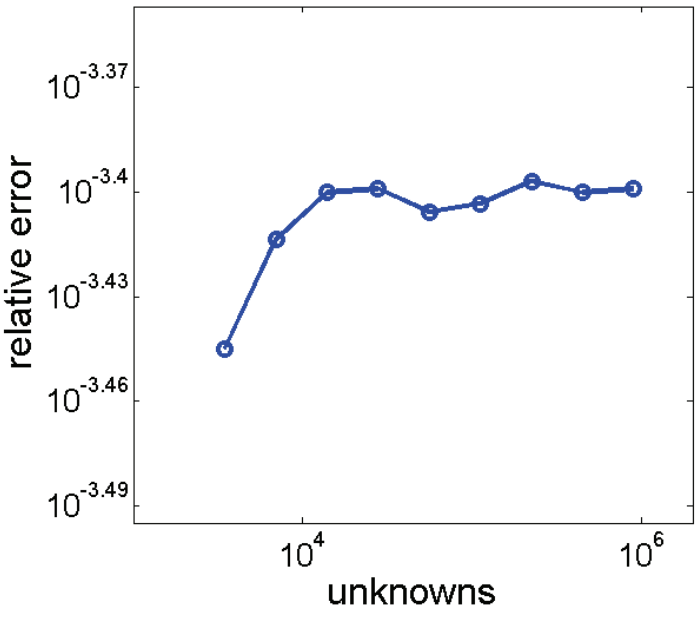
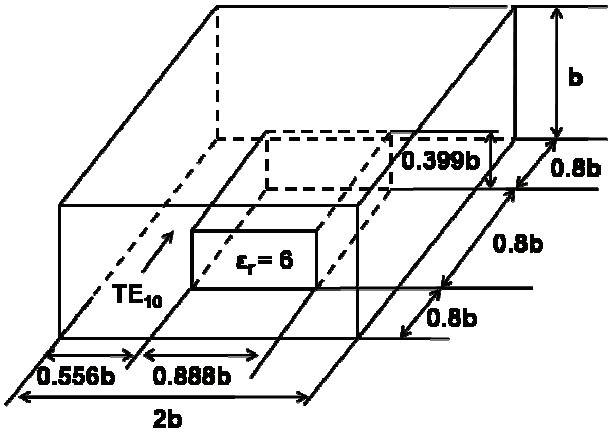
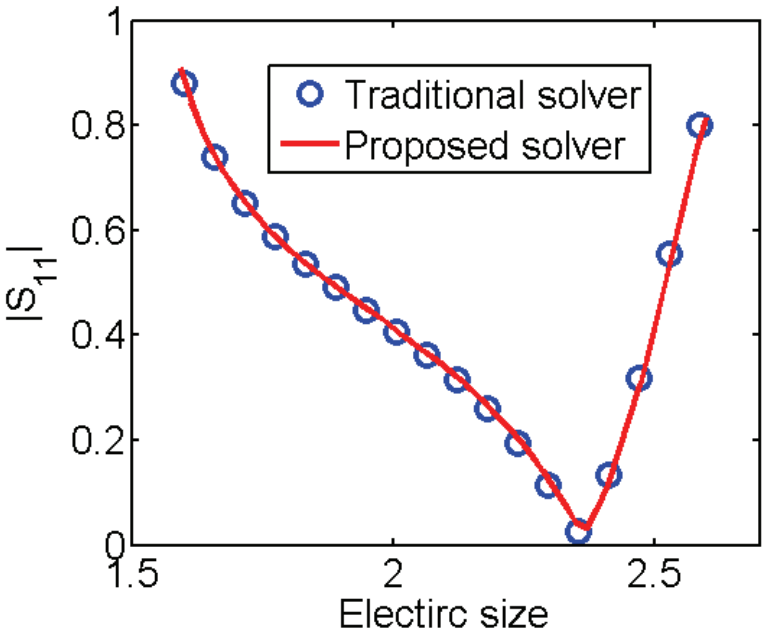

relative error
$10^{-3.37}$
$10^{-3.4}$
$10^{-3.43}$
$10^{-3.46}$
$10^{-3.49}$
10^{4}
10^{6}
unknowns
(c)

FIGURE 4

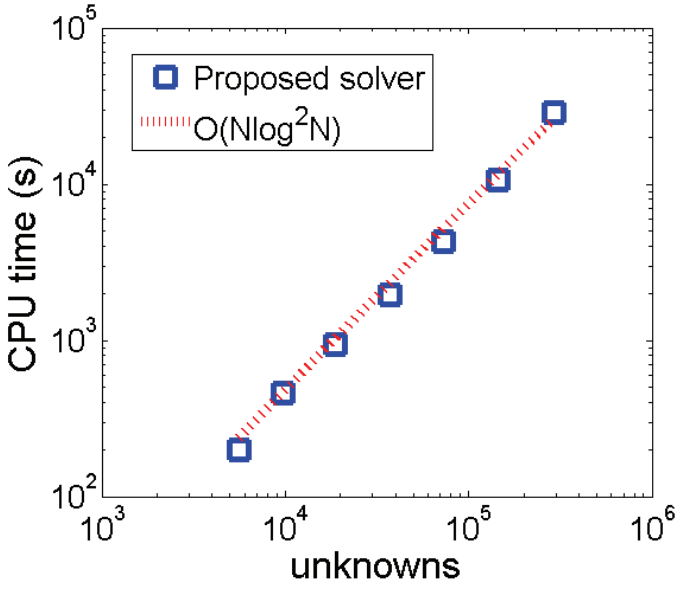

b
0.399b
0.8b
0.8b
0.8b
$\varepsilon_r = 6$
TE_{10}
0.556b
0.888b
2b

(a)

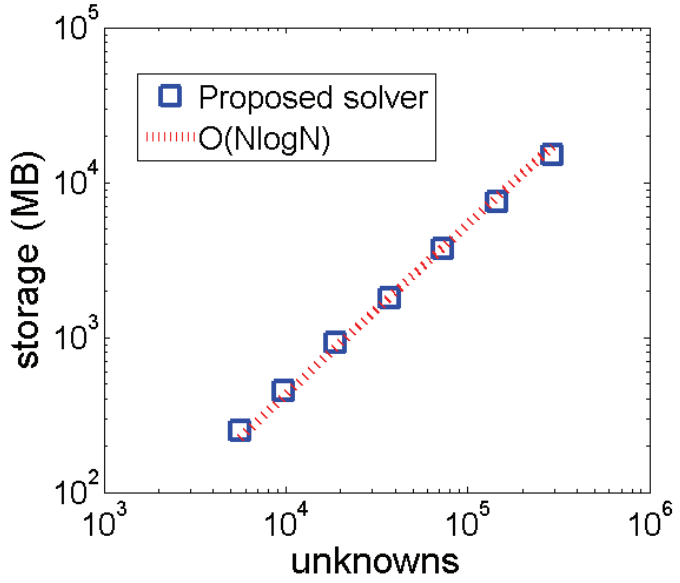

Traditional solver
Proposed solver
$|S_{11}|$
1
0.8
0.6
0.4
0.2
0
1.5
2
2.5
Electirc size

(b)

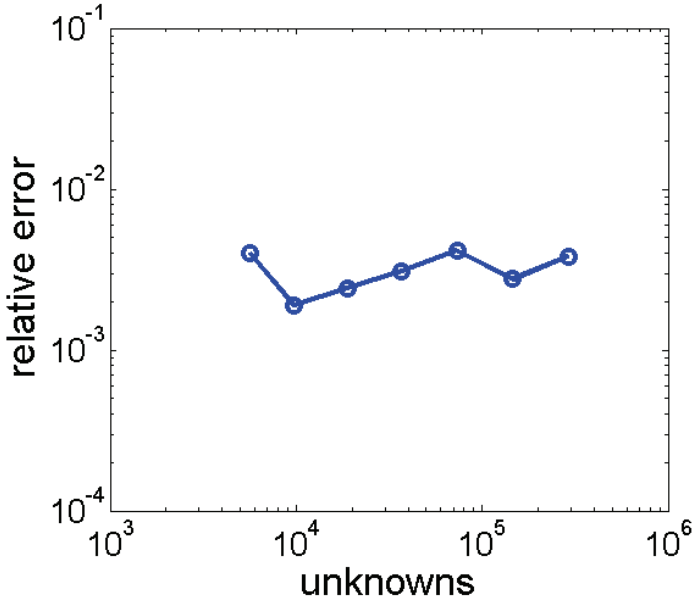
FIGURE 5


Proposed solver
O(Nlog2N)
CPU time (s)
unknowns
10^5
10^4
10^3
10^2
10^3
10^4
10^5
10^6

(a)

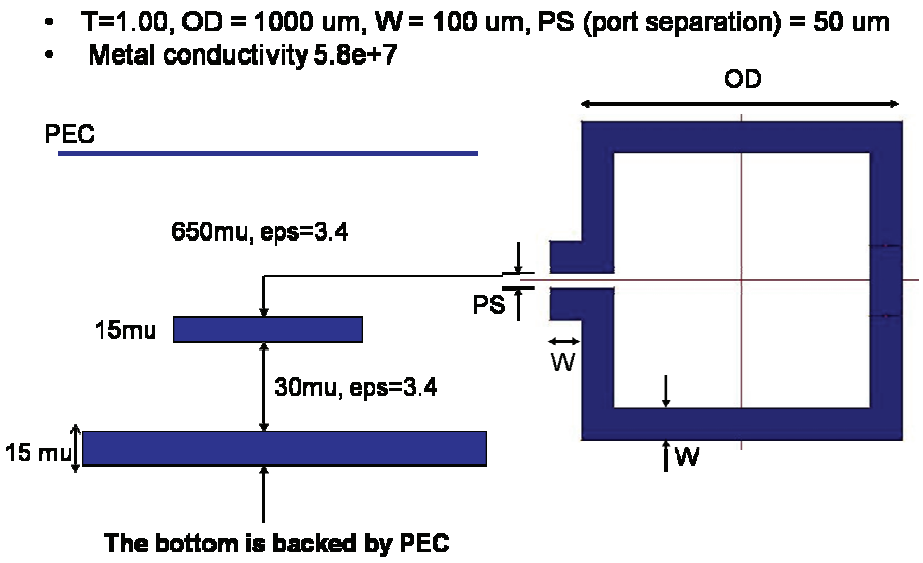

Proposed solver
O(NlogN)
storage (MB)
unknowns
10^5
10^4
10^3
10^2
10^3
10^4
10^5
10^6

(b)

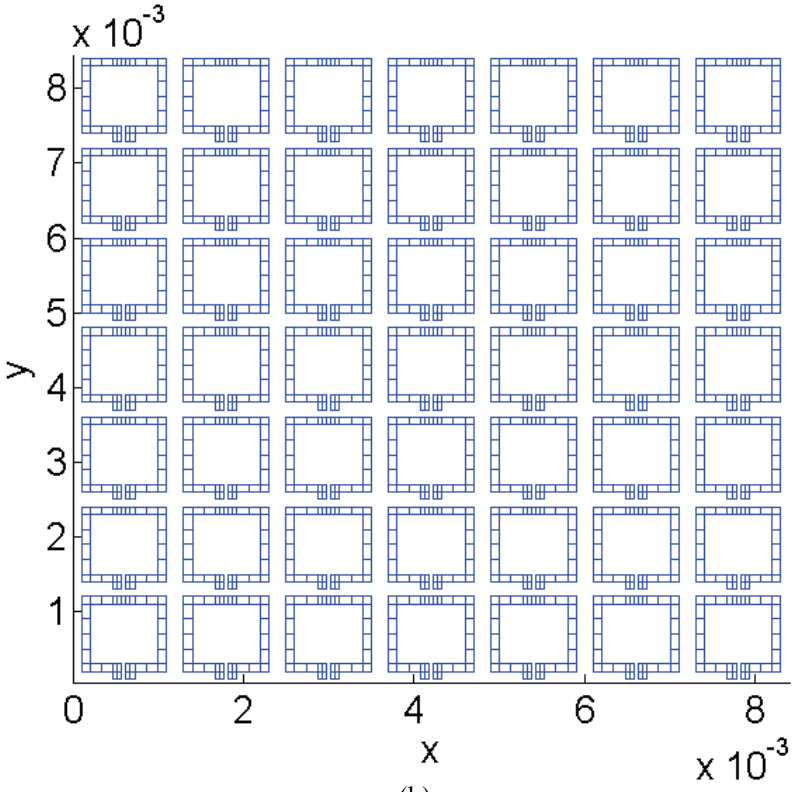


(c)

FIGURE 6

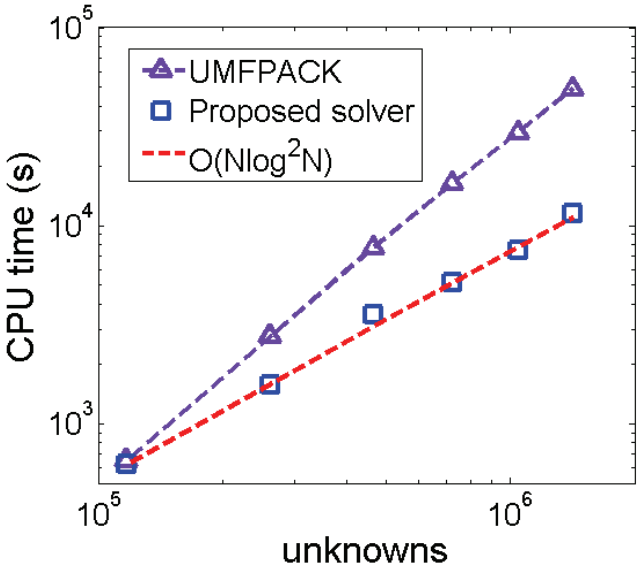

• T=1.00, OD = 1000 um, W = 100 um, PS (port separation) = 50 um
• Metal conductivity 5.8e+7
OD
PEC
650mu, eps=3.4
PS
15mu
W
30mu, eps=3.4
15 mu
W
The bottom is backed by PEC

(a)

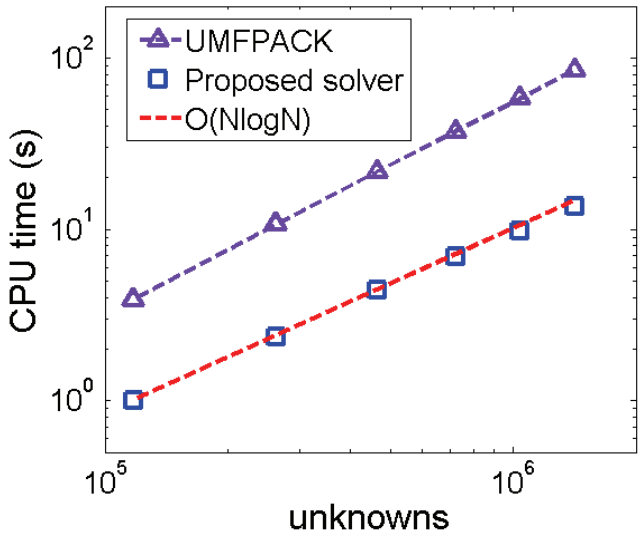

x 10^-3
8
7
6
5
y
4
3
2
1
0
2
4
6
8
x
x 10^-3

(b)

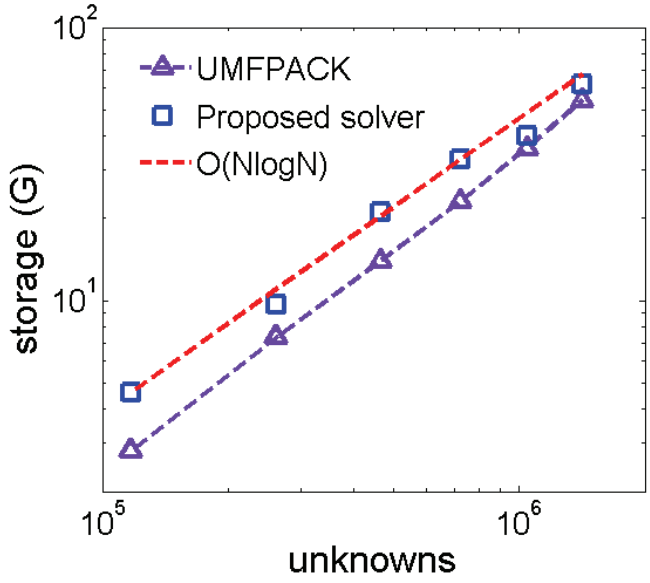
FIGURE 7


UMFPACK
Proposed solver
O(Nlog2N)
CPU time (s)
unknowns
10^5
10^4
10^3
10^5
10^6

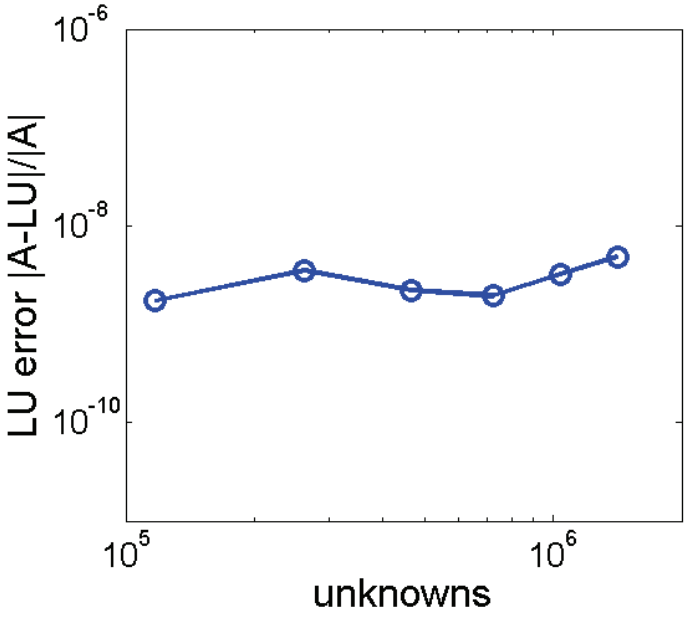
(a)


UMFPACK
Proposed solver
O(NlogN)
CPU time (s)
unknowns
10^2
10^1
10^0
10^5
10^6

(b)


10^2
10^1
storage (G)
UMFPACK
Proposed solver
O(NlogN)
10^5
10^6
unknowns

(c)


10^{-6}
10^{-8}
10^{-10}
LU error |A-LU|/|A|
10^5
10^6
unknowns

(d)

TABLE

Table 1. Rank distribution across the tree level for a 7 ×7 inductor array that has 1,415,127 unknowns.

Tree Level	Minimum k	Maximum k
1	No admissible blocks	
2		
3		
4	2	3
5	2	8
6	1	9
7	3	18
8	3	20
9	6	9
10	2	16
11	1	12
12	1	74
13	1	70
14	1	20
15	1	20
16	1	20
17	1	20
18	1	31
19	1	35
20	4	20
21	5	20