SIAM J. SCI. COMPUT. © 2007 Society for Industrial and Applied Mathematics
Vol. 29, No. 4, pp. 1476–1494

INCOMPLETE LU PRECONDITIONING WITH THE MULTILEVEL FAST MULTIPOLE ALGORITHM FOR ELECTROMAGNETIC SCATTERING[*]

TAHİR MALAS[†] AND LEVENT GÜREL[†‡]

Abstract. Iterative solution of large-scale scattering problems in computational electromagnetics with the multilevel fast multipole algorithm (MLFMA) requires strong preconditioners, especially for the electric-field integral equation (EFIE) formulation. Incomplete LU (ILU) preconditioners are widely used and available in several solver packages. However, they lack robustness due to potential instability problems. In this study, we consider various ILU-class preconditioners and investigate the parameters that render them safely applicable to common surface integral formulations without increasing the $\mathcal{O}(n \log n)$ complexity of MLFMA. We conclude that the no-fill ILU(0) preconditioner is an optimal choice for the combined-field integral equation (CFIE). For EFIE, we establish the need to resort to methods depending on drop tolerance and apply pivoting for problems with high condition estimate. We propose a strategy for the selection of the parameters so that the preconditioner can be used as a black-box method. Robustness and efficiency of the employed preconditioners are demonstrated over several test problems.

Key words. preconditioning, incomplete LU preconditioners, multilevel fast multipole algorithm, electromagnetic scattering

AMS subject classifications. 31A10, 65F10, 78A45, 78M05

DOI. 10.1137/060659107

1. Introduction.

A popular approach in studying wave scattering phenomena in computational electromagnetics (CEM) is to solve discretized surface integral equations, which give rise to large, dense, complex systems in the form of $\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$. Two kinds of surface integral equations are commonly used. The electric-field integral equation (EFIE) can be used for both open and closed geometries, but it results in poorly conditioned systems, especially when the geometry is large in terms of the wavelength. On the other hand, the combined-field integral equation (CFIE) produces well-conditioned systems, but it is applicable to closed geometries only [21].

For the solution of such dense systems, direct methods based on Gaussian elimination are still widely used due to their robustness [34]. However, the large problem sizes confronted in computational electromagnetics prohibit the use of these methods which have $\mathcal{O}(n^2)$ memory and $\mathcal{O}(n^3)$ computational complexity for n unknowns. On the other hand, by making use of the multilevel fast multipole algorithm (MLFMA) [12], the dense matrix-vector products required at least once in each step of the iterative solvers can be performed in $\mathcal{O}(n \log n)$ time and by storing only the sparse near-field matrix elements, rendering these solvers very attractive for large problems.

However, the iterative solver may not converge, or convergence may require too many iterations. We need to have a suitable preconditioner to reach convergence in

[*]Received by the editors May 3, 2006; accepted for publication (in revised form) January 19, 2007; published electronically June 21, 2007. This work was supported by the Scientific and Technical Research Council of Turkey (TUBITAK) under research grant 105E172, by the Turkish Academy of Sciences in the framework of the Young Scientist Award Program (LG/TUBA-GEBIP/2002-1-12), and by contracts from ASELSAN and SSM.

http://www.siam.org/journals/sisc/29-4/65910.html

[†]Department of Electrical and Electronics Engineering, Bilkent University, TR-06800, Bilkent, Ankara, Turkey (tmalas@ee.bilkent.edu.tr, lgurel@bilkent.edu.tr).

[‡]Computational Electromagnetics Research Center (BiLCEM), Bilkent University, TR-06800, Bilkent, Ankara, Turkey.

a reasonable number of iterations and retain the $\mathcal{O}(n \log n)$ complexity of MLFMA. We can use the sparse near-field matrix $\overline{\mathbf{N}}$ to construct a preconditioner. Since this matrix is the best available approximation to the coefficient matrix $\overline{\mathbf{A}}$, it makes sense to use the near-field matrix as a preconditioner and solve (for example) the left-preconditioned system

$$\overline{\mathbf{N}}^{-1} \cdot \overline{\mathbf{A}} \cdot \mathbf{x} = \overline{\mathbf{N}}^{-1} \cdot \mathbf{b}. \tag{1.1}$$

The inversion of the near-field matrix $\overline{\mathbf{N}}$ can be accomplished using direct methods which decompose the matrix into a product of a unit lower-triangular matrix $\overline{\mathbf{L}}$ and an upper-triangular matrix $\overline{\mathbf{U}}$. However, during the factorization of sparse matrices, in general, fill-in occurs and the resulting factors lose their sparsity [28]. This may make it difficult to preserve the $\mathcal{O}(n \log n)$ complexity of MLFMA. Nevertheless, we can discard part of the fill-in and partially incorporate the robustness of the LU factorization into the iterative method by using the incomplete factors of $\overline{\mathbf{N}}$ as a preconditioner. This is the general idea behind the incomplete LU (ILU) preconditioners.

In a general setting, depending on the dropping strategy, we can talk about two kinds of ILU-class preconditioners. The first one depends on the matrix structure and the entries are dropped by their position. A "levels of fill-in" concept is introduced and stronger preconditioners can be constructed by increasing the level of fill-in [31]. Since this technique does not consider numerical values, it becomes ineffective in predicting the locations of the largest entries, particularly for matrices that are far from being diagonally dominant and indefinite [13]. This is the case for the matrices arising from the EFIE formulation. Alternatively, one can drop the matrix elements depending on their magnitudes, and the zero pattern is generated dynamically during the factorization. Among such methods, ILUT(τ, p) proposed by Saad has been successful for many general systems [3]. During the factorization, ILUT drops matrix elements that are smaller than τ times the 2-norm of the current row; and of all the remaining entries no more than the p largest ones are kept. ILUT is known to yield more accurate factorizations than the level-of-fill methods with the same amount of fill-in [13].

Although ILUT is more robust than its counterparts depending on the level of fill-in, it may occasionally encounter problems of instability for real-life problems. Even when factorization terminates normally, the resulting incomplete factors may sometimes be unstable. The common reasons of instability are in general excessive dropping and small pivots [13]. If the problem is related to the small pivots, one can significantly increase the quality of the ILUT preconditioner by using partial pivoting as in the complete factorization case. The resulting preconditioner is called ILUTP [31].

In order to understand the quality of the preconditioner, or to understand the reason for failure when it occurs, we can use $\|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1} \cdot \mathbf{e}\|_\infty$, where $\mathbf{e}$ is the vector of ones. This statistic is called *condest* (for condition estimate) and it provides an upper bound for $\|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1}\|_\infty$ [13]. If the *condest* value is not very high, but the preconditioner still does not work, one can deduce that fill-in should be increased to achieve a successful preconditioner. On the other hand, if the *condest* value is high, one can first try pivoting to remedy the situation instead of including more elements in the incomplete factors.

Considering the remarkable success of ILU-class preconditioners for general nonsymmetric and indefinite systems [13] and the wide availability of ILU-class preconditioners in various packages [2, 24, 6], the present study aims to develop a strategy for both selecting the most appropriate ILU-class preconditioner and determining their parameters to use them as black-box preconditioners for CFIE and EFIE formulations. We perform tests on canonical, quasi-canonical, and real-life problems with increasing number of unknowns and show that when these preconditioners fail for the reasons stated earlier, the failure can be circumvented using pivoting strategies without increasing the memory cost. We also show that the *condest* value is very useful for determining the quality of the resulting factorization before starting the iterative solution.

ILU-class preconditioners have been tested for electromagnetic problems in [8, 32, 26]. In [8], ILU(0) was tried on systems resulting from EFIE formulation with discouraging results in all test problems. Sertel and Volakis [32] tried ILU(0) on two model problems. For the very small problem of 480 unknowns, ILU(0) was successful with the EFIE and CFIE formulations, but clearly such a small problem is not representative of large-scale CEM simulations. They presented only CFIE results for the 50,000-unknown problem; in this case, ILU(0) was quite successful in reducing the number of iterations. Probably the most impressive results are those of Lee, Zhang, and Lu [26], who tried the ILUT preconditioner on hybrid surface-volume integral equations and showed it to be successful on many test problems. However, they neither tried commonly used EFIE or CFIE formulations nor applied pivoting or any other techniques to increase the effectiveness of the preconditioner.

The rest of the paper is organized as follows. In the next section, we outline the surface integral equations of CEM, their discretizations, resulting matrix equations, and MLFMA. Spectral properties of such matrices are analyzed in section 3. Then, in section 4, we comment on the preconditioning of the systems arising from integral-equation formulations. In section 5, we present experimental results for the CFIE and EFIE formulations. In the last section, we make some suggestions for the effective and safe usage of ILU-class preconditioners with EFIE and CFIE.

2. Surface integral equations and fast solvers. EFIE and CFIE are surface integral equations that are formed from the application of physical boundary conditions; they are formulated to solve the radiation and scattering problems of arbitrarily shaped geometries. This leads to the reduction of a three-dimensional problem into a two-dimensional problem, but the resulting matrix equation becomes fully populated.

The boundary condition stating that the total tangential electric field should vanish on a conducting surface can be mathematically expressed to obtain EFIE as

$$\hat{\boldsymbol{t}} \cdot \int_{S'} \mathrm{d}\mathbf{r}' \overline{\mathbf{G}}(\mathbf{r},\mathbf{r}') \cdot \mathbf{J}(\mathbf{r}') = \frac{i}{k\eta} \hat{\boldsymbol{t}} \cdot \mathbf{E}^{inc}(\mathbf{r}), \tag{2.1}$$

where $\mathbf{E}^{inc}$ represents the incident electric field, S' is the surface of the object, $\hat{\boldsymbol{t}}$ is any tangential unit vector on S', $\mathbf{J}(\mathbf{r}')$ is the unknown induced current residing on the surface,

$$\overline{\mathbf{G}}(\mathbf{r},\mathbf{r}') = \left[\overline{\mathbf{I}} + \frac{\nabla\nabla}{k^2} \right] g(\mathbf{r},\mathbf{r}') \tag{2.2}$$

is the dyadic Green's function, and

$$g(\mathbf{r},\mathbf{r}') = \frac{e^{ik|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|} \tag{2.3}$$

is the scalar Green's function for the three-dimensional scalar Helmholtz equation. Green's function represents the response at $\mathbf{r}$ due to a point source located at $\mathbf{r}'$.

Similarly, the boundary condition for the tangential magnetic field on a conducting surface is used to derive the magnetic-field integral equation (MFIE) [22, 14] as

$$\mathbf{J}(\mathbf{r}) - \hat{\boldsymbol{n}} \times \int_{S'} \mathrm{d}\mathbf{r}' \mathbf{J}(\mathbf{r}') \times \nabla' g(\mathbf{r}, \mathbf{r}') = \hat{\boldsymbol{n}} \times \mathbf{H}^{inc}(\mathbf{r}), \tag{2.4}$$

where $\hat{\boldsymbol{n}}$ is any unit normal on S' and $\mathbf{H}^{inc}(\mathbf{r})$ is the incident magnetic field. It should be noted that EFIE is valid for both open and closed geometries, whereas MFIE is valid only for closed surfaces [18, 16].

Combining EFIE and MFIE, we obtain CFIE, i.e.,

$$\text{CFIE} = \alpha\text{EFIE} + (1 - \alpha)\text{MFIE}, \tag{2.5}$$

where α is a parameter between 0 and 1. CFIE is free from the internal resonance problems of both EFIE and MFIE and leads to well-conditioned systems [19]. On the other hand, CFIE is not applicable to open geometries since it contains MFIE. Therefore, CFIE is preferred over MFIE for closed geometries, but EFIE is the only choice for open geometries. CFIE can also be extended for geometries containing both closed and open surfaces [23].

Surface integral equations can be converted to linear systems of equations using the method of moments (MOM). In fact, MOM can be used to reduce any linear-operator equation to a matrix equation. Using a linear operator L, we can represent EFIE and CFIE as

$$L\{\mathbf{f}\} = \mathbf{g}, \tag{2.6}$$

where $\mathbf{f}$ is the unknown vector function, and $\mathbf{g}$ is the excitation. To discretize this equation, $\mathbf{f}$ is first approximated by a set of vector basis functions via the expansion

$$\mathbf{f} \approx \sum_{j=1}^{n} x_j \mathbf{f}_j, \tag{2.7}$$

where x_j is the unknown coefficient of the jth basis function. Then this discretized equation is tested at n points and a linear system is obtained, i.e.,

$$\sum_{j=1}^{n} x_j \langle \mathbf{t}_i, L\{\mathbf{f}_j\}\rangle = \langle \mathbf{t}_i, \mathbf{g}\rangle, \qquad i = 1, \ldots, n. \tag{2.8}$$

In (2.8), $\mathbf{t}_i$ is the vector testing function and the inner product is defined as

$$\langle \mathbf{t}, \mathbf{f}\rangle = \int_S \mathrm{d}\mathbf{r}\, \mathbf{t}(\mathbf{r}) \cdot \mathbf{f}(\mathbf{r}). \tag{2.9}$$

By defining the elements of the coefficient matrix (or the so-called impedance matrix) as

$$A_{ij} = \langle \mathbf{t}_i, L\{\mathbf{f}_j\}\rangle \tag{2.10}$$

and the elements of the right-hand-side vector (or the excitation vector) as

$$b_i = \langle \mathbf{t}_i, \mathbf{g}\rangle, \tag{2.11}$$

the linear system of equations can be expressed as

$$\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}. \tag{2.12}$$

In order to implement a Galerkin approach for discretization, we use Rao–Wilton–Glisson (RWG) functions [29] for both basis and testing functions. RWG functions are linearly varying vector functions defined on planar triangular domains. They are widely used in MOM applications to discretize surface current distributions.

Surfaces of geometries are meshed with planar triangles in accordance with the RWG functions. Each basis function is associated with an edge; hence the number of unknowns n is equal to the total number of edges in a triangulation. For high accuracy, triangle sizes have to be small, but this leads to problems with large numbers of unknowns. As a rule of thumb, we choose the average size of the mesh about one-tenth of the wavelength.

Since the integrodifferential operator L for both EFIE and CFIE involves long-distance interactions, the resulting coefficient matrix is dense, which is expensive to store and to solve. However, when we use iterative solvers, direct solution is replaced with a series of matrix-vector multiplications, and we have the opportunity to perform the matrix-vector product with $\mathcal{O}(n \log n)$ complexity using MLFMA.

MLFMA can be described as the multilevel application of the fast multipole method (FMM). An important concept, on which FMM relies, is the diagonalized factorization of the Green's function, i.e.,

$$g(\mathbf{r}, \mathbf{r}') = \frac{e^{ik|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|} = \frac{e^{ik|\mathbf{D}+\mathbf{d}|}}{4\pi|\mathbf{D}+\mathbf{d}|} \approx \frac{1}{4\pi} \int d^2\hat{\boldsymbol{k}} e^{i\hat{\boldsymbol{k}}\cdot\mathbf{d}} T_L(k, D, \psi), \tag{2.13}$$

where $D = |\mathbf{D}| < d = |\mathbf{d}|$, the integration is on the unit sphere, $\hat{\boldsymbol{k}}$ is the unit vector normal to the unit sphere, and T_L is the truncated sum

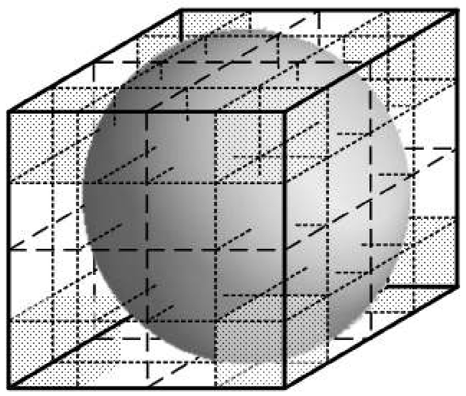
$$T_L(k, D, \psi) = \frac{ik}{4\pi} \sum_{l=0}^{L} i^l (2l+1) h_l^{(1)}(kD) P_l \cos(\psi), \tag{2.14}$$

which is called the translation function [17]. In (2.14), $h_l^{(1)}(x)$ is the spherical Hankel function of the first kind, P_l is the Legendre polynomial, and ψ is the angle between unit vectors $\hat{\boldsymbol{k}}$ and $\hat{\boldsymbol{D}}$. The truncation number L is determined by the formula given in [12],

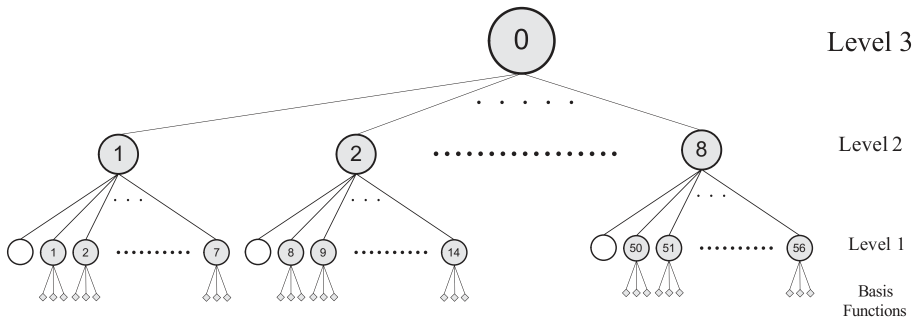
$$L \approx kd + 1.8 d_0^{2/3} (kd)^{1/3}, \tag{2.15}$$

where d_0 is the number of accurate digits required in the matrix-vector multiplication.

In a physical setting, we can think of a matrix-vector multiplication as a set of electromagnetic interactions between the basis and testing functions. When the basis and testing functions are clustered according to the proximity of their locations in space, the same translation function can be used for all interactions between pairs of functions in any two clusters, instead of calculating the interactions separately. For this purpose, radiation patterns of the basis functions $\mathbf{f}_j$ weighted with the corresponding coefficients x_j are evaluated on the unit sphere at K directions. Next, in each basis group, the radiation patterns are added at each direction, a process called aggregation. For each pair of basis and testing groups, a translation function is defined. The overall radiation pattern of each basis group is multiplied by a translation function and thereby translated to the center of a testing cluster. Following



(a)



(b)

FIG. 2.1. (a) *Multilevel partitioning of the scatterer for the case of a sphere with diameter* 1λ. *The shaded boxes are empty.* (b) *Tree structure of MLFMA for the sphere. Unfilled nodes correspond to empty boxes.*

the translations, radiation patterns of the basis groups are added at the center point of each testing group. Finally, the total radiation from all basis functions at each testing cluster is disaggregated onto individual testing functions in order to complete the testing process. Since the factorization of the Green's function is not valid for basis and testing functions that are close to each other, the near-field interactions are calculated directly. In a multilevel scheme, MLFMA requires interpolations and anterpolations to pass from one level to another [15].

In MLFMA, to perform a multilevel application of FMM, the whole geometry is placed in a cube and then this cube is recursively divided into smaller ones until the smallest cube contains only a few basis or testing functions. However, during the partitioning, if any cube becomes empty, recursion stops there. This partitioning defines the levels of MLFMA and corresponds to a tree structure. An example of a sphere with a diameter of one wavelength (1λ) is shown in Figure 2.1.

With the partitioning scheme described, we can easily determine the near-field and far-field clusters. On any level, pairs of same-size cubes touching at any point are in the near-field zone of each other and the others are in the far-field zone. In the MLFMA tree, we refer to the filled cubes as clusters. A cluster contains either some basis functions (in the lowest level) or low-level clusters. In MLFMA, the replacement of the element-to-element interactions with cluster-to-cluster interactions is made in

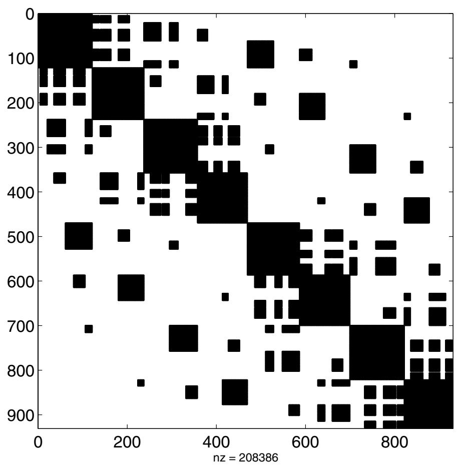


FIG. 2.2. *Near-field pattern of the* 930*-unknown sphere problem.*

a multilevel scheme.

In the lowest level, interactions between the near-field clusters are computed exactly and stored in the sparse near-field matrix. When the ordering of the unknowns is in accordance with the clustering, the near-field matrix is composed of small blocks, but the distribution of the blocks does not exhibit any structured pattern. For the sphere geometry shown in Figure 2.1, the nonzero pattern of the near-field matrix is presented in Figure 2.2. Self-interactions of the second-level clusters can easily be observed in the diagonal blocks of the nonzero pattern.

Interactions among the far-field clusters are computed approximately (but with controllable accuracy) and efficiently. For each level excluding the highest two, radiated fields of each cluster are aggregated at the centers of the clusters. Then, for each pair of far-field clusters whose parents are in the near-field zone of each other, cluster-to-cluster interaction is computed via a single translation. If the parents of the pair of clusters are not in the near-field of each other, the cluster-to-cluster interaction is included in a higher-level translation. Finally, after the translations, the summed values at the centers of the testing clusters are disaggregated towards the testing functions, completing the matrix-vector multiplication.

3. Spectral analysis.

For electromagnetic scattering problems, iterative methods with MLFMA provide new opportunities to solve large-scale problems that were previously unsolvable. However, EFIE, MFIE, and CFIE formulations produce indefinite systems for which convergence becomes an issue. Moreover, since we consider the solution of very large problems with millions of unknowns, the condition of the matrix deteriorates (especially for EFIE) as the problem sizes grow. Hence, effective preconditioning is indispensable for attaining convergence in a reasonable time.

Though they are indefinite and nonhermitian, CFIE produces well-conditioned systems that are close to being diagonally dominant. As a consequence, the number of iterations required for convergence has been limited with a simple block-Jacobi preconditioner even for large-scale problems [19]. Nonetheless, for some geometries, the number of iterations is still large. Considering the dominant cost of matrix-vector product for large problems, better preconditioners for CFIE are still desirable.

Systems resulting from EFIE are much more difficult to solve. In addition to being

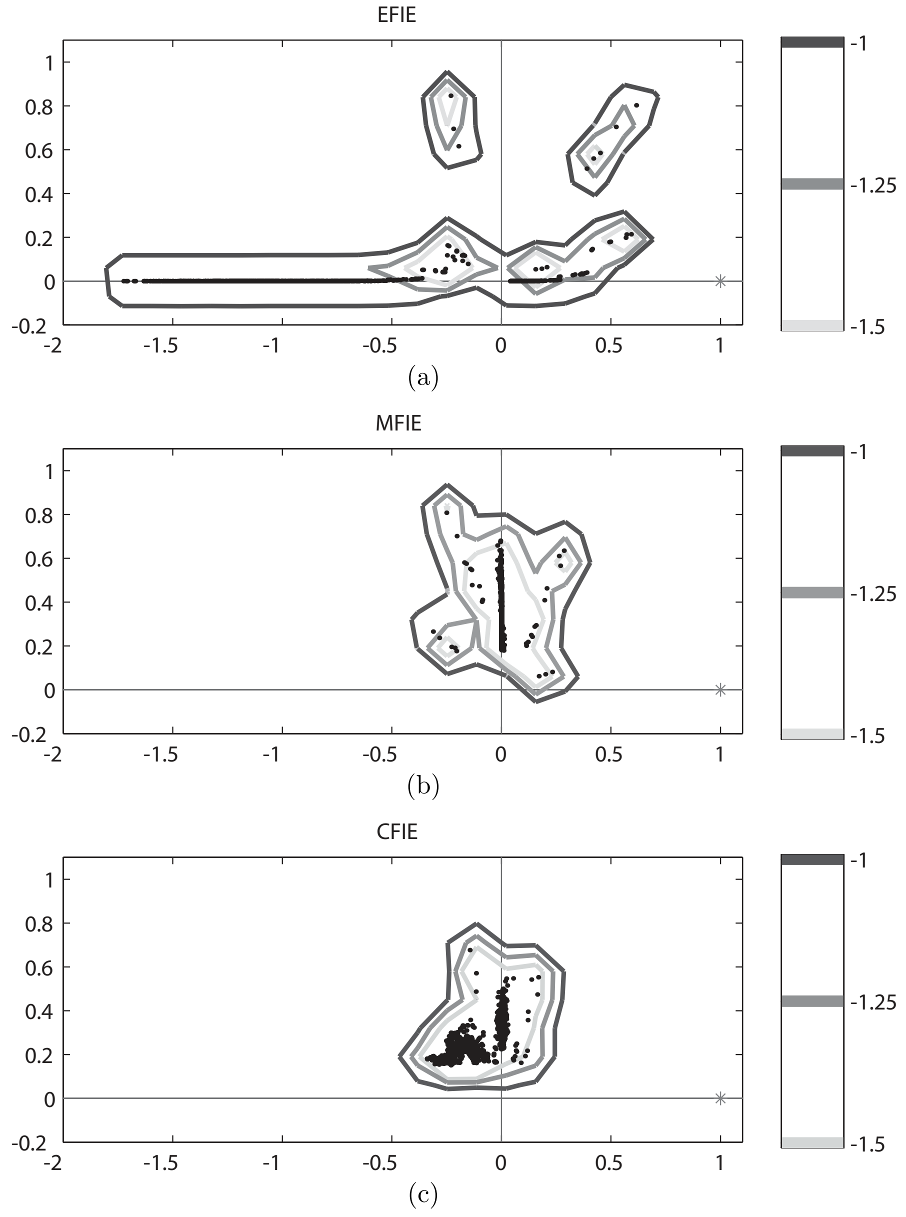


FIG. 3.1. *Pseudospectra of the EFIE, MFIE, and CFIE formulations for three ϵ values, i.e.,* 10^{-1}, $10^{-1.25}$, *and* $10^{-1.5}$. *The black dots denote the exact eigenvalues of the unperturbed matrices.*

indefinite and nonhermitian, EFIE matrices may have large elements away from the diagonal, and some of the nonstored far-field interactions may be stronger than the near-field interactions.

For a better understanding of the properties of the systems resulting from surface integral equations, we show in Figure 3.1 both the eigenvalues and the pseudospectra of the EFIE, MFIE, and CFIE matrices for the 930-unknown sphere problem. For nonnormal matrices, information obtained from eigenvalues may be misleading, since they may become highly unstable [33]. More reliable and telling information can be obtained using the ϵ-pseudospectrum, $\wedge_\epsilon(\overline{\mathbf{A}})$, which can be defined as

$$\wedge_\epsilon(\overline{\mathbf{A}}) = \left\{ z \mid z \in \mathbb{C}^n,\ \|(z\overline{\mathbf{I}} - \overline{\mathbf{A}})^{-1}\|_2 \geq \frac{1}{\epsilon} \right\}. \tag{3.1}$$

Denoting the spectrum of $\overline{\mathbf{A}}$ with $\wedge(\overline{\mathbf{A}})$, if an eigenvalue $\lambda \in \wedge(\overline{\mathbf{A}})$, then $\|(z\overline{\mathbf{I}} - \overline{\mathbf{A}})^{-1}\|_2 = \infty$, so $\wedge(\overline{\mathbf{A}}) \subset \wedge_\epsilon(\overline{\mathbf{A}})$ for any $\epsilon > 0$. The pseudospectrum represents the topology of the eigenvalues of the perturbed matrices associated with the exact matrix $\overline{\mathbf{A}}$, and thus gives an idea about the nonnormality.

The ultimate aim in preconditioning is to move all eigenvalues towards the point (1,0). However, if the matrix is close to normal, a spectrum clustered away from the origin also implies rapid convergence for Krylov subspace methods [3, 25]. Comparison of the EFIE and CFIE pseudospectra in Figure 3.1 indicates that combining EFIE with MFIE (to obtain CFIE) has the effect of clustering the distributed eigenvalues and moving them towards the right half-plane. In contrast, most of the eigenvalues of EFIE are scattered in the left half-plane. Moreover, the 0.1-pseudospectrum of the EFIE matrix contains the origin, signaling the near-singularity of the matrix. Hence, effective preconditioning for EFIE becomes more difficult, and also more crucial.

4. ILU-class preconditioners.

4. ILU-class preconditioners. Various preconditioners have been used for the solution of CEM problems. For CFIE, a block-Jacobi preconditioner is frequently used. In an MLFMA setting, a block-Jacobi preconditioner can be constructed from the self-interactions of the lowest-level clusters. Since there are $\mathcal{O}(n)$ such blocks and each block is composed of a fixed number of unknowns, both the construction and the application of the preconditioner scale with $\mathcal{O}(n)$. Because of its optimal complexity and success with many problems, this simple preconditioner is a common choice for CFIE. However, probably due to the weaker diagonal dominance and indefiniteness of EFIE, the block-Jacobi preconditioner performs even worse than the no-preconditioner case. Sparse approximate inverse (SAI) preconditioners depending on a fixed a priori pattern have been thoroughly studied in some recent works [9, 10, 11, 27, 1]. The electromagnetics community has started to use SAI preconditioners more frequently because of ease of parallelization. However, the construction cost of SAI can become prohibitively large unless one chooses the prefiltering and postfiltering threshold parameters carefully [27]. Hence, it is not suitable for use as a black-box preconditioner; there is still the need for a more easily attainable preconditioner, particularly for sequential implementations.

As an alternative, the ILU-class preconditioners have been widely used and included in several solver packages. They were historically developed for positive-definite and structured matrices arising from the discretization of partial differential equations. For general systems, the failure rate of ILU-class preconditioners is still high. Nonetheless, there have been many improvements to increase their robustness [13, 4, 5].

For iterative solvers utilizing MLFMA, the near-field matrix $\overline{\mathbf{N}}$ is the natural candidate to generate the incomplete factors. Consider an incomplete factorization of the near-field matrix, $\overline{\mathbf{N}} \approx \overline{\mathbf{L}} \cdot \overline{\mathbf{U}}$. If we let the sparsity patterns of $\overline{\mathbf{N}}$ and $\overline{\mathbf{L}} + \overline{\mathbf{U}}$ be the same, that is, if we retain nonzero values of $\overline{\mathbf{L}}$ and $\overline{\mathbf{U}}$ only at the nonzero positions of $\overline{\mathbf{N}}$, we end up with the no-fill LU method, or ILU(0). This simple idea works successfully for well-conditioned matrices [31]. Denser and potentially more effective preconditioners can be obtained by increasing the level of fill-in, but this strategy is unsuccessful in determining the largest entries, particularly for matrices that are indefinite and far from being diagonally dominant. A more robust strategy is to drop the nonzero elements by comparing the magnitudes during the factorization. Such a strategy discards elements that are small with respect to a suitably chosen drop tolerance τ.

One of the disadvantages of the dropping strategy, which depends on the size of matrix entries, is the difficulty in predicting storage. For this purpose, a dual threshold strategy can be used [30]. The resulting preconditioner, called ILUT(τ, p), retains no more than p elements in the incomplete factors after dropping all the elements that are smaller than τ times the 2-norm of the current row. The threshold parameter τ determines the CPU time and p determines the storage requirement of the preconditioner. This preconditioner is known to be quite powerful and robust.

Despite ILUT's good reputation, there are two important drawbacks preventing its use as a black-box and general library software. The first problem is determining the appropriate parameters. For our specific applications and in the context of MLFMA, we propose to select a small drop tolerance and then set the parameter p so that the preconditioner will have approximately the same number of nonzero elements as the near-field matrix $\overline{\mathbf{N}}$. Once the near-field matrix is generated, this value can easily be found. With this strategy, we aim to obtain a powerful preconditioner with modest storage and low complexity.

Probably the more problematic aspect of threshold-based ILU-class preconditioners is their potential inaccuracy and/or instability. Accuracy refers to how close the incomplete factors are to $\overline{\mathbf{A}}$; this is measured by the norm of the error matrix, i.e., *accuracy* $= \|\overline{\mathbf{E}}\| = \|\overline{\mathbf{A}} - \overline{\mathbf{L}} \cdot \overline{\mathbf{U}}\|$. Stability refers to how close the preconditioned matrix is to the identity matrix and is measured by the norm of the preconditioned error, i.e., *stability* $= \|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1} \cdot \overline{\mathbf{E}}\|$. If $\|\overline{\mathbf{L}}^{-1}\|$ or $\|\overline{\mathbf{U}}^{-1}\|$ are extremely large, a factorization may turn out to be accurate but unstable; in that case, the preconditioner may not work even if fill-in is increased [13]. Thus, for general matrices, stability is a more informative measure of the preconditioning quality.

Although we cannot compute these metrics with MLFMA, a rough estimate of $\|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1}\|$, called *condest*, gives a clue about the instability of the triangular factors [13]. This condition estimate is defined as

$$\|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1} \cdot \mathbf{e}\|_\infty, \qquad \mathbf{e} = [1, \ldots, 1]^T. \tag{4.1}$$

One can easily compute *condest* before the iterations, by using a forward substitution followed by a backward substitution, and it provides a strong indicator of the quality of the ILU preconditioner.

When the incomplete factors turn out to be unstable, there are some remedies that can be utilized depending on the cause. Preprocessing steps such as diagonal perturbation, reordering, and scaling can be applied on the coefficient matrix to stabilize the preconditioner. To increase the stability, diagonal perturbations can be used to make the LU factors more diagonally dominant, but quite large perturbations may be required for indefinite systems, and such large perturbations may introduce too much inaccuracy into the preconditioner. Diagonal shifts have already been tried on EFIE systems to increase the robustness of ILU, but the effect of the shift is undetermined, and furthermore it is difficult to select suitable shift parameters [7]. Reorderings aimed at improving the condition of the incomplete factors are widely studied. Indeed, some reordering schemes significantly improve the convergence of the Krylov methods [4]. However, the effect of ordering becomes significant when the incomplete factors are allowed to be denser than the original matrix [3]. We usually prefer to keep the memory required by the preconditioner bounded by the storage needed for the near-field matrix. Hence, this remedy is not a good candidate because of the storage considerations. Finally, for threshold-based ILU-class preconditioners, it is recommended to scale the matrix so that each column has unit 2-norm, and then scale it again so that each row has unit 2-norm. This suggestion is not applicable in the context of MLFMA, since the preconditioner is constructed from a sparse portion of the coefficient matrix and the coefficient matrix is not explicitly available.

On the other hand, if the instability is caused by the small pivots, partial pivoting is helpful. This is a well-known and much simpler method. Column pivoting can be applied in a rowwise factorization with negligible cost. The resulting preconditioner is known as ILUTP [31]. In some cases, it may be useful to include a permutation tolerance *permtol* and perform the permutation for the ith row when $permtol \times |a_{ij}| > |a_{ii}|$. It is best not to select a very small value for *permtol*; 0.5 is accepted as a good choice [31].

5. Results.

In this section, we show the effectiveness of ILU-class preconditioners for electromagnetic scattering problems. We first identify the most appropriate ILU-class preconditioner for the problem type (i.e., open geometries vs. closed geometries, EFIE vs. CFIE), then compare the selected ILU preconditioner with other commonly used preconditioners. For this purpose, we implement an SAI preconditioner, whose sparsity pattern is chosen to be the same as the near-field matrix. In this way, it has the same storage cost as that of ILU(0).

Instead of giving several results with varying parameters for ILUT, we adopt the following strategy for the selection of the parameters. We set the drop tolerance τ to a low value such as 10^{-6} and set p, the maximum number of nonzero elements per row, such that the memory cost of factorization does not exceed that of the no-fill ILU preconditioner. We accomplish this by simply letting p be the average number of nonzero elements in a row of the near-field matrix. In this way, we obtain robust preconditioners with modest computational requirements.

For the specific implementation of the MLFMA considered here, we set the size of the smallest clusters to 0.25λ, the number of accurate digits d_0 to 3, and the α parameter of CFIE to 0.2. The CPU times reported in this section were obtained on a 64-bit server with 1.8 GHz AMD Opteron 244 processors and 4 GB of memory. In addition to performing numerical experiments involving ILU-class preconditioners, we also obtain the exact solution of the near-field matrix to use it as a benchmark preconditioner. This solution, which is denoted by LU, is performed on another 64-bit server with 24 GB of memory. Due to its excessive computational requirements, this LU preconditioner is presented merely for comparison purposes.

For the iterative solver, starting with the zero initial guess, we try to reduce the initial residual norm by 10^{-6} and set the maximum number of iterations at 1,500. We use the generalized minimal residual method (GMRES) with no-restart and apply right-preconditioning in order to minimize the true residual norm. For CFIE solution of closed geometries, the performance of other Krylov subspace methods, such as conjugate gradient squared (CGS), biconjugate gradient (Bi-CG) or biconjugate gradient stabilized (Bi-CGSTAB), approximates GMRES in terms of the number of matrix-vector products. However, for EFIE, other solvers are less robust and do not always converge with preconditioning. Even when they converge, they require more matrix-vector multiplications than GMRES. Though GMRES with no-restart brings extra CPU and memory costs with increasing number of iterations, reduction in the number of matrix-vector products significantly decreases the overall solution time due

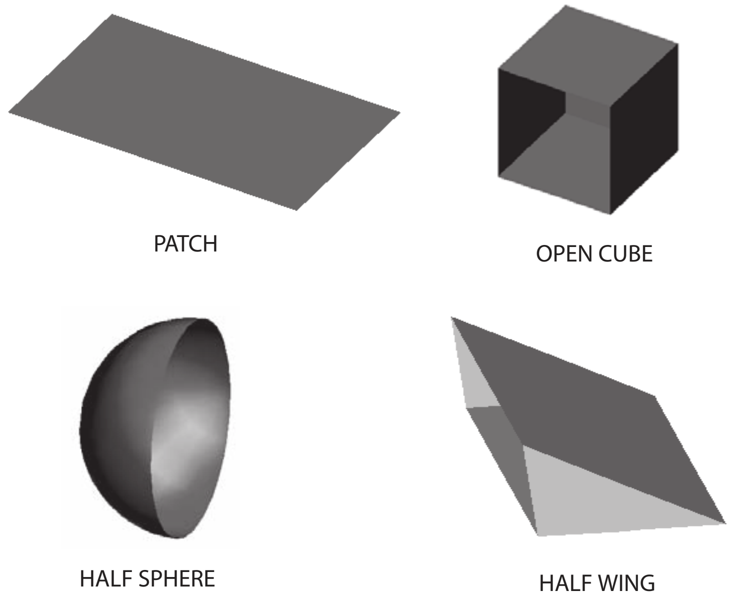


Fig. 5.1. *Open geometries.*

to the high cost of matrix-vector multiplications, particularly for large problems.

5.1. Open geometries. Figure 5.1 displays the open geometries used in the numerical experiments, i.e., a patch (P), a half sphere (HS), an open prism (OP), and an open cube (OC). These geometries are solved at various frequencies, requiring different meshes and numbers of unknowns as shown in Table 5.1. In Table 5.1 the "Size" column stands for the diameter for the spheres, and the maximum side length for others. The subsection sizes of different meshes are consistently selected as one-tenth of the wavelength. As mentioned in section 2, EFIE is the only choice for these geometries.

We compare the ILU-class preconditioners in Table 5.2. The summary of our observations are as follows:

(i) It is easily noticed that as the number of unknowns increases, ILU(0) produces highly unstable and hence useless factorizations. ILU(0) works well for small problems due to the fact that the near-field matrices used to generate the preconditioner and consequently the incomplete factors are nearly dense for such problems.

(ii) ILUT produces stable factors for all geometries except HS3. When we use 0.5 pivoting tolerance (ILUTP5) or 1.0 pivoting tolerance (ILUTP), we overcome the problem. However, for HS3, ILUTP yields a larger *condest* value and requires more iterations compared to ILUTP5. A similar situation is also encountered in some other experiments and the high value of *condest* for full pivoting is related to a poor pivoting sequence [13].

(iii) From the results, we also see a strong relationship between *condest* and the usefulness of the preconditioner. When the *condest* value is very high (i.e., higher than 10^5), the iterative method either requires too many iterations or does not converge at all.

Since ILUTP5 is robust for all our geometries, in Table 5.3 we compare it with other commonly used preconditioners. Block-Jacobi preconditioning performs poorer

TABLE 5.1
Information about the open geometries.

Problem	Frequency (MHz)	Size (λ)	n
P1	2,000	2	1,301
P2	6,000	6	12,249
P3	20,000	20	137,792
OC1	313	1.0	1,690
OC2	781	2.6	16,393
OC3	2,370	7.9	171,655
OP1	683	2.3	1,562
OP2	2,270	7.6	14,705
OP3	6,820	22.7	163,871
HS1	750	1.5	1,101
HS2	2,310	4.6	9,911
HS3	7,890	15.8	116,596

TABLE 5.2
ILU results for open geometries.

		ILU(0)		ILUT		ILUTP5		ILUTP	
Problem	n	*condest*	*iter*	*condest*	*iter*	*condest*	*iter*	*condest*	*iter*
P1	1,301	189	37	83	22	73	21	69	21
P2	12,249	60,855	228	712	42	309	39	5,606	56
P3	137,792	6.3E+09	-	1,398	82	1,350	81	2,545	78
OC1	1,690	59	76	14	37	12	35	11	33
OC2	16,393	2,154	333	52	110	48	109	44	109
OC3	171,655	9.6E+05	-	192	377	240	376	2,892	376
OP1	1,562	198	65	41	27	50	26	151	39
OP2	14,705	1.3E+05	416	161	98	164	92	151	91
OP3	163,871	5.3E+05	-	948	268	835	253	2,424	251
HS1	1,101	47	48	26	26	31	24	27	23
HS2	9,911	990	248	1,095	73	126	46	95	45
HS3	116,596	6.3E+05	-	1.5E+15	-	582	110	22,755	156

than the no-preconditioner case, so the Jacobi preconditioner is used instead. We emphasize the following observations:

(i) Although we use a robust solver, for a simple preconditioner such as Jacobi, either the number of iterations turns out to be very high or convergence is not attained in 1,500 iterations. This is in good agreement with the conclusions derived in section 4.

(ii) ILUTP5 reduces iteration numbers by an order of magnitude compared to the Jacobi preconditioner. Moreover, the iteration numbers of ILUTP5 are not extremely higher than those of LU, indicating that the ILUTP5 preconditioners provide good approximations to the near-field matrices.

(iii) We see that the setup cost of SAI is prohibitively large, proving its inappropriateness for sequential implementations. Moreover, except for OC3, ILUTP5 yields fewer number of iterations compared to SAI.

(iv) Furthermore, the iteration counts reveal that the algebraic scalability of

Table 5.3
Comparison of preconditioners for open geometries.

		LU	Jacobi		ILUTP5			SAI		
Problem	*n*	*iter*	*iter*	*time*	*iter*	*setup*	*time*	*iter*	*setup*	*time*
P1	1,301	15	201	15	21	1	3	25	101	103
P2	12,249	26	431	503	39	33	88	45	1,524	1,573
P3	137,792	53	833	16,209	81	661	2,167	92	19,955	21,384
OC1	1,690	28	224	26	35	4	9	35	569	574
OC2	16,393	97	617	854	109	141	273	114	15,040	15,167
OC3	171,655	332	-	-	376	2,243	9,833	354	207,436	213,619
OP1	1,562	18	315	39	26	2	5	48	663	668
OP2	14,705	78	991	1,894	92	97	224	173	12,301	12,524
OP3	163,871	195	-	-	253	996	6,883	396	57,606	66,093
HS1	1,101	17	187	15	24	3	5	26	165	167
HS2	9,911	38	490	748	46	107	186	61	1,712	1,813
HS3	116,596	93	1052	25,947	110	1,353	3,579	156	22,079	25,066

ILUTP5 is favorable for open geometries. For two orders of increase in the number of unknowns, the iteration numbers increase approximately four times for patch and half sphere, and 10 times for the open cube and open prism.

5.2. Closed geometries.

As mentioned in section 2, both EFIE and CFIE can be used for closed geometries. However, CFIE yields better-conditioned systems, so it is usually preferred over EFIE. Nonetheless, we will present some of the results obtained from EFIE for comparison purposes.

Figure 5.2 shows the model problems that we consider for the numerical experiments. These include two canonical geometries, i.e., a sphere (S) and a cube (C); two quasi-canonical geometries, i.e., a thin box (TB) and a wing (W); and two real-life problems, i.e., a helicopter (H) and Flamme (F), which is a stealth target [20]. Table 5.4 presents the operating frequency and the size of the geometries in terms of the wavelength. For Flamme and the helicopter, “Size” denotes the length of the objects in longitudinal direction.

Due to the well-conditioning of CFIE, ILU(0) is expected to be free from instability problems. In Table 5.5, we compare ILU(0) and ILUT by presenting the *condest* values and corresponding number of iterations for the systems obtained with CFIE. Pivoting does not change the iteration counts, and hence is not included in this case.

We note that ILU(0) and ILUT produce very similar preconditioners for CFIE. This is also observed for regular problems arising from the discretization of partial differential equations [35]. Since ILU(0) has a lower computational cost and is easier to implement compared to ILUT, we conclude that ILU(0) is the most appropriate choice among ILU-class preconditioners for CFIE.

When we use EFIE with closed geometries, it becomes even more difficult to solve the linear systems. Table 5.6 shows the *condest* values and iteration numbers for ILU-class preconditioners. W3 does not converge in 1,500 iterations, and for H2 memory limitation is exceeded during the iterations. All other problems converge with ILUTP5, but with higher iteration counts compared to open geometries.

In Table 5.7, for CFIE, ILU(0) is compared to the block-Jacobi preconditioner (where only the self-interactions of the smallest MLFMA clusters are used in the

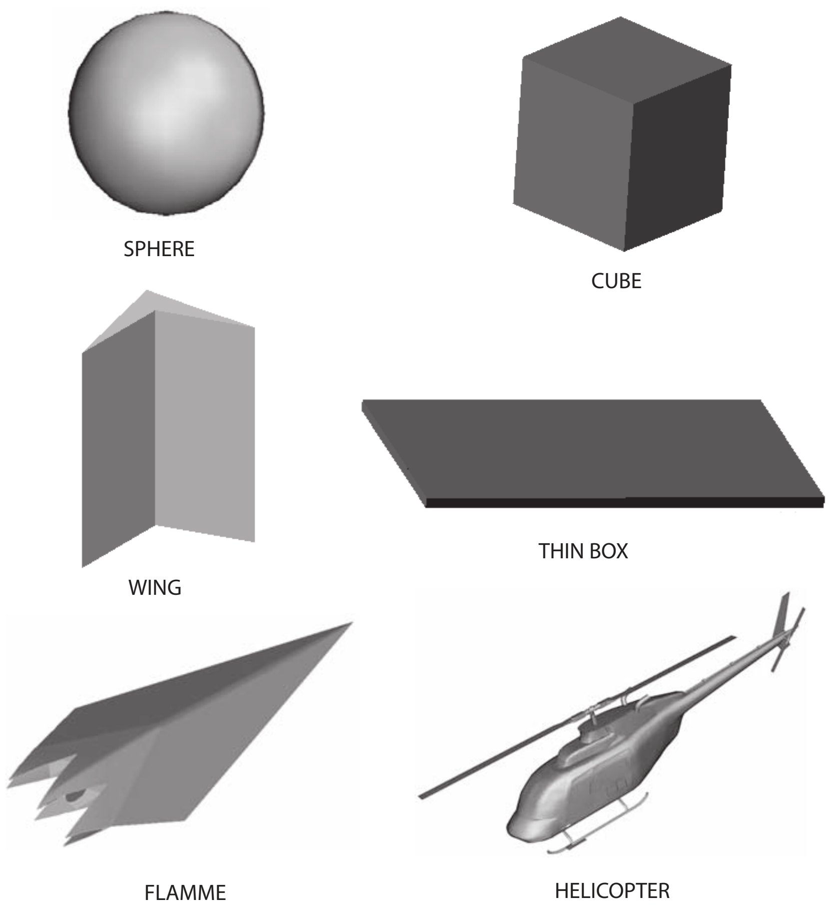


FIG. 5.2. *Closed geometries.*

near-field matrices) and the SAI preconditioner. We summarize the results as follows:

(i) For canonical geometries, compared to the block-Jacobi preconditioner, ILU(0) decreases the iteration numbers slightly. However, due to the larger setup time of ILU(0), total solution times become comparable. For multiple right-hand-side solutions, ILU(0) may be still preferable.

(ii) For quasi-canonical geometries and real-life problems, ILU(0) performs remarkably better compared to the block-Jacobi preconditioner. The number of iterations are halved for the largest problems, and more than halved for smaller sizes. Also, total solution times are significantly smaller.

(iii) Even though SAI has iteration numbers similar to ILU(0), the setup time of SAI is too large.

(iv) ILU(0) iteration numbers are very close to those of LU. Hence, among sequential-algebraic preconditioners for CFIE, ILU(0) emerges as the optimal choice for preconditioning MLFMA in the context of this study.

(v) Finally, even though the near-field matrix becomes sparser as the number of unknowns gets larger, we observe that the algebraic scalability of ILU(0) is surprisingly favorable. For the canonical geometries, as n increases two orders of magnitude, the iteration numbers only double. For quasi-canonical geometries, the iteration numbers

TABLE 5.4
Information about the closed geometries.

Problem	Frequency (MHz)	Size (λ)	n
S1	500	1	930
S2	1,500	3	8,364
S3	6,000	12	132,003
C1	210	0.7	918
C2	600	2.0	8,046
C3	2,410	8.0	131,436
W1	390	1.3	1,050
W2	1,200	4.0	10,512
W3	4,000	13.3	117,945
TB 1	188	1.9	1,650
TB 2	600	6.0	10,122
TB 3	2,400	24.0	147,180
F1	4,000	8	12,750
F2	6,000	12	28,866
F3	10,000	20	78,030
H1	222	9.6	33,423
H2	636	27.6	183,546

TABLE 5.5
ILU results for closed geometries using CFIE.

		ILU(0)		ILUT	
Problem	n	*condest*	*iter*	*condest*	*iter*
S1	930	8	13	3	13
S2	8,364	24	21	9	20
S3	132,003	108	29	108	29
C1	918	3	11	8	12
C2	8,046	9	20	24	20
C3	131,436	34	26	33	26
TB1	1,650	13	11	12	10
TB2	10,122	13	23	14	22
TB3	147,180	97	45	96	42
W1	1,050	8	10	8	9
W2	10,512	26	16	26	15
W3	117,945	83	32	82	32
F1	12,750	174	24	150	23
F2	28,866	198	34	207	33
F3	78,030	327	66	325	65
H1	33,423	6	30	6	30
H2	183,546	18	44	18	44

TABLE 5.6
ILU results for closed geometries using EFIE. "MLE" stands for "memory limitation exceeded."

		ILU(0)		ILUT		ILUTP(0.5)		ILUTP	
Problem	*n*	*condest*	*iter*	*condest*	*iter*	*condest*	*iter*	*condest*	*iter*
S1	930	65	46	61	37	16	28	20	29
S2	8,364	4.3E+04	416	3.4E+04	246	65	108	427	116
S3	132,003	1.9E+06	-	1.5E+131	-	381	572	1,346	589
C1	918	22	-	9	32	7	30	7	29
C2	8,046	3.1E+05	-	30	77	37	75	99	77
C3	131,436	2.9E+07	-	210	574	181	563	800	557
TB1	1,650	22	37	49	26	72	30	22	27
TB2	10,122	1.8E+05	-	414	169	166	151	7.5E+05	-
TB3	147,180	1.1E+14	-	48,600	1090	13,410	1084	867	709
W1	1,050	128	-	38	23	38	22	43	30
W2	10,512	6.6E+04	-	240	102	88	95	106	91
W3	117,945	4.5E+07	-	538	-	540	-	1,455	-
F1	12,750	4.7E+06	-	1,043	184	623	159	1,006	170
F2	28,866	9.5E+07	-	2,011	421	2,012	393	2,071	440
F3	78,030	1.5E+09	-	2,830	1106	3,066	1042	85,812	1131
H1	33,423	1,359	469	38	206	39	203	59	206
H2	183,546	29,184	MLE	61	MLE	71	MLE	1,799	MLE

TABLE 5.7
Comparisons of preconditioners for closed geometries using CFIE.

		LU	Block-Jacobi		ILU(0)			SAI		
Problem	*n*	*iter*	*iter*	*time*	*iter*	*setup*	*time*	*iter*	*setup*	*time*
S1	930	13	17	1	13	0	1	14	229	230
S2	8,364	20	23	23	21	6	23	21	1,453	1,474
S3	132,003	29	32	684	29	23	665	29	23,102	23,722
C1	918	11	18	0	11	1	1	12	941	941
C2	8,046	20	25	17	20	2	16	21	2,170	2,184
C3	131,436	26	28	419	26	28	485	27	25,066	25,489
TB1	1,650	9	44	3	11	2	3	20	132	135
TB2	10,122	21	60	40	23	6	22	33	10,468	10,489
TB3	147,180	37	106	1,290	45	271	1,025	64	298,479	299,301
W1	1,050	9	30	1	10	1	1	13	970	971
W2	10,512	15	39	31	16	7	21	21	14,445	14,462
W3	117,945	31	52	779	32	46	542	37	73,100	73,587
F1	12,750	21	77	89	24	11	40	40	22,930	22,976
F2	28,866	32	81	264	34	20	130	45	42,214	42,389
F3	78,030	63	115	1,096	66	43	694	76	96,369	96,369
H1	33,423	30	125	326	30	40	142	51	94,150	94,282
H2	183,546	42	106	3,081	44	145	1,739	61	234,614	236,463

increase by a factor of only 3 or 4. For Flamme, as the number of unknowns increases 15 times, the iteration number only triples. For the helicopter, the increase in the iteration number is 1.4 compared to 5.5 times increase in the number of unknowns.

6. Conclusion.

For iterative solvers, ILU-class preconditioners have been intensively studied and widely used. However, potential instability is still a shortcoming that reduces their reliability. We show that this drawback can be eliminated when it occurs, and ILU-class preconditioners can be safely applied to CEM problems employing MLFMA.

For open geometries, EFIE is the only choice of formulation. For the resulting systems, ILUT works remarkably well (about 10 times faster than Jacobi and with disproportionately lower setup time compared to SAI), but sometimes incomplete factors turn out to be unstable. We show that this situation can be handled by pivoting without incurring significant CPU costs; 0.5 pivoting tolerance gives the best results. We also show that the *condest* value is a strong indicator of the resulting preconditioner. Hence, considering the extra cost of pivoting (though not very significant), we propose the following strategy for the solution of problems involving open geometries. Before the iterations begin, compute *condest* for ILUT. If the condition estimate is not high (such as less than 10^4), use ILUT as the preconditioner. Otherwise, switch to ILUTP5. With this strategy, we have obtained robust and effective preconditioners for all our test problems.

CFIE can be used for closed geometries and yields linear systems that are well-conditioned. ILU(0) and ILUT produce very similar factorizations, and therefore cheaper ILU(0) should be preferred. With ILU(0), overall solution times have been decreased by at least one-half compared to the commonly used block-Jacobi preconditioner for real-life problems. Iteration numbers obtained with ILU(0) are very close to those of the exact solution of the near-field matrix, showing that ILU(0) is the optimum preconditioner in the context of this study. Though EFIE can also be used with closed geometries, it becomes harder to obtain fast convergence even with the exact solution of the near-field matrix.

REFERENCES

[1] G. Alléon, M. Benzi, and L. Giraud, *Sparse approximate inverse preconditioning for dense linear systems arising in computational electromagnetics*, Numer. Algorithms, 16 (1997), pp. 1–15.

[2] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith, *PETSc Users Manual*, Technical report ANL-95/11, Revision 2.1.5, Argonne National Laboratory, Argonne, IL, 2004.

[3] M. Benzi, *Preconditioning techniques for large linear systems: A survey*, J. Comput. Phys., 182 (2002), pp. 418–477.

[4] M. Benzi, D. B. Szyld, and A. van Duin, *Orderings for incomplete factorization preconditioning of nonsymmetric problems*, SIAM J. Sci. Comput., 20 (1999), pp. 1652–1670.

[5] M. Bollhöfer, *A robust and efficient ILU that incorporates the growth of the inverse triangular factors*, SIAM J. Sci. Comput., 25 (2003), pp. 86–103.

[6] M. Bollhöfer and Y. Saad, *ILUPACK—Preconditioning Software Package*, 2004; available online from http://www.math.tu-berlin.de/ilupack/.

[7] B. Carpentieri, *Sparse Preconditioners for Dense Complex Linear Systems in Electromagnetic Applications*, Ph.D. thesis, Institut National Polytechnique de Toulouse, Toulouse, France, 2002.

[8] B. Carpentieri, I. S. Duff, and L. Giraud, *Experiments with Sparse Preconditioning of Dense Problems from Electromagnetic Applications*, Technical report TR/PA/00/04, CERFACS, Toulouse, France, 1999.

[9] B. Carpentieri, I. S. Duff, and L. Giraud, *Robust preconditioning of dense problems from electromagnetics*, in Revised Papers from the Second International Conference on

Numerical Analysis and Its Applications, L. Vulkov, J. Waśniewski, and P. Yalamov, eds., Springer, London, 2000, pp. 170–178.

[10] B. Carpentieri, I. S. Duff, and L. Giraud, *Sparse pattern selection strategies for robust Frobenius-norm minimization preconditioners in electromagnetism*, Numer. Linear Algebra Appl., 7 (2000), pp. 667–685.

[11] B. Carpentieri, I. S. Duff, L. Giraud, and M. M. Made, *Sparse symmetric preconditioners for dense linear systems in electromagnetism*, Numer. Linear Algebra Appl., 11 (2004), pp. 753–771.

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

[13] E. Chow and Y. Saad, *Experimental study of ILU preconditioners for indefinite matrices*, J. Comput. Appl. Math., 86 (1997), pp. 387–414.

[14] Ö. Ergül and L. Gürel, *Improved testing of the magnetic-field integral equation*, IEEE Microwave Wireless Comp. Lett., 15 (2005), pp. 615–617.

[15] Ö. Ergül and L. Gürel, *Enhancing the accuracy of the interpolations and anterpolations in MLFMA*, IEEE Trans. Antennas Propagat. Lett., 5 (2006), pp. 467–470.

[16] Ö. Ergül and L. Gürel, *Improving the accuracy of the magnetic-field integral equation with the linear-linear basis functions*, Radio Sci., 41 (2006), article RS4004 (doi: 10.1029/2005RS003307).

[17] Ö. Ergül and L. Gürel, *Optimal interpolation of translation operator in multilevel fast multipole algorithm*, IEEE Trans. Antennas and Propagation, 54 (2006), pp. 3822–3826.

[18] Ö. Ergül and L. Gürel, *The use of curl-conforming basis functions for the magnetic-field integral equation*, IEEE Trans. Antennas and Propagation, 54 (2006), pp. 1917–1926.

[19] Ö. Ergül, *Fast Multipole Method for the Solution of Electromagnetic Scattering Problems*, Master's thesis, Bilkent University, Ankara, Turkey, 2003.

[20] L. Gürel, H. Bağci, J.-C. Castelli, A. Cheraly, and F. Tardivel, *Validation through comparison: Measurement and calculation of the bistatic radar cross section of a stealth target*, Radio Science, 38 (2003), pp. 1046–1058.

[21] L. Gürel and Ö. Ergül, *Comparisons of FMM implementations employing different formulations and iterative solvers*, in Proceedings of the IEEE Antennas and Propagation Society International Symposium, Vol. 1, 2003, pp. 19–22.

[22] L. Gürel and Ö. Ergül, *Singularity of the magnetic-field integral equation and its extraction*, IEEE Antennas Wireless Propagat. Lett., 4 (2005), pp. 229–232.

[23] L. Gürel and Ö. Ergül, *Extending the applicability of the combined-field integral equation to geometries containing open surfaces*, IEEE Antennas Wireless Propagat. Lett., 5 (2006), pp. 515–516.

[24] M. A. Heroux and J. M. Willenbring, *Trilinos Users Guide*, Technical report SAND2003-2952, Sandia National Laboratories, Albuquerque, NM, 2003.

[25] I. C. F. Ipsen and C. D. Meyer, *The idea behind Krylov methods*, Amer. Math. Monthly, 105 (1998), pp. 889–899.

[26] J. Lee, J. Zhang, and C.-C. Lu, *Incomplete LU preconditioning for large scale dense complex linear systems from electromagnetic wave scattering problems*, J. Comput. Phys., 185 (2003), pp. 158–175.

[27] J. Lee, J. Zhang, and C.-C. Lu, *Sparse inverse preconditioning of multilevel fast multipole algorithm for hybrid integral equations in electromagnetics*, IEEE Trans. Antennas and Propagation, 52 (2004), pp. 2277–2287.

[28] G. Meurant, *Computer Solution of Large Linear Systems*, Stud. Math. Appl. 28, North-Holland, Amsterdam, 1999.

[29] S. M. Rao, D. R. Wilton, and A. W. Glisson, *Electromagnetic scattering by surfaces of arbitrary shape*, IEEE Trans. Antennas and Propagation, 30 (1982), pp. 409–418.

[30] Y. Saad, *ILUT: A dual threshold incomplete LU factorization*, Numer. Linear Algebra Appl., 1 (1994), pp. 387–402.

[31] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[32] K. Sertel and J. L. Volakis, *Incomplete LU preconditioner for FMM implementation*, Microw. Opt. Tech. Lett., 28 (2000), pp. 265–267.

[33] L. N. Trefethen, *Computation of pseudospectra*, Acta Numer., 8 (1999), pp. 247–295.

[34] L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[35] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge Monogr. Appl. Comput. Math. 13, Cambridge University Press, Cambridge, UK, 2003.