Efficient Preconditioning Strategies for the Multilevel Fast Multipole Algorithm

Levent Gürel[1,2], **Tahir Malas**[1], **and Özgür Ergül**[1]

[1]Department of Electrical and Electronics Engineering, Bilkent University
TR-06800, Bilkent, Ankara, Turkey
[2]Computational Electromagnetics Research Center (BiLCEM), Bilkent University
TR-06800, Bilkent, Ankara, Turkey

Abstract— For the iterative solutions of the integral equation methods employing the multilevel fast multipole algorithm (MLFMA), effective preconditioning techniques should be developed for robustness and efficiency. Preconditioning techniques for such problems can be broadly classified as fixed preconditioners that are generated from the sparse near-field matrix and variable ones that can make use of MLFMA with the help of the flexible solvers. Among fixed preconditioners, we show that an incomplete LU preconditioner depending on threshold (ILUT) is very successful in sequential implementations, provided that pivoting is applied whenever the incomplete factors become unstable. For parallel preconditioners, sparse approximate inverses (SAI) can be used; however, they are not as successful as ILUT for the electric-field integral equation. For a remedy, we employ variable preconditioning, and we iteratively solve the near-field system in each major iteration. However, for very large systems, neither of these methods succeeds to reduce the iteration counts as desired because of the thinning of the near-field matrices for increasing problem sizes. Considering this fact, we develop a preconditioner using MLFMA, with which we solve an approximate system. Respective advantages of these different preconditioners are demonstrated on a variety of problems ranging in both geometry and size.

1. INTRODUCTION

In this paper, we consider fast iterative solutions of the integral-equation methods, which yield dense $n \times n$ linear systems in the form of $\bar{\mathbf{Z}} \cdot \mathbf{x} = \mathbf{b}$. These methods are widely used to solve scattering and radiation problems in computational electromagnetics (CEM) because of their applicability to complex geometries and high accuracy. With the help of the multilevel fast multipole algorithm (MLFMA) [1], iterative methods can be used with $O(n \log n)$ computational complexity provided that iteration counts are limited with proper preconditioners. This important problem is addressed successfully by only a few CEM researchers [2–4].

Two formulations of the integral equations are widely used for the radiation and scattering problems. The electric-field integral equation (EFIE) is used for geometries involving open surfaces. On the other hand, the combined-field integral equation (CFIE) is the appropriate formulation for targets with closed surfaces since it is free from the internal resonance problem that EFIE may suffer. Moreover, CFIE yields linear systems that are easier to solve iteratively compared to EFIE.

In each step, iterative methods require matrix-vector products with the coefficient matrix $\bar{\mathbf{Z}}$. MLFMA provides this operation to the solver by computing the near-field entries exactly and the far-field entries approximately but with controllable error. Hence, only the near-field part of the coefficient matrix $\bar{\mathbf{Z}}^{\mathrm{NF}}$ is stored and the interactions of the far-field entries are computed "on the fly". For this purpose, the computational domain is located in a cube, and then the cube is divided recursively into smaller ones. This partitioning of the computational domain defines the levels of MLFMA. The near-field part corresponds to the interactions of the smallest cubes with their neighbors and their self-interactions.

In this paper, we present our work for preconditioning of the integral-equation methods. First, we consider sequential implementations of MLFMA and concentrate on the incomplete LU (ILU) preconditioners. We show that ILU preconditioners can be very successful, provided that pivoting is applied in case of instability. Because of the difficulty of the parallelization of the ILU preconditioners, we construct an efficient sparse approximate inverse (SAI) preconditioner for the parallel implementations. For EFIE, SAI is not as successful as ILU in reducing the iteration counts; hence we use it as a preconditioner for the solution of the near-field system. Then, the solution of the near-field system is used as the preconditioner of the original system. For very large problems, on the other hand, due to their increasing sparseness, the near-field matrices do not serve as good approximations to the dense system matrices. Therefore, we use an incomplete version of MLFMA to

obtain a more powerful preconditioner. With these efforts, we are able to solve large-scale problems using a 16-processor computer in moderate solution times.

2. INCOMPLETE LU PRECONDITIONING

During factorization of sparse matrices, sparsity is lost in general. However, by sacrificing some of the nonzero elements of the exact LU factorization, incomplete LU (ILU) preconditioners can be constructed. This idea is the most established preconditioning method with many freely available implementations [9] and it has proven to be successful in the iterative solution of sparse linear systems [6].

Consider an incomplete factorization of the near-field matrix, $\bar{\mathbf{Z}}^{\mathrm{NF}} = \bar{\mathbf{L}} \cdot \bar{\mathbf{U}}$. If we retain the nonzero values of $\bar{\mathbf{L}}$ and $\bar{\mathbf{U}}$ only at the nonzero positions of $\bar{\mathbf{Z}}^{\mathrm{NF}}$, we end up with the no-fill ILU method, or ILU(0). For problems that are not far from being diagonally dominant, such as the ones resulting from CFIE, this simple idea usually works well. On the other hand, since ILU(0) does not consider the numerical values of the entries, it becomes ineffective in predicting the locations of the largest entries for matrices that are far from being diagonally dominant and highly indefinite.

For such problems, a second class of ILU preconditioners is developed based on the principle of dropping matrix elements depending on their magnitudes. Among such methods, ILUT (τ, p) [6] drops matrix elements that are smaller than τ times the 2-norm of the current row and of all the remaining entries at most p largest ones are kept. ILUT is known to yield more accurate and stable factorizations compared to ILU(0) with the same amount of fill-in [7]. However, for some cases, although the factorization terminates normally, the incomplete factors sometimes turn out to be unstable. When the problem is related to the small pivots, this problem can be avoided using partial pivoting as in the complete factorization case.

Table 1: ILU results for CFIE.

Problem	Number of unknowns	LU	No PC		BJ		ILU(0)		
		Iter	Iter	Time	Iter	Time	Iter	Setup	Time
Sphere	132,003	29	49	1,103	32	684	29	23	665
Thin box	147,180	37	158	1,965	106	1,290	45	271	1,025
Wing	117,945	31	86	1,110	52	779	32	46	542
Flamme	78,030	63	229	2,138	115	1,096	66	43	694
Helicopter	183,546	42	253	7,338	106	3,081	44	145	1,739

Table 2: ILU results for EFIE.

Problem	Number of unknowns	LU	Jacobi		ILUTP		
		Iter	Iter	Time	Iter	Setup	Time
Patch	137,792	53	833	16,209	81	661	2,167
Open cube	171,655	332	-	-	376	2,243	9,833
Open prism	163,871	195	-	-	253	996	6,883
Half sphere	116,596	93	1,052	25,947	110	1,353	3,579

We realize that ILU(0) and ILUT works very well for the CFIE and EFIE systems, respectively. In Table 1 and Table 2, we show the number of iteration counts and solution times with GMRES no-restart on a solver containing Opteron 244 processors. Zero initial guess is used and 10^{-6} residual error is aimed in maximum 1500 iterations. We compare the ILU preconditioners with the following ones.

- **LU:** Denotes the exact solution of the near-field matrix, which is used as a benchmark preconditioner. Due to its excessive computational requirements, LU is not a practical preconditioner; it is presented merely for comparison of iteration numbers.
- **Block-Jacobi (BJ):** The blocks in this widely used preconditioner correspond to self-interactions of the smallest clusters in MLFMA. Exact inverse of each small block is obtained by LU decomposition. There are n such blocks with fixed sizes; hence, the cost of this preconditioner is $O(n)$.

- **Jacobi:** Contains only the diagonal of the near-field matrix. In Table 2, this preconditioner is included instead of BJ, since the latter performs poorer for EFIE.

From Table 1, we recognize that particularly for real-life problems, such as Flamme [8] and the helicopter, ILU(0) decreases the iteration counts and the solution times significantly. Moreover, the iteration counts turn out to be very close to that of LU, signaling the optimality of ILU(0). For open geometries presented in Table 2, first we note that for EFIE systems, even with a robust solver, either the numbers of iterations are very high or no convergence is seen at all. ILUTP refers to ILUT preconditioner with pivoting, which is required for the robustness of ILUT. The parameters of ILUTP are chosen so that the memory requirement does not exceed that of the near-field matrix. From the table one can see that ILUTP reduces the iteration numbers by an order of magnitude compared to Jacobi when the latter succeeds to converge.

3. SPARSE APPROXIMATE INVERSES FOR PARALLEL COMPUTING

The main disadvantage of the ILU preconditioners is the lack of parallelizability of the factorization and the application phases. To overcome this limitation, SAI preconditioners are developed, which are based on approximating the inverse of the matrix directly [3, 4]. For this purpose, an approximate inverse $\bar{\mathbf{M}} \approx \left(\bar{\mathbf{Z}}^{\mathrm{NF}}\right)^{-1}$ is explicitly constructed and stored. Then, application of the preconditioner is carried out by a sparse matrix-vector multiplication, whose parallelization can be performed efficiently.

The results in Table 3, obtained on a 16-processor parallel computer, indicate that SAI is much more successful in reducing the iteration counts and solution times compared to BJ in closed geometries. Even though total solution times are higher, only a small number of right-hand-side (RHS) vectors suffice to compensate setup time of SAI. Hence, for multiple-illumination problems involving real-life closed targets, SAI should be preferred over BJ.

Table 3: Comparison of parallel SAI and BJ.

CFIE		BJ				SAI				
Problem	Number of unknowns	Iter	Setup	Solution Time	Total Time	Iter	Setup	Solution Time	Total Time	RHS
Sphere	829,881	40	0.18	882	882	37	532	838	1,370	12
Thin box	147,180	104	0.14	352	352	63	304	218	522	3
Wing	117,945	53	0.05	98	98	38	192	73	264	8
Flamme	895,407	185	0.68	8,096	8,097	151	3,239	6,588	9,827	3
Helicopter	739,404	107	0.30	3,698	3,699	77	1,840	2,739	4,578	2

However, SAI preconditioners are not as successful as ILU preconditioners for EFIE. For this purpose, we consider using the SAI preconditioner for the iterative solution of the near-field system, and then the near-field solution is used as a preconditioner to the original system. For the iterative near-field preconditioner, the preconditioning operation is not fixed. Hence, a flexible solver should be used for the system matrix, allowing the preconditining operation change from iteration to iteration. We use FGMRES [6] in our experiments. For the near-field system solution, we use GMRES because of its ability to reduce the residual error quickly on the first iterations.

This scheme yields a forward-type preconditioner such as the ILU preconditioner. The difference is that, in ILU preconditioning, the preconditioner is already in factorized form, i.e., $\bar{\mathbf{M}} = \bar{\mathbf{L}} \cdot \bar{\mathbf{U}} \approx \bar{\mathbf{Z}}^{\mathrm{NF}}$, and for a given vector $\mathbf{y}$, the system $\bar{\mathbf{M}} \cdot \mathbf{x} = \mathbf{y}$ is solved by using backward and forward solves. On the other hand, for the preconditioning scheme described, which we call the iterative near-field preconditioner, the preconditioner is the exact near-field matrix, i.e., $\bar{\mathbf{M}} = \bar{\mathbf{Z}}^{\mathrm{NF}}$, but we approximately solve the system $\bar{\mathbf{M}} \cdot \mathbf{x} = \mathbf{y}$ by an iterative method.

In Table 4, we compare LU, SAI, and iterative near-field (NF/SAI) preconditioners. With NF/SAI, we achieve very close iteration counts to that of LU. We note that, SAI-preconditioned GMRES accelerates the convergence of the near-field system dramatically. We set the maximum number of iterations to 5, and this suffices for an effective preconditioner.

Table 4: Comparison of LU, SAI, and NF/SAI for EFIE. 'MLE' stands for 'memory limitation is exceeded'.

EFIE		LU	SAI Setup	SAI		NF/SAI	
Problem	Number of unknowns	Iter		Iter	Solution Time	Iter	Solution Time
Patch	137,792	53	52	91	336	59	253
	719,000	MLE	214	190	4,091	141	3,369
Open prism	127,925	112	156	172	628	120	520
	409,514	MLE	336	389	4,601	209	3,476
Half sphere	9,911	38	7	60	24	40	17
	116,596	93	77	156	510	103	383
Reflector antenna	356,439	MLE	952	125	878	71	646

4. USING MLFMA FOR EFFECTIVE PRECONDITIONING

For problems involving large numbers of unknowns, near-field matrices become increasingly sparser and, beyond some level, it becomes nearly impossible to achieve low iteration counts with the preconditioners obtained from the near-field matrices. Therefore, for effective preconditioning of very large problems, we need more information than that is provided by the near-field matrix.

Table 5: Comparison of incomplete MLFMA preconditioner with NF/SAI and SAI.

EFIE		SAI setup	LU	SAI		NF/SAI			IMLFMA/SAI		
Problem	Number of unknwns		Iter	iter	Solution Time	Iter		Solution Time	Iter		Solution Time
						outer	Inner		outer	Inner	
Patch	137,792	52	53	91	336	59	176	253	14	134	172
	719,000	214	-	190	4,091	141	421	3,369	32	313	2,152
Open prism	127,925	156	-	172	628	120	360	520	30	300	442
	409,514	336	-	389	4,601	209	627	3,476	54	540	2,561
Half sphere	9,911	7	38	60	24	40	120	17	8	76	14
	116,596	77	93	156	510	103	309	383	21	208	307
Reflector Antenna	356,439	952	-	125	878	71	213	646	17	165	478

Table 6: Comparison of incomplete MLFMA preconditioner with others for CFIE.

CFIE		Block Jacobi			SAI			IMLFMA/SAI		
Problem	Number of unknwns	Setup	Iter	Solution Time	Setup	Iter	Solution Time	Iter		Solution Time
								Outer	Inner	
Thin box	147,180	0.14	104	352	304	63	218	11	103	120
Wing	117,945	0.05	53	98	191.536	38	73	8	49	62
Helicopter	739,404	0.30	107	3,698	1,840	77	2,739	13	118	1,161
Flamme	895,407	0.68	185	8,096	3,239	151	6,588	26	255	2,491

This information can only be provided by MLFMA in the form of matrix-vector products. Following this idea, in each step of the iterative solver, we solve a nearby system using MLFMA. This preconditioning scheme yields a nesting of the solvers. The inner solver is used for preconditioning purpose, hence we need to solve only an approximate system. In order to perform the matrix-vector multiplication faster, we lower the sampling densities of MLFMA compared to the high-accuracy MLFMA. In this way, it is possible to reduce the cost of the preconditioning operation, and thereby the total solution time significantly.

In Table 5 and Table 6, we present the total solution times of aforementioned parallel preconditioners and the preconditioner obtained from approximate MLFMA, for some of the large problems shown in Table 3 and Table 4. IMLFMA/SAI is the preconditioner obtained with an incomplete MLFMA. As can be seen from the tables, using MLFMA for preconditioning purposes reduces the

total solution time so significantly that even the solutions of large-scale problems can be obtained in less than an hour.

5. CONCLUSION

In this work we show that, for the success of large-scale iterative solutions of integral equation methods, the key is effective preconditioning. Even with systems resulting from CFIE, when the size of the problem gets larger or the target geometry is complex as in real-life problems, preconditioning is required to achieve solutions with less time. On the other hand, severely ill-conditioned EFIE systems sometimes do not converge even with robust solvers such as the no-restart GMRES.

Our experiments with the sequential programs reveal that ILU preconditioners can be safely applied to integral equation methods, provided that pivoting is applied to ILUT for EFIE systems. A comparison with the exact solution of the near-field matrix shows the near optimality of them. Furthermore, ILU is the most established preconditioning technique, whose implementations are available in solver packages, such as PETSc [9]. Hence, we strongly recommend their use for sequential problems.

For larger problems, on the other hand, one should resort to SAI preconditioners. Though SAI works well up to certain problem sizes, more effective preconditioning strategies, such as the use of iterative solution of the near-field system or the use of incomplete MLFMA for preconditioning help to solve even larger systems with low-memory and solution-time requirements.

ACKNOWLEDGMENT

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.

REFERENCES

1. Lu, C.-C. and W. C. Chew, "Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects," *IEEE Trans. Antennas Propagat.*, Vol. 45, No. 10, 1488–1493, 1997.
2. Lee, J., J. Zhang, and C.-C. Lu, "Incomplete LU preconditioning for large scale dense complex linear systems from electromagnetic wave scattering problems," *J. Comput. Phys.*, Vol. 185, 158–175, 2003.
3. Lee, J., 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*, Vol. 52, No. 9, 158–175, 2004.
4. Carpentieri, B., I. S. Duff, L. Giraud, and G. Sylvand, "Combining fast multipole techniques and an approximate inverse preconditioner for large electromagnetism calculations," *SIAM J. Sci. Comput.*, Vol. 27, No. 3, 774–792, 2005.
5. Benzi, M., "Preconditioning techniques for large linear systems: a survey," *J. Comput. Phys.*, Vol. 182, No. 2, 418–477, 2002.
6. Saad, Y., *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, ABJ, 2003.
7. Chow, E. and Y. Saad, "Experimental study of ILU preconditioners for indefinite matrices," *J. Comput. Appl. Math.*, Vol. 86, No. 2, 387–414, 1997.
8. Gürel, L., H. Bağcı, 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*, Vol. 38, No. 3, 1046–1058, 2003.
9. Balay, S., W. D. Gropp, L. C. McInnes, and B. F. Smith, "PETSc users manual," Tech. Report ANL-95/11 Revision 2.1.5, Argonne National Laboratory, 2004.