

EFFECTIVE PRECONDITIONERS FOR ITERATIVE
SOLUTIONS OF LARGE-SCALE
SURFACE-INTEGRAL-EQUATION PROBLEMS

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Tahir Malas

March 2010

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Levent Gürel (Supervisor)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Cevdet Aykanat

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Ayhan Altıntaş

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Vakur Ertürk

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assist. Prof. Dr. Ergün Şimşek

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

EFFECTIVE PRECONDITIONERS FOR ITERATIVE SOLUTIONS OF LARGE-SCALE SURFACE-INTEGRAL-EQUATION PROBLEMS

Tahir Malas

Ph.D. in Electrical and Electronics Engineering

Supervisor: Prof. Dr. Levent Gürel

March 2010

A popular method to study electromagnetic scattering and radiation of three-dimensional electromagnetics problems is to solve discretized surface integral equations, which give rise to dense linear systems. Iterative solution of such linear systems using Krylov subspace iterative methods and the multilevel fast multipole algorithm (MLFMA) has been a very attractive approach for large problems because of the reduced complexity of the solution. This scheme works well, however, only if the number of iterations required for convergence of the iterative solver is not too high. Unfortunately, this is not the case for many practical problems. In particular, discretizations of open-surface problems and complex real-life targets yield ill-conditioned linear systems. The iterative solutions of such problems are not tractable without preconditioners, which can be roughly defined as easily invertible approximations of the system matrices.

In this dissertation, we present our efforts to design effective preconditioners for large-scale surface-integral-equation problems. We first address incomplete LU (ILU) preconditioning, which is the most commonly used and well-established

preconditioning method. We show how to use these preconditioners in a black-box form and safe manner. Despite their important advantages, ILU preconditioners are inherently sequential. Hence, for parallel solutions, a sparse-approximate-inverse (SAI) preconditioner has been developed. We propose a novel load-balancing scheme for SAI, which is crucial for parallel scalability. Then, we improve the performance of the SAI preconditioner by using it for the iterative solution of the near-field matrix system, which is used to precondition the dense linear system in an inner-outer solution scheme. The last preconditioner we develop for perfectly-electric-conductor (PEC) problems uses the same inner-outer solution scheme, but employs an approximate version of MLFMA for inner solutions. In this way, we succeed to solve many complex real-life problems including helicopters and metamaterial structures with moderate iteration counts and short solution times. Finally, we consider preconditioning of linear systems obtained from the discretization of dielectric problems. Unlike the PEC case, those linear systems are in a partitioned structure. We exploit the partitioned structure for preconditioning by employing Schur complement reduction. In this way, we develop effective preconditioners, which render the solution of difficult real-life problems solvable, such as dielectric photonic crystals.

Keywords: Preconditioning, incomplete-LU preconditioners, sparse-approximate-inverse preconditioners, flexible solvers, variable preconditioning, computational electromagnetics, surface integral equations, multilevel fast multipole algorithm, electromagnetic scattering, parallel computing.

ÖZET

BÜYÜK ÖLÇEKLİ YÜZEY İNTEGRAL DENKLEMİ PROBLEMLERİNİN İTERATİF ÇÖZÜMLERİ İÇİN ETKİN ÖNİYİLEŞTİRİCİLER

Tahir Malas

Elektrik ve Elektronik Mühendisliği Bölümü Doktora

Tez Yöneticisi: Prof. Dr. Levent Gürel

Mart 2010

Üç boyutlu elektromanyetik saçılım ve ısıtım problemlerinin çalışılmasında yoğun doğrusal sistemlere yol açan ayrıklaştırılmış yüzey integral denklemlerini çözmek yaygın bir yöntemdir. Çözümün karmaşıklığının azalmasından dolayı, bu doğrusal denklemlerin Krylov altuzayı ve çok seviyeli hızlı çokkutup (ÇSHÇY) yöntemleri kullanılarak iteratif çözümü son derece çekici hale gelmiştir. Fakat bu yaklaşım sadece yakınsama için gereken iterasyon sayısı aşırı derecede yüksek olmadığı sürece işe yaramaktadır. Maalesef, pek çok pratik durumda bu geçerli olmamaktadır. Özellikle, açık yüzey ve karmaşık gerçek hayat problemleri kötü koşullu doğrusal sistemlere yol açmaktadır. Bu tarz problemlerin iteratif çözümleri, kabaca sistem matrislerine yaklaşan tersi alınabilir matrisler olarak tanımlanan öniyeleştiriciler olmadan mümkün olmamaktadır.

Bu doktora tezinde, büyük ölçekli yüzey integral denklemi problemleri için geliştirdiğimiz etkin öniyeleştiricileri sunmaktayız. İlk olarak, en yaygın ve oturmuş bir öniyeleştirme yöntemi olan eksik LU (ELU) öniyeleştirmesini ele aldık. Bu öniyeleştiricilerin nasıl bir kara kutu formunda ve güvenli olarak kullanılabileceğini gösterdik. Önemli avantajlarına rağmen, ELU öniyeleştiricileri

temel olarak sıralı bir yapıda oldukları için, paralel çözümlerde kullanılmak üzere bir seyrek yaklaşık ters (SYT) öniyileştiricisi geliştirdik. Ayrıca, paralel ölçeklenebilirlik için önemli olan özgün bir yük dengeleme yöntemi önerdük. Daha sonra SYT öniyileştiricilerini, yoğun sistemi bir iç-dış çözümü şeklinde öniyileştiren yakın alan matris sisteminin iteratif çözümünde kullanarak geliştirdik. Mükemmel iletkenler için geliştirdiğimiz son öniyileştirici, benzer bir iç-dış çözümü kullanmakta, ama iç çözümler için ÇSHY'nin yaklaşık bir versiyonunu kullanmaktadır. Bu yolla, helikopterler ve metamateryaller içeren çok sayıda karmaşık gerçek hayat problemini makul iterasyon sayılarında çözmeyi başardık.

Son olarak, dielektrik problemlerinin ayrıklaştırılmasından elde edilen doğrusal sistemlerin öniyileştirilmelerini hedefledik. Mükemmel iletkenlerden farklı olarak, bu sistemler bölünmüş yapıdadırlar. Schur tümleyenine indirgemeyle bu bölünmüş yapıyı öniyileştirme için kullandık. Bu yaklaşımla, dielektrik fotonik kristaller gibi, çözümü zor gerçek hayat problemlerinin makul sürelerde çözümünü mümkün kılan etkin öniyileştiricilerin geliştirilmesi mümkün olmuştur.

Anahtar Kelimeler: Öniyileştirme, eksik LU öniyileştiricileri, seyrek yaklaşık ters öniyileştiricileri, esnek çözümler, değişken öniyileştirme, bilişimsel elektromanyetik, yüzey integral denklemleri, çok seviyeli hızlı çokkutup yöntemi, elektromanyetik saçılım, paralel hesaplama.

ACKNOWLEDGMENTS

I would like to express my eternal gratitude to my supervisor, Prof. Levent Gürel. I have learned fundamental and very valuable skills from him. I have no doubt that these will assist me a lot in my future academic career. I also would like to thank him for his guidance and support throughout my Ph.D. study.

I would like to thank the committee members, Prof. Cevdet Aykanat, Prof. Ayhan Altıntaş, Assoc. Prof. Vakur Ertürk, and Assist. Prof. Ergün Şimşek for reading and commenting on this dissertation.

I was lucky to work with BiLCEM researchers, Dr. Özgür Ergül, Alp Manyas, Burak Tiryaki, and Seçil Kılınç. I thank them for their collaboration and friendship.

My final gratitude is to my wife and family, who gave me endless support in my Ph.D. study.

This work was supported by the Scientific and Technical Research Council of Turkey (TÜBİTAK) through a Ph. D. scholarship.

Contents

1	Preliminaries	1
1.2	Introduction	2
1.3	Notation	3
1.4	Surface Integral-Equation Formulations	4
1.4.1	The Electric-Field Integral Equation (EFIE)	4
1.4.2	The Magnetic-Field Integral Equation (MFIE)	5
1.4.3	The Combined-Field Integral Equation (CFIE)	5
1.5	Discretization of the Surface Formulations	6
1.5.1	Method of Moments	6
1.5.2	Discretization of EFIE	7
1.5.3	Discretization of MFIE	9
1.5.4	Discretization of CFIE	10
1.5.5	Computation of the RHS Vectors	10
1.6	The Multilevel Fast Multipole Algorithm (MLFMA)	10

1.6.1	Clustering	11
1.6.2	Factorization of the Green's Function	13
1.6.3	Far-Field Interactions	15
1.7	Iterative Solvers	17
1.7.1	Krylov Subspace Methods	18
1.7.2	The generalized minimal residual method (GMRES)	21
1.7.3	Convergence of GMRES	22
1.8	Preconditioning	23
1.9	Spectral Analysis of the Surface Formulations	25
1.10	Contributions	26
1.11	Computational Resources	29
1.12	Organization	30
2	Incomplete-LU (ILU) Preconditioners	32
2.1	Introduction	32
2.2	Preconditioners based on Incomplete LU Factorization	35
2.3	Improving Stability of ILU Preconditioners	37
2.4	Numerical Results	39
2.4.1	Open Geometries	40
2.4.2	Closed Geometries	44

2.5	Conclusion	49
3	Sparse-Approximate-Inverse (SAI) Preconditioners	51
3.1	Introduction	51
3.2	Brief Review of SAI	54
3.2.1	Methods Derived from the Frobenius Norm Minimization	54
3.3	Parallel Implementation Details	56
3.3.1	Pattern Selection and Filtering	58
3.3.2	Communication Phase and Enlarging the Local Submatrix	61
3.3.3	Load Balancing of SAI	63
3.3.4	Construction of the Preconditioner	65
3.3.5	Application of the Preconditioner	65
3.4	Results	66
3.4.1	Parallel Performance of the Construction Phase	67
3.4.2	EFIE Results	68
3.4.3	CFIE Results	72
3.4.4	Solutions of Metamaterial Structures	74
3.5	Conclusion	76
4	The Iterative Near-Field (INF) Preconditioner	78
4.1	Introduction	78

4.2	Near-Field versus Full-Matrix Preconditioners	80
4.3	The Iterative Near-Field Preconditioner	81
4.4	Numerical Results	82
4.4.1	EFIE Results	83
4.4.2	CFIE Results	86
4.5	Conclusion	90
5	Preconditioners Utilizing More Than the Near-Field Matrix	93
5.1	Introduction	93
5.2	The Approximate Multilevel Fast Multipole Algorithm	96
5.3	Iterative Preconditioning Based on the Approximate MLFMA	98
5.3.1	Preconditioning Operator	99
5.3.2	Inner Solver and the Secondary Preconditioner	99
5.3.3	Inner Stopping Criteria	99
5.4	Numerical Results	100
5.4.1	EFIE Results	101
5.4.2	CFIE Results	102
5.5	Conclusion	103
6	Schur Complement Preconditioners For Dielectric Problems	105
6.1	Introduction	106

6.2	Surface Integral-Equation Methods for Dielectric Problems	112
6.2.1	The Combined Tangential Formulation (CTF)	112
6.2.2	The Combined Normal Formulation (CNF)	114
6.2.3	The Modified Normal Müller Formulation (MNMF)	115
6.2.4	The Electric and Magnetic Current Combined-Field For- mulation (JMCFIE)	115
6.2.5	Comparison of the Integral-Equation Formulations for Di- electrics	116
6.3	Discretization of the Surface Formulations of Dielectric Problems .	118
6.4	Preconditioning with Schur Complement Reduction	119
6.4.1	Schur Complement Reduction	120
6.5	Approximate Schur Complement Preconditioners	121
6.5.1	Approximations of the Solutions Involving the (1, 1) Par- tition and the Schur complement	123
6.6	Iterative Schur Complement Preconditioners	131
6.6.1	Iterative Solutions Involving the (1, 1) Partition	133
6.6.2	Iterative Solutions Involving the Schur Complement	135
6.6.3	Stopping Criteria for Inner Solutions	138
6.7	Numerical Results	139
6.7.1	The Sphere Problem	141
6.7.2	The Lens Problem	149

6.7.3	Periodic Slabs (PS)	153
6.7.4	The Perforated-Waveguide Problem	158
6.7.5	Accuracy of the Solutions of the Photonic Crystal Waveguide	162
6.8	Conclusions	163
7	Conclusions and Future Work	166

List of Figures

1.1	Illustration of the oct-tree partitioning of the computational domain in MLFMA.	12
1.2	(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.	13
1.3	Sparse near-field matrices for (a) $N = 930$, (b) $N = 1,302$, and (c) $N = 3,723$	14
1.4	Comparison of the direct and iterative solvers for work performed and acquired error levels.	18
1.5	The GMRES method. ϵ is a predetermined stopping threshold.	22
1.6	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.	27
2.1	Open geometries used to compare ILU preconditioners.	41
2.2	Closed geometries used to compare ILU preconditioners.	45

3.1	Reduction of a 10×10 matrix for the generation of the 4th row of SAI.	58
3.2	Filtering algorithm.	59
3.3	The pseudocode that finds the rows to be sent by the process P_k	61
3.4	The pseudocode that finds the column indices of the sparse rows to be received by the process P_k	62
3.5	The pseudocode that exchanges the sparse rows.	62
3.6	Redistribution of the SAI rows according to the near-field partitioning. R_k^{NF} and R_k^{SAI} denote the row indices of process k with respect to the near-field and SAI partitionings, respectively.	64
3.7	The pseudocode for the sparse matrix-vector multiplication used for right preconditioning. \mathbf{IA} , \mathbf{JA} , and \mathbf{VA} are respectively row-index, column-index, and value arrays of $\overline{\mathbf{M}}_k$, which is stored in CSR format.	66
3.8	Speedup curves for the patch, half sphere, and Flamme problems.	67
3.9	Load imbalance of the Flamme problem for (a) unbalanced and (b) balanced cases.	68
3.10	Open geometries used in EFIE problems.	69
3.11	Approximate eigenvalues of the RA4 problem on the complex plane.	72
3.12	Closed-surface geometries used in CFIE problems.	73
3.13	Unit cells that are used to construct various metamaterial walls: (a) SRR, (b) thin wires, and (c) a combination of SRR and thin wires.	75

3.14	Processing time including the iterative solution and the setup of the preconditioner for the $18 \times 11 \times 4$ SRR wall. “NP” represents the no-preconditioner case.	77
4.1	Nested solvers for iterative near-field preconditioning.	81
4.2	Closed-surface geometries formulated with CFIE.	87
4.3	Total solution times of the helicopter problem. The lines fit the solution times in a least-squares sense.	90
5.1	Inner-outer solution scheme that use an approximate version of MLFMA.	95
6.1	Illustration of the solution of dielectric problems using MLFMA and approximate Schur complement preconditioners. As explained in Section 6.5, $\overline{\mathbf{M}}_{11}$ is an approximate inverse for $\overline{\mathbf{A}}_{11}^{NF}$ and $\overline{\mathbf{M}}_S$ is an approximate inverse for the Schur complement. \mathbf{w}'_1 and \mathbf{w}'_2 take different forms depending on the type of the preconditioner. .	110
6.2	Illustration of the solution of dielectric problems using MLFMA and iterative Schur complement preconditioners. Here, instead of direct solves, iterative solutions are employed to find approximate solutions to reduced systems. As explained in Section 6.6, $\widetilde{\mathbf{S}}$ is an approximation to the Schur complement and \mathbf{w}'_1 and \mathbf{w}'_2 take different forms depending on the type of the preconditioner. . . .	111
6.3	Eigenvalues of $\overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{11}^{NF}$ for different formulations and increasing dielectric constants of 4, 8, and 12.	125

6.4	Incomplete matrix-matrix multiplication of $\overline{\mathbf{C}} = \overline{\mathbf{D}} \cdot \overline{\mathbf{E}}$, where $\overline{\mathbf{C}}$, $\overline{\mathbf{D}}$, and $\overline{\mathbf{E}}$ are block near-field matrices with the same sparsity pattern. $\overline{\mathbf{C}}_{ij}$ denotes the block of the near-field matrix $\overline{\mathbf{C}}$ that corresponds to the interaction of cluster i with cluster j . $\mathcal{N}(i)$ denotes the clusters that are in the near-field zone of cluster i	129
6.5	Eigenvalues of preconditioned Schur complement $\overline{\mathbf{S}}$ for increasing dielectric constants of 4, 8, and 12. CTF is preconditioned with $\overline{\mathbf{M}}_{MF}$, whereas $\overline{\mathbf{M}}_{BD}$ is used as the preconditioner for the other formulations.	130
6.6	Eigenvalues of $\overline{\mathbf{M}}_{22} \cdot \overline{\mathbf{S}}$ for different formulations and increasing dielectric constants of 4, 8, and 12.	131
6.7	Eigenvalues of $\overline{\mathbf{M}}_S \cdot \overline{\mathbf{S}}$ for different formulations and increasing dielectric constants of 4, 8, and 12.	132
6.8	Comparison of iterative solutions of (6.33) without a preconditioner and using $\overline{\mathbf{M}}_{11}$ for a periodic-slabs problem involving 262,920 unknowns.	134
6.9	Comparison of iterative solutions of (6.33) without a preconditioner and using $\overline{\mathbf{M}}_{11}$ for perforated waveguide involving 162,420 unknowns.	135
6.10	Iterative solutions of (6.46) with various preconditioners for a periodic-slabs problem involving 262,920 unknowns.	137
6.11	Iterative solutions of (6.46) with various preconditioners for a perforated waveguide involving 162,420 unknowns.	138
6.12	Comparisons of iteration counts for the sphere problem.	148

6.13	Illustration of the filtering capability of the periodic slabs problem. At 250 MHz and 350 MHz, the power transmission is unity in the transmission region on the left-hand side of the structure. On the other hand, a shadowing occurs at 300 MHz and the device becomes opaque.	155
6.14	(a) A perforated photonic crystal waveguide. (b) Near-zone magnetic fields of the problem when illuminated by a Hertzian dipole.	159
6.15	Near-zone magnetic fields for a perforated waveguide (PW3 in Table 6.19) illuminated by a Hertzian dipole.	162
6.16	Near-zone magnetic fields for a perforated PhC waveguide involving 7×10 holes illuminated by a Hertzian dipole. Solutions are obtained with (a) CTF and $\lambda/20$ triangulation, (b) JMCFIE and $\lambda/20$ triangulation, (c) CTF and $\lambda/40$ triangulation, and (d) JMCFIE and $\lambda/40$ triangulation.	163
7.1	Decision chart for the selection of preconditioners for PEC problems.	168

List of Tables

2.1	Information about the open geometries used to compare ILU preconditioners.	42
2.2	ILU results for open geometries.	42
2.3	Comparison of ILU preconditioners for open geometries.	43
2.4	Information about the closed geometries used to compare ILU preconditioners.	44
2.5	ILU results for closed geometries using CFIE.	46
2.6	ILU results for closed geometries using EFIE. “MLE” stands for “Memory Limitation Exceeded.”	47
2.7	Comparisons of ILU preconditioners for closed geometries using CFIE.	48
3.1	Quantitative features of the open geometries.	69
3.2	The solutions with no preconditioning for open geometries formulated by EFIE.	70
3.3	Comparison of SAI preconditioners for open geometries formulated by EFIE.	71

3.4	Quantitative features of the closed geometries.	73
3.5	The solutions with BDP for closed-surface problems formulated by CFIE.	73
3.6	Comparison of SAI preconditioners for closed-surface problems formulated by CFIE.	74
4.1	Experimental results for comparing the SAI and INF precondi- tioners to NF-LU.	84
4.2	Quantitative features of the open-surface geometries used for the numerical experiments.	85
4.3	Experimental results for comparing the SAI and INF precondi- tioners.	86
4.4	Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.	87
4.5	Quantitative features of the closed-surface geometries used for the numerical experiments.	88
4.6	Experimental results for comparing the INF preconditioner with DP, BDP, and the SAI preconditioner for closed-surface problems.	89
4.7	Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.	90
5.1	Electromagnetics problems involving open metallic objects.	101
5.2	Processing time (seconds) and the number of iterations* for the solution of electromagnetics problems involving open metallic ob- jects.	102

5.3	Electromagnetics problems involving closed metallic objects. . . .	103
5.4	Processing time (seconds) and the number of iterations* for the solution of electromagnetics problems involving closed metallic objects.	103
6.1	Number of iterations for the solution of $\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}'_1$ to reduce the residual error by 10^{-6}	134
6.2	Number of iterations for the solution of $\widetilde{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}'_2$ to reduce the residual error by 10^{-6}	137
6.3	Number of iterations for the sphere problem involving 65,724 unknowns using the iterative Schur complement preconditioners with varying inner tolerances. For 10^{-6} inner tolerance, extra inner solves for the Schur complement and for the RHS of ISP are used.	139
6.4	Salient features of the sphere problems investigated in this study.	141
6.5	Setup times (in minutes) of ILU-type preconditioners and SAIs of $\overline{\mathbf{A}}_{11}^{NF}$ and the Schur complement matrix $\overline{\mathbf{S}}$ for the sphere problem.	142
6.6	Performances of the 4PBDP and ILU(0) preconditioners and No PC on the sphere problem.	144
6.7	Performances of the Approximate Schur complement preconditioners on the sphere problem.	145
6.8	Performances of the Schur complement preconditioners on the sphere problem.	147
6.9	Memory requirements of the Schur complement preconditioners, MLFMA, and solutions with the no-restart GMRES for the sphere problems.	149

6.10	Salient features of the lens problems investigated in this study. . .	150
6.11	Performances of the 4PBDP and ILU(0) preconditioners and No PC on the lens problems.	151
6.12	Performances of the Approximate Schur complement precondi- tioners on the lens problems.	152
6.13	Number of iterations obtained with NF-LU for the lens problems.	153
6.14	Memory requirements of the Schur complement preconditioners, MLFMA, and solutions with the no-restart GMRES for the lens problems.	154
6.15	Salient features of the periodic-slab problems investigated in this study.	155
6.16	Comparison of ILU and simple preconditioners for the periodic slab problems.	156
6.17	Comparison of the Schur complement preconditioners for the PS problems.	157
6.18	Memory requirements of the Schur complement precondition- ers, MLFMA, and solutions with the no-restart GMRES for the periodic-slab problems.	159
6.19	Salient features of the perforated waveguide (PW) investigated in this study.	160
6.20	Comparison of the preconditioners for the PW1 and PW2 problems.	161
6.21	Comparison of the Schur complement preconditioners for PW3 and PW4 problems.	161

Dedicated to my wife and my family . . .

Chapter 1

Preliminaries

It is widely recognized that preconditioning is the most critical ingredient in the development of efficient solvers for challenging problems in scientific computation, and that the importance of preconditioning is destined to increase even further.

Michele Benzi. *Journal of Computational Physics*, Vol. 182, 2002.

The first chapter starts with an introduction of the dissertation, which motivates the use of preconditioners for the solutions of computational-electromagnetics (CEM) problems. After defining the notation that we adopt, we continue with the background information about the surface integral-equation formulations, their discretization, the multilevel fast multipole algorithm (MLFMA), iterative solvers, and preconditioning. We note that surface integral-equation formulations we define in this chapter are related to perfectly-electric-conductor (PEC) objects. We postpone the information about dielectric problems to related chapter. Then, we state contributions of the Ph.D. to the CEM community. Since we extensively compare the performances of the preconditioners that we develop with each other and also with previously developed

ones, we explain our hardware and software resources. Finally, we conclude with the organization of the dissertation.

1.2 Introduction

A popular approach to study electromagnetic scattering and radiation of three-dimensional (3-D) CEM problems is to solve discretized surface integral equations, which give rise to large, dense, and complex linear systems. For the solution of such dense systems, direct methods based on Gaussian elimination have been preferred in the past due to their robustness [1]. However, the large problem sizes confronted in CEM prohibit the use of these methods, which have the $\mathcal{O}(N^2)$ memory and $\mathcal{O}(N^3)$ computational complexity for N unknowns.

On the other hand, iterative solutions of linear systems using Krylov subspace methods make it possible to solve large-scale scientific problems with modest computing requirements [1, 2, 3, 4]. Krylov subspace methods access the system matrix through matrix-vector multiplications (MVMs). Even though the system matrix is dense in our case, the MVMs can be performed in $\mathcal{O}(N \log N)$ time and memory complexity using MLFMA. Hence, iterative solution of such dense systems with MLFMA has been a very attractive approach for large CEM problems [5].

This approach works well, however, only if the number of iterations required for convergence of the iterative solver is not too high. Unfortunately, this is not the case for many practical problems. In particular, discretizations of open-surface problems and complex real-life targets yield ill-conditioned linear systems. Also, there are many other practical problems that degrades the conditioning of the system matrix. These include non-uniformity of the surface meshes that cannot be avoided for some complex objects and fine discretizations of the surface

mesh, which can be obliged because of the geometry or to increase solution accuracy. For such problems, iterative solvers may even not converge, or convergence may require too many iterations.

At this point, preconditioners, which can be roughly defined as easily invertible approximations of the system matrices, come into the picture. With the help of the preconditioners, we try to render the system matrix have spectral properties that favor iterative convergence. In this dissertation, we present our efforts to design effective preconditioners for large-scale surface-integral-equation problems.

1.3 Notation

In general, we adopt the style of the CEM community in our notation. Greek or roman letters with an italic font are used for scalars. Bold-face, italic, capital letters with an over bar, such as $\overline{\mathbf{A}}$, are used to denote matrices. We reserve $\overline{\mathbf{A}}$ for the system (coefficient) matrix. Vectors are denoted with bold-face, italic, small letters without a bar, such as \mathbf{x} . For matrix-matrix or matrix-vector products, we use a dot between the matrices or vectors to differentiate them from scalar products. Unit vectors are denoted with a hat over the vector, such as $\hat{\mathbf{n}}$. For complexity estimates, we use the calligraphic capital letter (\mathcal{O}) to indicate a worst-case running time [6].

1.4 Surface Integral-Equation Formulations

Surface integral equations are extensively used in CEM for solving scattering and radiation problems [7, 8, 9]. Integral-equation formulations can be obtained by defining equivalent currents on the surface of an arbitrary 3-D geometry and applying boundary conditions. Various integral-equation formulations can be derived by employing different sets of boundary conditions and the testing procedure [10]. For PEC problems, we consider the most commonly used electric-field integral equation (EFIE), magnetic-field integral equation (MFIE), and combined-field integral equation (CFIE) formulations.

1.4.1 The Electric-Field Integral Equation (EFIE)

EFIE is based on a physical boundary condition, which states that the total tangential electric field vanishes on a conducting surface. Mathematically, EFIE can be expressed as

$$\hat{\mathbf{t}} \cdot \int_{S'} d\mathbf{r}' \overline{\mathbf{G}}(\mathbf{r}, \mathbf{r}') \cdot \mathbf{J}(\mathbf{r}') = \frac{i}{k\eta} \hat{\mathbf{t}} \cdot \mathbf{E}^{inc}(\mathbf{r}), \quad (1.1)$$

where $\mathbf{E}^{inc}(\mathbf{r})$ represents the incident electric field, S' is the surface of the object, $\hat{\mathbf{t}}$ is any tangential unit vector on S' , $\mathbf{J}(\mathbf{r}')$ is the unknown induced current residing on the surface, and $\eta = \sqrt{\mu/\epsilon}$ is the intrinsic impedance of the medium.

In (1.1), $\overline{\mathbf{G}}(\mathbf{r}, \mathbf{r}')$ is the dyadic Green's function defined as

$$\overline{\mathbf{G}}(\mathbf{r}, \mathbf{r}') = \left[\overline{\mathbf{I}} + \frac{\nabla \nabla}{k^2} \right] g(\mathbf{r}, \mathbf{r}'), \quad (1.2)$$

where

$$g(\mathbf{r}, \mathbf{r}') = \frac{e^{ik|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|} \quad (1.3)$$

is the scalar Green's function for the 3-D scalar Helmholtz equation. The scalar Green's function represents the response at the observation point \mathbf{r} due to a point source located at \mathbf{r}' . In (1.1), (1.2), and (1.3), k denotes the wavenumber ($k = 2\pi/\lambda$, where λ is the wavelength).

EFIE belongs to the class of first-kind integral equations, which have a weakly-singular kernel. Due to the weak singularity of the kernel, the integral equation acts as a smoothing operator and provides high accuracy with low-order basis functions, such as the commonly used Rao-Wilton-Glisson (RWG) basis functions [7]. On the other hand, because of the weak singularity of the kernel, matrices obtained with the discretization of EFIE tend to be ill-conditioned [11, 12].

1.4.2 The Magnetic-Field Integral Equation (MFIE)

Using the boundary condition for the tangential magnetic field on a conducting surface, MFIE can be expressed as

$$-\mathbf{J}(\mathbf{r}) + \hat{\mathbf{n}} \times \int_{S'} d\mathbf{r}' \mathbf{J}(\mathbf{r}') \times \nabla' g(\mathbf{r}, \mathbf{r}') = -\hat{\mathbf{n}} \times \mathbf{H}^{inc}(\mathbf{r}), \quad (1.4)$$

where $\hat{\mathbf{n}}$ is any unit normal vector on S' and $\mathbf{H}^{inc}(\mathbf{r})$ is the incident magnetic field. In (1.4), note that the boundary condition for the magnetic field is tested via the unit normal vector $\hat{\mathbf{n}}$. This is necessary to obtain stable solutions using a Galerkin scheme [10].

Unlike EFIE, MFIE is a second-kind integral equation that leads to diagonally dominant and well-conditioned matrices [13]. However, due to the singularity of its kernel, the accuracy of MFIE is significantly lower than that of EFIE [12, 14, 15, 16]. The identity term that results from the $\mathbf{J}(\mathbf{r})$ term in (1.4) is also another source for error [17].

1.4.3 The Combined-Field Integral Equation (CFIE)

CFIE is a more accurate second-kind integral equation than MFIE. It is obtained by linearly combining EFIE and MFIE, i.e.,

$$\text{CFIE} = \alpha \text{EFIE} + (1 - \alpha) \text{MFIE}, \quad (1.5)$$

where α is a parameter between 0 and 1. It is shown that $\alpha = 0.2$ or $\alpha = 0.3$ yields minimum iteration counts [18]. Among the three integral equations considered in this study, CFIE is the only formulation that is free from internal-resonance problems [13]. Furthermore, CFIE leads to well-conditioned systems, particularly for simple objects [19]. Currently, the solution of a sphere problem involving more than 200 million unknowns has been reported, where the solution is obtained in only 25 iterations with a simple block-diagonal preconditioner [20]. On the other hand, CFIE is not applicable to open geometries since it contains MFIE. Therefore, CFIE is preferred to MFIE for closed geometries, but EFIE, which produces ill-conditioned linear systems, particularly for large problems [17], is the mandatory choice for geometries with open surfaces.

1.5 Discretization of the Surface Formulations

Following a simultaneous discretization of the integral-equation formulations and geometry surfaces, electromagnetics problems involving complicated targets can be discretized and solved numerically. In this section, we present the details of the discretization procedures.

1.5.1 Method of Moments

We can convert the surface integral equations described in Section 1.4 to dense linear systems using the method of moments (MOM). Using a linear operator \mathcal{L} , these integral equations can be denoted as

$$\mathcal{L}\{\mathbf{J}\} = \mathbf{G}, \tag{1.6}$$

where \mathbf{G} is one of the known right-hand-side (RHS) vectors in (1.1) or (1.5). Projecting (1.6) onto the N -dimensional space $\text{span}\{\mathbf{j}_1, \mathbf{j}_2, \dots, \mathbf{j}_N\}$ formed by

the divergence-conforming RWG basis functions, we obtain

$$\langle \mathbf{j}_m, \mathcal{L}\{\mathbf{J}\} \rangle = \langle \mathbf{j}_m, \mathbf{G} \rangle, \quad m = 1, 2, \dots, N, \quad (1.7)$$

where

$$\langle \mathbf{f}, \mathbf{g} \rangle = \int d\mathbf{r} \mathbf{f}(\mathbf{r}) \cdot \mathbf{g}(\mathbf{r}) \quad (1.8)$$

denotes the inner product of two vector functions \mathbf{f} and \mathbf{g} . Then, adopting Galerkin's approach, we expand the unknown current using the same set of basis functions, i.e.,

$$\mathbf{J} \approx \sum_{n=1}^N x_n \mathbf{j}_n. \quad (1.9)$$

Hence, the coefficient vector \mathbf{x} becomes the solution of the $N \times N$ linear system

$$\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}, \quad (1.10)$$

where

$$(\overline{\mathbf{A}})_{mn} = \langle \mathbf{j}_m, \mathcal{L}\{\mathbf{j}_n\} \rangle, \quad (\mathbf{b})_m = \langle \mathbf{j}_m, \mathbf{G} \rangle, \quad m, n = 1, 2, \dots, N. \quad (1.11)$$

A matrix entry $(\overline{\mathbf{A}})_{mn}$ defined in (1.11) can be interpreted as an electromagnetic interaction between the m th testing function and the n th basis function.

The RWG basis functions are defined on planar triangles. Therefore, surfaces of CEM problems are meshed accordingly using planar triangles. Each RWG basis function is associated with an edge; hence the number of unknowns for a problem becomes equal to the total number of edges in the mesh, except for the boundary edges of an open surface.

1.5.2 Discretization of EFIE

After the discretization of EFIE defined in (1.1) with MOM, the matrix entries can be derived as

$$\begin{aligned} (\overline{\mathbf{A}}^{EFIE})_{mn} &= \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \int_{S_n} d\mathbf{r}' \mathbf{b}_n(\mathbf{r}') g(\mathbf{r}, \mathbf{r}') \\ &\quad - \frac{i}{k^2} \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \int_{S_n} d\mathbf{r}' \mathbf{b}_n(\mathbf{r}') \cdot [\nabla \nabla' g(\mathbf{r}, \mathbf{r}')], \end{aligned} \quad (1.12)$$

where \mathbf{t}_m denotes a testing function and \mathbf{b}_n denotes a basis function. Due to the double differentiation of the scalar Green's function, EFIE is highly singular in this form. However, using the divergence-conforming feature of RWG basis functions, it is possible to distribute the two differential operators onto the basis and testing functions and obtain [7]

$$\begin{aligned} (\overline{\mathbf{A}}^{EFIE})_{mn} &= ik \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \int_{S_n} d\mathbf{r}' \mathbf{b}_n(\mathbf{r}') g(\mathbf{r}, \mathbf{r}') \\ &\quad - \frac{i}{k^2} \int_{S_m} d\mathbf{r} \nabla \cdot \mathbf{t}_m(\mathbf{r}) \int_{S_n} d\mathbf{r}' \nabla' \cdot \mathbf{b}_n(\mathbf{r}') g(\mathbf{r}, \mathbf{r}'). \end{aligned} \quad (1.13)$$

The outer integrals in (1.13) can be evaluated numerically by employing Gaussian quadrature rules [21]. The inner integrals can be evaluated as

$$\int_{S_n} d\mathbf{r}' \left\{ \begin{array}{c} 1 \\ x' \\ y' \end{array} \right\} g(\mathbf{r}, \mathbf{r}') = I_1 + I_2, \quad (1.14)$$

where

$$I_1 = \frac{1}{4\pi} \int_{S_n} d\mathbf{r}' \left\{ \begin{array}{c} 1 \\ x' \\ y' \end{array} \right\} \frac{\exp(ikR) - 1}{R} \quad (1.15)$$

and

$$I_2 = \frac{1}{4\pi} \int_{S_n} d\mathbf{r}' \left\{ \begin{array}{c} 1 \\ x' \\ y' \end{array} \right\} \frac{1}{R}. \quad (1.16)$$

For I_1 , an adaptive integration method or a Gaussian quadrature rule can be used [5]. Furthermore, for accurate computations, singularity extraction techniques are employed by sufficiently subtracting the singular parts of the integrands. The integral I_2 can be evaluated analytically [22, 23].

1.5.3 Discretization of MFIE

The discretization of MFIE in (1.4) with RWG basis functions and a Galerkin scheme leads to

$$\begin{aligned} (\overline{\mathbf{A}}^{MFIE})_{mn} &= - \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \mathbf{b}_n(\mathbf{r}') \\ &+ \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \hat{\mathbf{n}} \times \int_{S_n} d\mathbf{r}' \mathbf{b}_n(\mathbf{r}') \times \nabla' g(\mathbf{r}, \mathbf{r}'). \end{aligned} \quad (1.17)$$

Since the second term in the RHS of (1.17) contains a singularity, we perform an efficient singularity extraction technique for the outer integral [14]. After the singularity extraction, (1.17) becomes

$$\begin{aligned} (\overline{\mathbf{A}}^{MFIE})_{mn} &= -\frac{1}{2} \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \mathbf{b}_n(\mathbf{r}') \\ &+ \int_{S_m, PV} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \hat{\mathbf{n}} \times \int_{S_n} d\mathbf{r}' \mathbf{b}_n(\mathbf{r}') \times \nabla' g(\mathbf{r}, \mathbf{r}'), \end{aligned} \quad (1.18)$$

where *PV* indicates the principal value of the integral. The double integral in the second RHS term of (1.18) can be modified as [23]

$$\int_{S_m} d\mathbf{r} (\mathbf{t}_m(\mathbf{r}) \times \hat{\mathbf{n}}) \cdot \mathbf{b}_n(\mathbf{r}) \times \int_{PV, S_n} d\mathbf{r}' \nabla' g(\mathbf{r}, \mathbf{r}'). \quad (1.19)$$

Note that only the principal values are required for (1.19) since the the limit part is extracted. Nonetheless, the singularity extraction is applied again to smooth the integrand before an adaptive integration. The inner integral in (1.19) can be calculated as

$$\int_{PV, S_n} d\mathbf{r}' \nabla' g(\mathbf{r}, \mathbf{r}') = I_1 + I_2 + I_3, \quad (1.20)$$

where

$$I_1 = \frac{1}{4\pi} \int_{PV, S_n} d\mathbf{r}' \nabla' \left(\frac{\exp(ikR) - 1 + 0.5k^2 R^2}{R} \right), \quad (1.21)$$

$$I_2 = \frac{1}{4\pi} \int_{PV, S_n} d\mathbf{r}' \nabla' \left(\frac{1}{R} \right), \quad (1.22)$$

and

$$I_3 = -\frac{k_l^2}{8\pi} \int_{PV, S_n} d\mathbf{r}' \nabla' R. \quad (1.23)$$

I_1 is calculated using an adaptive integration method or a Gaussian quadrature, whereas I_2 and I_3 are evaluated analytically [24, 22].

1.5.4 Discretization of CFIE

Since CFIE is a linear combination of EFIE and MFIE, both formulations should be discretized to form CFIE. Once these formulations are discretized, the elements of CFIE matrices can be derived as

$$(\overline{\mathbf{A}}^{CFIE})_{mn} = \alpha (\overline{\mathbf{A}}^{EFIE})_{mn} + (1 - \alpha) (\overline{\mathbf{A}}^{MFIE})_{mn}. \quad (1.24)$$

1.5.5 Computation of the RHS Vectors

Elements of the RHS vector for EFIE are obtained by testing the incident electric field in the RHS of (1.1), i.e.,

$$(\mathbf{b})_m^{EFIE} = -\frac{i}{k\eta} \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \mathbf{E}^{inc}(\mathbf{r}). \quad (1.25)$$

Similarly, the RHS vector for MFIE can be found using

$$(\mathbf{b})_m^{MFIE} = - \int_{S_m} d\mathbf{r} \mathbf{t}_m(\mathbf{r}) \cdot \hat{\mathbf{n}} \times \mathbf{H}^{inc}(\mathbf{r}). \quad (1.26)$$

Then the RHS vectors for CFIE can be calculated as the linear combination of (1.25) and (1.26), i.e.,

$$(\mathbf{b})_m^{CFIE} = \alpha (\mathbf{b})_m^{EFIE} + (1 - \alpha) (\mathbf{b})_m^{MFIE}. \quad (1.27)$$

1.6 The Multilevel Fast Multipole Algorithm (MLFMA)

The discretization of the surface formulations with MOM leads to dense linear systems due to the nonlocal nature of the electromagnetic interactions between the basis and testing functions. Surfaces of objects are usually meshed with one-tenth of the wavelength for accuracy. Hence, for high frequencies, where

the scatterer or the radiator sizes become large in terms of the wavelength, the system matrix also becomes large. For solving such matrix systems, direct solution methods become too expensive due to their high computational complexity. Iterative methods may be preferred as a more viable option provided that the number of iterations remains limited even for large numbers of unknowns. However, iterative methods require matrix-vector multiplications, which have $\mathcal{O}(N^2)$ complexity for $N \times N$ dense matrices. Although lower than the $\mathcal{O}(N^3)$ complexity of direct solvers, $\mathcal{O}(N^2)$ complexity is still prohibitive for large problems. As a result, in addition to effective preconditioners, iterative solutions of real-life CEM problems require acceleration methods for performing fast matrix-vector multiplications with low-complexity. In this context, MLFMA is a method of choice since it renders the solution of large CEM problems possible by reducing the complexity of matrix-vector multiplications to $\mathcal{O}(N \log N)$. The main components of MLFMA are outlined in the following.

1.6.1 Clustering

In order to compute the interactions between the basis and testing functions in a multilevel scheme, an oct-tree strategy is employed. For this purpose, the whole geometry is placed inside a cube, which is recursively divided into smaller cubes until the smallest cubes contain only a few basis functions, as illustrated in Fig. 1.1. If any of the cubes becomes empty during the partitioning, recursion stops there. An example of the clustering and the corresponding oct-tree for a sphere problem is shown in Fig. 1.2.

In 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 lowest level (Level 1 in Fig. 1.2), interactions between the near-field clusters, including the self interactions, constitute the near-field matrix and the remaining far-field interactions constitute the far-field matrix. In the course of an iterative solution,

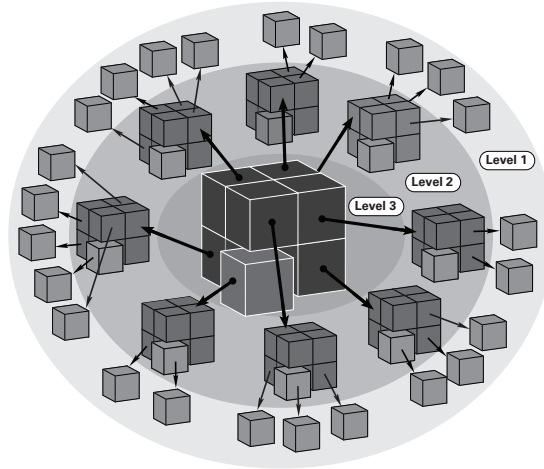


Figure 1.1: Illustration of the oct-tree partitioning of the computational domain in MLFMA.

MLFMA decomposes matrix-vector multiplications as

$$\overline{\mathbf{A}} \cdot \mathbf{x} = \overline{\mathbf{A}}^{NF} \cdot \mathbf{x} + \overline{\mathbf{A}}^{FF} \cdot \mathbf{x}. \quad (1.28)$$

In (1.28), $\overline{\mathbf{A}}^{NF}$ denotes the near-field matrix, which is calculated directly as described in Section 1.5 and stored in memory to perform the partial matrix-vector multiplication $\overline{\mathbf{A}}^{NF} \cdot \mathbf{x}$. Examples for $\overline{\mathbf{A}}^{NF}$ are depicted in Fig. 1.3. Note that these matrices are composed of small blocks, which correspond to the near-field interactions of the lowest-level clusters. However, the matrices do not exhibit any structured sparsity pattern, except for the apparent larger diagonal blocks. Those diagonal blocks are formed from the interactions of the lowest-level clusters that have the same parent cluster. $\overline{\mathbf{A}}^{FF} \cdot \mathbf{x}$ denotes the multiplication with far-field interactions, which will be detailed in Section 1.6.3. To achieve $\mathcal{O}(N \log N)$ complexity, this stage is performed approximately but with controllable error, i.e., with the desired level of accuracy.

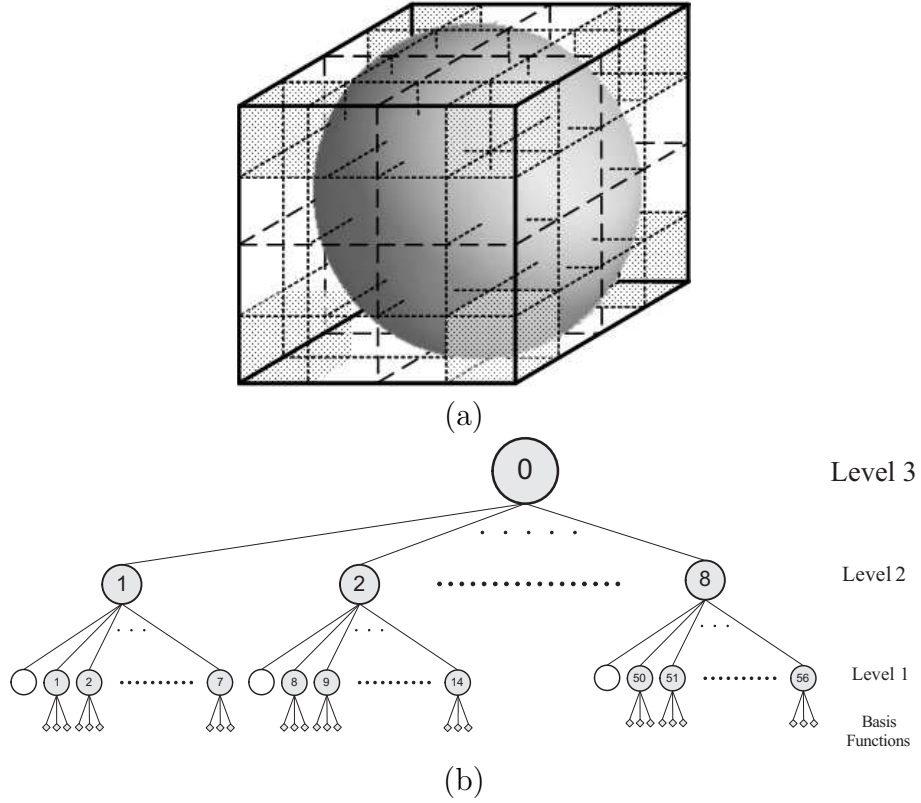


Figure 1.2: (a) Multilevel partitioning of the scatterer for the case of a sphere with diameter 1λ . The shaded boxes are empty. (b) Tree structure of MLFMA for the sphere. Unfilled nodes correspond to empty boxes.

1.6.2 Factorization of the Green's Function

MLFMA is proposed as a multilevel extension of the single-level fast multipole method (FMM) [25, 26], and the factorization of the Green's function is at the core of FMM.

Consider two far-zone clusters that are defined with the reference points C' and C . For the interactions between the basis functions that are clustered around C' and testing functions that are clustered around C , the scalar Green's function can be factorized as [27]

$$g(\mathbf{r}, \mathbf{r}') = \frac{e^{ik|\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|} = \frac{e^{ik|\mathbf{D}+\mathbf{d}|}}{4\pi|\mathbf{D}+\mathbf{d}|} \approx \frac{1}{4\pi} \int d^2\hat{\mathbf{k}} e^{i\hat{\mathbf{k}}\cdot\mathbf{d}} \alpha_T(k, D, \hat{\mathbf{D}} \cdot \hat{\mathbf{k}}), \quad (1.29)$$

where $D = |\mathbf{D}|$ represents the distance between C' and C . The integration in (1.29) is performed on the unit sphere and $\hat{\mathbf{k}}$ is the unit vector normal to the

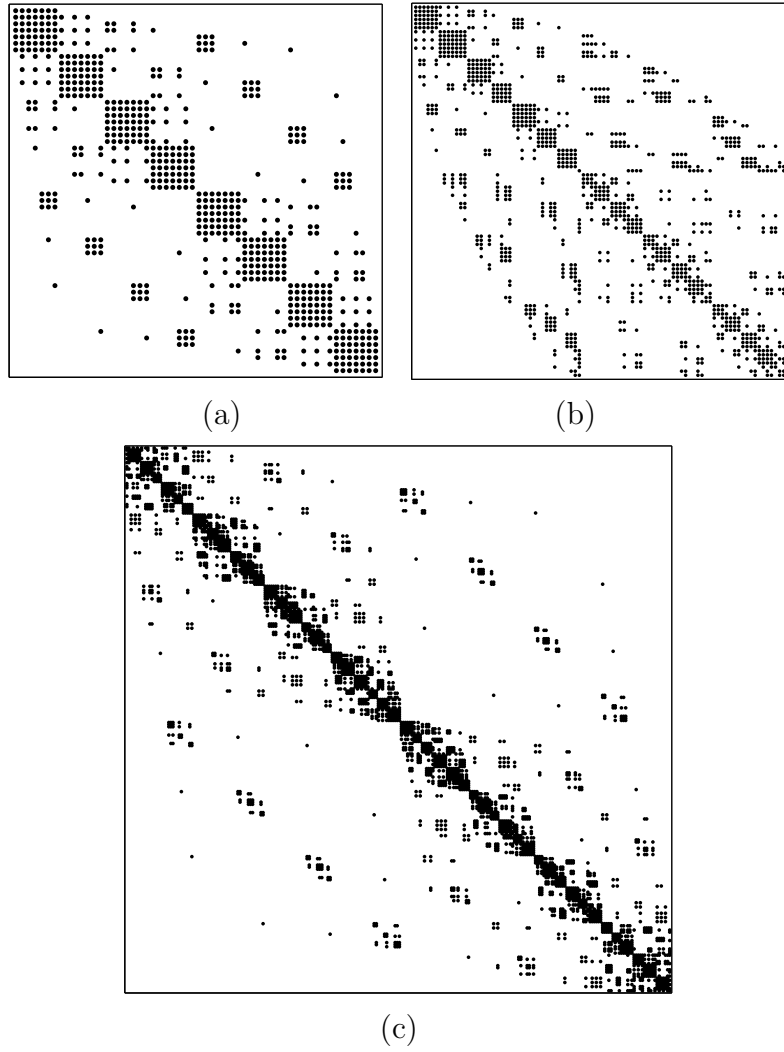


Figure 1.3: Sparse near-field matrices for (a) $N = 930$, (b) $N = 1,302$, and (c) $N = 3,723$.

unit sphere. The translation function

$$\alpha_T(k, D, \hat{\mathbf{D}} \cdot \hat{\mathbf{k}}) = \sum_{t=0}^T i^t (2t+1) h_t^{(1)}(kD) P_t(\hat{\mathbf{D}} \cdot \hat{\mathbf{k}}) \quad (1.30)$$

involves the spherical Hankel function of the first kind $h_t^{(1)}$ and the Legendre polynomial P_t . The translation function defined in (1.30) can be used to evaluate the group interactions between the basis and testing functions clustered around C' and C , instead of calculating the interactions separately.

By diagonalizing [28] the scalar Green's function as in (1.29) and (1.30), single-level interactions can be derived as

$$(\overline{\mathbf{A}})_{mn} = \left(\frac{ik}{4\pi}\right)^2 \int d^2\hat{\mathbf{k}} \overline{\mathbf{F}}_{C_m}^{rec}(\hat{\mathbf{k}}) \cdot \alpha_T(k, D, \hat{\mathbf{D}} \cdot \hat{\mathbf{k}}) \overline{\mathbf{F}}_{C'_n}^{rad}(\hat{\mathbf{k}}), \quad (1.31)$$

where $\overline{\mathbf{F}}_{C_m}^{rec}$ represents the receiving pattern of the m th testing function with respect to the reference point C and $\overline{\mathbf{F}}_{C'_n}^{rad}$ represents the radiation pattern of the n th basis function with respect to the reference point C' .

In any MLFMA level l , radiation and receiving patterns are defined and sampled at $\mathcal{O}(T_l^2)$ angular points, where T_l is the truncation number for the series in (1.30). Since we set the minimum cluster size at the lowest level as 0.25λ , the cluster size at level l is $a_l = 2^{l-3}\lambda$. For a cluster of size a_l , the truncation number is determined by using the excess bandwidth formula [29] for the worst-case scenario and the one-box-buffer scheme [30], i.e.,

$$T_l \approx 1.73ka + 2.16(d_0)^{2/3}(ka_l)^{1/3}, \quad (1.32)$$

where d_0 is the number of accurate digits desired.

1.6.3 Far-Field Interactions

In MLFMA, far-field interactions are calculated in a multilevel scheme and in a group-by-group manner. For this purpose, the aggregation, translation, and

disaggregation stages are performed in each matrix-vector multiplication. These stages are described below.

- *Aggregation:* Radiated fields of clusters are calculated from the bottom of the tree structure to the highest level. At the lowest level, radiation patterns of basis functions are multiplied with the elements of the input vector provided by the iterative solver. Then, the radiated field of a cluster is determined by combining the radiation patterns inside the cluster. At higher levels, the radiated field of a cluster is obtained by combining the radiated fields of the clusters in the lower levels. Between two consecutive levels, interpolations are employed to match the different sampling rates of the fields using a local interpolation method [31, 32].
- *Translation:* For each pair of far-field clusters, whose parents are in the near-field zone of each other, the cluster-to-cluster interaction is computed via a translation. Note that the sizes of the cubic clusters are identical in each level. Hence, the number of translation operators is reduced to $\mathcal{O}(1)$ using the symmetry. For those clusters whose parents are in the far-field zone of each other, the cluster-to-cluster interaction is performed in a higher-level translation.
- *Disaggregation:* Total incoming fields at the cluster centers are calculated from the top of the tree structure to the lowest level. The total incoming field for a cluster is obtained by combining incoming fields due to translations and the incoming field from its parent cluster, if it exists. Incoming fields to the center of a cluster are shifted to the centers of the clusters in the lower levels by using transpose interpolations, or antinterpolations [33]. Finally, in the lowest level, incoming fields are received by the testing functions via angular integrations.

1.7 Iterative Solvers

In this section, we give a brief introduction to Krylov subspace iterative solvers and the generalized minimal residual (GMRES) method, which has been preferred in our numerical experiments for the reasons that will be explained. For a comprehensive introduction, we refer to books [1, 2, 4]. A shorter and yet neat explanation of iterative methods and GMRES can be found in [34].

The main motivation behind the development of iterative methods is the prohibitive $\mathcal{O}(N^3)$ complexity of direct methods. If we use a direct method and need to increase the size of the problem from thousands to say just ten thousands, then the solution time increases by an order of 10^3 . That is, if we can solve the small problem in a few hours, we should wait for days for the solution of the larger problem. If we need a more radical increase, e.g., to millions, than the waiting period can be tremendous. Another limitation of direct methods is their memory use. Even though the memory consumption of direct methods is proportional to $\mathcal{O}(N^2)$, this requirement can still easily exceed the available memory. Ideally, we would like to have a linear increase in both memory and solution time with respect to problem size, which means an $\mathcal{O}(N)$ - or $\mathcal{O}(N \log N)$ -complexity solver.

The iterative algorithms can approach that ideal complexity by exploiting the structure of matrices. For finite-difference or finite-element methods, the structure is in the form of sparsity, i.e., most of the matrix entries are zero. Sparsity is preserved in iterative methods since they access the system matrix in the form of matrix-vector products. That is, regarding the system matrix, an iterative method requires only the ability to determine the product $\bar{\mathbf{A}} \cdot \mathbf{z}$ for a given \mathbf{z} . This property can also be used for some dense matrices, which can be stored and multiplied in a “data-sparse form”. The solution of discretized integral equations by MLFMA is a typical example.

Another advantage of the iterative solvers is that they provide an approximate solution that is “accurate enough” and satisfies practical needs. For instance, for the computation of radar cross sections in CEM, a residual error ϵ about 10^{-3} demonstrates remarkable consistency with the analytical solutions obtained with the Mie series. If the number of iterations can be kept constant or increase slowly with number of unknowns, the iterative solvers can reach the optimal solution complexity. However, in order to guarantee a low-iteration count, preconditioning is in general required. Direct solution methods, on the other hand, require N steps and in each step they require $\mathcal{O}(N^2)$ floating point operations, for a total work of $\mathcal{O}(N^3)$ to achieve a solution to machine precision $\epsilon_{machine}$, which is about 10^{-15} for double precision arithmetic. We illustrate these ideas in Fig. 1.4.

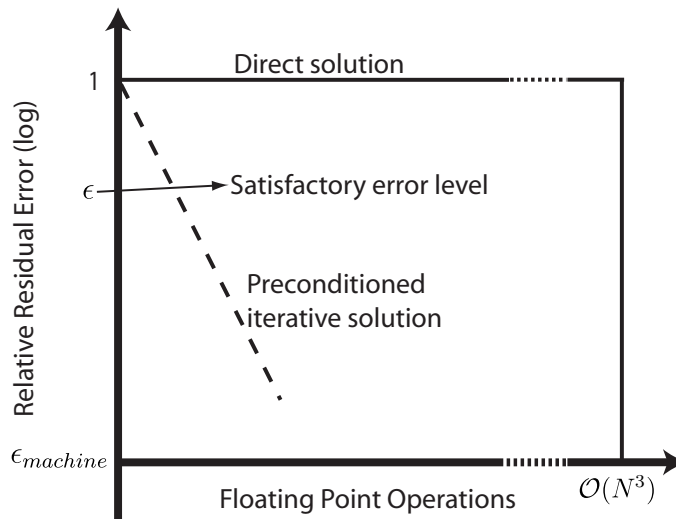


Figure 1.4: Comparison of the direct and iterative solvers for work performed and acquired error levels.

1.7.1 Krylov Subspace Methods

Krylov subspace methods start with an initial guess, mostly a zero-vector. Then, the true solution is approximated from a Krylov subspace, which is augmented at each iteration. The Krylov subspace generated by $\overline{\mathbf{A}}$ and the right-hand-side

(RHS) vector \mathbf{b} at the k th iteration is defined as

$$K_k(\overline{\mathbf{A}}, \mathbf{b}) = \text{span}\{\mathbf{b}, \overline{\mathbf{A}} \cdot \mathbf{b}, \dots, \overline{\mathbf{A}}^{k-1} \cdot \mathbf{b}\}. \quad (1.33)$$

Krylov subspace is a suitable space to search an approximation to $\mathbf{x} = \overline{\mathbf{A}}^{-1} \cdot \mathbf{b}$ because of the Cayley-Hamilton theorem [35], which allows us to express $\overline{\mathbf{A}}^{-1}$ in terms of powers of $\overline{\mathbf{A}}$.

The Krylov subspace $K_k(\overline{\mathbf{A}}, \mathbf{b})$ is generated by a process known as Arnoldi iteration, which forms an orthonormal subspace. For Hermitian positive definite (HPD) matrices, the Lancsoz iteration is used instead. The approximate solution at iteration k is found by solving a reduced $k \times k$ system, which corresponds to projection of the N -dimensional system $\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$ into the k -dimensional Krylov subspace $K_k(\overline{\mathbf{A}}, \mathbf{b})$. For HPD matrices, it is possible to construct a well-conditioned orthonormal subspace with a three-term recurrence, hence, the reduced system is tridiagonal. For non-Hermitian matrices, on the other hand, it is shown that such a short-term recurrence relation does not exist [36]. Hence, for such matrices, one needs to construct the k th subspace with a k -term recurrence relation, and the projected system becomes Hessenberg (i.e., an upper triangular matrix with an additional off-diagonal below the diagonal) [4].

The aforementioned approach leads to the conjugate gradient (CG) method for HPD matrices and to the generalized minimal residual method (GMRES) for non-Hermitian matrices. These solvers are optimal in the sense that they guarantee a non-increasing residual norm and convergence in N iterations ignoring finite precision effects. Because of the three-term recurrence, per iteration cost of CG is constant both in terms of memory and CPU time. As a result, it is always the method of choice for HPD systems. On the other hand, GMRES needs to call the Arnoldi's method at each iteration to orthonormalize the current Krylov vector against all previous Krylov vectors. Hence, its CPU and memory costs increase linearly with iteration number. To eliminate this, the optimality is sacrificed, and CG-like solvers are developed with short-term recurrence

relations, such as the conjugate gradient squared (CGS), bi-conjugate gradient (BiCG), and its stabilized version, bi-conjugate gradient stabilized (BiCG-Stab). For an overview of such methods, see [36]. Another approach to limit the cost of GMRES is to use its restarted or truncated versions [2], again with the cost of forgoing its robustness.

We have used GMRES without a restart or truncation in our numerical experiments. We notice that for systems originated from the first-kind integral equations, such as EFIE, there is a severe difference in the performances of GMRES and other non-optimal solvers. This is valid for both preconditioned and unpreconditioned solutions. For sparse linear systems, the long recurrences associated with GMRES may be significant, and solutions with non-optimal solvers may require shorter CPU times, even though numbers of matrix-vector multiplications increase compared to GMRES (e.g., see [1]-Chapter 39.) In our case, on the other hand, the cost of GMRES is much less than that of MLFMA, both in terms of CPU time and memory. One reason for this is the high constant term hidden in the $\mathcal{O}(N \log N)$ complexity of MLFMA, and the other is that we usually set a modest upper limit for number of iterations, typically 1,000. A comparison of the memory of MLFMA and GMRES can be found in the results sections of Chapter 4 and 6. Another reason of the use of GMRES is that with a small modification, it leads to a flexible version, for which the preconditioner is allowed to change from iteration to iteration. This feature allows us to use the iterative solution of the near-field matrix and the MLFMA itself as effective preconditioners. These methods will be described in Sections 4 and 5.

Next, we present a high-level description of GMRES and comment briefly on its convergence. A detailed explanation can be found in books [1] and [2].

1.7.2 The generalized minimal residual method (GMRES)

As mentioned, GMRES picks the best approximation \mathbf{x}_k from the orthonormalized Krylov subspace $K_k(\overline{\mathbf{A}}, \mathbf{b})$. (We assume a zero initial guess for simplicity.) The selection is performed by minimizing the norm of the k th residual $\mathbf{r}_k = \mathbf{b} - \overline{\mathbf{A}} \cdot \mathbf{x}_k$, i.e., GMRES solves the least-squares problem

$$\min_{\mathbf{x}_k \in K_k(\overline{\mathbf{A}}, \mathbf{b})} \|\mathbf{b} - \overline{\mathbf{A}} \cdot \mathbf{x}_k\|. \quad (1.34)$$

For this purpose, a set of orthonormal vectors $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k\}$ that span the subspace $K_k(\overline{\mathbf{A}}, \mathbf{b})$ is constructed by means of the Arnoldi iteration. Let $\overline{\mathbf{Q}}_k$ denote the $N \times k$ matrix obtained by collecting those vectors. Since $\mathbf{x}_k \in K_k(\overline{\mathbf{A}}, \mathbf{b})$, we have $\mathbf{x}_k = \overline{\mathbf{Q}}_k \cdot \mathbf{z}$ for some k -length vector \mathbf{z} and the least-squares problem in (1.34) is reduced to $N \times k$ system

$$\min_{\mathbf{z}} \|\mathbf{b} - \overline{\mathbf{A}} \cdot \overline{\mathbf{Q}}_k \cdot \mathbf{z}\|. \quad (1.35)$$

This problem can be further reduced to $(k+1) \times k$ size as follows. The Krylov subspace property $\overline{\mathbf{A}} \cdot K_k(\overline{\mathbf{A}}, \mathbf{b}) \in K_{k+1}(\overline{\mathbf{A}}, \mathbf{b})$ implies that there is a $(k+1) \times k$ Hessenberg matrix $\overline{\mathbf{H}}_k$ such that

$$\overline{\mathbf{A}} \cdot \overline{\mathbf{Q}}_k = \overline{\mathbf{Q}}_{k+1} \cdot \overline{\mathbf{H}}_k. \quad (1.36)$$

Using this equality, (1.35) can be transformed to

$$\min_{\mathbf{z}} \|\mathbf{b} - \overline{\mathbf{Q}}_{k+1} \cdot \overline{\mathbf{H}}_k \cdot \mathbf{z}\|. \quad (1.37)$$

Note that \mathbf{b} is the first member of the $K_k(\overline{\mathbf{A}}, \mathbf{b})$, hence

$$\mathbf{b} = \|\mathbf{b}\| \mathbf{q}_1 = \|\mathbf{b}\| \overline{\mathbf{Q}}_{k+1} \cdot \mathbf{e}_1, \quad (1.38)$$

where \mathbf{e}_1 is the first vector of the $(k+1) \times (k+1)$ identity matrix. Combining (1.37) and (1.38), we have

$$\min_{\mathbf{z}} \|\overline{\mathbf{Q}}_{k+1} \cdot (\|\mathbf{b}\| \mathbf{e}_1 - \overline{\mathbf{H}}_k \cdot \mathbf{z})\|. \quad (1.39)$$

Having orthonormal vectors, multiplying a vector by $\overline{\mathbf{Q}}_{k+1}$ leave the norm unchanged. Therefore, an equivalent problem to (1.39) is

$$\min_{\mathbf{z}} \left\| \|\mathbf{b}\| \mathbf{e}_1 - \overline{\mathbf{H}}_k \cdot \mathbf{z} \right\|, \quad (1.40)$$

which is $(k+1) \times k$ size. Once we find the least-error solution \mathbf{z} , k th-approximation to \mathbf{x} is $\mathbf{x}_k = \overline{\mathbf{Q}}_k \cdot \mathbf{z}$. We outline the method with a pseudocode shown in Fig. 1.5. Note that convergence can be checked without the need to compute \mathbf{x}_k .

```

k = 0
||r_k|| = ||b||
while ||r_k||/||b|| > ε do
    k = k + 1
    Find  $\overline{\mathbf{Q}}_k$  and  $\overline{\mathbf{H}}_k$  with Arnoldi iteration
    Solve  $\min_{\mathbf{z}} \|\mathbf{r}_k\| = \min_{\mathbf{z}} \|\|\mathbf{b}\| \mathbf{e}_1 - \overline{\mathbf{H}}_k \cdot \mathbf{z}\|$ 
endwhile
 $\mathbf{x}_k = \overline{\mathbf{Q}}_k \cdot \mathbf{z}$ 

```

Figure 1.5: The GMRES method. ϵ is a predetermined stopping threshold.

The least-squares solution of $\min_{\mathbf{z}} \|\|\mathbf{b}\| \mathbf{e}_1 - \overline{\mathbf{H}}_k \cdot \mathbf{z}\|$ can be obtained in $\mathcal{O}(k)$ time using Givens rotation [2]. Hence, both the memory and CPU costs of GMRES increase linearly with iteration number.

1.7.3 Convergence of GMRES

GMRES demonstrates a monotonic decrease of the residual norm, and in exact arithmetic convergence is obtained in at most N steps. Of course, this bound does not have a practical importance for large-scale problems. In practice, however, convergence to satisfactory threshold can be achieved for some $k \ll N$, particularly if a suitable preconditioner is used.

Unlike the CG solver, convergence of GMRES is mostly governed by the settlement of eigenvalues on the complex plane, not on the condition number of

$\bar{\mathbf{A}}$. The reliability of the eigenvalues, on the other hand, depends on the normality of $\bar{\mathbf{A}}$, which can be measured with the condition number of the matrix composed of eigenvectors of $\bar{\mathbf{A}}$. Another tool that can be used to measure normality is the pseudospectrum [37]. A comparison of pseudospectra of the matrices obtained from surface integral equations will be presented in Section 1.9. Regarding the convergence of GMRES, one can say that a clustered spectrum that does not contain the origin, and for which the eigenvalues are not too close to the origin results in rapid convergence [1, 38].

1.8 Preconditioning

Preconditioners can be broadly classified as one of two types: forward (or implicit) and inverse (or explicit). Forward preconditioning (implicit) refers to finding an easily invertible operator $\bar{\mathbf{M}}$ for the system $\bar{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$, while $\bar{\mathbf{M}}$ approximates $\bar{\mathbf{A}}$ in some sense. Then, instead of the original system, the preconditioned system

$$\bar{\mathbf{M}}^{-1} \cdot \bar{\mathbf{A}} \cdot \mathbf{x} = \bar{\mathbf{M}}^{-1} \cdot \mathbf{b} \quad (1.41)$$

is solved. For inverse preconditioning (explicit), $\bar{\mathbf{M}}$ directly approximates the inverse of the system matrix, and the preconditioned system becomes

$$\bar{\mathbf{M}} \cdot \bar{\mathbf{A}} \cdot \mathbf{x} = \bar{\mathbf{M}} \cdot \mathbf{b}. \quad (1.42)$$

The idea is based on the observation that as $\bar{\mathbf{M}}$ approximates $\bar{\mathbf{A}}$, the product $\bar{\mathbf{M}}^{-1} \cdot \bar{\mathbf{A}}$ approximates the identity matrix (for forward preconditioners), and convergence can be attained in fewer iterations.

In Equations (1.41) and (1.42), we apply *left preconditioning*. We can also apply *right preconditioning*, in which case we should solve the systems

$$\bar{\mathbf{A}} \cdot \bar{\mathbf{M}}^{-1} \cdot \mathbf{y} = \mathbf{b}, \quad \mathbf{x} = \bar{\mathbf{M}}^{-1} \cdot \mathbf{y} \quad (1.43)$$

and

$$\overline{\mathbf{A}} \cdot \overline{\mathbf{M}} \cdot \mathbf{y} = \mathbf{b}, \quad \mathbf{x} = \overline{\mathbf{M}} \cdot \mathbf{y} \quad (1.44)$$

for forward and inverse preconditioning, respectively. At step k , GMRES minimizes the true residual norm $\|\mathbf{r}_k\|$ with right preconditioning, instead of the preconditioned residual norm $\|\overline{\mathbf{M}}^{-1} \cdot \mathbf{r}_k\|$. This is a desired situation, especially if there is an instability issue with the preconditioner.

For a useful preconditioner $\overline{\mathbf{M}}$, in addition to approximating the system matrix $\overline{\mathbf{A}}$, the construction (or setup) and application of $\overline{\mathbf{M}}$ should be performed efficiently. By application, we mean the solution of the system

$$\overline{\mathbf{M}} \cdot \mathbf{v} = \mathbf{w} \quad (1.45)$$

for implicit preconditioning, and the computation of the product

$$\mathbf{v} = \overline{\mathbf{M}} \cdot \mathbf{w} \quad (1.46)$$

for explicit preconditioning. The two requirements, i.e., approximation of $\overline{\mathbf{A}}$ and the fast construction and application, are in competition with each other. The better the approximation, the faster the convergence, but the more costly setup and application. Hence, useful preconditioners satisfy both requirements in a balanced way. In particular, we limit ourselves with the $\mathcal{O}(N \log N)$ complexity of MLFMA for the construction and application of a preconditioner.

Since the convergence of the non-Hermitian systems of surface integral equations mostly depends on the distribution of eigenvalues, we try to change the distribution in a way that favors convergence. The desired distribution is, in general, a clustered spectrum around the point $(1, 0)$. The matrices obtained from surface integral equations are indefinite, meaning that some of the eigenvalues have a negative real-part, i.e., they are scattered in the left side of the complex-plane. A successful preconditioner must move these eigenvalues towards $(1, 0)$, but because of the approximations errors that are intentionally made to render the construction efficient, some of the eigenvalues may be arbitrarily close to the

origin. In that case, the convergence may slow down compared to no preconditioning or a cheaper preconditioner. We have also faced with this phenomena for the first-kind integral-equation formulations. This is a severe difficulty in preconditioning of such indefinite systems.

1.9 Spectral Analysis of the Surface Formulations

In this section, we present a brief analysis about the algebraic properties of the matrices obtained from the discretization of surface integral equations.

Though they are indefinite and non-hermitian, 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-diagonal preconditioner even for large-scale problems [5]. Nonetheless, for some complex geometries, the number of iterations is still large. Considering the dominant cost of matrix-vector product for large problems, stronger preconditioners for CFIE are still desirable.

Systems resulting from EFIE are much more difficult to solve. In addition to being indefinite and non-hermitian, EFIE matrices may have large elements away from the diagonal, and some of the non-stored 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 Fig. 1.6 both the eigenvalues and the pseudospectra of the EFIE, MFIE, and CFIE matrices for the 930-unknown sphere problem. For non-normal matrices, information obtained from eigenvalues may be misleading, since they may become highly unstable [37]. More reliable and insightful information can be obtained using the ϵ -pseudospectrum, $\Lambda_\epsilon(\overline{\mathbf{A}})$,

which can be defined as

$$\Lambda_\epsilon(\overline{\mathbf{A}}) = \left\{ z \mid z \in \mathbb{C}^n, \|(z\overline{\mathbf{I}} - \overline{\mathbf{A}})^{-1}\|_2 \geq \frac{1}{\epsilon} \right\}. \quad (1.47)$$

Denoting the spectrum of $\overline{\mathbf{A}}$ with $\Lambda(\overline{\mathbf{A}})$, if an eigenvalue $\lambda \in \Lambda(\overline{\mathbf{A}})$, then $\|(z\overline{\mathbf{I}} - \overline{\mathbf{A}})^{-1}\|_2 = \infty$, so $\Lambda(\overline{\mathbf{A}}) \subset \Lambda_\epsilon(\overline{\mathbf{A}})$ for any $\epsilon > 0$. 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 non-normality.

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 [38, 39]. Comparison of the EFIE and CFIE pseudospectra in Fig. 1.6 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.

In Chapter 3.4.2, we also present some spectral information of unpreconditioned and preconditioned matrices using the approximate eigenvalues, which are obtained during GMRES iterations as a byproduct [1].

1.10 Contributions

Our contributions to integral-equation methods have been two folds. First, we adapted some of the well-proven algebraic preconditioning techniques used in scientific computing to CEM problems. Second, we propose some application-specific preconditioners that are more effective than general-purpose algebraic preconditioners. We summarize these studies as follows:

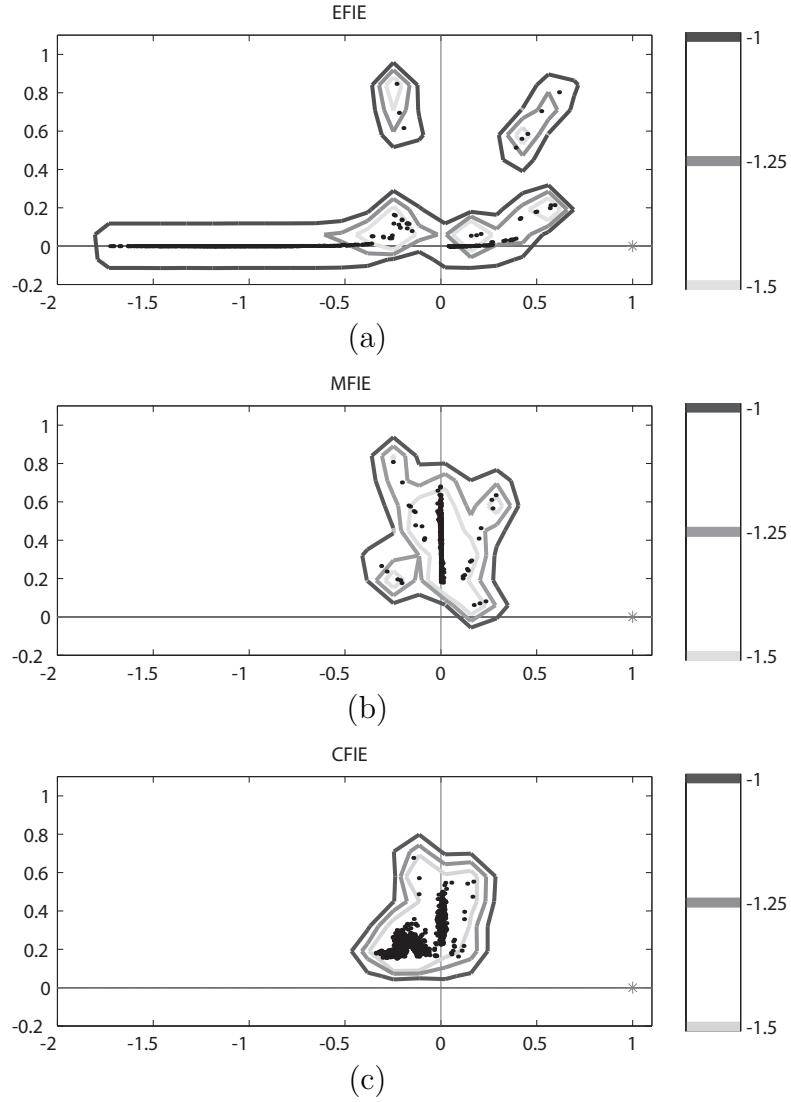


Figure 1.6: Pseudospectra of the EFIE, MFIE, and CFIE formulations for three ϵ values, i.e., 10^{-1} , $10^{-1.25}$, and $10^{-1.5}$. The black dots denote the exact eigenvalues of the unperturbed matrices.

- For sequential implementations of integral equations and MLFMA, we adapted incomplete LU (ILU) preconditioners to CEM problems [40, 41, 42, 43]. Among various ILU preconditioners, we demonstrated that ILU(0) fits best for the CFIE formulation and ILUT fits best to the EFIE formulation. We also showed that the reason behind the instability that occurs for some open-surface problems can be circumvented using partial pivoting.
- We introduced an efficient implementation of the sparse approximate inverse (SAI) preconditioner for the solution of large-scale electromagnetic

problems with parallel MLFMA. Thanks to our effective load-balancing algorithm, we obtain high scalability up to 128 processors [44, 45]. Furthermore, we have been able to solve ill-conditioned open-surface problems up to 33 millions of unknowns [46, 47, 48]. SAI preconditioner has also been applied to metamaterial problems with a high success [49, 50, 51].

- ILU and SAI preconditioners are constructed from the near-field matrix, which is a sparse portion of the dense coefficient matrix. For difficult problems, we observed that SAI is not as effective as ILU [52, 53]. For this reason, we proposed to use the iterative solution of the near-field matrix as a preconditioner, which provides faster convergence compared to SAI [54, 55, 56]. We call this approach as the iterative near-field (INF) preconditioner.
- Using an inner-outer solution scheme similar to INF, we introduced preconditioners that can make use of the far-field elements of MLFMA. We propose an approximate version of MLFMA [57, 58] to be used for the inner iterations, which performs a much faster matrix-vector multiplication compared to the regular MLFMA. Thanks to this effective strategy, we have been able to solve extremely large and ill-conditioned problems with modest computational requirements [59, 60, 48, 61, 62, 63].
- Finally, we proposed novel preconditioners based on the Schur complement reduction for partitioned linear systems arising from integral-equation formulations of dielectric problems. We proposed non-iterative and iterative versions of those preconditioners. For the reduced systems obtained from the Schur complement reduction, we constructed effective SAIs to be used as an approximate direct inverse, or, as preconditioners to accelerate inner solves. Real-life problems show that those preconditioners either render many difficult problems solvable, or significantly decrease the solution times [64, 65, 66, 67, 68].

1.11 Computational Resources

The performance of an implementation of a numerical technique depends on the chosen hardware and software components, in addition to its algorithmic features. For example, commonly used numerical kernels, such as basic linear algebra subprograms (BLAS) and linear algebra package (LAPACK), significantly affect the setup time of our SAI preconditioner. Similarly, for parallel programs on a computer cluster, the speed of the interconnect network affects the parallel performance (e.g., speedup) of an algorithm. In addition to devising high-performance solvers, we have given utmost importance to optimize our hardware and software resources by selecting suitable components.

The information related to computers on which numerical experiments performed will be presented in the results sections of the Chapters 2–7. Here, we present a brief information about the selected software resources.

Compilers: We have implemented our preconditioners using Fortran and used Intel Fortran compilers for this purpose. With this choice, we received highest performance among other compilers, on both Intel and AMD servers.

Numeric Kernels: Intel’s math kernel library (MKL) [69], which includes high-performance implementations of BLAS and LAPACK, are used as numeric kernels in our programs.

Message Passing Interface (MPI): MPI is a message passing standard designed to ease the development parallel programs. We have compared Intel MPI [70], Open MPI [71], and MVAPICH MPI [72], which are high-performance implementations of MPI over InfiniBand network connect. Our observations impelled us to use MVAPICH, which demonstrated the most successful results on our clusters considering both latency and bandwidth.

Iterative Solvers and ILU Preconditioners: We borrowed GMRES and ILU preconditioners from PETSc, which stands for portable, extensible toolkit for scientific computation [73, 74]. PETSc employs MPI for parallel programming and is developed for large-scale application projects. It also supports complex numbers and Fortran language.

1.12 Organization

This dissertation involves seven chapters, first of which is this introduction. In Chapter 2, we address ILU preconditioners, which is the most commonly used and well-established preconditioning method. We show how to use these preconditioners in CEM problems in a black-box form and in a safe manner. Despite their important advantages, ILU preconditioners are inherently sequential. Hence, for parallel solutions, a SAI preconditioner has been developed. We explain the parallel implementation details, such as load-balancing and pattern selection, in Chapter 3. We improve the performance of the SAI preconditioner by using it for the iterative solution of the near-field matrix system, which is used to precondition the dense system in an inner-outer solution scheme. This preconditioner, which we call the INF preconditioner, is explained in Chapter 4. The last preconditioner we develop for PEC problems uses the same inner-outer solution scheme, but employs an approximate version of MLFMA for the inner solutions. We give the details about the MLFMA approximations and tuning the inner solutions for efficiency in Chapter 5.

Chapter 6 is about preconditioning of matrix systems obtained from the discretization of dielectric problems. Unlike the PEC case, those matrix systems are in a partitioned structure. We exploit the partitioned structure for preconditioning by employing Schur complement reduction. In this way, we develop effective preconditioners, which render the solution of difficult real-life problems possible.

We conclude the dissertation in Chapter 7 by stating some concluding remarks and listing some of the future research areas that we foresee in preconditioning of CEM problems.

Chapter 2

Incomplete-LU (ILU)

Preconditioners

By not applying these algorithms blindly, for example, by looking at the structure of the matrix in this case, we were able to make ILU work.

E. Chow and Y. Saad, *Journal of Computational and Applied Mathematics*, Vol. 86, 1997.

2.1 Introduction

Since the sparse near-field matrix $\overline{\mathbf{A}}^{NF}$ 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

$$(\overline{\mathbf{A}}^{NF})^{-1} \cdot \overline{\mathbf{A}} \cdot \mathbf{x} = (\overline{\mathbf{A}}^{NF})^{-1} \cdot \mathbf{b}. \quad (2.1)$$

The inversion of the near-field matrix $\overline{\mathbf{A}}^{NF}$ 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 [75]. 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 $\overline{\mathbf{A}}^{NF}$ 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 [2]. 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 [76]. 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 [38]. 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 [76].

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 [76]. 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 [2].

In order to understand the quality of the preconditioner, or to understand the reason for failure when it occurs, we can use $\|(\bar{\mathbf{L}} \cdot \bar{\mathbf{U}})^{-1} \cdot \mathbf{e}\|_\infty$, where \mathbf{e} is the vector of ones. This statistic is called *condest* (for condition estimate) and it provides an upper bound for $\|(\bar{\mathbf{L}} \cdot \bar{\mathbf{U}})^{-1}\|_\infty$ [76]. 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 [76] and the wide availability of ILU-class preconditioners in various packages [73, 77, 78], 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 [79, 80, 81]. In [79], ILU(0) was tried on systems resulting from EFIE formulation with discouraging results in all test problems. Sertel and Volakis [80] 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 50000-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 *et al.* [81], 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 did they apply pivoting or any other techniques to increase the effectiveness of the preconditioner.

In the next section we briefly review ILU preconditioners. Then, in Section 2.3, we discuss the stability of ILU preconditioners and the ways to improve their conditioning. In Section 2.4, we compare open-surface and closed-surface problems, formulated with EFIE and CFIE, respectively. Then, we conclude in Section 2.5, where we propose a strategy for the selection of appropriate preconditioner among ILU-class preconditioners and suitable parameters that render them robust for CEM problems.

2.2 Preconditioners based on Incomplete LU Factorization

Various preconditioners have been used for the solution of CEM problems. For CFIE, a block-diagonal (or block-Jacobi) preconditioner (BDP) is frequently used. In an MLFMA setting, a BDP 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 BDP 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 [82, 83, 84, 85, 86].

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 pre-filtering and post-filtering threshold parameters carefully [85]. 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 [76, 87, 88].

For iterative solvers utilizing MLFMA, the near-field matrix $\overline{\mathbf{A}}^{NF}$ is the natural candidate to generate the incomplete factors. Consider an incomplete factorization of the near-field matrix, $\overline{\mathbf{A}}^{NF} \approx \overline{\mathbf{L}} \cdot \overline{\mathbf{U}}$. If we let the sparsity patterns of $\overline{\mathbf{A}}^{NF}$ 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 $\overline{\mathbf{A}}^{NF}$, we end up with the no-fill LU method, or ILU(0). This simple idea successfully works for well-conditioned matrices [2]. 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 [89]. 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 $\overline{\mathbf{A}}^{NF}$. 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.

2.3 Improving Stability of ILU Preconditioners

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.,

$$accuracy = \|\overline{\mathbf{E}}\| = \|\overline{\mathbf{A}} - \overline{\mathbf{L}} \cdot \overline{\mathbf{U}}\|. \quad (2.2)$$

Stability refers to how close the preconditioned matrix to the identity matrix and is measured by the norm of the preconditioned error, i.e.,

$$stability = \|(\overline{\mathbf{L}} \cdot \overline{\mathbf{U}})^{-1} \cdot \overline{\mathbf{E}}\|. \quad (2.3)$$

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 [76]. 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 $\|(\bar{\mathbf{L}} \cdot \bar{\mathbf{U}})^{-1}\|$, called *condest*, gives a clue about the instability of the triangular factors [76]. This condition estimate is defined as

$$\|(\bar{\mathbf{L}} \cdot \bar{\mathbf{U}})^{-1} \cdot \mathbf{e}\|_{\infty}, \quad \mathbf{e} = [1, \dots, 1]^T. \quad (2.4)$$

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 in the preconditioner. Diagonal shifts are already 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 [90]. 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 [87]. However, the effect of ordering becomes significant when the incomplete factors are allowed to be denser than the original matrix [38]. 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 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 a much simpler method. Column pivoting can be applied in a row-wise factorization with negligible cost. The resulting preconditioner is known as ILUTP [2]. In some cases, it may be useful to include a permutation tolerance $permtol$, and perform the permutation for the i th 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 [2].

2.4 Numerical 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 a 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 three, and the α parameter of CFIE to 0.2. The CPU times reported in this section are 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 1500. 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 number of 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 to the high cost of matrix-vector multiplications, particularly for large problems.

2.4.1 Open Geometries

Fig. 2.1 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 2.1. In Table 2.1 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

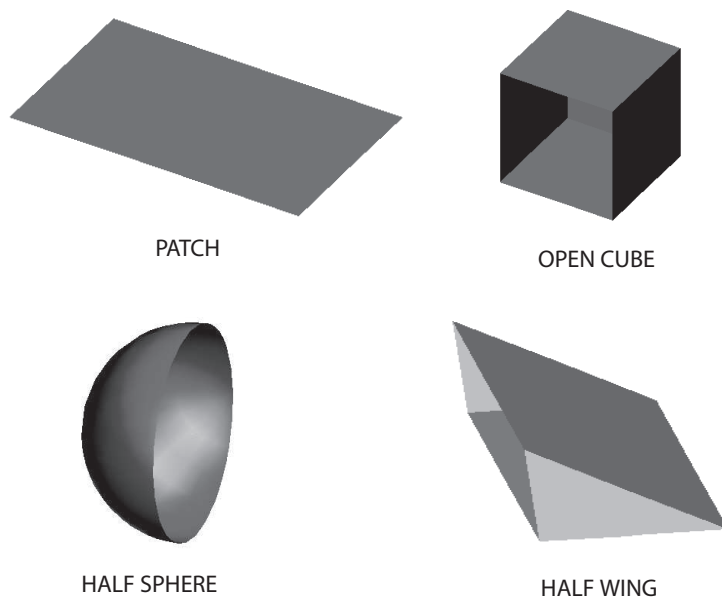


Figure 2.1: Open geometries used to compare ILU preconditioners.

the wavelength. As mentioned in Section 1.4, EFIE is the only choice for these geometries.

We compare the ILU-class preconditioners in Table 2.2. The summary of our observations are as follows:

- 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.
- 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 [76].

Table 2.1: Information about the open geometries used to compare ILU preconditioners.

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 2.2: ILU results for open geometries.

Problem	N	ILU(0)		ILUT		ILUTP5		ILUTP	
		<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>
P1	1301	189	37	83	22	73	21	69	21
P2	12249	60,855	228	712	42	309	39	5,606	56
P3	137792	6.3E+09	-	1,398	82	1,350	81	2,545	78
OC1	1690	59	76	14	37	12	35	11	33
OC2	16393	2,154	333	52	110	48	109	44	109
OC3	171655	9.6E+05	-	192	377	240	376	2,892	376
OP1	1562	198	65	41	27	50	26	151	39
OP2	14705	1.3E+05	416	161	98	164	92	151	91
OP3	163871	5.3E+05	-	948	268	835	253	2,424	251
HS1	1101	47	48	26	26	31	24	27	23
HS2	9911	990	248	1,095	73	126	46	95	45
HS3	116596	6.3E+05	-	1.5E+15	-	582	110	22,755	156

- 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 do not converge at all.

Since ILUTP5 is robust for all our geometries, in Table 2.3, we compare it with other commonly used preconditioners. BDP preconditioning performs poorer than the no-preconditioner case, hence the diagonal (or Jacobi) preconditioner (DP) is used instead. We emphasize the following observations:

Table 2.3: Comparison of ILU preconditioners for open geometries.

Problem	N	LU	DP		ILUTP5			SAI		
		<i>iter</i>	<i>iter</i>	<i>time</i>	<i>iter</i>	<i>setup</i>	<i>time</i>	<i>iter</i>	<i>setup</i>	<i>time</i>
P1	1301	15	201	15	21	1	3	25	101	103
P2	12249	26	431	503	39	33	88	45	1,524	1,573
P3	137792	53	833	16,209	81	661	2,167	92	19,955	21,384
OC1	1690	28	224	26	35	4	9	35	569	574
OC2	16393	97	617	854	109	141	273	114	15,040	15,167
OC3	171655	332	-	-	376	2,243	9,833	354	207,436	213,619
OP1	1562	18	315	39	26	2	5	48	663	668
OP2	14705	78	991	1,894	92	97	224	173	12,301	12,524
OP3	163871	195	-	-	253	996	6,883	396	57,606	66,093
HS1	1101	17	187	15	24	3	5	26	165	167
HS2	9911	38	490	748	46	107	186	61	1,712	1,813
HS3	116596	93	1052	25,947	110	1,353	3,579	156	22,079	25,066

- Although we use a robust solver, for a simple preconditioner such as DP, either the number of iterations turns out to be very high, or convergence is not attained in 1500 iterations. This is in good agreement with the conclusions derived in Section 2.2.
- ILUTP5 reduces iteration numbers by an order of magnitude compared to DP. 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.
- 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.
- Furthermore, the iteration counts reveal that the algebraic scalability of 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.

Table 2.4: Information about the closed geometries used to compare ILU preconditioners.

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

2.4.2 Closed Geometries

As mentioned in Section 1.4, both EFIE and CFIE can be used for closed geometries. However, CFIE yields better-conditioned systems, hence it is usually preferred to EFIE. Nonetheless, we will present some of the results obtained from EFIE for comparison purposes.

Fig. 2.2 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 [91]. Table 2.4 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 2.5, we compare ILU(0) and ILUT, by presenting

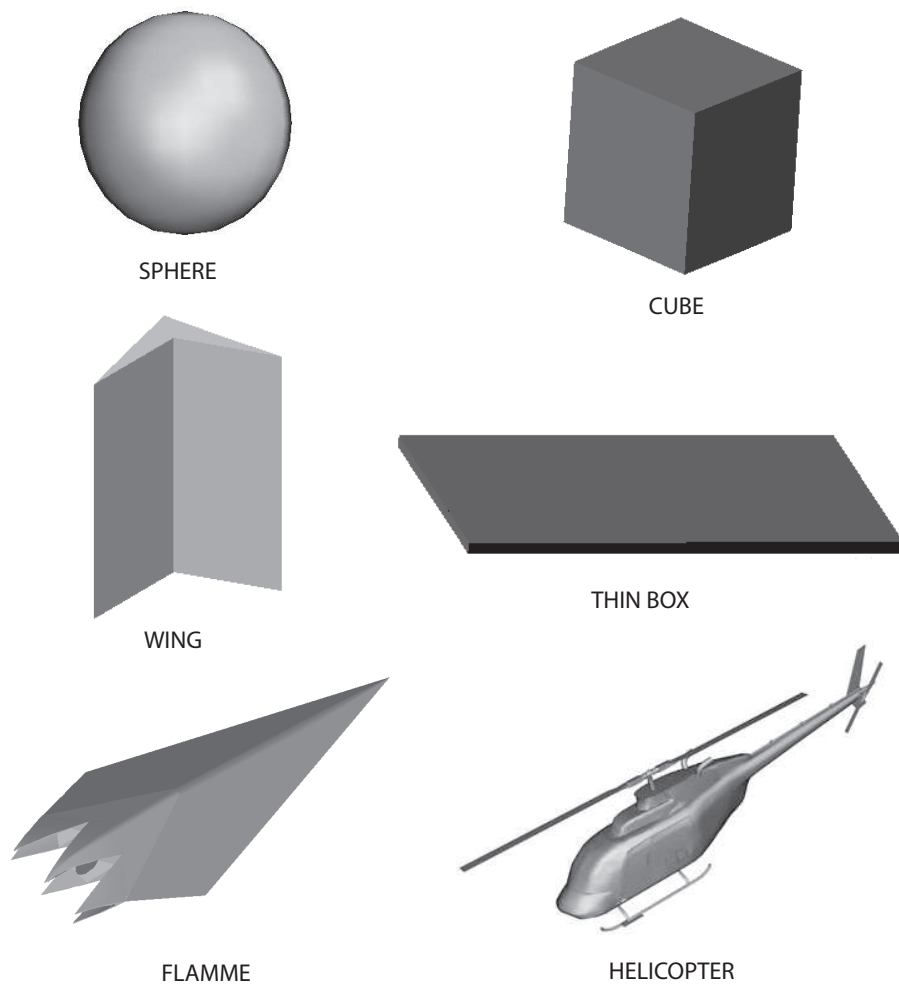


Figure 2.2: Closed geometries used to compare ILU preconditioners.

Table 2.5: ILU results for closed geometries using CFIE.

Problem	N	ILU(0)		ILUT	
		<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>
S1	930	8	13	3	13
S2	8364	24	21	9	20
S3	132003	108	29	108	29
C1	918	3	11	8	12
C2	8046	9	20	24	20
C3	131436	34	26	33	26
TB1	1650	13	11	12	10
TB2	10122	13	23	14	22
TB3	147180	97	45	96	42
W1	1050	8	10	8	9
W2	10512	26	16	26	15
W3	117945	83	32	82	32
F1	12750	174	24	150	23
F2	28866	198	34	207	33
F3	78030	327	66	325	65
H1	33423	6	30	6	30
H2	183546	18	44	18	44

the *condest* values and corresponding number of iterations for the systems obtained with CFIE. Pivoting does not change the iteration counts, 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 [4]. 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 2.6 shows the *condest* values and iteration numbers for ILU-class preconditioners. W3 does not converge in 1500 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.

Table 2.6: ILU results for closed geometries using EFIE. “MLE” stands for “Memory Limitation Exceeded.”

Problem	N	ILU(0)		ILUT		ILUTP(0.5)		ILUTP	
		<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>	<i>condest</i>	<i>iter</i>
S1	930	65	46	61	37	16	28	20	29
S2	8364	4.3E+04	416	3.4E+04	246	65	108	427	116
S3	132003	1.9E+06	-	1.5E+131	-	381	572	1,346	589
C1	918	22	-	9	32	7	30	7	29
C2	8046	3.1E+05	-	30	77	37	75	99	77
C3	131436	2.9E+07	-	210	574	181	563	800	557
TB1	1650	22	37	49	26	72	30	22	27
TB2	10122	1.8E+05	-	414	169	166	151	7.5E+05	-
TB3	147180	1.1E+14	-	48,600	1090	13,410	1084	867	709
W1	1050	128	-	38	23	38	22	43	30
W2	10512	6.6E+04	-	240	102	88	95	106	91
W3	117945	4.5E+07	-	538	-	540	-	1,455	-
F1	12750	4.7E+06	-	1,043	184	623	159	1,006	170
F2	28866	9.5E+07	-	2,011	421	2,012	393	2,071	440
F3	78030	1.5E+09	-	2,830	1106	3,066	1042	85,812	1131
H1	33423	1,359	469	38	206	39	203	59	206
H2	183546	29,184	MLE	61	MLE	71	MLE	1,799	MLE

In Table 2.7, for CFIE, ILU(0) is compared to the BDP (where only the self interactions of the smallest MLFMA clusters are used in the near-field matrices) and the SAI preconditioner. We summarize the results as follows:

- For canonical geometries, compared to the BDP 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-RHS solutions, ILU(0) may be still preferable.
- For quasi-canonical geometries and real-life problems, ILU(0) performs remarkably better compared to the BDP 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.
- Even though SAI has iteration numbers similar to ILU(0), the setup time of SAI is too large.

Table 2.7: Comparisons of ILU preconditioners for closed geometries using CFIE.

Problem	N	LU	BDP		ILU(0)			SAI		
		<i>iter</i>	<i>iter</i>	<i>time</i>	<i>iter</i>	<i>setup</i>	<i>time</i>	<i>iter</i>	<i>setup</i>	<i>time</i>
S1	930	13	17	1	13	0	1	14	229	230
S2	8364	20	23	23	21	6	23	21	1,453	1,474
S3	132003	29	32	684	29	23	665	29	23,102	23,722
C1	918	11	18	0	11	1	1	12	941	941
C2	8046	20	25	17	20	2	16	21	2,170	2,184
C3	131436	26	28	419	26	28	485	27	25,066	25,489
TB1	1650	9	44	3	11	2	3	20	132	135
TB2	10122	21	60	40	23	6	22	33	10,468	10,489
TB3	147180	37	106	1,290	45	271	1,025	64	298,479	299,301
W1	1050	9	30	1	10	1	1	13	970	971
W2	10512	15	39	31	16	7	21	21	14,445	14,462
W3	117945	31	52	779	32	46	542	37	73,100	73,587
F1	12750	21	77	89	24	11	40	40	22,930	22,976
F2	28866	32	81	264	34	20	130	45	42,214	42,389
F3	78030	63	115	1,096	66	43	694	76	96,369	96,369
H1	33423	30	125	326	30	40	142	51	94,150	94,282
H2	183546	42	106	3,081	44	145	1,739	61	234,614	236,463

- 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.
- 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 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.

2.5 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 DP 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 BDP preconditioner for real-life problems. Iteration numbers obtained with ILU(0) are very close 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.

Despite the success of ILU techniques, both construction and application of those preconditioners are inherently sequential. Therefore, it is difficult to obtain parallel scalability with ILU. For this reason, we develop other preconditioners that provides high scalability for parallel solutions. These will be the subject of the next three chapters.

Chapter 3

Sparse-Approximate-Inverse (SAI) Preconditioners

Nevertheless, it is often the case that many of the entries in the inverse of a sparse matrix are small in absolute value, thus making the approximation of $\overline{\mathbf{A}}^{-1}$ with a sparse matrix possible.

Michele Benzi, *Journal of Computational Physics*, Vol. 182, 2002.

3.1 Introduction

Iterative solutions of linear systems using Krylov subspace methods make it possible to solve large-scale scientific problems with modest computing requirements [1]. Effective parallelization of the matrix-vector multiplication, which is used at least once in an iteration, and the iterative solvers are possible, allowing even larger systems to be solved with cost-effective parallel computers [2]. However, iterative solvers usually require preconditioning in order to be effective.

Most preconditioners use methods similar to direct solution techniques, rendering their parallelization a difficult task. As a result, preconditioning is currently an important bottleneck for the solution of large scientific problems [38].

Constructing parallel and efficient preconditioners for CEM applications can also be difficult. MLFMA stores only the near-field matrix, which is composed of the interactions of the neighboring (touching) boxes or clusters in the lowest level of the tree structure. When the ordering of the unknowns is in accordance with the cluster membership, the near-field matrix takes a block structure. In Fig. 1.3, we show the near-field patterns of three problems involving sphere geometries of different sizes. Both the maximum size of the blocks and the maximum number of the nonzero blocks per row are fixed. Therefore, as the problem size increases, the near-field matrix becomes sparser. Since preconditioners are usually built from near-field matrices, effective preconditioning of CEM problems may become a challenge, particularly for large problem sizes.

Nonetheless, effective utilization of the near-field matrix provides strong preconditioners for problems up to certain large sizes. Each element of the matrix represents the electromagnetic interaction of a basis function and a testing function. The Green's function used for the computation of the matrix elements decays with $1/R$, where R is the distance between the pair of basis and testing functions under consideration. Due to this rapid decay of the Green's function, magnitudes of the matrix elements display a variety. The general trend of this variety obeys physical proximity, i.e., basis and testing functions that are close to each other are expected to have strong electromagnetic coupling, resulting in matrix elements with relatively larger magnitudes. Therefore, the sparse near-field matrix is likely to retain the most relevant contributions of the dense matrix. The exact inverse of such a sparse matrix is, in general, a dense matrix. Nevertheless, the inverse matrix also displays a similar variation among the magnitudes

of its elements. Hence, the inverse matrix can also be approximated by a sparse matrix.

In this work, we consider sparse approximate inverse (SAI) preconditioners for large CEM problems. This is partly because an efficient parallelization of the more standard ILU preconditioners [40] is difficult for matrices with unstructured sparsity patterns [38]. Application of the SAI preconditioners to CEM problems in the context of MLFMA has been analyzed by the CERFACS group [92] and by Lee et al. [85]. Here, we present an effective construction scheme with an effective load-balancing method that produces high parallel efficiency. We also propose to use the near-field pattern for the approximate inverse with filtered matrices and then compare different filtering strategies. Moreover, for conductor problems, the earlier work [92] considered only the electric-field integral equation (EFIE). However, for conducting geometries with closed surfaces, the combined-field integral equation (CFIE) should also be considered. Even though EFIE can also be used in such problems, this has no practical use since CFIE can solve the closed-surface problems much faster. Furthermore, we show that CFIE solutions of large real-life problems with closed surfaces can benefit more from SAI than from simple preconditioners, such as the block-diagonal preconditioner.

This chapter is organized as follows. After presenting a brief summary of the SAI preconditioners in the next section, we dwell upon the implementation details in Section 3.3. In particular, we explain pattern selection and filtering strategies. For a parallel implementation, we present a load-balancing algorithm and show how the communication in the construction phase can be efficiently performed. The results section analyzes CFIE and EFIE problems separately. Then, in Section 3.5, we discuss some conclusions.

3.2 Brief Review of SAI

In each step of an iterative method, preconditioning is performed by backward and forward solves for ILU preconditioners. In contrast, SAI preconditioners are based on approximating the inverse of the matrix directly. For this purpose, an approximate inverse is explicitly constructed and stored. Then, the preconditioner is applied by a sparse matrix-vector multiplication. In the context of MLFMA, we use $\overline{\mathbf{A}}^{NF}$ to generate the preconditioner and our approximation is of the form $\overline{\mathbf{M}} \approx (\overline{\mathbf{A}}^{NF})^{-1}$.

In this work, we concentrate on the SAI preconditioners derived from the Frobenius norm minimization. There are two other classes of approximate inverses that have been proposed in the literature [93]. One of the classes involves the factorized sparse approximate inverses. Two important members of this group, FSAI [94] and AINV [95], have already been tried on CEM problems and their performances have been discouraging [79]. The third group of SAI preconditioners are the inverse ILU techniques, which consist of approximately inverting an incomplete factorization of the matrix. Because of the initial incomplete factorization phase, the inverse ILU methods have some serious drawbacks for parallel computing [93].

3.2.1 Methods Derived from the Frobenius Norm Minimization

For this class of preconditioners, the approximate inverse of the near-field matrix is computed by minimizing

$$\left\| \overline{\mathbf{I}} - \overline{\mathbf{M}} \cdot \overline{\mathbf{A}}^{NF} \right\|_F. \quad (3.1)$$

The approximation is implemented by forcing $\overline{\mathbf{M}}$ to be sparse. With the Frobenius norm choice, the minimization can be performed independently for each row

by using the identity

$$\left\| \bar{\mathbf{I}} - \bar{\mathbf{M}} \cdot \bar{\mathbf{A}}^{NF} \right\|_F^2 = \sum_{i=1}^N \left\| \mathbf{e}_i - \mathbf{m}_i \cdot \bar{\mathbf{A}}^{NF} \right\|_2^2, \quad (3.2)$$

where \mathbf{e}_i is the i th unit row vector and \mathbf{m}_i is the i th row of the preconditioner.

Various preconditioners have been developed with different pattern selection and minimization techniques. Saad and Chow proposed to solve each equation

$$\left(\bar{\mathbf{A}}^{NF} \right)^T \cdot \mathbf{m}_i^T = \mathbf{e}_i^T \quad (3.3)$$

iteratively and approximately [96]. One way to do this is to use the first few iterations of the generalized minimal residual (GMRES) solver. However, the cost of this method is of order $\mathcal{O}(N^2)$ for sparse matrices, assuming a fixed number of nonzero elements per row. To avoid this high cost, the authors proposed to keep the iterates and other vectors sparse, as well as the matrix. This is done by filtering iterates as they become denser. Then, matrix-vector multiplications are carried out in sparse-sparse mode. However, such a multiplication scheme is not efficiently implemented with MLFMA.

Considering the difficulty in finding a suitable nonzero pattern for the approximate inverse, Grote and Huckle [97] proposed to find the sparsity pattern adaptively starting with an initial sparsity pattern. Construction time of this preconditioner can be very high [93], hence it should be used only if simpler methods fail.

On the other hand, the nonzero structure of the near-field matrix itself is a natural candidate for the nonzero pattern of the SAI preconditioner. The storage scheme used for the block-sparse matrices consumes less memory than regular sparse matrices. Moreover, as noted in [92], when using the block structure of the near-field matrix, QR factorization involved in the least-squares solutions of (3.2) can be done once for each diagonal block, which corresponds to self interactions of the last-level clusters in MLFMA. In this way, construction time of

the preconditioner can be reduced substantially. If filtering is required, however, the block structure is distorted and both the setup time and memory consumption of the preconditioner can be even higher than the no-filtering case. Moreover, in a parallel implementation, load balancing should be ensured and communications in the construction phase should be carefully performed, since this phase involves all-to-all exchanges of rows. In the next section, we analyze these issues in more detail.

3.3 Parallel Implementation Details

In this work, we adopt K-way row-wise conformable partitionings of the near-field matrix $\overline{\mathbf{A}}^{NF}$, the approximate inverse $\overline{\mathbf{M}}$, and the right-hand-side (RHS) vector \mathbf{b} as

$$\overline{\mathbf{A}}^{NF} = \begin{bmatrix} \overline{\mathbf{A}}_1^{NF} \\ \vdots \\ \overline{\mathbf{A}}_k^{NF} \\ \vdots \\ \overline{\mathbf{A}}_K^{NF} \end{bmatrix}, \quad \overline{\mathbf{M}} = \begin{bmatrix} \overline{\mathbf{M}}_1 \\ \vdots \\ \overline{\mathbf{M}}_k \\ \vdots \\ \overline{\mathbf{M}}_K \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_k \\ \vdots \\ \mathbf{b}_K \end{bmatrix}, \quad (3.4)$$

where $\overline{\mathbf{A}}_k^{NF}$ and $\overline{\mathbf{M}}_k$ are $N_k \times N$ submatrices, \mathbf{b}_k is an $N_k \times 1$ subvector, and

$$\sum_{k=1}^K N_k = N. \quad (3.5)$$

Process P_k holds $\overline{\mathbf{A}}_k^{NF}$, \mathbf{b}_k , and $\overline{\mathbf{M}}_k$. However, we use a different partitioning for $\overline{\mathbf{M}}$ during the generation of SAI, as explained in Section 3.3.3.

The construction of the SAI preconditioner is accomplished by solving

$$\mathbf{m}_i \cdot \overline{\mathbf{A}}^{NF} = \mathbf{e}_i \quad \text{for } i = 1, 2, \dots, N \quad (3.6)$$

subject to sparsity conditions. Left-preconditioning is consistent with row-wise decomposition, because in this scheme the computations involve the rows of the

original matrix, and each row-block $\overline{\mathbf{M}}_k$ of the approximate inverse is generated by a different process. However, for the EFIE formulation, which produces symmetric complex matrices, right-preconditioning is also viable. This can be accomplished by a transpose matrix-vector multiplication operation in the application phase, as will be detailed in Section 3.3.5.

In a row-wise decomposition of the matrix, each process P_k solves part of (3.6). For a given row $i \in P_k$, let $I \subset \{1, 2, \dots, N\}$ denote the set of column indices j for which $\mathbf{m}_i(j)$ is nonzero. Then, only the rows of the near-field matrix included in this set affect the solution. Therefore, the i th minimization problem is reduced to

$$\mathbf{m}_i(I) \cdot \overline{\mathbf{A}}^{NF}(I, :) = \mathbf{e}_i. \quad (3.7)$$

This step incurs a communication among the processes, because not all of the rows in I belong to P_k . Hence, P_k requires some sparse rows (*i.e.*, nonzero values and column indices) from other processes. Once $\overline{\mathbf{A}}^{NF}(I, :)$ is formed, because of the sparsity of the near-field matrix, some of the columns of $\overline{\mathbf{A}}^{NF}(I, :)$ will be zero. Denoting the indices of the nonzero columns by J , the $N \times N$ problems in (3.6) are reduced to $n_1 \times n_2$ problems

$$\mathbf{m}_i(I) \cdot \overline{\mathbf{A}}^{NF}(I, J) = \mathbf{e}_i(J) \quad \text{for } i = 1, 2, \dots, N, \quad (3.8)$$

where n_1 and n_2 are the number of elements in the sets I and J , respectively. An example of reduction of a 10×10 sparse matrix for the 4th row is illustrated in Fig. 3.1. In this example, the sparsity pattern of SAI is the same as that of the original matrix.

The $n_2 \times n_1$ problems

$$\overline{\mathbf{A}}^{NF}(I, J)^T \cdot \mathbf{m}_i(I)^T = \mathbf{e}_i(J)^T \quad (3.9)$$

can be solved, by first computing the reduced QR factorization

$$\overline{\mathbf{A}}^{NF}(I, J)^T = \overline{\mathbf{Q}} \cdot \overline{\mathbf{R}}, \quad (3.10)$$

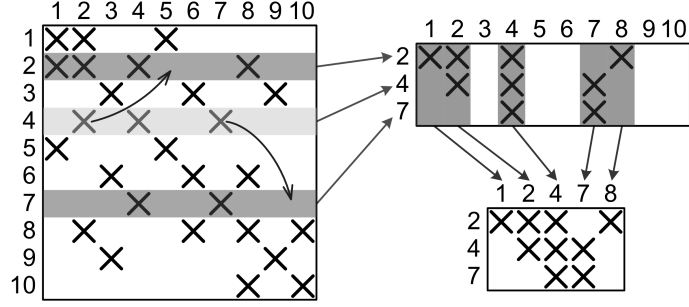


Figure 3.1: Reduction of a 10×10 matrix for the generation of the 4th row of SAI.

and then obtaining the solution as

$$\mathbf{m}_i(I)^T = \overline{\mathbf{R}}^{-1} \cdot \overline{\mathbf{Q}}^H \cdot \mathbf{e}_i(J)^T. \quad (3.11)$$

For the near-field matrix, the maximum number of nonzero elements in a row or a column is fixed for a given problem irrespective of the problem size. Therefore, a constant number of rows are involved in (3.7), i.e., $n_1 = \mathcal{O}(1)$. In $\overline{\mathbf{A}}^{NF}(I, :)$, each row again contains a fixed number of nonzero entries. The worst case occurs when the locations of the nonzero elements do not coincide for all rows $i \in I$. Even in that case, $n_2 = \mathcal{O}(1)\mathcal{O}(1) = \mathcal{O}(1)$. This makes the complexity of the SAI preconditioner $\mathcal{O}(N)$. On the other hand, the QR factorization used in the solution of the $n_2 \times n_1$ least-squares problem requires asymptotically $n_2 n_1^2$ flops, causing the setup time of the SAI preconditioner be high, even though it has a low complexity.

Because of this possible high construction cost, the implementation of the SAI preconditioner deserves close attention. The following subsections will detail the main steps for the generation and application of the preconditioner.

3.3.1 Pattern Selection and Filtering

In a Frobenius-norm minimization technique that depends on a fixed inverse pattern, the main issue for an efficient and effective preconditioner is the selection

of an appropriate sparsity pattern. Because of the possible high cost of the SAI preconditioner, filtering is used in general. Filtering refers to dropping small elements from the original matrix. Then, the preconditioner is constructed from this sparser matrix. In our work, we have used the algorithm detailed in Fig. 3.2 for filtering.

```

find the largest entry  $max_k$  of  $\overline{\mathbf{A}}_k^{NF}$ 
 $global\_max = \text{reduce\_all\_maximum}(max_k)$ 
for each  $a_{ij} \in \overline{\mathbf{A}}_k^{NF}$  do
    if  $a_{ij}/global\_max < threshold$  then
        drop  $a_{ij}$ 
    endif
endfor

```

Figure 3.2: Filtering algorithm.

Filtering only decreases n_2 if a different pattern from the filtered matrix is used for the approximate inverse. In that case, n_1 is determined by the pattern of the approximate inverse. However, filtering causes smaller n_1 and n_2 values if the pattern of the filtered matrix is used for the approximate inverse.

Considering MLFMA and the special structure of the near-field matrix, we think that the following pattern selection and filtering strategies are appropriate for low-cost SAI generation:

No Filtering

In this strategy, no filtering is applied to the near-field matrix and the same pattern is used for the approximate inverse. Because of the block structure of $\overline{\mathbf{A}}^{NF}$, all rows of the SAI preconditioner that reside in the same diagonal block require the same rows of $\overline{\mathbf{A}}^{NF}$ for their computation; hence, the reduced matrices $\overline{\mathbf{A}}^{NF}(I, :)$ and $\overline{\mathbf{A}}^{NF}(I, J)$ become the same. Therefore, QR factorization

is done only once for each diagonal block and the least-squares solution is obtained for multiple RHSs. Since the least-squares solution is dominated by the QR factorization, substantial savings can be achieved.

Preserving Block Structure in Filtering

Even though the near-field matrices become sparser as the number of unknowns increases, filtering may be required. This may be due to the possible high cost of SAI construction or because of some special problems, such as densely packed meta-material structures [49], which produce denser near-field matrices. To be able to use the advantage of the block structure, we suggest using the near-field pattern for the approximate inverse and filtering only the near-field matrix, from which the approximate inverse is generated. In this way, the row size n_1 does not change, but n_2 can be much smaller. Hence, we expect cheaper construction time compared to the no-filtering case.

Using a Filtered Pattern for the Approximate Inverse

If we use the block structure for the approximate inverse, the memory requirement of the preconditioner will be the same as the near-field matrix, which is the largest data in MLFMA. One way to reduce the memory cost is to use a filtered pattern for the preconditioner. On the other hand, with this strategy, we will not be able to use the advantage of the block structure. Therefore, we have to perform N factorizations instead of N/m , where m is the average size of the diagonal blocks. Hence, substantial filtering should be employed in order to decrease the memory and construction costs with this scheme.

Block Filtering

Another strategy to take advantage of the block structure can be to drop an entire block, instead of only the nonzero entries, with the hope that dropped blocks do not carry significant information. To determine which blocks to drop, the Frobenius norm of each one is computed; those having a relative norm smaller than a prescribed tolerance are dropped.

3.3.2 Communication Phase and Enlarging the Local Submatrix

After a suitable pattern is selected for SAI, each process P_k exchanges some rows of $\overline{\mathbf{A}}_k^{NF}$ with others. In this way, they enlarge their local submatrix $\overline{\mathbf{A}}_k^{NF}$, so that no communication is required during the generation of $\overline{\mathbf{M}}_k$. For this purpose, P_k scans the nonzero pattern of $\overline{\mathbf{M}}_k$ and decides which rows it needs for the generation of the k th block. Then, after an all-to-all communication, each process learns the row identities it has to send. This communication pattern is detailed in Fig. 3.3.

```
for each  $m_{ij} \in \overline{\mathbf{M}}_k$  do
  if  $j$  is not marked then
     $p = \text{findProcId}(j)$ 
    append  $j$  into rowRecvList[ $p$ ]
    mark  $j$ 
  endif
endfor
send rowRecvList; receive into rowSendList           ! All-to-all communication
```

Figure 3.3: The pseudocode that finds the rows to be sent by the process P_k .

However, the information obtained is not sufficient for the exchange of rows because the processes do not yet know the column indices of the nonzero entries

```

for each row  $i \in rowSendList$  do
    append column indices of row  $i$  to  $sendColIndices$ 
endfor
send  $sendColIndices$ ; receive into  $recvColIndices$            ! All-to-all communication

```

Figure 3.4: The pseudocode that finds the column indices of the sparse rows to be received by the process P_k .

of the rows to be received. Hence, another scan and exchange of data regarding to the column indices is performed. Finally, the values are exchanged. These two steps are illustrated in Fig. 3.4 and Fig. 3.5.

```

for each row  $i \in rowSendList$  do
    append  $a_{ij}$  to  $sendColValues$ 
endfor
send  $sendColValues$ ; receive into  $recvColValues$            ! All-to-all communication

```

Figure 3.5: The pseudocode that exchanges the sparse rows.

The near-field matrix and SAI are held in compressed sparse row (CSR) format. This has two advantages. First, access to memory is minimized for the sparse matrix-vector multiplications. More importantly, with CSR storage, the access of the matrix is done by rows, hence the communications in the last two steps are done in-place.

There could be another way to exchange the rows, in which the communication is done during the construction of the SAI preconditioner. This approach allows communications and computations to overlap by exchanging data for the next row while computing the current row. However, in this method, a row can be exchanged many times. Moreover, as shown in Section 3.4.1, our implementation produces superior parallel performance; hence, we did not need to try this alternative strategy.

3.3.3 Load Balancing of SAI

Load Balancing for the Generation Phase

The computation of the nonzero elements of the near-field matrix constitutes an expensive part of the setup phase in MLFMA. In this part, the cost of a row is proportional to the number of nonzero elements in that row. To ensure load balancing, the rows of the near-field matrix are distributed among the processes so that each process acquires approximately an equal number of nonzero elements. Since the application of the near-field matrix in the iterative phase is also proportional to the number of nonzero elements in a submatrix, this approach serves the load balancing of the near-field matrix-vector multiplication as well.

On the other hand, the cost of the generation of the i th row \mathbf{m}_i of SAI is proportional to $n_2 n_1^2$, where n_1 and n_2 are the dimensions of the reduced matrix. Note that n_1 is the number of the nonzero elements in that row if filtering is not applied. If the near-field partitioning is also used for SAI, this high cost can cause the SAI generation to be unbalanced. For this reason, we repartition the near-field matrix in accordance with the load-balancing scheme of the SAI setup.

After the pattern of SAI is decided, we can quickly determine the cost of each row by finding n_1 and n_2 values. Then, the workload of SAI is distributed among the processes so that each process has approximately equal amount of work. Alternatively, it is also possible to apply an incremental partitioning to existing near-field partitioning to decrease the overhead of repartitioning, as detailed in [98]. We follow the former approach, where we use a separate partitioning for the SAI generation that is different from the partitioning of the near-field matrix. This way, we obtain a better load balance, and we can still limit the overhead of repartitioning by overlapping the communications with computations, as explained in the next section.

```

for each row  $i \in R_k^{NF}$  do
  if row  $i \notin R_k^{SAI}$  then
     $p = \text{findProcId}(i)$ 
    start the reception of  $m_i$  from  $p$            ! Non-blocking communication
  endif
endfor
for each row  $i \in R_k^{SAI}$  do
  if row  $i \notin R_k^{NF}$  then
     $p = \text{findProcId}(i)$ 
    generate  $m_i$  and start the transfer to  $p$    ! Non-blocking communication
  endif
endfor
for each row  $i \in R_k^{SAI}$  do
  if row  $i \in R_k^{NF}$  then
    generate  $m_i$ 
  endif
endfor
finish all non-blocking communications

```

Figure 3.6: Redistribution of the SAI rows according to the near-field partitioning. R_k^{NF} and R_k^{SAI} denote the row indices of process k with respect to the near-field and SAI partitionings, respectively.

Load Balancing for the SAI Application

To ensure load balancing for the application phase, we have to redistribute the rows of SAI according to the near-field partitioning. The overhead of this data transfer can be eliminated by overlapping communications with computations, as detailed in Fig. 3.6. In the first loop, all processes initiate the receptions of the rows that they should have with respect to the near-field partitioning, but they do not generate. Then, all processes generate those rows in their SAI partitioning that do not belong to themselves and initiate their transfers to appropriate processes. While the communications take place, local computations, *i.e.*, the generation of the rows that belong to process k with respect to both near-field and SAI partitionings, are performed. Finally, all processes wait for the non-blocking communications to finish.

3.3.4 Construction of the Preconditioner

For the generation of the i th row \mathbf{m}_i , first a map of length N is prepared to map the sets I and J to $\bar{I} = \{1, 2, \dots, n_1\}$ and $\bar{J} = \{1, 2, \dots, n_2\}$, respectively. Then, we form the $n_2 \times n_1$ dense matrix

$$\overline{\mathbf{A}}^{NF}(\bar{I}, \bar{J})^T = \overline{\mathbf{A}}^{NF}(\bar{J}, \bar{I}). \quad (3.12)$$

Finally, we solve the least-squares problem

$$\overline{\mathbf{A}}^{NF}(\bar{J}, \bar{I}) \cdot \mathbf{m}_i(I)^T = \mathbf{e}_i(J)^T \quad (3.13)$$

via QR factorization and generate the i th row of $\overline{\mathbf{M}}_k$.

3.3.5 Application of the Preconditioner

The application of the preconditioner is performed with the sparse matrix-vector multiplication $\mathbf{y}_k = \overline{\mathbf{M}}_k \cdot \mathbf{x}$. Since P_k computes \mathbf{x}_k , an expand operation, *i.e.*, $\mathbf{x} = \text{expand}(\mathbf{x}_k)$ (also known as “gather-all”), is required before the multiplication so that all processes possess the entire \mathbf{x} vector.

For EFIE, using the symmetry of the near-field matrix, we have

$$\left\| \bar{\mathbf{I}} - \overline{\mathbf{M}} \cdot \overline{\mathbf{A}}^{NF} \right\|_F = \left\| \bar{\mathbf{I}} - (\overline{\mathbf{M}} \cdot \overline{\mathbf{A}}^{NF})^T \right\|_F = \left\| \bar{\mathbf{I}} - \overline{\mathbf{A}}^{NF} \cdot \overline{\mathbf{M}}^T \right\|_F. \quad (3.14)$$

Therefore, right-preconditioning can be achieved with the operation $\mathbf{y}^k = (\overline{\mathbf{M}}_k)^T \cdot \mathbf{x}_k$, or equivalently $(\mathbf{y}^k)^T = (\mathbf{x}_k)^T \cdot \overline{\mathbf{M}}_k$, where $\mathbf{y} = \sum_{k=1}^K \mathbf{y}^k$. This multiplication can be done in CSR format using the outer product form of matrix-vector multiplication, *i.e.*,

$$\mathbf{y}^k = \sum_{i=1}^{N_k} \mathbf{x}_k(i) \overline{\mathbf{M}}_k(i, :). \quad (3.15)$$

We outline this operation in Fig. 3.7. Finally, a fold operation, *i.e.*, $\mathbf{y}_k = \text{fold}(\mathbf{y}^k)$ (also known as “reduce-scatter”) is required so that partial sums \mathbf{y}^k are summed across the processes, and each process P_k ends up with the k th subvector \mathbf{y}_k of \mathbf{y} .

```

 $\mathbf{y}^k = 0$ 
for  $i = 1$  to  $N_k$  do
     $xval = \mathbf{x}_k(i)$ 
     $kStart = \mathbf{IA}(i); kEnd = \mathbf{IA}(i + 1) - 1$ 
    for  $k = kStart$  to  $kEnd$  do
         $j = \mathbf{JA}(k)$ 
         $\mathbf{y}^k(j) = \mathbf{y}^k(j) + xval * \mathbf{VA}(k)$ 
    endfor
endfor

```

Figure 3.7: The pseudocode for the sparse matrix-vector multiplication used for right preconditioning. \mathbf{IA} , \mathbf{JA} , and \mathbf{VA} are respectively row-index, column-index, and value arrays of $\overline{\mathbf{M}}_k$, which is stored in CSR format.

3.4 Results

In this section, we present the parallel performance of the generation phase of the SAI preconditioner. Then, for EFIE and CFIE formulations, we compare different versions of SAI with other preconditioners.

The solutions presented in this section are obtained on a 16-node cluster connected with an Infiniband network. Each node includes two quad-core Intel Xeon processes and 16 GB of RAM. All of the results are obtained on 32 cores (4 processes on each node). For robustness, we use the generalized minimal residual method (GMRES) with no restart as the solver. Contrary to results presented in [92], orthogonalization cost of GMRES is negligible, compared to the time spent on the matrix-vector multiplications. For example, the largest problem shown in this study involves 3,838,496 unknowns. For the solution of this problem, the time spent on GMRES orthogonalization is only 2.3% of the time spent on matrix-vector multiplications by MLFMA. We use zero as the initial guess and set the stopping criteria as a six orders of magnitude relative decay in the initial residual or a maximum of 1,000 iterations. In our MLFMA implementation, we use the Rao-Wilton-Glisson (RWG) functions [7] for both

basis and testing functions. We set the size of the smallest clusters to 0.25λ and the number of accurate digits to three. Three digits of accuracy has proven to yield accurate results, as shown in [99] by comparing the numerical results with the analytical ones for the sphere problem formulated with CFIE. For the patch problem formulated with EFIE, accuracy is demonstrated by comparing the numerical solution with a physical optics solution that gives accurate results at some specific observation angles for high frequencies [48].

3.4.1 Parallel Performance of the Construction Phase

In Fig. 3.8, we show the speedup curves for the construction of SAI with no filtering for a patch geometry with 344,000 unknowns, a half sphere with 408,064 unknowns, and for the stealth target Flamme [91], which has 312,120 unknowns. To show the worst-case performance, the processes are distributed so that the inter-node communications are maximized. Thanks to our efficient parallelization scheme and the load-balancing method, we obtain superior speedups for all problems.

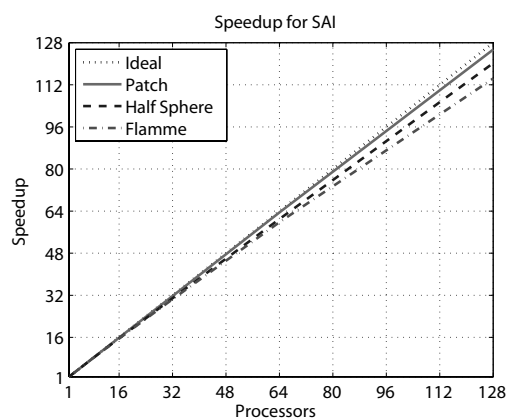


Figure 3.8: Speedup curves for the patch, half sphere, and Flamme problems.

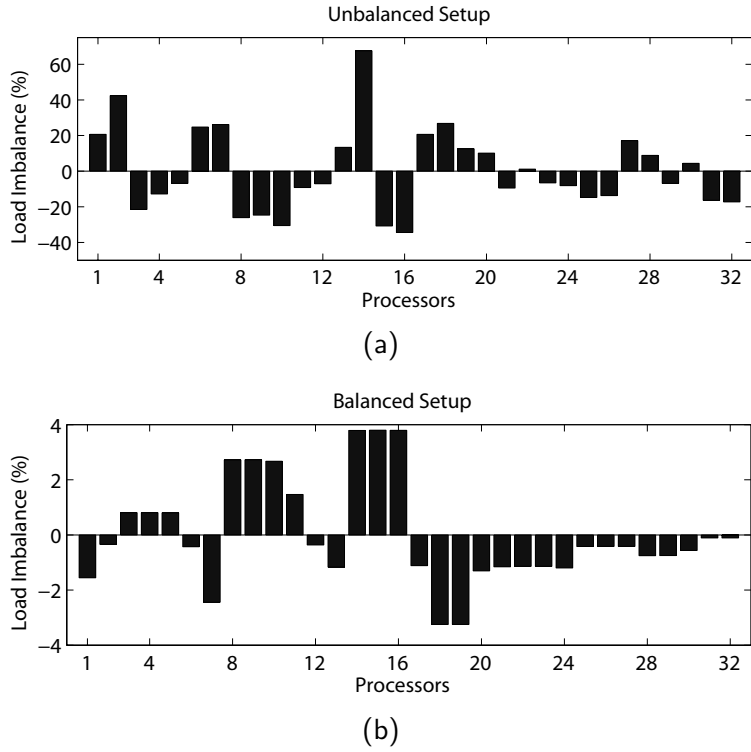


Figure 3.9: Load imbalance of the Flamme problem for (a) unbalanced and (b) balanced cases.

The effect of the load-balancing algorithm is demonstrated in Fig. 3.9 on the Flamme problem. The load imbalance ε_k of process k is defined as

$$\varepsilon_k = \frac{time_k - time_{avg}}{time_{avg}}. \quad (3.16)$$

In (3.16), $time_k$ is the setup time of SAI for process k , and $time_{avg}$ is the average setup time. Particularly for complex geometries, such as Flamme, adopting the same partitioning of the near-field for SAI can cause significant imbalance and inefficiency. Using the proposed load-balancing method, we reduce the average imbalance of 18.5% to 1.5% and achieve high efficiency.

3.4.2 EFIE Results

The sample geometries that are solved with EFIE in this paper are illustrated in Fig. 3.10, and their quantitative features are listed in Table 3.1. Only open

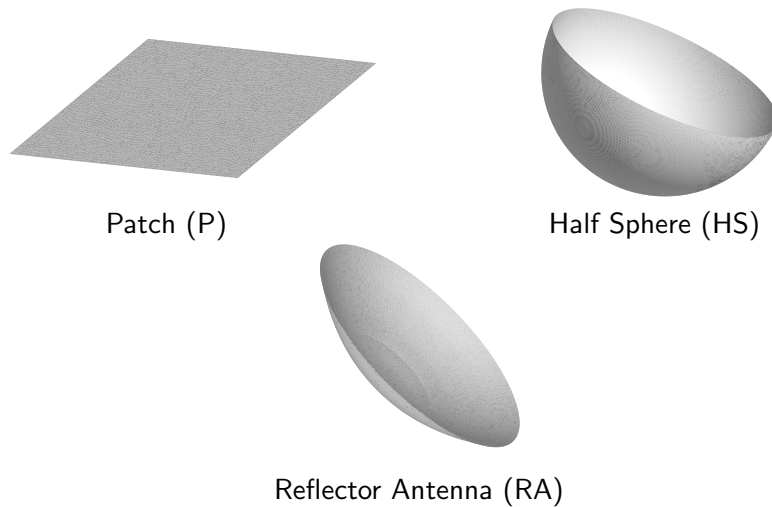


Figure 3.10: Open geometries used in EFIE problems.

Table 3.1: Quantitative features of the open geometries.

Problem	Frequency (GHz)	Size (λ)	MLFMA Levels	N
P1	16	16	7	85,840
P2	32	32	8	344,000
P3	64	64	9	1,377,280
P4	96	96	10	3,062,400
HS1	16	32	8	101,888
HS2	32	64	9	408,064
HS3	64	96	10	1,633,280
HS4	96	192	10	3,838,496
RA1	4	13	7	47,870
RA2	8	27	8	187,144
RA3	16	53	9	748,024
RA4	32	107	10	2,991,067

geometries are solved with EFIE since closed geometries can be solved more easily with CFIE. In Table 3.1, patch is abbreviated with P, half sphere with HS, and reflector antenna with RA. The “Size” value stands for the diameter of the half sphere and the reflector antenna, and it is the length of one side for the square patch. Mesh lengths are chosen as one-tenth of the corresponding wavelength.

The experimental results indicate that there is no significant difference between left and right preconditioning of SAI. For consistency with the CFIE results, we prefer left preconditioning with EFIE. The results for the no-preconditioning case and comparisons of the three types of SAI preconditioners are shown in Tables 3.2 and 3.3, respectively. We omit the results with the

Table 3.2: The solutions with no preconditioning for open geometries formulated by EFIE.

Problem	Iterations	Time (s)
P1	814	346
P2	> 1,000	-
P3	> 1,000	-
P4	> 1,000	-
HS1	913	1,363
HS2	> 1,000	-
HS3	> 1,000	-
HS4	> 1,000	-
RA1	795	446
RA2	> 1,000	-
RA3	> 1,000	-
RA4	> 1,000	-

block-filtering version of SAI, because it performs worse than other SAI preconditioners. We also omit the results with the block-diagonal preconditioner (BDP), because it deteriorates the convergence rate, compared to no preconditioning. In Table 3.3, “Ratio” stands for the ratio of the sparsity of the filtered near-field matrix to the original near-field matrix. “Setup” stands for the generation time of SAI and “Time” for the solution time, both in seconds. For EFIE, we set the threshold of filtering at 0.5%.

We outline our observations as follows:

- Without an effective preconditioner, EFIE solutions converge only for small problems. On the other hand, SAI preconditioners solve all problems within reasonable iteration counts. Even for those that converge without preconditioning, SAI with no filtering decreases the iteration counts by an order of magnitude for the patch and the reflector antenna, and by six times for the half sphere.
- When we apply filtering, using the block structure of the near-field matrix for the approximate inverse decreases setup times significantly. Even for the reflector antenna, for which 90% of the near-field entries are dropped with filtering, setup times of SAI preconditioners that use the near-field

Table 3.3: Comparison of SAI preconditioners for open geometries formulated by EFIE.

Problem	Filtered Pattern				Near-Field Pattern			No Filtering		
	Ratio	Setup	Iter	Time	Setup	Iter	Time	Setup	Iter	Time
P1	51%	14	94	41	2	91	38	2	74	31
P2	50%	62	139	224	10	132	209	10	109	174
P3	49%	370	194	1,431	45	190	1,384	48	157	1,147
P4	50%	1,495	243	7,849	129	231	7,368	132	194	6,225
HS1	73%	32	159	246	4	146	217	5	132	196
HS2	73%	133	266	1,762	19	246	1,583	20	221	1,424
HS3	71%	703	426	12,382	87	392	11,235	92	351	10,046
HS4	59%	2,512	599	30,828	343	570	28,295	350	480	23,458
RA1	6%	0	599	336	1	228	127	2	63	36
RA2	7%	1	859	1,890	9	557	1,230	9	93	204
RA3	36%	80	171	1,539	33	173	1,552	37	139	1,266
RA4	13%	1,142	598	22,269	148	303	11,144	201	200	7,276

Notes: “Ratio” is the ratio of the sparsity of the SAI to that of the near-field matrix. “Setup” and “Time” denote the setup and solution times, given in seconds. “Iter” denotes the number of iterations.

pattern are much smaller. However, for large simulations, memory savings can be an important motivation to use the filtered pattern. For example, for RA4, SAI with a filtered pattern requires 840 MB of RAM, whereas SAI with the near-field pattern and SAI with no filtering require 4.2 GB of RAM. However, we note that there should be considerable filtering to provide memory gain, because the format used in a block-structured sparse matrix is more economical than regular sparse matrices.

- In terms of the solution times, SAI with no filtering produces the best results for all geometries. The setup times of the filtered SAI that uses the near-field pattern are the lowest, except for RA1 and RA2; however, SAI with no filtering is more successful in reducing the iteration counts and solution times.
- We observe superior algebraic scalability for the unfiltered SAI preconditioner. For all targets, the largest problem is approximately 64 times larger than the smallest, whereas the iteration count of the P4 is only 2.6 times that of P1, that of HS4 is 3.6 times that of HS1, and that of RA4 is only 3.2 times that of RA1.

Finally, in Fig. 3.11, we demonstrate the Arnoldi estimates for the eigenvalues of the RA4 problem. These estimates are found as a byproduct of the GMRES solver, and they are known to approximate the bounding eigenvalues of the spectrum [1]. SAI with filtered pattern leaves some of the eigenvalues in the left half-plane, and there are many small eigenvalues around the origin, accounting for its slow convergence. If filtered pattern is used for the approximate inverse or filtering is not applied at all, then all of the eigenvalues are clustered in the right half-plane. However, SAI with no filtering produces also smaller radius for the spectrum, hence converges faster.

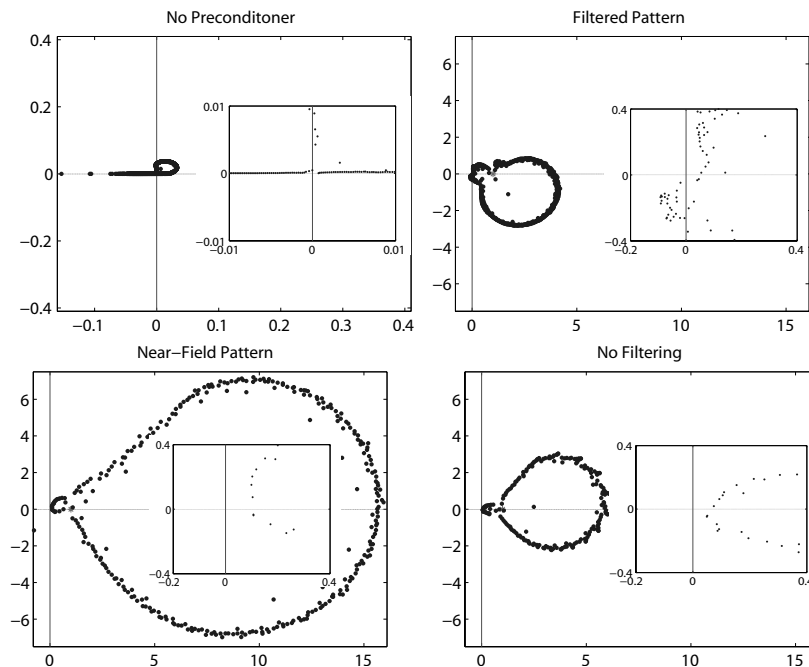


Figure 3.11: Approximate eigenvalues of the RA4 problem on the complex plane.

3.4.3 CFIE Results

Many real-life problems confronted in CEM involve complicated structures enclosing a volume. Due to its favorable properties, CFIE is the preferred integral-equation formulation for those targets with closed surfaces. In Fig. 3.12, we illustrate two such geometries, a helicopter (H) and the Flamme (F). We solve these problems at increasing frequencies, as detailed in Table 3.4.

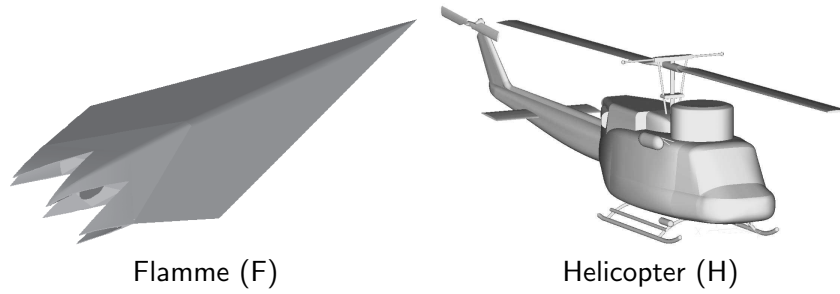


Figure 3.12: Closed-surface geometries used in CFIE problems.

Table 3.4: Quantitative features of the closed geometries.

Problem	Frequency (GHz)	Size (λ)	MLFMA Levels	N
F1	10	20	8	78,030
F2	20	40	9	312,120
F3	40	80	10	1,248,480
F4	60	120	10	3,166,272
H1	0.3	14	8	46,383
H2	0.6	28	9	185,532
H3	1.2	56	10	742,128
H4	2.4	112	11	2,968,512

Contrary to EFIE, BDP is the commonly used preconditioner for CFIE problems. BDP has negligible setup time and is easily parallelized. In addition, BDP enables fast convergence for a variety of problems due to the diagonal-dominance behavior of CFIE matrices to some extent. Hence, we first provide the solutions of the closed-surface problems with BDP in Table 3.5, and then compare different versions of SAI preconditioners in Table 3.6. With CFIE, we use a smaller threshold value for filtering, *i.e.*, 0.05%, because such a small threshold causes significant filtering due to the diagonal-dominance feature of CFIE matrices.

Table 3.5: The solutions with BDP for closed-surface problems formulated by CFIE.

Problem	Iterations	Time (s)
F1	116	122
F2	122	563
F3	211	4,451
F4	347	11,734
H1	99	73
H2	109	391
H3	121	1,939
H4	138	10,192

Table 3.6: Comparison of SAI preconditioners for closed-surface problems formulated by CFIE.

Problem	Filtered Pattern				Near-Field Pattern			No Filtering		
	Ratio	Setup	Iter	Time	Setup	Iter	Time	Setup	Iter	Time
F1	25%	31	99	133	14	81	95	17	76	90
F2	21%	55	113	583	43	96	490	53	97	493
F3	19%	196	198	4,253	124	181	3,916	163	174	3,836
F4	33%	1,889	318	10,767	328	297	9,112	374	316	9,530
H1	11%	7	93	76	8	51	45	12	51	48
H2	6%	3	110	391	20	84	326	41	59	241
H3	6%	43	123	2,019	78	97	1,619	152	80	1,403
H4	6%	1,176	152	11,701	366	114	8,649	644	97	7,515

Notes: “Ratio” is the ratio of the sparsity of the SAI to that of the near-field matrix. “Setup” and “Time” denote the setup and solution times, given in seconds. “Iter” denotes the number of iterations.

We summarize our observations and comments about the CFIE results as follows:

- We observe that both the filtered SAI that uses the near-field pattern and the unfiltered SAI decrease the number of iterations and total solution times with respect to BDP for both problems. For instance, if we compare the largest targets, the solution times of F4 and H4 are shortened by 20% and 26%, respectively.
- In terms of the solution time, the unfiltered SAI is the most successful preconditioner, except for F2 and F4. Surprisingly, for these problems, obtaining the preconditioner from a sparser matrix instead of the original near-field matrix improves performance.

3.4.4 Solutions of Metamaterial Structures

We finish this section with the solutions of metamaterial structures. Metamaterials are artificial structures that are constructed by periodically arranging unit cells, such as split-ring resonators (SRRs) and thin wires. Due to the resonant nature of the cells, electromagnetic properties of the host medium, i.e.,

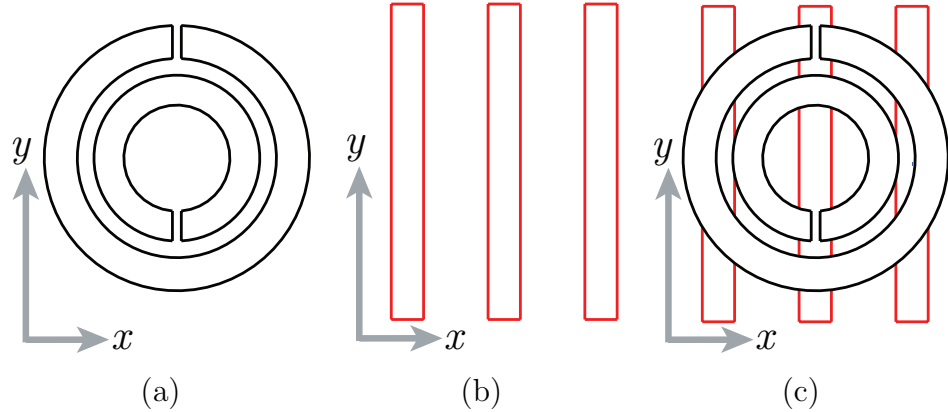


Figure 3.13: Unit cells that are used to construct various metamaterial walls: (a) SRR, (b) thin wires, and (c) a combination of SRR and thin wires.

permittivity, permeability, or both, can effectively become negative for some frequencies. Because of these unique features, metamaterials have been utilized in various applications, such as sub-wavelength focusing [100],[101], cloaking [102], and designing improved antennas [103]. It is possible to solve scattering problems involving three-dimensional metamaterials hundreds and even thousands of unit cells at a time with MLFMA. Since the conductor surfaces are modeled by perfectly conducting sheets with zero thicknesses, it is mandatory to use EFIE for metamaterials.

Fig. 3.13 presents the unit cells that are used to construct the metamaterial structures investigated in this study. A single SRR, which is depicted in Fig. 3.13(a), has dimensions in the order of microns. The SRR resonates at about 100 GHz, when it is located in a medium with a relative permittivity of 4.8 [104]. Around the resonance frequency, the SRR stimulates negative effective permeability in the medium. Dimensions of the thin wires depicted in Fig. 3.13(b) are compatible with the dimensions of the SRRs and they exhibit negative effective permittivity in a wide range of frequencies, including 100 GHz. Finally, as depicted in Fig. 3.13(c), we also consider composite metamaterials (CMMs) by combining SRRs and thin wires in the same medium to obtain a double-negative property.

In Fig. 3.14, we present the solutions of an $18 \times 11 \times 4$ SRR wall discretized with 64,944 unknowns. Fig. 3.14 depicts the solution times required as a function of frequency. For a fair comparison, we include the setup time required by the preconditioners, together with the solution time. Without using a preconditioner, the processing time is less than 400 seconds at all frequencies, except for 95 GHz. Due to a numerical resonance, the processing time increases to 1700 seconds at 95 GHz. Using BDP, the processing time is reduced at ordinary frequencies, as opposed previous problems presented. This is due to a finer mesh size used to model small unit cells. Nonetheless, the solution at 95 GHz is again decelerated, compared to the no-preconditioner case. On the other hand, using a sparse-approximate-inverse preconditioner with a threshold parameter of 0.05 and with a filtered sparsity pattern, the solution is accelerated at all frequencies, including the resonance frequency, again compared to the no-preconditioner case. The processing time is reduced to 800 seconds at 95 GHz, corresponding to less than half the time required without preconditioning. The gain obtained by using the sparse-approximate-inverse preconditioner is more significant for larger metamaterial problems.

3.5 Conclusion

In this work, we analyze SAI preconditioning for dense linear systems arising from the discretization of integral equations. We describe in detail practical issues, such as pattern selection, filtering, and load balancing to obtain a highly parallel and efficient preconditioner.

For large open-surface problems that are modelled by EFIE, linear systems can be challenging to solve. We conclude that, for such problems, it is better to avoid filtering and to construct a SAI preconditioner that has the same block structure as the near-field matrix. The use of the block structure has advantages

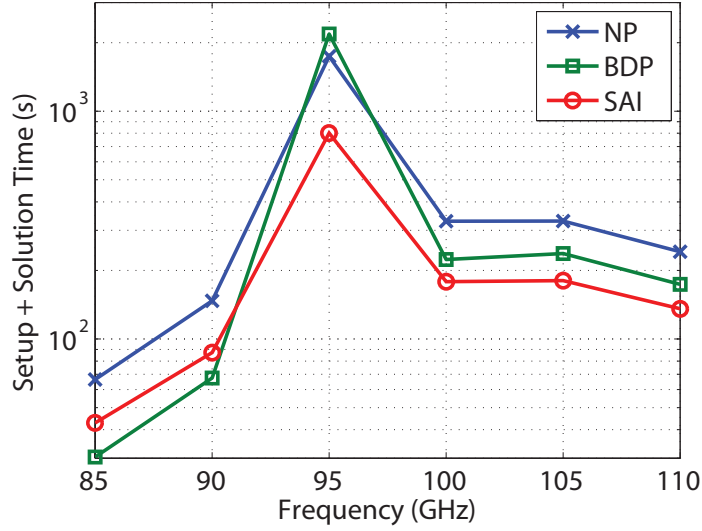


Figure 3.14: Processing time including the iterative solution and the setup of the preconditioner for the $18 \times 11 \times 4$ SRR wall. “NP” represents the no-preconditioner case.

in reducing the setup cost and memory requirement of the preconditioner. However, if filtering is required for further memory saving, we show that our filtering strategy is robust.

For complex closed-surface problems that can make use of the well-conditioned CFIE, we show that SAI is more beneficial than the commonly used BDP. The benefit will be even more dominant for the computation of backscattering with different incident angles, which requires the solution of linear systems involving many RHSs.

Chapter 4

The Iterative Near-Field (INF) Preconditioner

Yet another possibility is to approximate the action of $\overline{\mathbf{S}}^{-1}$ on a vector \mathbf{v} by performing a few steps of an iterative method on the Schur complement system $\overline{\mathbf{S}} \cdot \mathbf{z} = \mathbf{v}$.

Michele Benzi, Gene H. Golub, and Jörg Liesen, *Acta Numerica*, Vol. 14, 2005.

4.1 Introduction

To achieve a strong preconditioner in surface-integral-equation methods employing MLFMA, the information provided by the near-field matrix should be effectively used. As mentioned before, using the exact factorization of the near-field matrix for preconditioning is too expensive because of fill-ins. In Chapter 2, we show that, among various ILU preconditioners, ILUT [89] and ILUTP [2] are successful in CEM problems and produce iteration counts that are very close to

those obtained by the exact factorization of the near-field matrix. However, because ILU methods are inherently sequential, we resorted to SAI preconditioners in parallel MLFMA implementations, as detailed in the previous chapter.

On the other hand, it is widely observed that SAI is not as successful as ILU in reducing the number of iterations and the solution times [93]. As an alternative strategy, the entire near-field matrix can be used in an iterative solver for preconditioning purposes. This can be accomplished with low cost and complexity since Krylov subspace solvers merely require matrix-vector multiplications and the near-field matrix is sparse. Therefore, the preconditioning solution can be obtained by another iterative process, nested in the outer solver, provided that the outer Krylov subspace solver is flexible. With this strategy, we propose to use the iterative solution of the near-field system as a preconditioner for the original system, which is also solved iteratively. Furthermore, we use a fixed preconditioner obtained from the near-field matrix as a preconditioner to the inner iterative solver. MLFMA solutions of several model problems establish the effectiveness of the proposed nested iterative near-field preconditioner, allowing us to report the efficient solution of electric-field and combined-field integral-equation problems involving difficult geometries and millions of unknowns.

In the next section, we compare the proposed preconditioner with preconditioners that can make use of dense system matrix through MLFMA, such as the one that will be described in the next chapter. Then, we detail the proposed preconditioning method in Section 4.3. In Section 4.4, we present the numerical results and comparisons of INF with SAI. We summarize our conclusions in Section 4.5.

4.2 Near-Field versus Full-Matrix Preconditioners

Since only the interactions corresponding to the lowest-level near-field clusters are kept in memory, it is common practice to construct preconditioners from $\overline{\mathbf{A}}^{NF}$, assuming that it is a good approximation to $\overline{\mathbf{A}}$. However, since the size of the lowest-level clusters is kept fixed in MLFMA, the number of nonzero elements in a row of $\overline{\mathbf{A}}^{NF}$ also remains constant. Therefore, $\overline{\mathbf{A}}^{NF}$ becomes increasingly sparser as the problem size grows. As a result, it has been shown that preconditioners that make use of the full $\overline{\mathbf{A}}$ matrix, as in some nested-solver schemes [105], are usually stronger than preconditioners that depend on only near-field interactions [92, 58].

Nonetheless, for matrices obtained from the discretizations of surface integral equations, magnitudes of matrix elements change with physical proximity, as a general trend. Therefore, the available near-field matrix $\overline{\mathbf{A}}^{NF}$ is likely to preserve the most relevant contributions of the dense system matrix. As Section 4.4 will reveal, the proposed preconditioner renders solutions of large EFIE problems possible with modest iteration counts by effectively using all information provided by the near-field matrix. The results will also reveal that the scaling of iteration counts with respect to increasing problem sizes is remarkably favorable, e.g., iteration counts increase less than three fold, even when problem sizes increase 36 fold in some cases. Furthermore, once a fixed preconditioner, such as SAI, is constructed, the proposed scheme has no extra costs in terms of setup time and memory. On the other hand, preconditioners that make use of the full matrix require less-accurate versions of MLFMA, which can be obtained using extra setup time and significant amounts of memory.

4.3 The Iterative Near-Field Preconditioner

It is known that SAI is not as successful as ILU with the same amount of memory [93]. We confirm this assertion by comparing SAI with the exact solution of the near-field matrix, which we name NF-LU. Though ILUT produces iteration counts very close to those of NF-LU [40], SAI deviates from this optimum behavior as the number of unknowns increases. For a remedy, increasing the density of the preconditioner is undesirable because of a possible high setup time and memory considerations.

On the other hand, an iterative solution of the near-field matrix can be used as a preconditioner, provided that the original system is solved using a flexible solver [2]. Since SAI is a good approximation to the inverse of the near-field matrix, the iterative solution of the near-field system can be accelerated using SAI as a preconditioner. This approach produces a nesting of the iterative solvers. For the outer solver that solves the original system, we use the flexible GMRES (FGMRES) method, which allows the preconditioner to change from iteration to iteration [2]. The preconditioner of this solver is another preconditioned Krylov subspace solver, which we call the inner solver. We solve the sparse near-field system in the inner solver using SAI as the fixed preconditioner. We illustrate this nested inner-outer preconditioning scheme in Figure 4.1.

Outer solver: FGMRES; solve $\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$
Matrix-vector product: MLFMA
Inner solver (Preconditioner): GMRES; solve $\overline{\mathbf{A}}^{NF} \cdot \mathbf{v} = \mathbf{w}$
Matrix-vector product: Sparse mat-vec
Fixed preconditioner: SAI

Figure 4.1: Nested solvers for iterative near-field preconditioning.

Since the inner solver is used for preconditioning purposes, a rough solution is adequate. We use GMRES as the inner solver since it provides a fast drop of the residual norm in early iterations.

The proposed scheme, which we name the iterative near-field (INF) preconditioner, yields a forward-type preconditioner, as the ILU preconditioner is. The difference is that, in ILU preconditioning, the preconditioner approximates the near-field matrix in factorized form, i.e., $\overline{\mathbf{M}} = \overline{\mathbf{L}} \cdot \overline{\mathbf{U}} \approx \overline{\mathbf{A}}^{NF}$, but the system $\overline{\mathbf{M}} \cdot \mathbf{v} = \mathbf{w}$ is solved exactly by using backward and forward solves for a given vector \mathbf{w} . On the other hand, for INF, the preconditioner is the exact near-field matrix, i.e., $\overline{\mathbf{M}} = \overline{\mathbf{A}}^{NF}$, but we approximately solve the system $\overline{\mathbf{M}} \cdot \mathbf{v} = \mathbf{w}$ with an iterative method.

4.4 Numerical Results

In this section, we compare the performances of the SAI and INF preconditioners since the SAI preconditioner has been widely used and proven to be successful in parallel implementations of integral-equation methods [86, 85, 83, 44]. Furthermore, when the near-field matrix pattern is selected as the nonzero pattern of the approximate inverse, the setup time of the SAI preconditioner can be lowered using the block structure, as shown in [92, 44]. Regarding the stopping criteria of the inner solver for the INF preconditioner, we conclude that the one-order residual drop provides a successful preconditioner that can be attained in a few iterations. Hence, we set the stopping criteria of the inner solver as a one-order residual drop from the initial residual norm or a maximum of five iterations, whichever is satisfied first.

4.4.1 EFIE Results

For small problems, we can evaluate the quality of the SAI and INF preconditioners by comparing them with a preconditioner obtained from the exact factorization of the near-field matrix. This preconditioner, which we call NF-LU, can be used only as a benchmark due to its excessive memory and setup costs. Nonetheless, it is useful for evaluation purposes since its iteration count is expected to be the minimum that can be achieved with a preconditioner constructed from the near-field matrix. Then, we can evaluate other preconditioners on the basis of how close their iteration counts are to those of NF-LU.

In Table 4.1, we present the solutions of three geometries with various preconditioners, i.e., the diagonal preconditioner (DP), SAI, INF, NF-LU, and the no-preconditioner case (No PC). Computations are performed on a 16-core parallel cluster constructed with eight dual-core AMD Opteron 870 processors in a symmetric multiprocessing (SMP) configuration. The geometries are depicted in Figure 2.1. We choose geometries with open surfaces, since closed-surface geometries can be solved more easily using CFIE. Mesh size is chosen as one-tenth of the wavelength at the frequency of operation. Due to its robustness, we use GMRES (FGMRES for INF) with no-restart as the iterative solver. We set the the initial guess as a vector of zeros and the stopping criterion as either a six-order-of-magnitude relative decay from the initial residual or a maximum of 1,000 iterations. In our MLFMA implementation, the size of the smallest clusters is fixed to 0.25 wavelength and the number of accurate digits to three.

The results presented in Table 4.1 show that the SAI preconditioner succeeds in accelerating the convergence of these relatively small problems since their solutions without a preconditioner or with DP require either several hundreds of or more than 1,000 iterations. On the other hand, the iteration counts are not close to those of NF-LU. This observation can be interpreted as that there is more room for improvement between an approximate inverse generated with the SAI

Table 4.1: Experimental results for comparing the SAI and INF preconditioners to NF-LU.

Geometry	N	No PC		DP		SAI		INF		NF-LU
		Iter	Time	Iter	Time	Iter	Time	Iter	Time	Iter
Patch	12,249	447	106	432	103	44	12	29	9	26
	137,792	894	3,241	851	3,087	91	336	59	253	53
Half Sphere	9,911	514	178	485	438	60	24	40	17	38
	116,596	-	3,257	-	3,259	156	510	103	383	93
Reflector Antenna	12,142	564	453	545	458	44	22	28	13	27
	105,570	-	4,285	-	4,288	80	344	51	236	49

Notes: “Iter” denotes the number of iterations and “Time” denotes the solution times. A dash “-” indicates that convergence is not attained in 1,000 iterations.

preconditioner and the exact inverse computed with NF-LU for benchmarking purposes. One can actually increase the density of the approximate inverses using two different tree structures for MLFMA and for the construction of the SAI preconditioner, as detailed in [92], but this comes at the cost of extra memory, which is a potential source of problem for large CEM computations. With the INF preconditioner, however, we achieve iteration counts that are very close to those of NF-LU. This means that the INF preconditioner makes good use of the available sparse near-field matrix and produces nearly optimal approximations for the inverse. In addition, these approximations are achieved in at most five iterations; hence the solution times are also decreased significantly.

To further assess the performance of the INF preconditioner, we solve larger instances of the problems in Figure 3.10 with increasing frequencies, as shown in Table 4.2. The solutions of these problems are carried out on 32 cores of an eight-node cluster interconnected with an Infiniband network. Each node of the cluster has two Intel Xeon 5345 quad-core processors and 32 GB of RAM. We note that none of the problems in Table 4.2 can be solved without an effective preconditioner even if the no-restart GMRES solver is used.

Iteration counts and timings pertaining to the solutions of the problems listed in Table 4.2 for the SAI and INF preconditioners are presented in Table 4.3. These results indicate that the proposed INF preconditioner consistently achieves

Table 4.2: Quantitative features of the open-surface geometries used for the numerical experiments.

Geometry	Frequency (GHz)	Size (λ)	MLFMA Levels	N
P1	32	32	8	344,000
P2	64	64	9	1,377,280
P3	96	96	10	3,062,400
P4	128	128	11	5,511,680
HS1	32	64	9	408,064
HS2	64	96	10	1,633,280
HS3	96	192	10	3,838,496
HS4	128	256	11	6,535,168
RA1	8	27	8	187,144
RA2	16	53	9	748,024
RA3	32	107	10	2,991,067
RA4	48	160	11	6,849,398
Notes: “Size” denotes the edge length for the patch and the diameter for the sphere. λ denotes the wavelength at the frequency of operation.				

better performance than the SAI preconditioner in all cases. The INF preconditioner decreases the solution times of the patch and reflector antenna problems by about 30% and those of the half-sphere problem by about 25%, with respect to the SAI preconditioner.

In each iteration, GMRES stores the preconditioned residual vector [2], hence its memory cost can be significant for large problems when the number of iterations is high. In Table 4.4, we present the parallel memory costs (per process) of GMRES for solutions with SAI and INF preconditioners. We also present the memory consumptions of MLFMA and the SAI setup. Since the sparsity pattern of SAI is the same as that of the near-field matrix, we do not need to store indexing arrays for SAI [44]. As a result, the amount of memory required by SAI is much less than that of MLFMA. On the other hand, memory amounts required by GMRES are significant and they are even higher than those of the SAI setup. INF reduces the iteration counts with respect to the SAI preconditioner, but

Table 4.3: Experimental results for comparing the SAI and INF preconditioners.

Geometry	SAI Setup	SAI		INF		
		Iter	Time	Inner Iter	Outer Iter	Time
P1	10	109	174	217	73	132
P2	48	157	1,147	316	106	812
P3	132	194	6,225	391	131	4,393
P4	308	234	27,902	478	160	19,620
HS1	20	221	1,424	480	160	999
HS2	92	351	10,046	780	260	7,258
HS3	350	480	23,458	1,101	367	18,374
HS4	839	546	66,778	1,218	406	51,285
RA1	9	93	204	184	62	136
RA2	37	139	1,266	272	95	832
RA3	201	200	7,276	408	138	5,138
RA4	671	252	31,784	509	172	22,404

Notes: “SAI Setup” denotes the construction time of SAI (in seconds) and applies to both SAI and INF. “Time” denotes the solution times, given in seconds. “Inner Iter” and “Outer Iter” denote the total number of inner and outer iterations.

GMRES memory for INF is larger than that of SAI, since for INF we use the flexible version of GMRES, whose memory cost is twice that of usual GMRES. Nonetheless, we note that the memory consumption of GMRES is much less than that of MLFMA.

4.4.2 CFIE Results

We investigate the performance of the INF preconditioner on two closed-surface problems formulated with CFIE. Even though CFIE is expected to produce better-conditioned systems compared to the EFIE formulation of open geometries, the two closed-surface problems are selected as particularly difficult real-life problems. These problems involve a wing geometry (W) and a helicopter (H), as illustrated in Figure 4.2. The wing geometry (W) has sharp edges and corners. The helicopter geometry (H) has a closed surface, but with very thin features and complicated surfaces, causing the deterioration of its condition numbers.

Table 4.4: Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.

Geometry	MLFMA	SAI/INF Setup	GMRES	
			SAI	INF
P1	78	16	9	12
P2	261	64	52	70
P3	430	139	142	191
P4	2,955	256	307	421
HS1	201	17	22	31
HS2	788	69	137	202
HS3	1,769	169	439	672
HS4	3,145	277	851	1,265
RA1	87	8	4	6
RA2	327	33	25	34
RA3	1,274	133	143	197
RA4	3,114	313	407	556

The quantitative features of the various numerical experiments are listed in Table 4.5. Both the wing (W) and the helicopter (H) problems are discretized with very large numbers of unknowns, 7.5 million and 13 million, respectively. Furthermore, the surface of the real-life helicopter (H) geometry is triangulated with three different mesh types, and each mesh type is created with three different mesh sizes, hence obtaining 9 different problems. For example, H3₁ in Table 4.5 denotes the third mesh size for the first mesh type. This is a very realistic approach since different mesh generators and different users of mesh generators produce different types of meshes, which, in turn, influence the condition of the resulting matrix equations. We will demonstrate the effectiveness of the INF preconditioner on these difficult real-life problems.

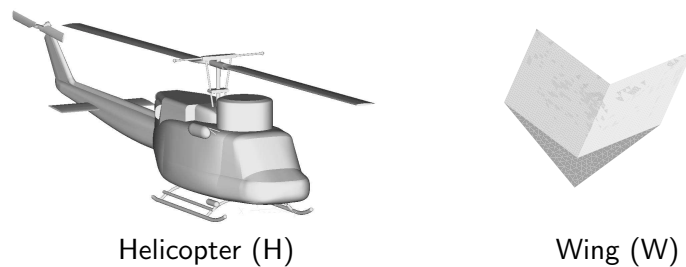


Figure 4.2: Closed-surface geometries formulated with CFIE.

Table 4.5: Quantitative features of the closed-surface geometries used for the numerical experiments.

Geometry	Frequency (GHz)	Size (λ)	MLFMA Levels	N
W1	4	13	7	117,945
W2	8	27	8	471,780
W3	16	53	9	1,887,120
W4	32	107	10	7,548,480
H1 ₁	1.3	74	10	556,515
H2 ₁	2.6	147	11	2,226,060
H3 ₁	5.2	295	12	8,904,240
H1 ₂	1.4	79	10	644,133
H2 ₂	2.8	159	11	2,576,532
H3 ₂	5.6	317	12	10,306,128
H1 ₃	1.6	91	10	817,260
H2 ₃	3.2	181	11	3,269,040
H3 ₃	6.4	363	12	13,076,160

Notes: “Size” denotes the largest dimension, i.e., edge length of the smallest cube enclosing the geometry. λ denotes the wavelength at the frequency of operation.

For the closed-surface problems, in addition to DP, SAI, and INF, we consider also the block-diagonal preconditioner (BDP), which is commonly used with the CFIE formulation. BDP is obtained by exactly solving the diagonal blocks that represent the self-interactions of the lowest-level clusters of the MLFMA tree structure (Figure 1.2). Iteration counts and timings of the solutions are compared in Table 4.6. With the INF preconditioner, we observe a significant decrease in the solution time. For the wing geometry, the gain is about 40% with respect to BDP and 25% to 35% with respect to the SAI preconditioner. For the real-life helicopter problem, which has thin and complicated surfaces, the gain is about 27% to 57% with respect to BDP, and 16% to 25% with respect to the SAI preconditioner.

We further analyze helicopter solutions in Figure 4.3, where we plot the total solution times, including the setup and solution times of the preconditioner. For all instances of problem sizes and mesh types, the INF preconditioner consistently provides faster solutions than the other preconditioners. Figure 4.3 shows that all

Table 4.6: Experimental results for comparing the INF preconditioner with DP, BDP, and the SAI preconditioner for closed-surface problems.

Geometry	DP		BDP		SAI	SAI		INF		
	Iter	Time	Iter	Time	Setup	Iter	Time	Inner	Iter	Time
W1	100	61	60	37	12	42	34	60	31	22
W2	127	300	78	186	33	57	150	78	40	111
W3	166	1,667	98	985	111	74	832	103	53	617
W4	211	8,951	131	5,559	576	96	4,139	212	65	3,166
H1 ₁	170	2,722	115	1,848	54	75	1,249	202	55	960
H2 ₁	170	12,581	115	8,490	172	92	7,026	222	74	5,771
H3 ₁	195	65,151	134	44,804	644	112	38,164	273	91	31,293
H1 ₂	169	2,996	110	1,871	62	77	1,374	197	57	1,045
H2 ₂	170	12,892	114	8,655	215	94	7,454	234	78	6,325
H3 ₂	205	74,821	136	48,416	856	117	42,836	283	94	35,072
H1 ₃	167	3,032	150	2,730	70	79	1,513	197	59	1,168
H2 ₃	177	14,209	160	12,886	267	98	8,270	240	80	6,915
H3 ₃	205	77,240	187	70,549	1,054	127	49,344	297	99	39,268

Notes: “SAI Setup” denotes the construction time of SAI (in seconds) and applies to both SAI and INF. “Time” denotes the solution times, given in seconds. “Iter” denotes the number of iterations and “Inner” denotes the total number of iterations of the inner solver.

solution times obey the $\mathcal{O}(N \log N)$ complexity of MLFMA, in general. As the problem sizes grow and MLFMA levels increase, it is well known that the solution times experience discrete jumps [106], without violating the general $\mathcal{O}(N \log N)$ complexity. For this reason, we plot the solution times in the three groups, corresponding to the three MLFMA levels, i.e., 10, 11, and 12. In each group, the solution times with the INF preconditioner are significantly lower than those with the other preconditioners, especially considering that the vertical axis in Figure 4.3 is scaled logarithmically.

Finally, in Table 4.7, we present the parallel memory costs for CFIE solutions. We include the group that contains the largest problem of the helicopter. Closed-surface problems can be solved with CFIE in fewer iterations, compared to open-surface problems solved with EFIE. Therefore, memory required by the GMRES solver is significantly less than those presented in Table 4.4. Even though the memory requirement of FGMRES employed by INF is higher than that of GMRES, the memory cost of INF is not significant compared to that of

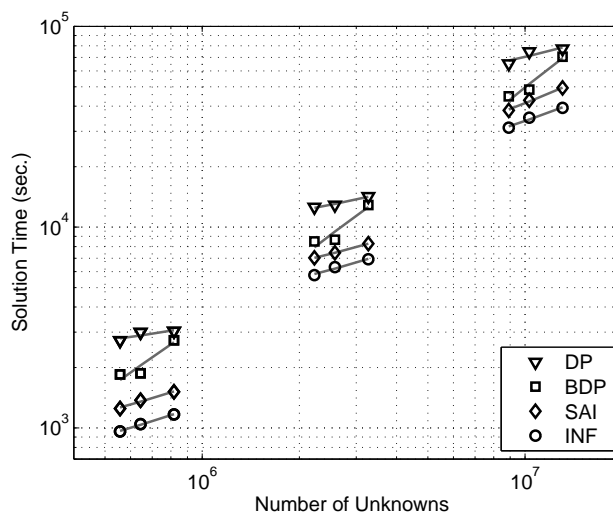


Figure 4.3: Total solution times of the helicopter problem. The lines fit the solution times in a least-squares sense.

Table 4.7: Memory costs (in MB) of MLFMA, SAI/INF setup, and GMRES solutions.

Geometry	MLFMA	SAI/INF Setup	GMRES			
			DP	BDP	SAI	INF
W1	68	7	3	2	1	2
W2	232	26	14	9	6	9
W3	774	97	75	44	33	48
W4	2,360	371	380	236	173	234
H1 ₃	437	46	33	29	15	23
H2 ₃	1,831	167	138	125	76	125
H3 ₃	7,431	637	639	583	396	617

MLFMA. We also note that memory costs of the SAI setup and GMRES are much less than that of MLFMA.

4.5 Conclusion

For the iterative solution of EFIE via MLFMA, designing preconditioners that effectively use the information provided by the sparse near-field matrix is crucial for fast convergence. Even though the CFIE formulation yields better-conditioned linear systems than EFIE, its use is limited to closed-surface problems. Furthermore, real-life problems usually involve thin and complex parts, and this causes

an increase in the iteration counts required for convergence, even with CFIE. Hence, iterative solutions of CFIE also benefit from preconditioning. ILU [40] and SAI [44] preconditioners are designed for this purpose. ILU preconditioners are not suitable for scalable parallel implementations. SAI preconditioners accelerate the iterative convergence to some extent, but they have limited success in taking full advantage of the available sparse near-field matrix, as demonstrated by the comparisons with the benchmark LU solutions (NF-LU) in Table 4.1. To increase their effectiveness, one can increase the density of the approximate inverses beyond that of the near-field matrix, but this is not the best solution because of the memory considerations. Moreover, the benefit obtained even with this costly solution is limited, as shown in [83].

In this work, we propose an alternative way to increase the efficiency using flexible solvers. In this scheme, the near-field matrix is iteratively solved and used as a preconditioner in addition to a fixed preconditioner, such as a SAI preconditioner, which is used to accelerate the inner iterative solver. This approach has the following advantages:

- By using the available SAI as the preconditioner of the inner system, only a few iterations suffice to achieve a strong preconditioner, and the iteration counts of the outer solver become very close to those obtained from the benchmark exact solution of the near-field system. Hence, the cost of applying the preconditioner is lowered, and the overall solution times are significantly decreased.
- The proposed INF preconditioner is demonstrated to provide faster (i.e., shorter CPU times and fewer iterations) and scalable solutions for problems involving as many as 13 million unknowns. The advantage of the INF preconditioner over the other near-field preconditioners is consistent and does not vanish as the problem size grows.

- The proposed preconditioner's parallel scalability is very good because the application of the preconditioner consists merely of repeated sparse matrix-vector multiplications, which are highly parallelizable.
- The only cost of the proposed scheme is the extra storage of the preconditioned residual vectors of FGMRES in memory, because of the variable preconditioning [2]. However, since the iteration counts are reduced, the required FGMRES memory is not significant, especially compared to the MLFMA memory.

Chapter 5

Preconditioners Utilizing More Than the Near-Field Matrix

For very large problems, the near-field matrix itself becomes insufficient to approximate the dense system matrix, and preconditioners generated from the near-field interactions cannot be effective.

Tahir Malas, Özgür Ergül, and Levent Gürel, 2007 Computational Electromagnetics Workshop (*CEM'07*).

5.1 Introduction

In the previous two chapters, we have investigated parallel preconditioners constructed from the sparse near-field matrix, and shown the effectiveness of these approaches for problems up to a few millions of unknowns. For some problems formulated with EFIE, however, iteration counts significantly increase as the problem size increases. Since we use the no-restart GMRES solver, in addition to increasing solution times, high iteration counts cause a significant memory consumption, such as the HS4 problem in Table 4.4. The increase in iteration

counts is, in part, due to the degrading conditioning of EFIE for large problems, and, in part, due to the increasing sparsity of near-field matrices. Hence, because of this second effect, the near-field preconditioners become insufficient to accelerate the iterative convergence as desired. This is the case, even if we use almost all of the near-field information efficiently by the INF preconditioner.

In addition to preconditioning, another option for reducing the cost of an iterative solution can be the relaxation methods. These methods aim to reduce the accuracy of the matrix-vector multiplications as the convergence takes place during iterations. For example, Bouras and Frayssé [107] propose decreasing the accuracy of the matrix-vector multiplications in the j th iteration by relating the relative error δ_j of the matrix-vector multiplication to the norm of the residual vector $\boldsymbol{\rho}_j$, i.e.,

$$\delta_j = \begin{cases} \epsilon, & \|\boldsymbol{\rho}_j\|_2 > 1 \\ \frac{\epsilon}{\|\boldsymbol{\rho}_j\|_2}, & \epsilon \leq \|\boldsymbol{\rho}_j\|_2 \leq 1 \\ 1, & \|\boldsymbol{\rho}_j\|_2 < \epsilon, \end{cases} \quad (5.1)$$

where $\epsilon \leq 1$ denotes the target residual error. Using (5.1), the relative error of the matrix-vector multiplication is relaxed from ϵ to 1 as the iterations proceed. However, using a similar relaxation strategy by adjusting the accuracy of MLFMA during the course of an iterative solution is not trivial. This is because the implementation details of MLFMA depend on the targeted accuracy. In principle, it is possible to construct a fixed number of MLFMA versions with various levels of accuracy. Nevertheless, each version increases the cost of the setup substantially. Moreover, a less-accurate MLFMA obtained by decreasing the number of accurate digits is not significantly cheaper than the ordinary MLFMA.

In this study, we use a similar idea, but use a “relaxed” version of MLFMA for preconditioning. We present an efficient inner-outer solution scheme [108], similar to INF. Outer iterative solutions are performed by using a flexible solver accelerated by an ordinary MLFMA. However, inner solutions are performed by

an approximate MLFMA (AMLFMA). We further accelerate the inner solutions using the SAI preconditioner. We illustrate the proposed solution strategy in Fig. 5.1. There are two explanations to describe the advantages of the proposed strategy:

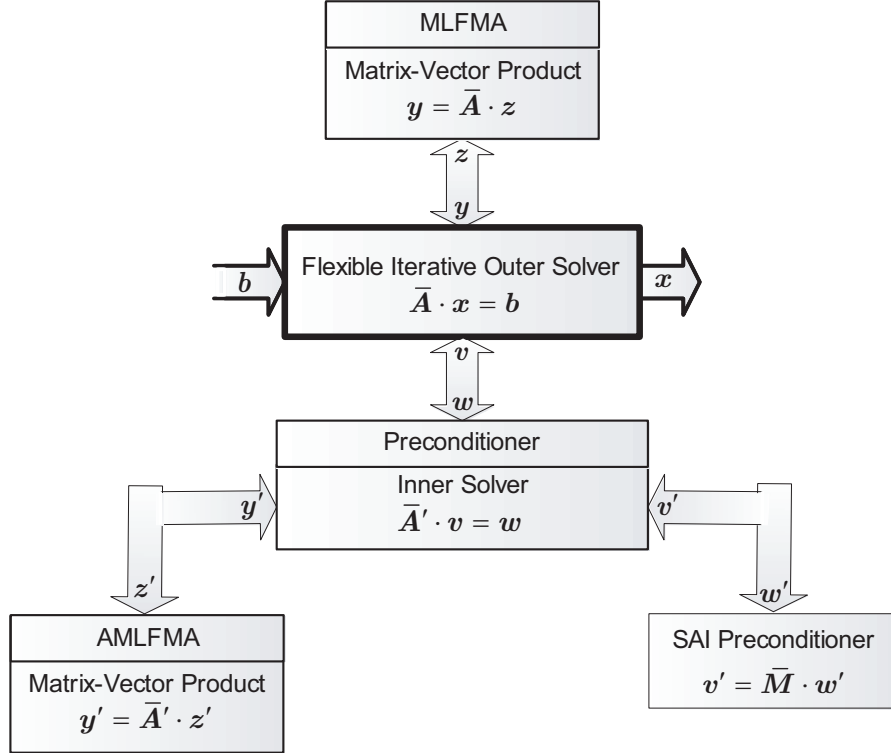


Figure 5.1: Inner-outer solution scheme that use an approximate version of MLFMA.

1. Matrix-vector products performed by an ordinary MLFMA are replaced with more efficient multiplications performed by AMLFMA. Different from the relaxation strategies, however, only a single specific implementation of AMLFMA is sufficient to construct an inner-outer scheme. In addition, a reasonable accuracy (without strict limits) is sufficient for the approximation.
2. Iterative solutions by an ordinary MLFMA are preconditioned with a very strong preconditioner that is constructed by approximating the full matrix instead of the sparse part of the matrix.

In this study, AMLFMA is introduced and proposed as a tool to construct effective preconditioners. We consider the iterative solutions of large-scale electromagnetics problems and demonstrate the acceleration provided by the proposed strategy based on AMLFMA, compared to the conventional near-field preconditioners.

The rest of the Chapter is organized as follows. In Section 5.2, we introduce the proposed AMLFMA scheme. Section 5.3 present some further details of the proposed preconditioning technique. Next, we provide numerical examples in Section 5.4, followed by our concluding remarks in Section 5.5.

5.2 The Approximate Multilevel Fast Multipole Algorithm

As explained in Section 1.6, MLFMA can perform a matrix-vector multiplication with a specific level of accuracy, which is controlled by the excess bandwidth formula, given in (1.32). To calculate the interactions between the clusters at a level l , radiation and receiving patterns are defined and sampled at $\mathcal{O}(T_l^2)$ angular points, where T_l is the truncation number for the series in (1.30). For a cluster of size $a_l = 2^{l-3}\lambda$ at level l , the truncation number is determined by using the excess bandwidth formula [29] for the worst-case scenario and the one-box-buffer scheme [30]. The work performed at each MLFMA level depends on

- the number of clusters in that level and
- the truncation number T_l , which depends on the size of the clusters in that level and the the number of accurate digits d_0 .

The maximum truncation number is linearly proportional to the electrical size of the object, hence the truncation number grows rapidly as a function of the

cluster size. For the highest MLFMA level L , $T_{max} = T_L = \mathcal{O}(kD)$, where k is the wave number and D is the size of the problem. However, the truncation number loosely depends on the value of d_0 for large clusters [32].

A direct way to construct a less-accurate MLFMA is to reduce the truncation numbers using (1.32). For example, in [92], a similar inner-outer solution scheme has been used where the ordinary MLFMA has four digits of accuracy, i.e., $d_0 = 4$, and the less-accurate MLFMA may have only one digit of accuracy. This strategy, however, has major disadvantages in terms of efficiency and memory use. First of all, a less-accurate MLFMA obtained in this way is not significantly faster than the ordinary MLFMA, because, the truncation number loosely depends on d_0 for large boxes at the higher levels of MLFMA [58, 48]. Moreover, a new set of patterns is required for the less-accurate MLFMA with the reduced truncation numbers.

In this work, we generate an inexpensive version of MLFMA by relaxing the accuracy in a flexible way. We balance the accuracy and efficiency by redefining the truncation number for level l as

$$L'_l = L_1 + a_f(L_l - L_1), \quad (5.2)$$

where L_1 is the truncation number defined for the first level, L_l is the original truncation number for the level l calculated by using the translation function given in (1.32). The approximation factor a_f is defined in the range from 0.0 to 1.0. As a_f increases from 0.0 to 1.0, the AMLFMA becomes more accurate but less efficient, while it corresponds to the full MLFMA when $a_f = 1$. Hence, this parameter provides us important flexibility in designing the preconditioner. Moreover, the truncation number of the lowest level is not modified, hence AMLFMA does not require extra computation load for the radiation and receiving patterns of the basis and testing functions when it is used in conjunction with MLFMA in a nested manner.

5.3 Iterative Preconditioning Based on the Approximate MLFMA

Preconditioners that are based on the near-field interactions can be insufficient to accelerate the iterative solutions of large-scale problems, especially those formulated with EFIE. For more efficient solutions, it is possible to use the far-field interactions in addition to the near-field interactions and construct more effective preconditioners. This can be achieved by using flexible solvers and employing approximate and ordinary versions of MLFMA in an inner-outer scheme. Using a reasonable approximation for the inner solutions, the number of outer iterations can be reduced substantially. In addition to more efficient solutions, the inner-outer scheme prevents numerical errors that arise because of the deviations of the computed residual from the true residual by significantly decreasing the number of outer iterations. This is because the “residual gap,” i.e., the difference between the true and computed residuals, increases with the number of iterations [109]. Another benefit of the reduction in iteration counts appears when the iterative solutions are performed with the generalized minimal residual (GMRES) algorithm, which is usually an optimal method for EFIE in terms of the processing time [92],[40]. Nested solutions by flexible variants of GMRES, namely, FGMRES [2] or GMRESR [110], require significantly less memory than the ordinary solutions by GMRES.

There are many factors that affect the performance of an inner-outer scheme, such as the primary preconditioning operator, the choice of the inner solver and the secondary preconditioner to accelerate the inner solutions, and the inner stopping criteria. Now, we discuss these factors in detail.

5.3.1 Preconditioning Operator

In an extreme case, one can use the full matrix itself as a preconditioner by employing the ordinary MLFMA to perform the matrix-vector multiplications for the inner solutions. On the other hand, an inner-outer scheme usually increases the total number of matrix-vector multiplications compared to the ordinary solutions [109]. In addition, an approximate solution instead of an ordinary solution can be sufficient to construct a robust preconditioner. As discussed in Section 5.2 and in [58], AMLFMA is an appropriate choice to perform the inner solutions. By using the approximation factor a_f , the accuracy of AMLFMA can be adjusted to achieve a maximum overall efficiency.

5.3.2 Inner Solver and the Secondary Preconditioner

For the inner solutions, GMRES is preferable due to its rapid convergence in a small number of iterations. The inner solutions are also accelerated by using a secondary preconditioner based on the near-field interactions. Among the various choices described in this dissertation, we prefer the SAI preconditioner, which effectively increases the convergence rate, especially in the early stages of the iterative solutions [92].

5.3.3 Inner Stopping Criteria

The relative residual error ϵ_{in} and the upper limit for the number of inner iterations j_{max}^{in} are also important parameters that affect the overall efficiency of the inner-outer scheme. Van den Eshof et al. [109] showed that fixing ϵ_{in} is nearly optimal if relaxation is not applied. However, even 0.1 (10%) residual error can cause a significant number of inner iterations for large-scale problems. Therefore, in addition to ϵ_{in} , the maximum number of iterations j_{max}^{in} should be set carefully

to avoid unnecessary iterations during the inner solutions. For large problems, a small value of j_{max}^{in} is more likely to keep the inner iteration counts under control than a large value of ϵ_{in} .

5.4 Numerical Results

Finally, we demonstrate the performance of the proposed inner-outer scheme using AMLFMA, compared to the solutions accelerated with BD, SAI, and INF preconditioners. Fortran 90 programming language is used for all implementations. Solutions are performed on a distributed-memory parallel computer containing Intel Xeon Harpertown processors with 3.0 GHz clock rate. A total of 32 cores located in 16 nodes (2 cores per node) are used, and the nodes are connected via an Infiniband network. Iterative solvers, namely, GMRES and FGMRES, are provided by the PETSc library [73]. In all solutions, matrix-vector multiplications are performed by MLFMA with three digits of accuracy. For the inner solutions with AMLFMA, the target residual error and the approximation factor are set to 10^{-1} and 0.2, respectively. To avoid unnecessary work, inner solutions are stopped at maximum 10th iteration ($j_{max}^{in}=10$). For the INF preconditioner, however, the target residual error for the inner solutions is in the range of 10^{-2} to 10^{-1} , whereas the maximum number of iterations is set to between 3 and 5, depending on the problem. A small number of inner iterations is usually sufficient for INF since the SAI preconditioner used to accelerate the inner solutions provides a good approximation to $\overline{\mathbf{A}}^{NF}$.

Parameters for the preconditioners are determined by testing the implementations on a wide class of problems and choosing the optimal combination to minimize the total processing time for each preconditioner. For example, by setting the approximation factor to 0.2 in AMLFMA, most of the matrix elements are calculated with less than 10% error [58, 48]. Then, using 10^{-1} residual error

Table 5.1: Electromagnetics problems involving open metallic objects.

Problem	Frequency (GHz)	Size (λ)	MLFMA Levels	Number of Unknowns
P1	96	96	8	3,062,400
P2	128	128	9	5,511,680
P3	192	192	9	12,253,440
HS1	96	192	8	3,838,496
HS2	128	256	9	6,535,168
HS3	192	384	9	15,356,992
RA1	32	107	8	2,991,067
RA2	48	160	9	6,849,398
RA3	64	214	9	11,967,620

for the inner iterations provides the best performance. Choosing a smaller error threshold leads to unnecessary iterations and a larger error threshold wastes the relatively high accuracy of AMLFMA. Setting the maximum number of iterations to more than 10 increases the processing time, even though the accuracy of the inner solutions is not improved significantly.

EFIE is notorious for generating ill-conditioned matrix equations, which are difficult to solve iteratively, especially when the problem size is large [79],[40]. Therefore, the proposed inner-outer scheme employing AMLFMA is particularly useful for open surfaces that must be formulated with EFIE. In addition, we show that the iterative solutions of complicated problems involving closed surfaces that are formulated with CFIE are also improved by the proposed method.

5.4.1 EFIE Results

We use the three different metallic objects involving open surfaces, namely, a patch (P), a half sphere (HS), and a reflector antenna (RA), which are depicted in Fig. 3.10. Discretizations of the objects for various frequencies lead to large matrix equations with millions of unknowns, as listed in Table 5.1. Dimensions of the objects in terms of the wavelength and the number of active levels (L) in MLFMA are also listed in Table 5.1.

Table 5.2: Processing time (seconds) and the number of iterations* for the solution of electromagnetics problems involving open metallic objects.

Problem	SAI** Setup	SAI		INF			AMLFMA		
		Outer	Time	Outer	Inner	Time	Outer	Inner	Time
P1	132	194	6,225	131	391	4,393	35	345	3,278
P2	308	234	27,902	160	478	19,620	43	425	9,860
P3	1,136	276	36,677	174	868	25,650	52	516	17,454
HS1	350	480	23,458	367	1,101	18,374	68	680	11,085
HS2	839	546	66,778	406	1,218	51,285	75	750	23,143
HS3	5,620	357	59,734	276	1,380	48,786	51	510	31,740
RA1	201	200	7,276	138	408	5,138	33	327	4,233
RA2	671	252	31,784	172	509	22,404	42	417	13,746
RA3	2,077	336	43,912	228	1,138	32,710	57	567	24,595
* The relative residual error is 10^{-3} for HS3 and RA3, and 10^{-6} for other problems. ** Setup of the SAI preconditioner is also required for INF and AMLFMA.									

Table 5.2 presents the number of iterations and the processing time for the solutions of the problems in Table 5.1. We observe that using the INF preconditioner accelerates the solutions significantly compared to the SAI preconditioner used alone. Employing the inner-outer scheme with AMLFMA further reduces the processing time, and we are able to solve the largest problem discretized with about 12 million unknowns in less than 7 hours. Although not shown in Table 5.2, the memory required for the iterative algorithm is also reduced substantially by the proposed method since the memory requirements of both GMRES and FGMRES increase with the number of iterations. As an example, for the solution of P3 with SAI, GMRES requires 2614 MB per processor. Using an inner-outer scheme and AMLFMA, the memory requirement is reduced to 820 MB.

5.4.2 CFIE Results

We use the two objects modelled with closed conducting surfaces, namely, a helicopter (H) and a stealth airborne target named Flamme (F) [91]. These objects are illustrated in Figure 3.12. Electromagnetics problems involving those objects are formulated with CFIE ($\alpha = 0.2$) and listed in Table 5.3. Table 5.4 presents the number of iterations and the processing time when the solutions are

Table 5.3: Electromagnetics problems involving closed metallic objects.

Problem	Frequency (GHz)	Size (λ)	MLFMA Levels	Number of Unknowns
F1	40	80	10	1,248,480
F2	60	120	10	3,166,272
F3	80	160	11	4,993,920
H1	1.7	80	10	1,302,660
H2	2.4	113	11	2,968,512
H3	3.4	160	11	5,210,640

accelerated with the BD and SAI preconditioners, as well as the inner-outer scheme using AMLFMA. We observe that the proposed method reduces the solution time significantly compared to both the BD and SAI preconditioners.

Table 5.4: Processing time (seconds) and the number of iterations* for the solution of electromagnetics problems involving closed metallic objects.

Problem	BD**		SAI*** Setup	SAI		AMLFMA		
	Outer	Time		Outer	Time	Outer	Inner	Time
H1	117	2,287	183	90	1,869	16	239	813
H2	138	10,192	644	97	7,515	19	137	2,562
H3	125	10,986	627	104	9,386	25	240	4,938
F1	211	4,451	163	174	3,836	40	305	1,917
F2	347	10,087	402	316	9,276	58	331	6,641
F3	724	66,094	505	706	64,365	125	733	28,593

* The relative residual error is 10^{-6} for all problems.

** The BD preconditioner has a negligible setup time.

*** Setup of the SAI preconditioner is also required for AMLFMA.

5.5 Conclusion

For very large electromagnetics problems, achieving a rapid convergence in a reasonable iteration count is only viable by means of robust preconditioners. In the context of MLFMA, the general trend is to develop sparse preconditioners by using the near-field interactions. However, as the problem size gets larger and the number of unknowns also increases, those preconditioners become sparser and they may not be sufficient to obtain an efficient solution.

In this study, we propose an inner-outer scheme to improve iterative solutions with MLFMA. For the inner solutions, matrix-vector multiplications are performed efficiently by AMLFMA, which is obtained by systematically reducing the accuracy of the ordinary MLFMA. We show that the resulting solver accelerates the iterative solutions of electromagnetics problems involving open and closed geometries formulated with EFIE and CFIE, respectively.

Chapter 6

Schur Complement

Preconditioners For Dielectric Problems

When applied to saddle point systems, on the other hand, standard algebraic preconditioners are often found to perform poorly. Because of the indefiniteness and lack of diagonal dominance, these preconditioners are often unstable. Even when the computation of the preconditioner does not suffer from some type of breakdown (e.g., zero pivots in an incomplete factorization), the quality of the resulting preconditioner is often not very satisfactory, and slow convergence is observed. Also, because of the absence of decay in $\overline{\mathbf{A}}^{-1}$, it is difficult to construct good sparse approximate inverse preconditioners for saddle point matrices.

Michele Benzi, Gene H. Golub, and Jörg Liesen, *Acta Numerica*, Vol. 14, 2005.

6.1 Introduction

Many real-life problems in CEM involve dielectrics, such as the development of effective lenses [111], simulations of photonic crystals [112], and optical analysis of blood for blood-related diseases [113]. Recently developed surface integral-equation formulations provide suitable mechanisms for the electromagnetic analysis of such dielectric problems. Examples include the combined tangential formulation (CTF), which is a first-kind integral equation, and the combined normal formulation (CNF), which is a second-kind integral equation [10]. For those formulations, similar to the PEC case, the multilevel fast multipole algorithm (MLFMA) can be applied to overcome the computational bottleneck of resulting dense matrices [114]. Hence, accurate analysis of three-dimensional, real-life dielectric problems become possible with low computational costs.

We analyze four types of surface formulations that are commonly used in CEM: the combined tangential formulation (CTF), the combined normal formulation (CNF), the modified normal Müller formulation (MNMF), and the electric and magnetic current combined-field integral equation (JMCFIE), which is derived from the combination of CTF and CNF, as shown in (6.17) [10, 115, 116]. Discretizations of those formulations with MOM yield 2×2 partitioned matrices in the form

$$\begin{bmatrix} \overline{\mathbf{A}}_{11} & \overline{\mathbf{A}}_{12} \\ \overline{\mathbf{A}}_{21} & \overline{\mathbf{A}}_{22} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}_J \\ \mathbf{x}_M \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \text{ or } \overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}, \quad (6.1)$$

where

$$\overline{\mathbf{A}} \in \mathbb{C}^{2N \times 2N} \text{ and } \overline{\mathbf{A}}_{11}, \overline{\mathbf{A}}_{12}, \overline{\mathbf{A}}_{21}, \overline{\mathbf{A}}_{22} \in \mathbb{C}^{N \times N}. \quad (6.2)$$

In (6.1), \mathbf{x}_J and \mathbf{x}_M are $N \times 1$ coefficient vectors of the Rao-Wilton-Glisson (RWG) [7] basis functions expanding the equivalent electric and magnetic electric currents, respectively, and $\mathbf{b}_{1,2}$ represent $N \times 1$ excitation vectors obtained by testing incident fields.

The resulting partitioned matrices are, in general, highly indefinite and have poor spectral properties [12]. Significant effort has been devoted to devising second-kind formulations that produce well-conditioned system matrices, such as the electric and magnetic current combined-field integral equation (JMCFIE) [116]. Another example is the modified normal Müller formulation (MNMF), which is a meticulously scaled version of the normal Müller formulation [117]. Nonetheless, strong preconditioners are still required for efficient solutions of dielectric problems because of the following reasons:

- It is known that the accuracy of second-kind integral equations is lower than that of first-kind integral equations [12]. For the PEC case, CFIE, which is a second-kind formulation, produces successful enough results even for very large problems [20]. In the case of dielectric problems, however, comparing solutions of a photonic crystal problem shows that the accuracy of JMCFIE can be much worse than that of CTF, and the results obtained with the former can be severely misleading [67].
- Even though second-kind formulations JMCFIE and MNMF produce more diagonally dominant and easier-to-solve systems than those of CTF, the diagonal dominance of these formulations disappears with an increasing dielectric constant. (Dielectric constant, ϵ_r , is defined as the ratio of the electric permittivity of the outer region of the object to that of the inner region.) As a result, the unpreconditioned solutions of such formulations can necessitate too many iterations for convergence [114, 64].
- Finally, we note that the well-conditioning of recently developed formulations, such as JMCFIE and MNMF, is shown by algebraic analysis of system matrices that belong to small-size canonical problems (e.g., [12]). However, recent studies [114, 64] show that real-life problems resulting in large matrix systems, such as periodic dielectric structures, may represent

a significant challenge in terms of convergence, even when those problems involve fairly small dielectric constants.

In the literature of scientific computing, preconditioning techniques for systems similar to (6.1) are usually studied in the context of generalized saddle-point problems [118, 119, 120, 121, 34, 122, 123, 124, 125, 126, 127]. By approximating the dense system matrix in (6.1) by a sparse near-field matrix, preconditioners developed for saddle-point problems can be adapted to integral-equation formulations of dielectric problems. The partitions in (6.1), however, do not satisfy any of the conditions that generally exist in saddle-point problems, such as symmetry or positive definiteness [118]. Moreover, contrary to our case, in many applications that lead to partitioned systems, the (2,2) partition is zero or has a much smaller dimension than other partitions. In general, preconditioners are tailored depending on the specific properties of the underlying problem [118]. Hence, preconditioners developed for other applications may not be readily applicable to surface integral-equation formulations.

In CEM, the effect of preconditioning on surface formulations of dielectric problems has been studied only in a few papers. In [10], the authors employ the diagonal preconditioner and a simple incomplete LU (ILU) preconditioner for sphere problems with a low ($\epsilon_r = 4$) and a very high dielectric constant ($\epsilon_r = 36 + 0.3i$). For the low-contrast sphere (i.e., with a low dielectric constant), these preconditioners slightly decrease the iteration counts, whereas for the high-contrast sphere, the preconditioners decelerate the convergence rate. In [114], a four-partition block-diagonal preconditioner (4PBDP) is proposed, which is constructed by using small diagonal blocks of each partition. 4PBDP reduces iteration counts only for low-contrast problems that are formulated by second-kind formulations. For problems formulated with CTF or involving a high contrast, 4PBDP either decelerates the convergence rate or fails to provide a significant improvement.

In this work, we consider preconditioners that are obtained with some approximations to Schur complement reduction. We use the sparse near-field matrix to construct preconditioners. The near-field matrix is formed naturally in the context of the multilevel fast multipole algorithm (MLFMA), which is employed to accelerate the dense matrix-vector multiplications (MVMs). The success of the Schur complement preconditioners depends on effective approximations for the solutions of systems involving the (1,1) partition and the Schur complement. Similar to the work in [119], we use sparse approximate inverses (SAIs) in these approximations. In [119], however, the authors use an iterative method [96] to generate the sparsity pattern of a SAI in the course of construction. In our case, the near-field pattern is a natural candidate for the sparsity pattern of a SAI, and this approach leads to successful preconditioners for the surface integral-equation formulations of PEC objects [44, 92]. Therefore, we employ the Frobenius-norm minimization technique and use the available near-field pattern for approximate inverses. The advantages of using SAIs over ILU-type preconditioners are robustness and ease of parallelization. Furthermore, by using the block structure of the near-field matrix, we eliminate the high setup time of SAI. The approximation for the Schur complement is more delicate than the (1,1) partition. In the literature, most of the proposed approaches are limited to cases in which the (2,2) partition is zero. We propose to obtain an approximate Schur complement via incomplete matrix-matrix multiplications that retain the near-field sparsity pattern. Then, we construct a SAI from the approximate Schur complement.

We also consider iterative Schur complement preconditioners. Instead of using SAIs directly to approximate the solutions of systems involving the (1,1) partition and the Schur complement, we solve these linear systems iteratively, and use SAIs as preconditioners for these “inner” solutions. The solution of dielectric problems with MLFMA and the proposed preconditioning schemes are illustrated in Figs. 6.2 and 6.1. For iterative Schur complement preconditioners (Fig. 6.2), iterative solvers are employed for preconditioning, hence, a flexible

solver must be used for the outer solver [2]. When iterative solvers are used for the reduced systems, it is in general possible to improve the direct approximations and obtain stronger preconditioners. For sparse partitioned systems, such a scheme may incur a significant application cost due to the extra MVMs needed for the inner solutions [118, 128]. In our case, however, the required sparse MVMs cause a small extra computational cost, since the computations related to the far-field matrix elements (which are performed by MLFMA) are usually dominant to near-field MVMs. Another advantage of the iterative Schur complement preconditioners is that when iterative solutions are employed for preconditioning, it is possible to use the same SAI for both the (1,1) partition and the Schur complement, hence reduce the memory cost of the Schur complement preconditioners shown in Fig. 6.1 by one half.

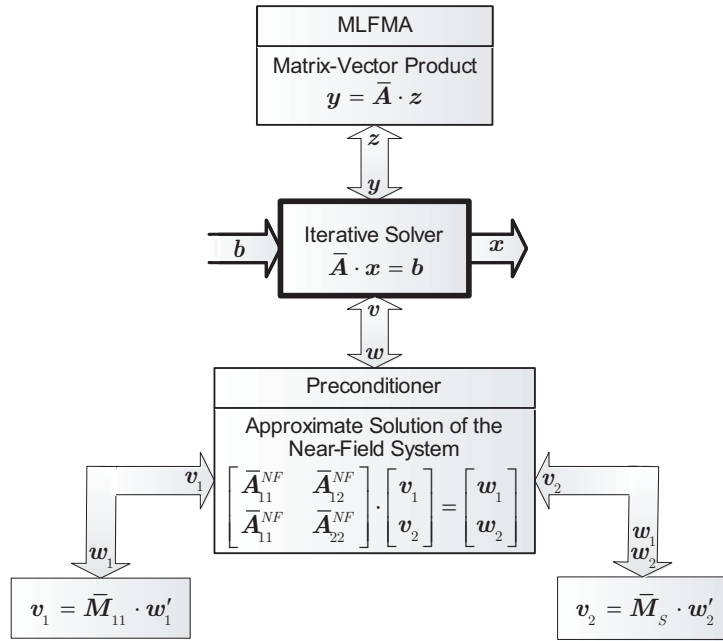


Figure 6.1: Illustration of the solution of dielectric problems using MLFMA and approximate Schur complement preconditioners. As explained in Section 6.5, $\bar{\mathbf{M}}_{11}$ is an approximate inverse for $\bar{\mathbf{A}}_{11}^{NF}$ and $\bar{\mathbf{M}}_S$ is an approximate inverse for the Schur complement. \mathbf{w}'_1 and \mathbf{w}'_2 take different forms depending on the type of the preconditioner.

This chapter is organized as follows. In the next section, we introduce some of the recently developed surface integral-equation formulations for dielectrics. However, we specifically concentrate on CTF, which produces the most accurate

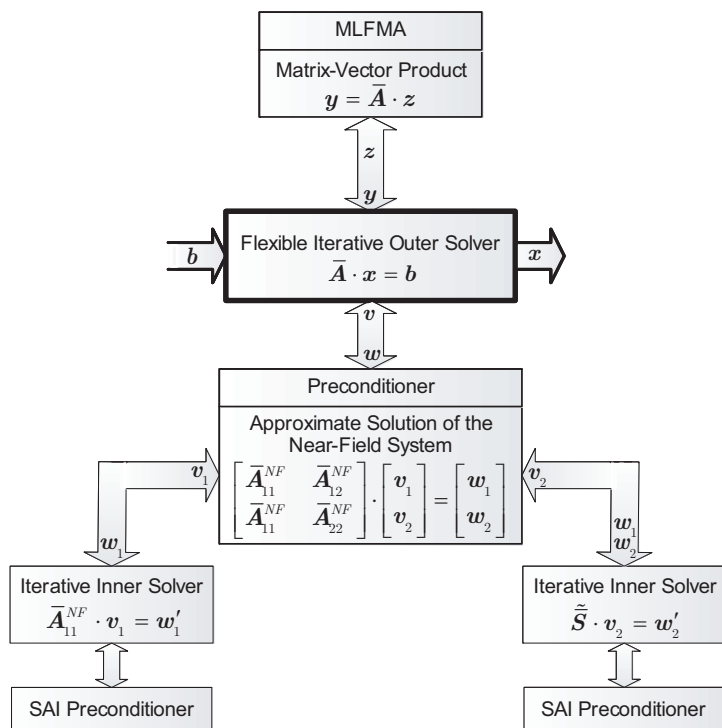


Figure 6.2: Illustration of the solution of dielectric problems using MLFMA and iterative Schur complement preconditioners. Here, instead of direct solves, iterative solutions are employed to find approximate solutions to reduced systems. As explained in Section 6.6, $\tilde{\mathbf{S}}$ is an approximation to the Schur complement and \mathbf{w}'_1 and \mathbf{w}'_2 take different forms depending on the type of the preconditioner.

results, and JMCIE, which produces the lowest iteration counts among recently developed surface formulations [114]. Then, we briefly review the discretization process for dielectric problems. We introduce our proposed approaches in Sections 6.5 and 6.6. Then, in the results section, we show the superiority of the proposed preconditioners both in terms of memory and solution times in various problems.

A note on the use of *partition* and *block*: Throughout this chapter, we will use the term *partition* to denote one of the submatrices of a 2×2 partitioned system, i.e., we call $\bar{\mathbf{A}}_{11}$ in (6.1) as the (1,1) partition of $\bar{\mathbf{A}}$. Partitions of the near-field matrix are composed of interactions between pairs of neighboring lowest-level MLFMA clusters. In the CEM community, the term *block* is used to denote

these interactions. We will adopt this convention and imply building blocks of a near-field partition by the term *block*.

6.2 Surface Integral-Equation Methods for Dielectric Problems

The surface integral-equation approach is an important class of numerical methods in electromagnetics scattering analyses of 3-D dielectric objects having arbitrary shapes [129]. Recently, significant progress has been made in devising new formulations that are well suited for iterative solutions [10, 115, 116]. In this section, we will briefly review these methods.

For all formulations, consider a closed homogeneous dielectric object that resides in a homogeneous medium. Let the electric permittivity and the electric permeability of the outer region of the object be ϵ_1, μ_1 and those of the inner region be ϵ_2, μ_2 , respectively. Using the equivalence principle, an equivalent electric current \mathbf{J} and an equivalent magnetic current¹ \mathbf{M} are defined on the surface S of the object. Depending on the testing procedure and the considered electromagnetic field, various integral-equation formulations can be derived.

6.2.1 The Combined Tangential Formulation (CTF)

If the boundary condition on the surface is tested directly, tangential electric-field and magnetic-field integral equations for the outer and the inner regions can be defined. For example, the tangential electric-field integral equation (T-EFIE)

¹Preconditioning matrices ($\overline{\mathbf{M}}$) and magnetic currents (\mathbf{M}) are denoted with similar symbols, following conventions. Since one of them is a matrix ($\overline{\mathbf{M}}$) and the other one is a vector (\mathbf{M}), they should be clearly distinguishable from the context.

for the outer region is defined as [130]

$$\hat{\mathbf{t}} \cdot \eta_1 \mathcal{T}_1 \{ \mathbf{J} \} - \hat{\mathbf{t}} \cdot \mathcal{K}_1 \{ \mathbf{M} \} - \hat{\mathbf{t}} \cdot \frac{1}{2} \hat{\mathbf{n}} \times \mathbf{M} = -\hat{\mathbf{t}} \cdot \mathbf{E}^{inc}, \quad (\text{T-EFIE-O}) \quad (6.3)$$

where $\hat{\mathbf{t}}$ is any tangential vector on the surface, $\eta_1 = \sqrt{\mu_1/\epsilon_1}$ is the impedance of the outer medium,

$$\mathcal{T}_l \{ \mathbf{X} \} = ik_l \int_S d\mathbf{r}' [\mathbf{X}(\mathbf{r}') + \frac{1}{k_l^2} \nabla' \cdot \mathbf{X}(\mathbf{r}') \nabla] g_l(\mathbf{r}, \mathbf{r}'), \quad (6.4)$$

and

$$\mathcal{K}_l \{ \mathbf{X} \} = \int_{PV,S} d\mathbf{r}' \mathbf{X}(\mathbf{r}') \times \nabla' g_l(\mathbf{r}, \mathbf{r}') \quad (6.5)$$

are the operators that can be defined for both the outer ($l = 1$) and inner ($l = 2$) regions, $\hat{\mathbf{n}}$ is the outward normal vector on the surface S , and \mathbf{E}^{inc} is the incident electric field on the object. In (6.4) and (6.5), k_l is the wavenumber in the corresponding medium, PV is the principal value of the integral, and

$$g_l(\mathbf{r}, \mathbf{r}') = \frac{e^{ik_l |\mathbf{r} - \mathbf{r}'|}}{4\pi |\mathbf{r} - \mathbf{r}'|} \quad (6.6)$$

is the scalar Green's function of the 3-D scalar Helmholtz equation for medium l , which represents the response at \mathbf{r} due to a point source located at \mathbf{r}' . For the inner region, the tangential electric-field integral equation is

$$\hat{\mathbf{t}} \cdot \eta_2 \mathcal{T}_2 \{ \mathbf{J} \} - \hat{\mathbf{t}} \cdot \mathcal{K}_2 \{ \mathbf{M} \} + \hat{\mathbf{t}} \cdot \frac{1}{2} \hat{\mathbf{n}} \times \mathbf{M} = 0, \quad (\text{T-EFIE-I}) \quad (6.7)$$

where η_2 is the impedance of the inner medium. Similar equations can also be obtained by testing the tangential magnetic fields. Respectively, the tangential magnetic-field integral equation (T-MFIE) for the outer and inner regions are

$$\hat{\mathbf{t}} \cdot \frac{1}{\eta_1} \mathcal{T}_1 \{ \mathbf{M} \} + \hat{\mathbf{t}} \cdot \mathcal{K}_1 \{ \mathbf{J} \} + \hat{\mathbf{t}} \cdot \frac{1}{2} \hat{\mathbf{n}} \times \mathbf{J} = -\hat{\mathbf{t}} \cdot \mathbf{H}^{inc} \quad (\text{T-MFIE-O}) \quad (6.8)$$

and

$$\hat{\mathbf{t}} \cdot \frac{1}{\eta_2} \mathcal{T}_2 \{ \mathbf{M} \} + \hat{\mathbf{t}} \cdot \mathcal{K}_2 \{ \mathbf{J} \} - \hat{\mathbf{t}} \cdot \frac{1}{2} \hat{\mathbf{n}} \times \mathbf{J} = 0. \quad (\text{T-MFIE-I}) \quad (6.9)$$

The four sets of integral equations, i.e., (6.3), (6.7), (6.8), and (6.9), can be combined in several ways to solve for the unknown currents \mathbf{J} and \mathbf{M} [129].

In particular, the combination of the outer and the inner equations produces internal-resonance-free formulations. Among such formulations, we consider the recently proposed CTF [10], which is defined as

$$\begin{aligned} & \frac{1}{\eta_1} \text{T-EFIE-O} + \frac{1}{\eta_2} \text{T-EFIE-I}, \\ & \eta_1 \text{T-MFIE-O} + \eta_2 \text{T-MFIE-I}. \end{aligned} \quad (6.10)$$

Note that in (6.10), the identity terms cancel each other and CTF turns out to be a first-kind integral equation. Also note that \mathbf{J} is well tested in T-EFIE and \mathbf{M} is well tested in T-MFIE [10], hence the combination used in CTF leads to a stable matrix equation. The scaling of the tangential equations further improves the condition of the formulation compared to its former variants [10], such as the tangential Poggio-Miller-Chang-Harrington-Wu-Tsai formulation [131, 132].

6.2.2 The Combined Normal Formulation (CNF)

Although CTF produces a stable formulation, it still suffers from slow convergence since it is a first-kind integral equation. Hence, several authors proposed second-kind and better-conditioned integral-equation formulations by making use of the normal formulations [12]. These formulations can be obtained by testing the fields after they are projected onto the surface via a cross-product by $\hat{\mathbf{n}}$. The normal outer and inner electric-field integral equations are, respectively,

$$-\hat{\mathbf{n}} \times \eta_1 \mathcal{T}_1 \{\mathbf{J}\} + \hat{\mathbf{n}} \times \mathcal{K}_1 \{\mathbf{M}\} - \frac{1}{2} \mathbf{M} = \hat{\mathbf{n}} \times \mathbf{E}^{inc} \quad (\text{N-EFIE-O}) \quad (6.11)$$

and

$$\hat{\mathbf{n}} \times \eta_2 \mathcal{T}_2 \{\mathbf{J}\} - \hat{\mathbf{n}} \times \mathcal{K}_2 \{\mathbf{M}\} - \frac{1}{2} \mathbf{M} = 0. \quad (\text{N-EFIE-I}) \quad (6.12)$$

For the magnetic field, normal formulations yield

$$\hat{\mathbf{n}} \times \frac{1}{\eta_1} \mathcal{T}_1 \{\mathbf{M}\} + \hat{\mathbf{n}} \times \mathcal{K}_1 \{\mathbf{J}\} - \frac{1}{2} \mathbf{J} = -\hat{\mathbf{n}} \times \mathbf{H}^{inc} \quad (\text{N-MFIE-O}) \quad (6.13)$$

and

$$-\hat{\mathbf{n}} \times \frac{1}{\eta_2} \mathcal{T}_2 \{\mathbf{M}\} - \hat{\mathbf{n}} \times \mathcal{K}_2 \{\mathbf{J}\} - \frac{1}{2} \mathbf{J} = 0. \quad (\text{N-MFIE-I}) \quad (6.14)$$

Then, similar to CTF, CNF is formed by the linear combinations of the outer and inner integral equations, i.e.,

$$\begin{aligned} & \text{N-MFIE-O} + \text{N-MFIE-I}, \\ & \text{N-EFIE-O} + \text{N-EFIE-I}. \end{aligned} \tag{6.15}$$

However, contrary to CTF, the identity terms do not cancel out in CNF, and a second-kind integral equation is obtained. When the Galerkin scheme is used to discretize (6.15), these well-tested identity operators appear on the diagonal partitions of the coefficient matrix and this results in more diagonally dominant linear systems than tangential formulations.

6.2.3 The Modified Normal Müller Formulation (MNMF)

In [115], the authors show that a scaled version of the normal Müller formulation [117] leads to a well-conditioned and stable formulation. Later, it is shown by the same authors that MNMF produces the lowest iteration counts for iterative solutions of dielectric problems compared to other stable formulations. Hence, we also consider MNMF, which is actually a scaled version of CNF. MNMF is defined as [115]

$$\begin{aligned} & \frac{\mu_1}{\mu_1 + \mu_1} \text{N-MFIE-O} + \frac{\mu_2}{\mu_1 + \mu_1} \text{N-MFIE-I}, \\ & \frac{\epsilon_1}{\epsilon_1 + \epsilon_1} \text{N-EFIE-O} + \frac{\epsilon_2}{\epsilon_1 + \epsilon_1} \text{N-EFIE-I}. \end{aligned} \tag{6.16}$$

6.2.4 The Electric and Magnetic Current Combined-Field Formulation (JMCFIE)

For non-dielectric PEC metallic objects, a combination of the electric-field integral equation and the magnetic-field integral equation yields the combined-field integral equation [8], which has favorable characteristics for iterative solutions [9]. In the dielectric case, a similar combination of CTF and CNF can be formed

as [116]

$$\text{JMCFIE} = \alpha\text{CTF} + \beta\text{CNF}, \quad (6.17)$$

where $0 \leq \alpha \leq 1$ and $\beta = 1 - \alpha$. Similar to the PEC case, the matrix systems of the JMCFIE formulation are more stable and can usually be solved in fewer iterations compared to those of CTF and CNF [114].

6.2.5 Comparison of the Integral-Equation Formulations for Dielectrics

All of the aforementioned integral-equation formulations have pros and cons in terms of storage, accuracy, and conditioning. In terms of memory use, CTF requires the least memory when MLFMA is applied to the solution. The reason is that CTF has identical diagonal partitions and the same set of far-field patterns for the inner and outer regions. CNF and JMCFIE also have identical diagonal partitions but they have different far-field patterns for each region. Finally, in addition to having different far-field patterns, MNMF also has different diagonal partitions due to different scaling of N-MFIE-O and N-EFIE-I in (6.16). These differences between the formulations can be remarkable, because the storage of the near-field matrix and the radiation patterns constitute the highest memory requirements in MLFMA. For example, the solution of a sphere geometry with approximately 413,000 unknowns leads to 1.1 GB difference of memory use between CTF and MNMF [114]. In that example, the sphere has a radius of 7.5λ , where λ denotes the wavelength in free space.

CTF is a first-kind integral-equation formulation, whereas the other formulations (CNF, MNMF, and JMCFIE) are all second-kind formulations. In CTF, the singularity of the hypersingular operator \mathcal{T} can be decreased by moving the differential operator from the Green's function to the testing function. Hence, CTF has a smoothing kernel, in contrast to other formulations with singular

kernels [12]. The smoothing property of the CTF kernel results in coefficient matrices that are far from being diagonally dominant and that have poor conditioning. On the other hand, due to the smoothing property of its kernel, CTF has a better solution accuracy compared to normal formulations (CNF and MNMF). JMCFIE includes CNF, therefore is also less accurate than CTF. Despite the accuracy drawbacks, the singular kernels and the identity terms of normal formulations and JMCFIE lead to more diagonally-dominant matrices and better conditioning than CTF.

To evaluate the integral-equation formulations, however, one should also consider two important parameters that seriously affect the accuracy and the stability of the resulting matrices: the dielectric constant (or relative permittivity) of the medium ($\epsilon_r = \epsilon_2/\epsilon_1$) and the shape of the geometry. Both the solution accuracy and the conditioning of second-kind integral equations decrease as the dielectric constant increases [12]. Irregularities of the geometry, i.e., surfaces having sharp edges and corners, also have a negative effect on the accuracy of second-kind integral equations. Therefore, when the dielectric constant is high and/or the surface of the object has non-smooth sections, the accuracy of second-kind integral equations can be much poorer than the accuracy of CTF [12]. Finally, integral equations of the second kind are also shown to be more sensitive to discretization quality of the surface and to the accuracy of the numerical integration than integral equations of the first kind.

From these discussions, it can be deduced that preconditioning is a critical issue for accurate and efficient electromagnetics simulations of dielectric objects. When the surface of the object has non-smooth regions or the dielectric constant of the object is high, the accuracy of second-kind equations can be unacceptable and one may have to employ CTF, for which the solutions are tough to obtain without effective preconditioning. Moreover, a high dielectric constant impairs

the conditioning of normal formulations, and this can necessitate applying effective preconditioners to these formulations.

6.3 Discretization of the Surface Formulations of Dielectric Problems

We can denote the surface integral equations described in Section 1.4 as

$$\begin{aligned}\mathcal{L}_{11}\{\mathbf{J}\} + \mathcal{L}_{12}\{\mathbf{M}\} &= \mathbf{G}_1 \\ \mathcal{L}_{21}\{\mathbf{J}\} + \mathcal{L}_{22}\{\mathbf{M}\} &= \mathbf{G}_2\end{aligned}\tag{6.18}$$

using linear operators \mathcal{L}_{kl} . Projecting each operator in (1.6) onto the N -dimensional space $\text{span}\{\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_N\}$ formed by the divergence-conforming RWG testing functions [7], we have

$$\begin{aligned}\langle \mathbf{f}_m, \mathcal{L}_{11}\{\mathbf{J}\} \rangle + \langle \mathbf{f}_m, \mathcal{L}_{12}\{\mathbf{M}\} \rangle &= \langle \mathbf{f}_m, \mathbf{G}_1 \rangle \\ \langle \mathbf{f}_m, \mathcal{L}_{21}\{\mathbf{J}\} \rangle + \langle \mathbf{f}_m, \mathcal{L}_{22}\{\mathbf{M}\} \rangle &= \langle \mathbf{f}_m, \mathbf{G}_2 \rangle\end{aligned}\quad 1 \leq m \leq N,\tag{6.19}$$

where

$$\langle \mathbf{f}, \mathbf{g} \rangle = \int d\mathbf{r} \mathbf{f}(\mathbf{r}) \cdot \mathbf{g}(\mathbf{r})\tag{6.20}$$

denotes the inner product of two vector functions \mathbf{f} and \mathbf{g} . This process is also known as “testing the integral equation.” By choosing the basis functions to be the same as the testing functions, we adopt a Galerkin scheme and seek the discrete solutions of

$$\mathbf{J} \approx \sum_{n=1}^N x_{Jn} \mathbf{f}_n\tag{6.21}$$

and

$$\mathbf{M} \approx \sum_{n=1}^N x_{Mn} \mathbf{f}_n\tag{6.22}$$

in the same N -dimensional space. As a result, the coefficient vectors \mathbf{x}_J and \mathbf{x}_M become the solution of the $2N \times 2N$ linear system

$$\begin{bmatrix} \overline{\mathbf{A}}_{11} & \overline{\mathbf{A}}_{12} \\ \overline{\mathbf{A}}_{21} & \overline{\mathbf{A}}_{22} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}_J \\ \mathbf{x}_M \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \quad (6.23)$$

where

$$(\overline{\mathbf{A}}_{kl})_{mn} = \langle \mathbf{f}_m, \mathcal{L}_{kl}\{\mathbf{f}_n\} \rangle, \quad (\mathbf{b}_i)_m = \langle \mathbf{f}_m, \mathbf{G}_i \rangle, \quad k, l = 1, 2, \quad m, n = 1, 2, \dots, N. \quad (6.24)$$

Since the RWG basis functions are defined on planar triangles, geometry surfaces are discretized accordingly, i.e., via planar triangulation. Each basis function is associated with an edge; hence the number of unknowns is equal to the total number of edges in a mesh. Unless dictated by the geometry, we set the average size of an edge about one-tenth of the wavelength as a rule of thumb.

6.4 Preconditioning with Schur Complement Reduction

Similar to the PEC case, the interactions among touching lowest-level clusters constitute the near-field matrix, whose entries are calculated directly using numerical integration techniques [21, 22, 14, 24] and stored in the memory for later use in MVMs. In this way, the dense system matrix is decomposed into its far-field and near-field parts as

$$\begin{bmatrix} \overline{\mathbf{A}}_{11} & \overline{\mathbf{A}}_{12} \\ \overline{\mathbf{A}}_{21} & \overline{\mathbf{A}}_{22} \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{A}}_{11}^{NF} & \overline{\mathbf{A}}_{12}^{NF} \\ \overline{\mathbf{A}}_{21}^{NF} & \overline{\mathbf{A}}_{22}^{NF} \end{bmatrix} + \begin{bmatrix} \overline{\mathbf{A}}_{11}^{FF} & \overline{\mathbf{A}}_{12}^{FF} \\ \overline{\mathbf{A}}_{21}^{FF} & \overline{\mathbf{A}}_{22}^{FF} \end{bmatrix}, \quad \text{or } \overline{\mathbf{A}} = \overline{\mathbf{A}}^{NF} + \overline{\mathbf{A}}^{FF}. \quad (6.25)$$

Since the lowest-level cluster is fixed to a certain size (i.e., 0.25λ) and the number of touching clusters is also fixed by the shape of the geometry, there are $\mathcal{O}(N)$ near-field interactions in each partition. In addition, the clustering of the geometry leads to a near-field matrix with block-structured partitions, where the blocks of partitions correspond to interactions of the lowest-level near-field clusters [40].

Since the whole matrix is not explicitly available in our case, we first approximate the dense system matrix with the sparse near-field matrix, i.e.,

$$\bar{\mathbf{A}} \approx \bar{\mathbf{A}}^{NF}. \quad (6.26)$$

In general, magnitudes of the elements of the matrix $\bar{\mathbf{A}}$ change with physical proximity [44]. Therefore, the near-field matrix $\bar{\mathbf{A}}^{NF}$ is likely to preserve the most relevant contributions of the dense system matrix.

For iterative solutions of partitioned linear systems, preconditioners are frequently based on segregated methods. In such methods, the unknown vectors are computed separately [118]. The main representative of the segregated approach is the Schur complement reduction method.

6.4.1 Schur Complement Reduction

Consider the 2×2 partitioned near-field system,

$$\begin{bmatrix} \bar{\mathbf{A}}_{11}^{NF} & \bar{\mathbf{A}}_{12}^{NF} \\ \bar{\mathbf{A}}_{21}^{NF} & \bar{\mathbf{A}}_{22}^{NF} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix}, \quad (6.27)$$

which can be rewritten as

$$\bar{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 + \bar{\mathbf{A}}_{12}^{NF} \cdot \mathbf{v}_2 = \mathbf{w}_1 \quad (6.28)$$

$$\bar{\mathbf{A}}_{21}^{NF} \cdot \mathbf{v}_1 + \bar{\mathbf{A}}_{22}^{NF} \cdot \mathbf{v}_2 = \mathbf{w}_{22}. \quad (6.29)$$

When $\bar{\mathbf{A}}_{11}^{NF}$ is nonsingular, from (6.28)

$$\mathbf{v}_1 = (\bar{\mathbf{A}}_{11}^{NF})^{-1} \cdot (\mathbf{w}_1 - \bar{\mathbf{A}}_{12}^{NF} \cdot \mathbf{v}_2). \quad (6.30)$$

If we insert (6.30) in (6.29) and rearrange, we can find \mathbf{v}_2 from

$$\bar{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}_2 - \bar{\mathbf{A}}_{21}^{NF} \cdot (\bar{\mathbf{A}}_{11}^{NF})^{-1} \cdot \mathbf{w}_1, \quad (6.31)$$

where

$$\bar{\mathbf{S}} = \bar{\mathbf{A}}_{22}^{NF} - \bar{\mathbf{A}}_{21}^{NF} \cdot (\bar{\mathbf{A}}_{11}^{NF})^{-1} \cdot \bar{\mathbf{A}}_{12}^{NF} \quad (6.32)$$

is the Schur complement. Once \mathbf{v}_2 is found from (6.31), \mathbf{v}_1 can be found using

$$\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}_1 - \overline{\mathbf{A}}_{12}^{NF} \cdot \mathbf{v}_2. \quad (6.33)$$

Schur complement reduction is an attractive solution technique if the order of the Schur complement $\overline{\mathbf{S}}$ is small and if linear systems with matrix $\overline{\mathbf{A}}_{11}^{NF}$ can be solved efficiently. Even when these requirements are not entirely satisfied, approximate solutions of (6.31) and (6.33) can serve as useful preconditioners. Hence, we consider the approximate solution of the system (6.27) as an important step of constructing and applying a preconditioner.

6.5 Approximate Schur Complement Preconditioners

Next, we describe four types of preconditioners derived from the Schur complement reduction with different approximations to the solutions of (6.33) and (6.31) [118].

Diagonal Approximate Schur Preconditioner (DASP)

The diagonal approximate Schur preconditioner (DASP) is derived with the approximations

$$\overline{\mathbf{A}}_{12}^{NF} = \overline{\mathbf{A}}_{12}^{NF} \approx 0 \quad (6.34)$$

performed in the right-hand sides (RHSs) of (6.33) and (6.31). Then, these equations reduce to

$$\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}_1 \quad (6.35)$$

and

$$\overline{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}_2. \quad (6.36)$$

Therefore, the preconditioning matrix of DASP is given by

$$\overline{\mathbf{M}}_{DASP} = \begin{bmatrix} \overline{\mathbf{A}}_{11}^{NF} & 0 \\ 0 & \overline{\mathbf{S}} \end{bmatrix}. \quad (6.37)$$

Upper Triangular Approximate Schur Preconditioner (UTASP)

If we set only one of the off-diagonal partitions $\overline{\mathbf{A}}_{12}^{NF}$ and $\overline{\mathbf{A}}_{21}^{NF}$ in the RHSs of (6.33) and (6.31) to zero, we obtain a partition triangular preconditioner. When we set $\overline{\mathbf{A}}_{21}^{NF} \approx 0$, we obtain the upper triangular approximate Schur preconditioner (UTASP). First, we have to solve for \mathbf{v}_2 from

$$\overline{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}_2. \quad (6.38)$$

Then, we can find \mathbf{v}_1 using \mathbf{v}_2 :

$$\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}_1 - \overline{\mathbf{A}}_{12}^{NF} \cdot \mathbf{v}_2. \quad (6.39)$$

Given the same RHS, UTASP finds the same \mathbf{v}_2 with DASP, but it computes a more accurate \mathbf{v}_1 . The preconditioning matrix of UTASP is defined as

$$\overline{\mathbf{M}}_{UTASP} = \begin{bmatrix} \overline{\mathbf{A}}_{11}^{NF} & \overline{\mathbf{A}}_{12}^{NF} \\ 0 & \overline{\mathbf{S}} \end{bmatrix}. \quad (6.40)$$

Lower Triangular Approximate Schur Preconditioner (LTASP)

If we set $\overline{\mathbf{A}}_{12}^{NF} \approx 0$ instead of $\overline{\mathbf{A}}_{21}^{NF}$, we obtain the lower triangular approximate Schur preconditioner (LTASP). In this case, we have to first solve for \mathbf{v}_1 from

$$\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}_1. \quad (6.41)$$

Then, we can find \mathbf{v}_2 using \mathbf{v}_1 :

$$\overline{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}_2 - \overline{\mathbf{A}}_{21}^{NF} \cdot (\overline{\mathbf{A}}_{11}^{NF})^{-1} \cdot \mathbf{w}_1 = \mathbf{w}_2 - \overline{\mathbf{A}}_{21}^{NF} \cdot \mathbf{v}_1. \quad (6.42)$$

Compared to DASP, LTASP finds the same \mathbf{v}_1 but a more accurate \mathbf{v}_2 for a given RHS. The preconditioning matrix of LTASP is defined as

$$\overline{\mathbf{M}}_{LTASP} = \begin{bmatrix} \overline{\mathbf{A}}_{11}^{NF} & 0 \\ \overline{\mathbf{A}}_{21}^{NF} & \overline{\mathbf{S}} \end{bmatrix}. \quad (6.43)$$

Approximate Schur Preconditioner (ASP)

In an effort to devise an effective preconditioner, it is also an option not to omit any of the off-diagonal blocks in $\overline{\mathbf{A}}^{NF}$. For efficiency, however, solutions of the systems involving $\overline{\mathbf{S}}$ and $\overline{\mathbf{A}}_{11}^{NF}$ should be performed approximately, as will be detailed in Section 6.5.1. Hence, we call this preconditioner the approximate Schur preconditioner (ASP), for which the preconditioning matrix is given by

$$\overline{\mathbf{M}}_{ASP} = \overline{\mathbf{A}}^{NF} = \begin{bmatrix} \overline{\mathbf{A}}_{11}^{NF} & \overline{\mathbf{0}} \\ \overline{\mathbf{A}}_{21}^{NF} & \overline{\mathbf{S}} \end{bmatrix} \cdot \begin{bmatrix} \overline{\mathbf{I}} & (\overline{\mathbf{A}}_{11}^{NF})^{-1} \cdot \overline{\mathbf{A}}_{12}^{NF} \\ \overline{\mathbf{0}} & \overline{\mathbf{I}} \end{bmatrix}. \quad (6.44)$$

6.5.1 Approximations of the Solutions Involving the (1, 1) Partition and the Schur complement

The performance of the preconditioners explained in the previous Section depends on the availability of fast and approximate solutions to

$$\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}'_1 \quad (6.45)$$

and

$$\overline{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}'_2, \quad (6.46)$$

where \mathbf{w}'_1 and \mathbf{w}'_2 take different forms depending on the type of preconditioner. Since the approximations performed in these solutions define a preconditioner for the linear system (6.1), accurate solutions are not required. On the other hand, very crude approximations of the exact solutions may deteriorate the quality of the preconditioner, and iteration counts may not be decreased as desired.

In the literature, several approximation strategies for the solutions of (6.45) and (6.46) have been proposed, but many of them are strongly problem dependent [118]. For surface integral-equation formulations, we discuss possible approximations and our approach for $\overline{\mathbf{A}}_{11}$ and $\overline{\mathbf{S}}$.

Approximating the Solutions Involving $\overline{\mathbf{A}}_{11}^{NF}$

For some specific problems, many efficient techniques are available for a fast and accurate solution of (6.45). For example, if the system matrix were obtained from the discretization of a differential operator, in many cases a few multigrid sweeps would yield efficient and yet sufficiently accurate solutions [133]. In general situations, however, one must resort to algebraic approaches, such as ILU factorizations, SAIs, or approximations by a few iterations of a Krylov subspace method.

In this work, we approximate the solution of the system (6.45) by a SAI of $\overline{\mathbf{A}}_{11}^{NF}$. We denote the SAI of $\overline{\mathbf{A}}_{11}^{NF}$ as $\overline{\mathbf{M}}_{11}$. Hence, our approximation becomes

$$(\overline{\mathbf{A}}_{11}^{NF})^{-1} \approx \overline{\mathbf{M}}_{11}. \quad (6.47)$$

SAI preconditioners have been successfully used in CEM for PEC problems [44, 86, 92, 85]. Two important advantages of SAI preconditioners over ILU-type preconditioners are robustness and ease of parallelization [93]. In our case, it is also possible to alleviate the high construction cost of SAI using the block structure of the near-field matrix [92, 44], as we describe in the following paragraph.

Approximate inverses of sparse matrices can be obtained in several ways [94, 95, 96, 97, 93]. Among these methods, we make use of the Frobenius-norm technique [93], which decouples the generation of an $N \times N$ SAI into N independent least-squares problems for each row. Then, each least-squares problem can be solved by employing a QR factorization and an upper-triangular system

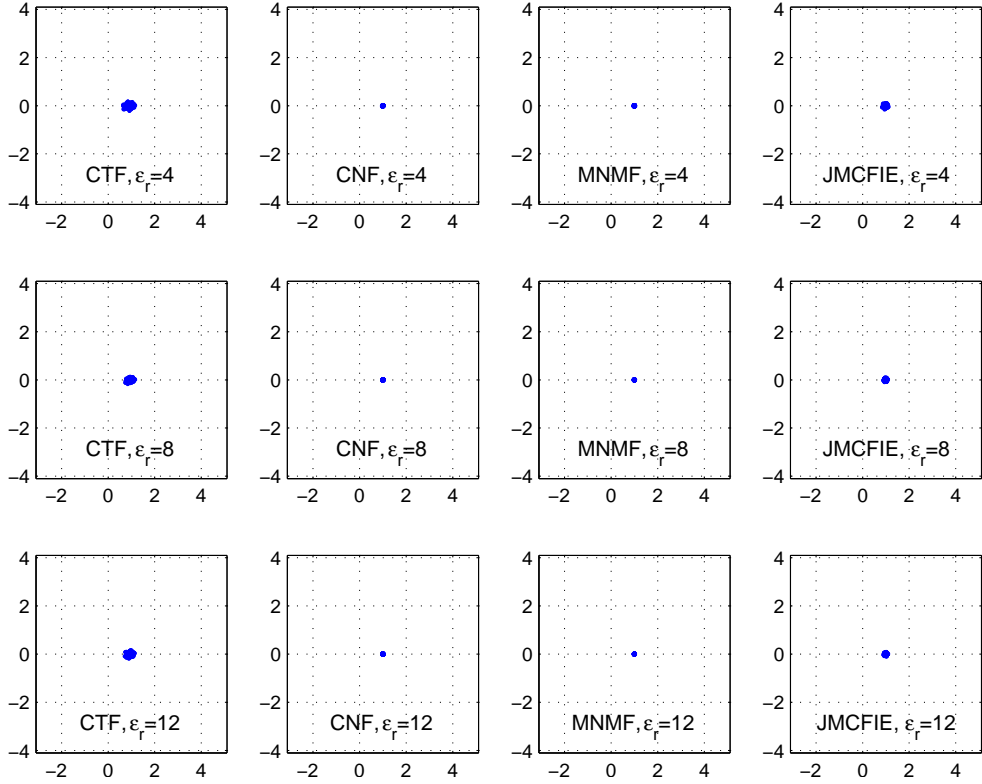


Figure 6.3: Eigenvalues of $\overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{11}^{NF}$ for different formulations and increasing dielectric constants of 4, 8, and 12.

solution [1]. On the other hand, due to the block structure of $\overline{\mathbf{A}}_{11}^{NF}$, we need to perform only N/m QR factorizations, where m is the average block size of $\overline{\mathbf{A}}_{11}^{NF}$. For a 0.25λ lowest-level box size and $\lambda/10$ mesh size, typical values of m lie between 20 and 50, depending on the geometry. Since the QR factorization constitutes the dominant cost in a least-squares solution, we significantly reduce the construction time of SAI.

We evaluate the approximation (6.47) in Fig. 6.3, where we depict eigenvalues of matrices $\overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{11}^{NF}$ for different formulations and increasing dielectric constants of 4, 8, and 12. The geometry is a 0.5λ sphere involving 1,860 unknowns. We see that eigenvalues are very tightly clustered around (1, 0) for normal formulations (CNF and MNMF). For CTF, we see a slightly looser clustering than CNF and MNMF. JMCFIE lies between the two cases. Also note that the spectra of $\overline{\mathbf{A}}_{11}^{NF}$ are unaffected by the increase of the dielectric constant.

Approximating the Solutions Involving $\overline{\mathbf{S}}$

The approximation involving the Schur complement matrix $\overline{\mathbf{S}}$ is more subtle than that of $\overline{\mathbf{A}}_{11}^{NF}$. Moreover, it is shown that the approximation quality provided to the system involving $\overline{\mathbf{S}}$ should accommodate the approximation level to the system involving $\overline{\mathbf{A}}_{11}^{NF}$ [120]. Therefore, we try to find an approximation for $\overline{\mathbf{S}}$ that is as good as the approximation for $\overline{\mathbf{A}}_{11}^{NF}$.

In the literature related to saddle-point problems, several choices exist when the system matrix $\overline{\mathbf{A}}$ is symmetric [118]. These choices include multigrid sweeps and low-order discretization of the related operator. Many purely algebraic approaches have also been proposed for the nonsymmetric case, in which the (2,2) partition is zero. Those approaches include approximating the inverse of the (1,1) partition in the Schur complement by the inverse of the diagonal or block-diagonal part of the (1,1) partition. Better approximations can be provided in the form of incomplete factors (e.g., [134]). However, a limited number of methods exist for the case of a nonzero (2,2) partition [118, 120, 119]. Perhaps one of the most applicable methods is to use a Krylov subspace solver to obtain an approximate solution of the system (6.46). MVMs with $\overline{\mathbf{S}}$ can be provided to the solver by multiplications with the (2,2) and off-diagonal partitions, and by another iterative solve with $\overline{\mathbf{A}}_{11}^{NF}$. The required solve with $\overline{\mathbf{A}}_{11}^{NF}$, however, can significantly increase the application cost of the preconditioner. Moreover, in many cases, a preconditioner for $\overline{\mathbf{S}}$ is still required to accelerate the Krylov subspace solver.

In this work, we consider the following strategies to approximate the inverse of $\overline{\mathbf{S}}$ for the solution of (6.46):

1. As a simple approach, we can approximate the inverse of $\overline{\mathbf{S}}$ using its block-diagonal part. Let $\overline{\mathbf{B}}_{ij}$ denote the block-diagonal part of the near-field partition (i, j) , which consists of the self-interactions of the lowest-level

clusters. Then, the approximation is

$$\overline{\mathbf{S}}^{-1} \approx \overline{\mathbf{M}}_{BD} = \left(\overline{\mathbf{B}}_{22} - \overline{\mathbf{B}}_{21} \cdot (\overline{\mathbf{B}}_{11})^{-1} \cdot \overline{\mathbf{B}}_{12} \right)^{-1}. \quad (6.48)$$

2. For normal formulations and JMCFIE, the resulting partitions and the Schur complement are likely to have some degree of diagonal dominance. Therefore, we expect to benefit from the approximation (6.48). On the other hand, CTF partitions are far from being diagonally dominant and indeed block-diagonal preconditioners decelerate the convergence rate of iterative solvers for tangential formulations of PEC problems [13]. Thus, for CTF, instead of the approximation in (6.48), we consider the modification formula [135] that expresses the inverse of $\overline{\mathbf{S}}$ as

$$\overline{\mathbf{S}}^{-1} = (\overline{\mathbf{A}}_{22}^{NF})^{-1} + (\overline{\mathbf{A}}_{22}^{NF})^{-1} \cdot \overline{\mathbf{A}}_{21}^{NF} \cdot \overline{\mathbf{S}}'^{-1} \cdot \overline{\mathbf{A}}_{12}^{NF} \cdot (\overline{\mathbf{A}}_{22}^{NF})^{-1}, \quad (6.49)$$

where

$$\overline{\mathbf{S}}' = \overline{\mathbf{A}}_{11}^{NF} - \overline{\mathbf{A}}_{12}^{NF} \cdot (\overline{\mathbf{A}}_{22}^{NF})^{-1} \cdot \overline{\mathbf{A}}_{21}^{NF}. \quad (6.50)$$

The modification formula is also known as the Woodbury matrix identity [35] or the matrix inversion lemma in control theory [136]. To obtain an approximate inverse for $\overline{\mathbf{S}}$, we discard the second term in $\overline{\mathbf{S}}'$ and approximate the inverses of $\overline{\mathbf{A}}_{11}^{NF}$ and $\overline{\mathbf{A}}_{22}^{NF}$ with SAIs, i.e.,

$$\overline{\mathbf{S}}^{-1} \approx \overline{\mathbf{M}}_{MF} = \overline{\mathbf{M}}_{22} + \overline{\mathbf{M}}_{22} \cdot \overline{\mathbf{A}}_{21}^{NF} \cdot \overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{12} \cdot \overline{\mathbf{M}}_{22} \quad (6.51)$$

$$= \overline{\mathbf{M}}_{22} \cdot (\overline{\mathbf{I}} + \overline{\mathbf{A}}_{21}^{NF} \cdot \overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{12}^{NF} \cdot \overline{\mathbf{M}}_{22}), \quad (6.52)$$

where $\overline{\mathbf{M}}_{22}$ denotes the SAI of $\overline{\mathbf{A}}_{22}^{NF}$. Note that $\overline{\mathbf{A}}_{22}^{NF} = \overline{\mathbf{A}}_{11}^{NF}$ for CTF, hence, we need to construct and store only one SAI. The application of (6.52) can be performed by sparse MVMs during the iterative solution of (6.1), without the need to store any matrices other than SAI.

3. We can approximate the inverse of the Schur complement matrix by

$$\overline{\mathbf{S}}^{-1} \approx (\overline{\mathbf{A}}_{22}^{NF})^{-1} \approx \overline{\mathbf{M}}_{22}, \quad (6.53)$$

assuming the first term in the RHS of (6.32) is the dominant term in the Schur complement matrix. $\overline{\mathbf{M}}_{22}$ denotes the SAI of $\overline{\mathbf{A}}_{22}^{NF}$. Again, we need to construct a second SAI only for MNMF.

4. Finally, by employing an incomplete matrix-matrix multiplication, we generate an explicit SAI for $\overline{\mathbf{S}}$ that involves both of its first and second terms. First, we compute a sparse approximation to $\overline{\mathbf{S}}$ in the form of

$$\widetilde{\overline{\mathbf{S}}} = \overline{\mathbf{A}}_{22}^{NF} - \overline{\mathbf{A}}_{21}^{NF} \odot \overline{\mathbf{M}}_{11} \odot \overline{\mathbf{A}}_{12}, \quad (6.54)$$

where \odot denotes an incomplete matrix-matrix multiplication obtained by retaining the near-field sparsity pattern and $\overline{\mathbf{M}}_{11}$ is the SAI of $\overline{\mathbf{A}}_{11}^{NF}$. Then, the approximation is performed as

$$\overline{\mathbf{S}}^{-1} \approx \widetilde{\overline{\mathbf{S}}}^{-1} \approx \overline{\mathbf{M}}_S, \quad (6.55)$$

where $\overline{\mathbf{M}}_S$ denotes a SAI approximation to the inverse of $\widetilde{\overline{\mathbf{S}}}$. In our implementation, the block entries of the near-field partitions are stored row-wise. Therefore, the incomplete matrix-matrix multiplication can be performed in $\mathcal{O}(N)$ time using the *ikj* loop order of the block matrix-matrix multiplication [35] so that the block entries of the matrices are accessed row-wise. Details of this operation are elucidated with a pseudocode in Fig. 6.4. Note that the “if statement” in the innermost loop ensures that a block $\overline{\mathbf{C}}_{ij}$ is updated only if clusters *i* and *j* are in the near-field zone of each other. In this way, the near-field sparsity pattern is preserved for the product matrix $\overline{\mathbf{C}}$.

We evaluate the aforementioned approximations in Figs. 6.5, 6.6, and 6.7, where we depict the eigenvalues of the preconditioned Schur complement matrices. We summarize our comments as follows:

- In Fig. 6.5, we depict $\overline{\mathbf{M}}_{MF} \cdot \overline{\mathbf{S}}$ for CTF and $\overline{\mathbf{M}}_{BD} \cdot \overline{\mathbf{S}}$ for other formulations. We see that the clustering (or localization) of the eigenvalues

```

 $\overline{\mathbf{C}} = \overline{\mathbf{0}}$ 
for each lowest-level cluster  $i$  do
  for each cluster  $k \in \mathcal{N}(i)$  do
    for each cluster  $j \in \mathcal{N}(k)$  do
      if  $j \in \mathcal{N}(i)$  then
         $\overline{\mathbf{C}}_{ij} = \overline{\mathbf{C}}_{ij} + \overline{\mathbf{D}}_{ik} \cdot \overline{\mathbf{E}}_{kj}$ 
      endif
    endfor
  endfor
endfor

```

Figure 6.4: Incomplete matrix-matrix multiplication of $\overline{\mathbf{C}} = \overline{\mathbf{D}} \cdot \overline{\mathbf{E}}$, where $\overline{\mathbf{C}}$, $\overline{\mathbf{D}}$, and $\overline{\mathbf{E}}$ are block near-field matrices with the same sparsity pattern. $\overline{\mathbf{C}}_{ij}$ denotes the block of the near-field matrix $\overline{\mathbf{C}}$ that corresponds to the interaction of cluster i with cluster j . $\mathcal{N}(i)$ denotes the clusters that are in the near-field zone of cluster i .

diminishes with the increasing dielectric constant, particularly for CTF and CNF. Even though the scattering (or spread) of the eigenvalues of CTF with $\overline{\mathbf{M}}_{BD}$ is much worse than of CNF (not shown here), interestingly, the spectra of JMCFIE are less affected from the increase in the dielectric constant than those of CTF and CNF. This can be related to the stronger diagonal dominance of matrices produced with combined formulations than those of tangential formulations [13]. Nonetheless, from the spectra in Fig. 6.5, we conclude that the approximations (6.48) and (6.52) are significantly poorer than (6.47) for all formulations.

- When we omit the second term of the Schur complement matrix in (6.32) and perform the approximation (6.53), we observe from Fig. 6.6 that the spectra of CNF are extensively scattered with an increasing dielectric constant. Even though not as much as those of CNF, the spectra of CTF are also scattered. JMCFIE, being a combination of CTF and CNF, is also affected from the scattering of CNF and CTF. Hence, we conclude that this approximation is problematic for high dielectric constants in CTF, CNF, and JMCFIE. MNMF, on the other hand, is less affected from the increase

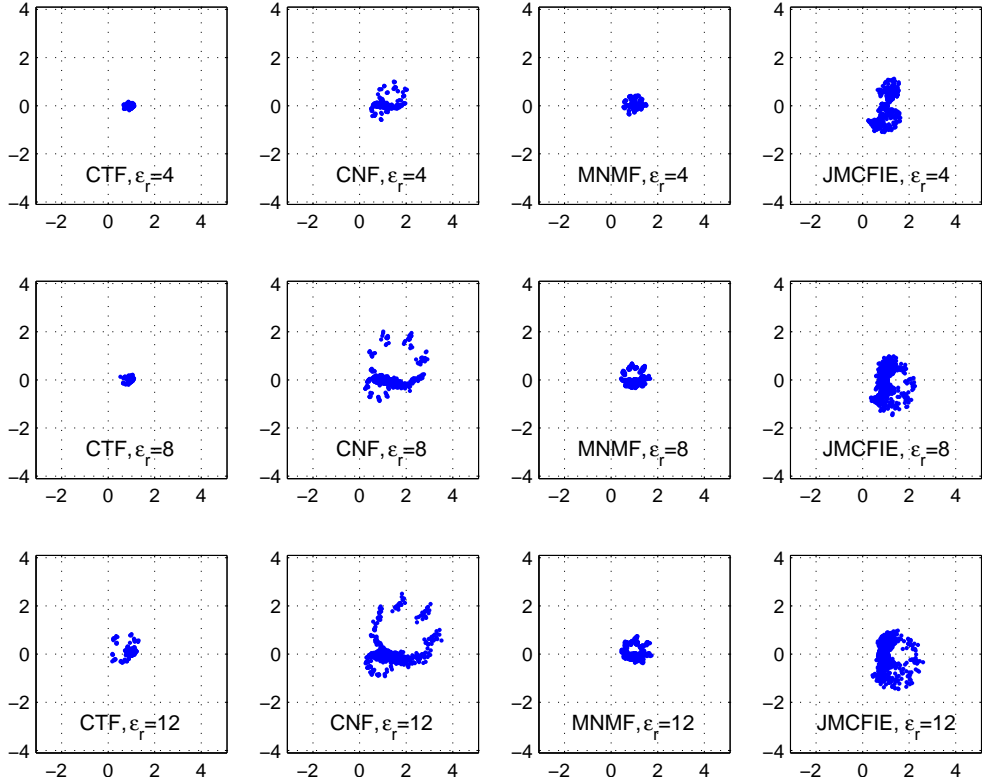


Figure 6.5: Eigenvalues of preconditioned Schur complement $\overline{\mathbf{S}}$ for increasing dielectric constants of 4, 8, and 12. CTF is preconditioned with $\overline{\mathbf{M}}_{MF}$, whereas $\overline{\mathbf{M}}_{BD}$ is used as the preconditioner for the other formulations.

in the dielectric constant. However, when we compare Figs. 6.6 and 6.3, we conclude that the approximation (6.53) is also significantly poorer than (6.47) for MNMF.

- From Fig. 6.7, it is clear that the best approximation for the Schur complement $\overline{\mathbf{S}}$ is provided by $\overline{\mathbf{M}}_S$. Clusterings of CTF, MNMF, and JMCFIE are tight, whereas CNF exhibits slightly looser clustering. When we compare Figs. 6.7 and 6.3, we observe that the approximation (6.47) is as good as (6.55) for CTF. For other formulations, clusterings in Fig. 6.7 are a little looser compared to those in Fig. 6.3.

From these discussions, we conclude that $\overline{\mathbf{M}}_S$ provides the most appropriate approximation to the inverse of the Schur complement matrix $\overline{\mathbf{S}}$. The other two approximate inverses $\overline{\mathbf{M}}_{BD}$ and $\overline{\mathbf{M}}_{22}$ have lower setup and memory costs, but

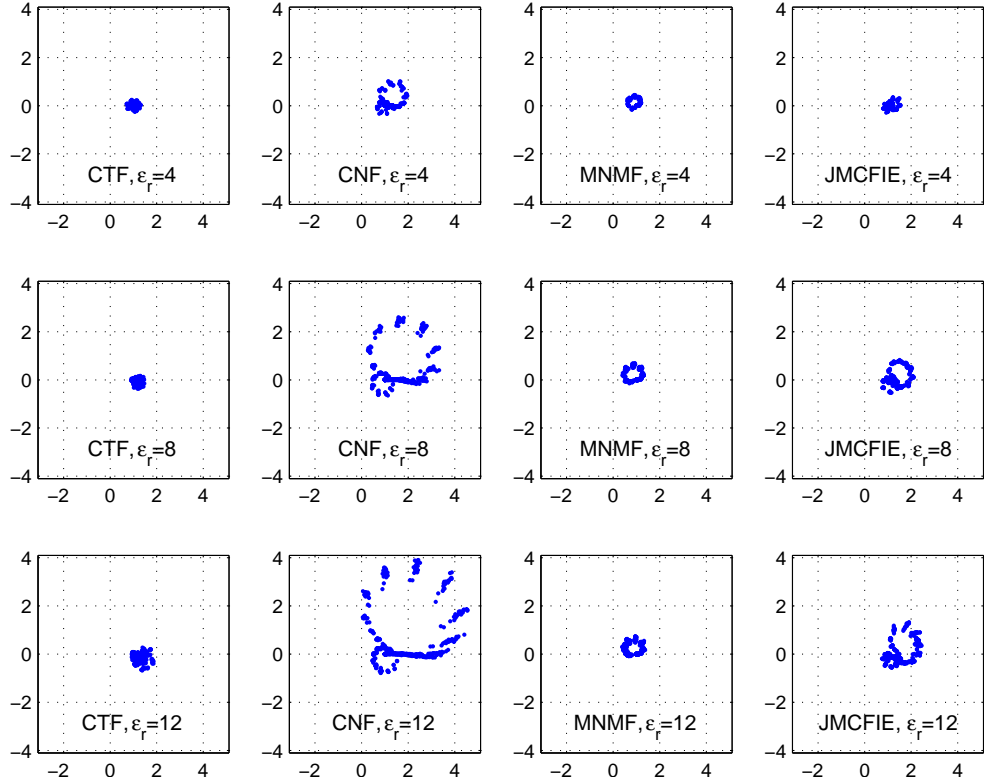


Figure 6.6: Eigenvalues of $\overline{\mathbf{M}}_{22} \cdot \overline{\mathbf{S}}$ for different formulations and increasing dielectric constants of 4, 8, and 12.

they are far from ensuring the requirement that the approximation for $\overline{\mathbf{S}}$ should be as good as that of $\overline{\mathbf{A}}_{11}$. On the other hand, in the context of a nested iterative solver (e.g., [56]), $\overline{\mathbf{M}}_S$ and other approximations, i.e., (6.48), (6.52), and (6.53), can also be utilized as inner preconditioners for iterative solutions of $\overline{\mathbf{S}}$, and this will be the subject of the next section.

6.6 Iterative Schur Complement Preconditioners

These preconditioners are formed by solving (6.45) and (6.46) iteratively and with controllable accuracy. We consider the following ones:

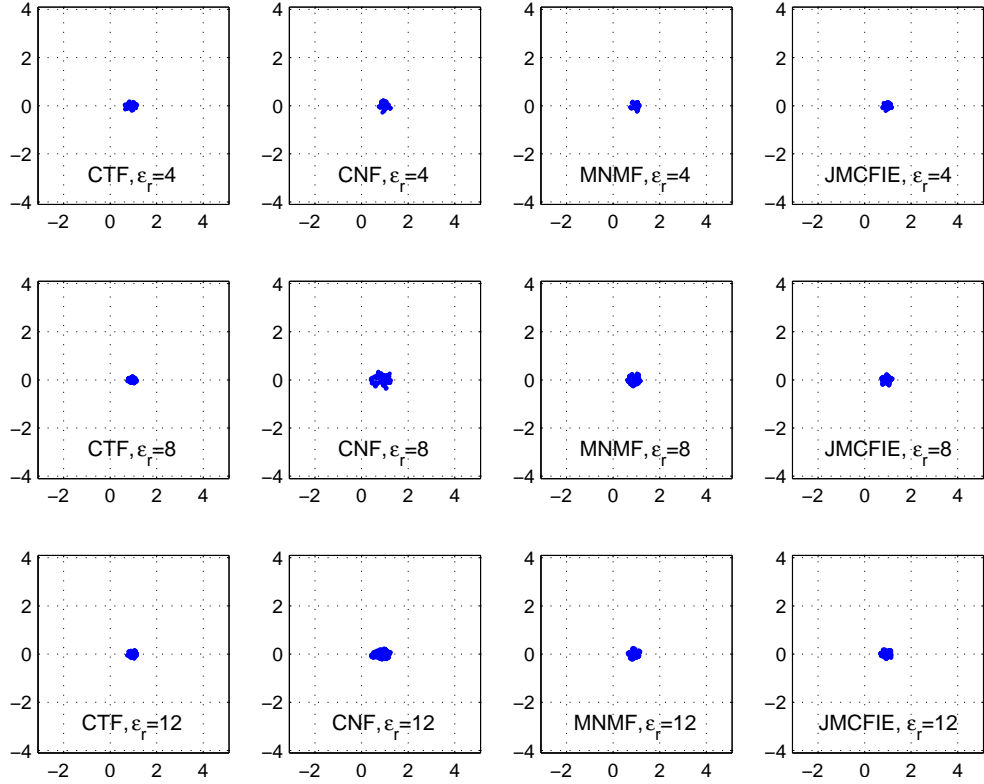


Figure 6.7: Eigenvalues of $\overline{\mathbf{M}}_S \cdot \overline{\mathbf{S}}$ for different formulations and increasing dielectric constants of 4, 8, and 12.

1. The upper triangular iterative Schur preconditioner (UTISP), which is formed by omitting $\overline{\mathbf{A}}_{21}^{NF}$ in the RHS of (6.31).
2. The lower triangular iterative Schur preconditioner (LTISP), which is formed by omitting $\overline{\mathbf{A}}_{12}^{NF}$ in (6.33).
3. The iterative Schur preconditioner (ISP), which is formed by iteratively solving (6.45) and (6.46) with some approximations that will be described.

These preconditioners, respectively, have the same form with UTASP, LTASP, and ASP introduced in Section 6.5. However, we solve those reduced systems via preconditioned Krylov subspace iterative methods. Sifert and de Sturler showed that there is no advantage in solving one system significantly better than the other [120]. Hence, we solve both systems using the same stopping tolerance. In this case, one should use a flexible solver, such as FGMRES, for the solution of the partitioned linear system [2]. The extra cost of using FGMRES is to store two sets

of preconditioned residual vectors instead of one, hence, the memory requirement of GMRES doubles. However, it is possible to use regular GMRES by ensuring a fixed number of inner iterations per solution for both systems. In the next two sections, we separately analyze the solutions of (6.45) and (6.46). Then, we determine an appropriate inner stopping tolerance for these two solutions by comparing the preconditioners with varying inner tolerances.

6.6.1 Iterative Solutions Involving the (1, 1) Partition

To reduce the application cost of the Schur complement preconditioners, approximate solutions of the reduced systems should be obtained in a few iterations. For the solutions of the systems involving $\overline{\mathbf{A}}_{11}^{NF}$, we use the SAI preconditioner that has been described in Section 6.5, which has proven to be successful both in EFIE and CFIE [44, 83].

In Table 6.1, we compare solutions of (6.45) obtained without a preconditioner (No PC) and using SAI. No-restart GMRES is used as the iterative solver. This problem involves a sphere illuminated with a plane wave. The problem size is increased by increasing the frequency and using $\lambda/10$ mesh size for all cases. We observe that SAI is very successful in reducing the iteration counts for all formulations. Furthermore, iteration counts obtained with the SAI preconditioner remain fixed as the number of unknowns is increased. Hence, its use significantly increases the efficiency of the Schur complement preconditioners. In this problem, the sphere involves a moderate relative dielectric constant of 4.0, however, as shown in Section 6.5, the approximation quality of SAI to $\overline{\mathbf{A}}_{11}^{NF}$ is not adversely affected from an increase in the dielectric constant.

We repeat the experiment on two photonic crystal problems: a periodic slabs with 262,920 unknowns and a perforated waveguide with 162,420 unknowns. For both problems RHSs are found using Hertzian dipoles for both structures.

Table 6.1: Number of iterations for the solution of $\overline{\mathbf{A}}_{11}^{NF} \cdot \mathbf{v}_1 = \mathbf{w}'_1$ to reduce the residual error by 10^{-6} .

N	CTF		CNF		JMCFIE		MNMF	
	No PC	SAI	No PC	SAI	No PC	SAI	No PC	SAI
1,860	167	9	13	4	38	7	13	4
7,446	195	10	14	3	37	6	14	3
29,742	217	10	13	3	38	6	13	3
65,724	243	10	14	3	39	6	14	3
264,006	294	10	14	3	41	6	14	3

In Figs. 6.8 and 6.9, we compare the no-restart GMRES solutions of the no-preconditioner case (No PC) and SAI ($\overline{\mathbf{M}}_{11}$). We analyze convergence for the first ten iterations since a rough solution, generally up to 0.1 residual error, is shown to be sufficient to yield a successful preconditioner [118, 65]. For both problems we observe that $\overline{\mathbf{M}}_{11}$ significantly accelerates the solutions of all formulations except MNMF. Note that convergence is too slow without a preconditioner for CTF solutions and for the JMCFIE solution of the perforated waveguide. On the other hand, it is possible to achieve 0.1 residual error in a few iterations when $\overline{\mathbf{M}}_{11}$ is employed, even though solutions stagnate later for CTF.

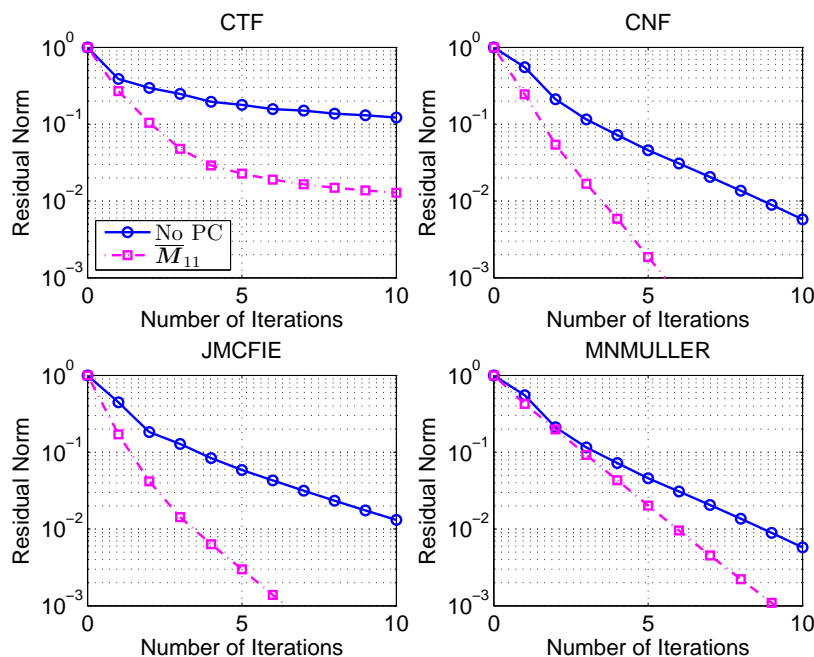


Figure 6.8: Comparison of iterative solutions of (6.33) without a preconditioner and using $\overline{\mathbf{M}}_{11}$ for a periodic-slabs problem involving 262,920 unknowns.

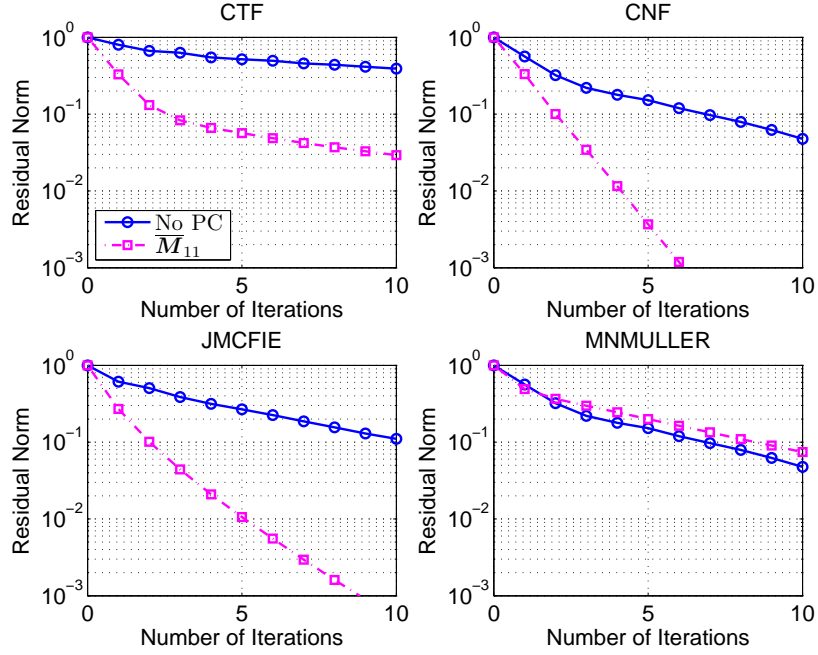


Figure 6.9: Comparison of iterative solutions of (6.33) without a preconditioner and using $\overline{\mathbf{M}}_{11}$ for perforated waveguide involving 162,420 unknowns.

6.6.2 Iterative Solutions Involving the Schur Complement

For solving the system in (6.46), we do not need to explicitly form the Schur complement $\overline{\mathbf{S}}$, because matrix-vector products required by Krylov subspace solvers can be performed by multiplications with $\overline{\mathbf{A}}_{22}^{NF}$, $\overline{\mathbf{A}}_{21}^{NF}$, $\overline{\mathbf{A}}_{12}^{NF}$, and solves with $\overline{\mathbf{A}}_{11}^{NF}$ [118]. However, we get rid of the solves with $\overline{\mathbf{A}}_{11}^{NF}$ by approximating the Schur complement in the form

$$\widetilde{\mathbf{S}} = \overline{\mathbf{A}}_{22}^{NF} - \overline{\mathbf{A}}_{21}^{NF} \cdot \overline{\mathbf{M}}_{11} \cdot \overline{\mathbf{A}}_{12}^{NF}, \quad (6.56)$$

where $\overline{\mathbf{M}}_{11}$ is the SAI of $\overline{\mathbf{A}}_{11}$. Hence, we solve

$$\widetilde{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}'_2, \quad (6.57)$$

instead of (6.46). In this way, we significantly reduce the application cost of the preconditioners. Note that $\overline{\mathbf{M}}_{11}$ is also used as a preconditioner for the iterative solution of (6.45), hence, this choice has no additional setup or memory

cost. We adopt the same approach for the RHS of ISP and approximate $\mathbf{w}'_2 = \mathbf{w}_2 - \overline{\mathbf{A}}_{21}^{NF} \cdot (\overline{\mathbf{A}}_{11}^{NF})^{-1} \cdot \mathbf{w}_1$ by

$$\widetilde{\mathbf{w}}'_2 = \mathbf{w}_2 - \overline{\mathbf{A}}_{21}^{NF} \cdot \overline{\mathbf{M}}_{11} \cdot \mathbf{w}_1 \quad (6.58)$$

to avoid an inner solve in each iteration.

The approximate inverses mentioned in Section 6.5 can be considered as preconditioners for the iterative solution of (6.56). Assuming the first term in the RHS of (6.32) is the dominant term and using the equality of the diagonal partitions, $\overline{\mathbf{M}}_{11}$, which is the approximate inverse of $\overline{\mathbf{A}}_{11}^{NF}$, can also be used as a preconditioner for the Schur complement. Even though it is shown that $\overline{\mathbf{M}}_{11}$ is not suitable for use as a direct inverse, the spectra shown in Fig. 6.6 reveal that it can still be used as an effective preconditioner. To verify this claim, we analyze the solution of (6.57) for No PC and SAI ($\overline{\mathbf{M}}_{11}$) in Table 6.2. The no-restart GMRES solver is used to solve the sphere problem with a 4.0 relative dielectric constant and the RHSs are found by plane-wave excitations. Similar to the solution of (6.45), we observe a remarkable decrease in iteration counts of all formulations. Furthermore, six-order reduction of initial residual norms can be obtained in merely 10 iterations for both CTF and JMCIE formulations. However, if the dielectric object has a very high dielectric constant, one may have to use a more robust preconditioner for the Schur complement. Such a preconditioner, called $\overline{\mathbf{M}}_S$, has been constructed in Section 6.5 by approximating the Schur complement using incomplete matrix-matrix multiplications and then approximately inverting the resulting sparse matrix.

In Figs. 6.10 and 6.11, we compare the solutions of (6.46) with photonic crystal problems for the no-preconditioner case, and three approximate inverses, i.e., $\overline{\mathbf{M}}_{11}$, $\overline{\mathbf{M}}_{BD}$, and $\overline{\mathbf{M}}_S$. First, observe that Schur complement solutions are more difficult to solve than the solutions involving $\overline{\mathbf{Z}}_{11}^{NF}$, shown in Figs. 6.8 and 6.9. In particular, it is not possible to attain fast convergence for normal formulations

Table 6.2: Number of iterations for the solution of $\tilde{\mathbf{S}} \cdot \mathbf{v}_2 = \mathbf{w}'_2$ to reduce the residual error by 10^{-6} .

N	CTF		CNF		JMCFIE		MNMF	
	No PC	SAI	No PC	SAI	No PC	SAI	No PC	SAI
1,860	166	10	26	17	40	10	18	11
7,446	193	10	26	17	40	10	18	11
29,742	213	10	26	16	43	9	19	11
65,724	238	9	27	16	44	9	20	11
264,006	282	9	29	16	45	9	21	11

CNF and MNMF. As a result, we prefer to use JMCFIE in photonic-crystal problems instead of CNF and MNMF. Note that JMCFIE produces more accurate solutions compared to CNF and MNMF [114]. Among the three approximate inverses, $\overline{\mathbf{M}}_S$ performs the best and accelerates the iterative solutions significantly. On the other hand, for the perforated waveguide problem that has a high contrast, attaining fast convergence is still difficult and solution starts to stagnate even with $\overline{\mathbf{M}}_S$ after first five iterations. Hence, it is wise to set a low maximum number of inner iterations for ISP to avoid wasted effort.

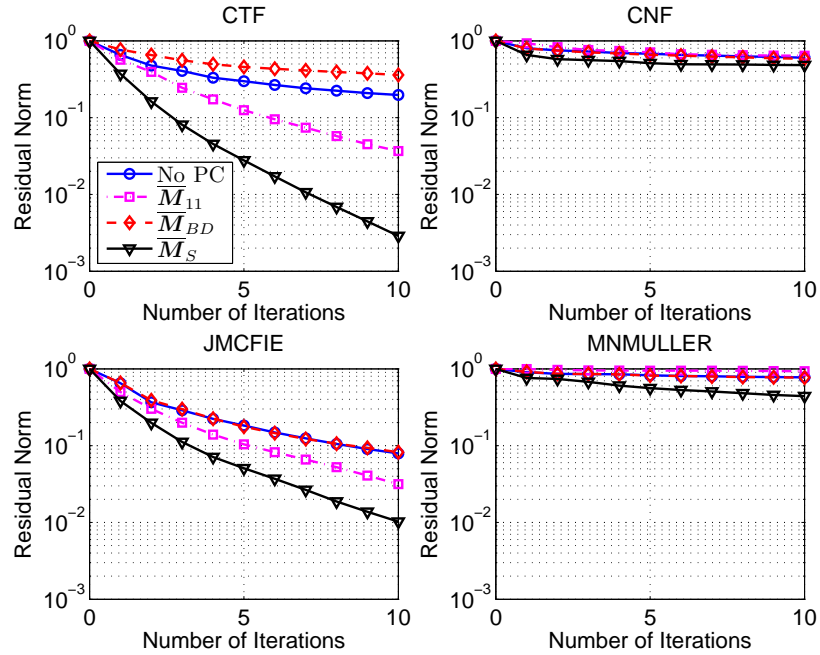


Figure 6.10: Iterative solutions of (6.46) with various preconditioners for a periodic-slabs problem involving 262,920 unknowns.

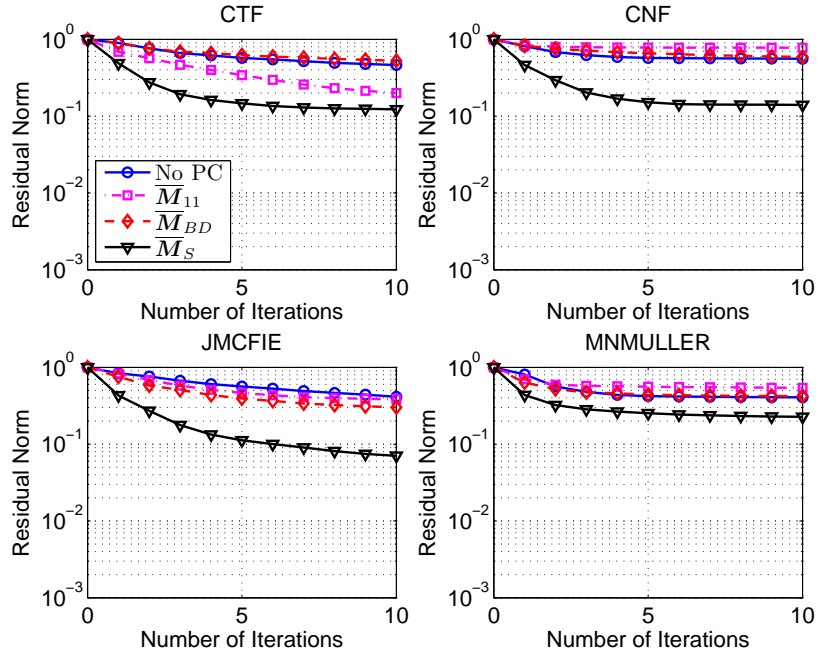


Figure 6.11: Iterative solutions of (6.46) with various preconditioners for a perforated waveguide involving 162,420 unknowns.

6.6.3 Stopping Criteria for Inner Solutions

Thanks to SAI, the fast convergence of the linear systems (6.45) and (6.57) can provide efficient solutions for preconditioning. Nonetheless, applying the iterative Schur complement preconditioners is more costly compared to simple preconditioners, such as 4PBDP [114], and the non-iterative Schur complement preconditioners [64]. For this purpose, we compare the following three versions of iterative Schur complement preconditioners, which are obtained by increasing the tolerances of inner solutions:

1. The stopping tolerances of (6.45) and (6.46) are set to 10^{-6} . Furthermore, extra solves are used to approximate the inverse of $\overline{\mathbf{A}}_{11}^{NF}$ in the Schur complement and for the RHS term of ISP. This benchmark preconditioner corresponds to an accurate solution of the near-field matrix and is expected to provide a lower bound for the Schur complement preconditioners in terms of iteration counts.

2. Stopping tolerances of (6.45) and (6.57) are set to 10^{-3} .
3. Stopping tolerances of (6.45) and (6.57) are set to 0.1.

In Table 6.3, we show the number of iterations obtained using these inner solutions for the sphere problem involving 65,724 unknowns. From the results, it is evident that accurate inner solves are not required for the Schur complement preconditioners. A low inner tolerance, such as 10^{-6} , can even increase the iteration counts of highly indefinite systems obtained from CTF. Also for other formulations, extra solves to approximate the inverse of $\overline{\mathbf{A}}_{11}^{NF}$ in the Schur complement and the RHS term of ISP are very costly and do not lead to a significant reduction in the outer iteration counts. Besides, the iteration counts corresponding to 10^{-3} and 0.1 inner tolerances are quite close, but applying preconditioners using a 0.1 inner tolerance is much cheaper. Hence, for the experiments reported in the next section, we set the inner tolerance at 0.1 residual error and a maximum of three or five inner iterations, depending on the difficulty of the problem.

Table 6.3: Number of iterations for the sphere problem involving 65,724 unknowns using the iterative Schur complement preconditioners with varying inner tolerances. For 10^{-6} inner tolerance, extra inner solves for the Schur complement and for the RHS of ISP are used.

SIE	LTISP			UTISP			ISP		
	10^{-6}	10^{-3}	0.1	10^{-6}	10^{-3}	0.1	10^{-6}	10^{-3}	0.1
CTF	87	87	89	79	77	83	68	65	69
CNF	59	59	60	46	46	51	38	38	43
JMCFIE	57	57	55	66	68	71	49	49	51

6.7 Numerical Results

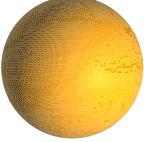
We use the following setup in our experiments:

- Computations are performed on an Intel Xeon 5355 processor with 16 GB of available memory.

- The generalized minimal residual method (GMRES) [2] with no restart is used as the iterative solver [73]. Even though it is not reported in detail here, contrary to findings in [137], we observe a significant difference between the performances of GMRES and other non-optimal solvers, such as conjugate gradient squared (CGS) or biconjugate gradient stabilized (BiCGStab). Comparisons of MVM counts for the sphere problem presented in Section 6.7.1 and in [114] demonstrate the superiority of GMRES. We note that the performance difference of GMRES and other non-optimal solvers is even more severe for the real-life problems of Sections 6.7.2 and 6.7.4.
- Iterations are performed until the norm of the initial residual is reduced by a factor of 10^{-3} . This error level is practical [26] and in accordance with the controllable error performed in MLFMA.
- Solutions are started with a zero initial guess and terminated if a maximum of 1,000 iterations is reached.

For comparison purposes, we provide solutions with the no-preconditioner case (No PC), with 4PBDP [114], and with an ILU-type preconditioner. 4PBDP is a simple preconditioner constructed by the inclusion of only self-interactions of the lowest-level clusters in each partition. Among several types of ILU preconditioners, the dual-threshold ILUT preconditioner [2] has been shown to be very ineffective in a finite-element implementation of the Navier-Stokes equations [119]. In CEM, however, ILU-type preconditioners have been successfully employed for surface integral-equation formulations of PEC problems [40]. For the ILUT preconditioner, we set the threshold values so that it uses up the same amount of memory as ILU(0) and the near-field matrix [40]. We also include, whenever possible, iterative counts obtained using the exact factorization of the whole near-field matrix by an LU factorization. This preconditioner, which we

Table 6.4: Salient features of the sphere problems investigated in this study.

Problem		Frequency (GHz)	Size (λ)	MLFMA Levels	Number of Unknowns
	S1	1.0	2	4	7,446
	S2	1.5	3	5	16,728
	S3	3.0	6	6	65,724
	S4	6.0	12	7	264,006
	S5	8.5	17	8	540,450
Note: λ denotes the wavelength at the frequency of operation.					

call NF-LU, has prohibitive memory and time requirements, and is used merely for benchmarking purposes.

We first evaluate the proposed preconditioners on a sphere problem, which has an inner dielectric constant of 4.0. The sphere is a widely used geometry in CEM since its analytical solutions are available via Mie-series solutions. Furthermore, since the sphere geometry is trivially reproducible, it is an important benchmarking problem, providing an opportunity for the evaluation of the performance of the proposed preconditioners with respect to other preconditioners. However, with possible high dielectric constants and complex shapes, real-life problems are more important for judging the quality of a preconditioner. Therefore, we also consider three real-life problems: a lens with a dielectric constant of 12.0 [111], a periodic dielectric structure (periodic slabs) with a dielectric constant of 4.8, and a photonic crystal waveguide with a dielectric constant of 11.56 [138].

6.7.1 The Sphere Problem

In Table 6.4, we present solution frequencies, diameters in terms of wavelength, number of MLFMA levels, and number of unknowns relating to the sphere problem. We deliberately solve problems with increasing sizes to make a reasonable judgment about the preconditioner, because near-field matrices become sparser as the number of MLFMA levels increases.

Setup Times

The setup of the Schur complement preconditioners is composed of the construction of $\overline{\mathbf{M}}_{11}$ (SAI of $\overline{\mathbf{A}}_{11}^{NF}$) and $\overline{\mathbf{M}}_S$ (SAI of the approximate Schur complement matrix $\overline{\mathbf{S}}$). In Table 6.5, we compare these setup times with those of ILUTP (ILUT with 0.5 pivoting tolerance) and ILU(0). The setup of 4PPBDP is negligible.

Table 6.5: Setup times (in minutes) of ILU-type preconditioners and SAIs of $\overline{\mathbf{A}}_{11}^{NF}$ and the Schur complement matrix $\overline{\mathbf{S}}$ for the sphere problem.

Problem	ILUTP	ILU(0)	$\overline{\mathbf{M}}_{11}$	$\overline{\mathbf{M}}_S$
S1	0.16	0.02	0.08	0.08
S2	0.56	0.05	0.18	0.19
S3	3.84	0.19	0.68	0.73
S4	20.54	0.77	2.82	3.05
S5	158.21	2.17	5.75	6.23

From Table 6.5, we see that setup times of ILUTP are disproportionately larger than those of the others, particularly for the S5 problem. The time required for the setup of ILU(0) is six to eight times less than that of the Schur complement preconditioners, which require the constructions of both $\overline{\mathbf{M}}_{11}$ and $\overline{\mathbf{M}}_S$. As the following tables will reveal, however, both of these times are insignificant compared to the iterative solution times of the problems. Finally, note that the setup time of $\overline{\mathbf{M}}_S$ is only slightly higher than that of $\overline{\mathbf{M}}_{11}$ because of the efficiently implemented incomplete matrix-matrix multiplication described in Fig. 6.4.

ILU-Type and Simple Preconditioners

For the first-kind integral formulation CTF, similar to the results of [119], we observe that the ILU-type preconditioners have an instability issue. In particular, with ILU(0), the condition estimates [76] turn out to be very high for some

large sphere problems. The same situation also arises for ILUT, but the instability can be removed in this case if pivoting with 0.5 tolerance is applied. Other formulations that are of the second-kind do not exhibit any instability and ILU(0) performs the best among the ILU-type preconditioners for those formulations. Therefore, we employ ILU(0) for formulations other than CTF, and ILUTP for CTF. Our comments on the results of No PC, 4PBDP and ILU-type preconditioners are as follows:

- For all formulations, the no-restart GMRES solves all sphere problems successfully. However, the number of iterations is very high in some instances, such as the CNF solution of S4. Moreover, some large instances of these problems cannot be solved with other non-optimal solvers. For example, the solutions of S5 do not converge with BiCGStab for CNF, MNMF, and JMCFIE.
- In accordance with the findings in [114], we observe that 4PBDP worsens the convergence behavior of CTF. In that paper, it is shown that for other formulations, CGS and BiCGStab solutions of the sphere geometry can significantly be improved with 4PBDP. Nevertheless, 4PBDP is, in general, less effective on the convergence of large problems when GMRES is employed as the iterative solver.
- Considering the solutions with CTF, ILUTP provides a significant improvement over No PC only for the S3 case. Solutions with CNF, on the other hand, significantly benefit from ILU(0). For better-conditioned JMCFIE and MNMF, ILU(0) provides minor improvements over 4PBDP.

Approximate Schur Complement Preconditioners

In Table 6.7, we present iteration counts and total solution times of approximate Schur complement preconditioners. We omit the results related to DASP since

Table 6.6: Performances of the 4PBDP and ILU(0) preconditioners and No PC on the sphere problem.

Problem	CTF						CNF					
	No PC		4PBDP		ILUTP		No PC		4PBDP		ILU(0)	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
S1	179	7	467	18	149	6	67	3	45	2	27	1
S2	167	21	668	85	138	19	140	18	89	11	46	6
S3	471	313	†	–	284	198	171	113	126	83	61	41
S4	291	912	†	–	268	851	968	3,065	516	1,894	161	515
S5	271	2,028	†	–	273	2,198	390	2,916	386	2,880	120	902
	MNMF						JMCFIE					
	No PC		4PBDP		ILU(0)		No PC		4PBDP		ILU(0)	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
S1	47	2	32	1	27	1	79	3	53	2	31	1
S2	71	9	51	7	39	5	93	12	62	8	36	5
S3	112	73	85	56	63	42	139	92	100	67	68	46
S4	192	605	161	504	116	368	223	706	141	444	102	326
S5	187	1,405	165	1,240	108	826	143	1,075	111	836	102	805
Notes: “iter” and “time” denote the number of iterations and total solution time in minutes. Nonconvergence is denoted by a dagger “†”.												

this preconditioner behaves consistently poorer than others do. We first note that per-iteration times of all Schur complement preconditioners are very close to each other. Even though the applications of UTASP and LTASP require three and the application of ASP requires four multiplications with $N \times N$ sparse partitions, the time required for these multiplications is much less than the time required for the far-field computations performed by MLFMA. Furthermore, the complexity of near-field partition is $\mathcal{O}(N)$, whereas MLFMA scales with $\mathcal{O}(N \log N)$. As a result, per-iteration times are dominated by the MLFMA operations and iteration times are in accordance with the iteration counts. For a certain formulation, when we can decide that some of the preconditioners behave worse than the others, we omit them for the largest S5 problem. For example, we omit UTASP and LTASP solutions of S5 for CTF.

Our comments on the results presented in Table 6.8, also compared to those in Table 6.6, are as follows:

Table 6.7: Performances of the Approximate Schur complement preconditioners on the sphere problem.

Problem	CTF						CNF					
	UTASP		LTASP		ASP		UTASP		LTASP		ASP	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
S1	53	2.2	48	2.0	43	1.8	33	1.4	30	1.3	27	1.2
S2	60	7.9	55	7.3	47	6.3	57	7.7	57	7.7	46	6.2
S3	147	98.7	121	81.5	103	69.8	64	43.8	59	40.5	55	38.0
S4	209	664.4	178	566.7	144	459.2	130	416.7	204	651.7	158	506.9
S5	*		*		147	1,109.7	97	747.8	*		97	741.0
	MNMF						JMCFIE					
	UTASP		LTASP		ASP		UTASP		LTASP		ASP	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
S1	45	1.9	33	1.4	29	1.3	33	1.4	35	1.5	29	1.3
S2	66	8.7	43	5.8	42	5.7	40	5.5	40	5.5	34	4.7
S3	123	83.3	67	46.0	70	48.0	63	43.4	76	52.1	55	38.1
S4	235	752.0	122	391.9	132	423.8	93	301.7	124	400.9	84	273.6
S5	*		103	788.7	128	982.3	*		*		77	595.7

Notes: “iter” and “time” denote the number of iterations and total solution time in minutes. An asterisk “*” denotes that the problem is not solved with that particular preconditioner.

- ASP is the best-performing preconditioner among the Schur complement preconditioners except for the S4 solution of CNF and the largest three problems of MNMF; it is possible that the indefiniteness of the matrices causes this [3]. While improving the preconditioner, the eigenvalues with a negative real part move progressively towards the point $(1, 0)$. Meanwhile, however, some eigenvalues may be very close to zero, slowing down the convergence.
- For CTF, ASP reduces solution times of the sphere problems by a factor of two to four, compared to ILUTP and No PC. For CNF, ASP provides a reduction by a factor of three to six with respect to 4PBDP. ILU(0) solves CNF systems as fast as ASP, but for S5, solutions with ASP converge faster. JMCFIE solutions are also obtained about two times faster than ASP than with 4PBDP. ILU(0) is better than 4PBDP for JMCFIE, but it is worse than ASP. Finally, MNMF benefits the least from the Schur complement preconditioners. Nonetheless, for large problems, LTASP provides

an approximate 30% reduction in time compared to 4PBDP. ILU(0) solves MNMF problems as fast as the Schur complement preconditioners do.

- When we compare the formulations considering their performances with ASP, we observe that JMCFIE systems are solved with the lowest and CTF systems are solved with the highest iteration counts. Although the iteration counts of MNMF are much less than those of CNF without a preconditioner, CNF benefits more from preconditioning. As a result, iteration counts of these formulations become close to each other when an ILU-type or a Schur complement preconditioner is employed.

Iterative Schur Complement Preconditioners

We compare non-iterative and iterative versions of Schur complement preconditioners among with a simple preconditioner (No PC for CTF and 4PBDP for other formulations) in Table 6.8. Our comments on the results are as follows:

- Among the iterative Schur complement preconditioners, ISP results in the lowest iteration counts, except for the S4 solution of CNF. However, for this case, the iteration count of UTISP is only slightly less than that of ISP.
- For CTF, ISP solves sphere problems at least three times faster than No PC, and also provides significant reductions (about 40%) in the solution times of ASP. For CNF, solutions are obtained four to five times faster than 4PBDP, and 20% faster than ASP. For JMCFIE, ISP provides a reduction of 50% with respect to 4PBDP, and a reduction of 10% with respect to ASP.
- Compared to those obtained with No PC and 4PBDP, the gap in the solution times between CTF and JMCFIE diminishes with ISP. Hence, it

becomes possible to obtain accurate CTF solutions with similar convergence rates of JMCFIE. We refer to [114] for a detailed comparison of the accuracy of CTF and JMCFIE on sphere problems.

Table 6.8: Performances of the Schur complement preconditioners on the sphere problem.

Prob- lem	CTF									
	No PC		ASP		UTISP		LTISP		ISP	
	iter	time	iter	time	iter	time	iter	time	iter	time
S1	179	7	43	2	41	2	38	2	27	1
S2	167	21	47	6	41	5	38	5	31	4
S3	471	313	103	70	89	60	83	57	69	47
S4	291	912	144	459	133	428	109	351	83	268
S5	271	2,028	147	1110	100	767	97	743	91	696
	CNF									
	4PBDP		ASP		UTISP		LTISP		ISP	
	iter	time	iter	time	iter	time	iter	time	iter	time
S1	45	2	27	1	33	1	28	1	22	1
S2	89	11	46	6	57	8	50	7	36	5
S3	126	83	55	38	60	41	51	35	43	30
S4	516	1,894	158	507	121	393	167	540	125	406
S5	386	2,880	97	741	88	672	98	748	80	612
	JMCFIE									
	4PBDP		ASP		UTISP		LTISP		ISP	
	iter	time	iter	time	iter	time	iter	time	iter	time
S1	53	2	29	1	30	1	33	1	27	1
S2	62	8	34	5	37	5	38	5	32	4
S3	100	67	55	38	55	38	71	49	51	35
S4	141	444	84	274	78	255	117	380	75	245
S5	111	836	77	596	71	546	98	751	67	515
Notes: “iter” and “time” denote number of iterations and total solution time in minutes.										

The maximum number of inner iterations is set to three for the iterative Schur complement preconditioners in Table 6.4. We note, however, that only the S4 solution of CNF is sensitive to the stopping criteria of inner solutions. For that problem, when we decrease the inner tolerance to 10^{-3} , ISP yields the best result and solves this problem in 85 iterations and 371 minutes. Other problems produce similar results with 10^{-3} inner tolerance. Hence, unless a very tight stopping criterion is selected, inner tolerance is not very important for the iterative Schur complement preconditioners.

Next, we compare the iteration counts of ASP and ISP with a simple preconditioner (No PC for CTF and 4PBDP for other formulations) and the exact solution of the near-field system (NF-LU) in Fig. 6.12. ASP and ISP significantly reduce the iteration counts compared to a simple preconditioner. However, the proposed preconditioning scheme, ISP, further accelerates the convergence rates and leads to iteration counts that are very close to those of NF-LU. This reveals that ISP uses almost all of the information provided by the near-field matrix efficiently for preconditioning.

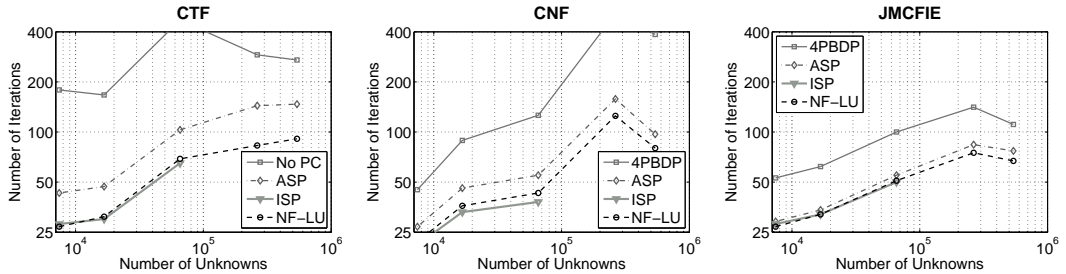


Figure 6.12: Comparisons of iteration counts for the sphere problem.

Memory Comparisons

Finally, we compare the preconditioned solutions in terms of their memory use. In Table 6.9, we present the memory requirements of the considered preconditioners, along with the memory requirement of MLFMA and no-restart GMRES. As mentioned in [114, 64], MLFMA requires more memory for CNF and JMCFIE than for CTF. Here, the difference in memory use becomes remarkable (1.1 GB) for the largest problem (S5). The memory requirement of both 4PBDP and ISP is modest, and even that of ISP is less than 10% of that of MLFMA. On the other hand, the memory consumed by GMRES is significant compared to the memory use of preconditioners. Note that ISP yields fewer iterations than ASP, but its memory use is higher due to the doubled memory requirement of FGMRES. However, ISP consumes less total memory than ASP does due to storing just one SAI (of $\overline{\mathbf{A}}_{11}^{NF}$) for preconditioning. When we compare preconditioned solutions

of CTF and JMCFIE in terms of total memory requirements, we observe that CTF uses significantly less memory than CNF or JMCFIE does.

Table 6.9: Memory requirements of the Schur complement preconditioners, MLFMA, and solutions with the no-restart GMRES for the sphere problems.

Problem	CTF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		No PC	ISP	No PC	ASP	ISP	No PC	ASP	ISP
S1	50	–	6	10	2	3	60	64	59
S2	117	–	13	21	6	8	138	148	138
S3	470	–	49	236	52	69	706	619	588
S4	1,931	–	197	586	290	334	2,517	2,615	2,462
S5	4,129	–	405	1,117	606	750	5,246	5,544	5,284
	CNF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
S1	66	2	6	3	2	2	71	79	74
S2	152	5	13	11	6	9	168	183	174
S3	607	18	49	63	28	43	688	732	699
S4	2,483	72	197	1,039	318	504	3,595	3,195	3,183
S5	5,279	149	405	1,592	400	660	7,020	6,488	6,343
	JMCFIE								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
S1	66	2	6	3	2	3	71	79	75
S2	152	5	13	8	4	8	165	182	173
S3	607	18	49	50	28	51	675	732	707
S4	2,483	72	197	284	169	302	2,839	3,046	2,982
S5	5,279	149	405	458	317	553	5,886	6,406	6,236
	MNMF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
S1	71	2	11	2	2	3	75	84	85
S2	164	5	25	7	5	9	175	195	199
S3	656	18	97	43	35	62	716	788	816
S4	2,680	72	394	324	266	443	3,077	3,340	3,517
S5	5,680	149	809	680	528	915	6,509	7,017	7,405


Notes: All values are in MB. “PC Memory” stands for the memory requirement of the preconditioner. For MNMF, the memory requirements of ASP and ISP are the same. For others, the memory requirement of ASP is twice that of ISP.

6.7.2 The Lens Problem

For radiometric remote sensing applications, delicate simulations of dielectric lenses are required for a wide spectrum, beginning from 30 GHz [111]. This application gives rise to large problems that are difficult to solve without preconditioning. In this section, we analyze preconditioned iterative solutions of this

important problem. We increase the frequency by 30 GHz intervals, up to 120 GHz. The resulting problems are listed in Table 6.10. The lens problem involves a dielectric half sphere with a high dielectric constant of 12.0.

Table 6.10: Salient features of the lens problems investigated in this study.

Problem		Frequency (GHz)	Size (λ)	MLFMA Levels	Number of Unknowns
	L1	30	2.5	6	38,466
	L2	60	5.0	7	158,286
	L3	90	7.5	7	353,646
	L4	120	10.0	8	632,172

Note: λ denotes the wavelength at the frequency of operation.

ILU-Type and Simple Preconditioners

CTF solutions of lens problems do not suffer from the instability of ILU-type preconditioners. ILU(0) performs better than ILUT and ILUTP for all formulations. Hence, for all formulations, we compare ILU(0) with No PC and 4PBDDP in Table 6.11. CNF solutions of L4 with No PC and 4PBDDP cannot be completed since the memory requirement cannot be met with the available memory after 500 GMRES iterations. Our comments on the results are as follows:

- We observe that CNF cannot solve L2, L3, and L4 problems without a preconditioner. CTF and JMCNMF converge with similar rates and MNMF converges the fastest. These results are in accordance with the discussion in Section 6.2.5. A high dielectric constant degrades the conditioning of normal formulations [12]. In addition, the spectra illustrated in Figs. 6.5, 6.6, and 6.7 reveal that CNF is negatively affected more than the others by an increase in the dielectric constant. As a combination of CTF and CNF, JMCNMF is also adversely affected by a high dielectric constant. Consequently, its iteration counts turn out to be close to those of CTF for the lens problem.

Table 6.11: Performances of the 4PBDP and ILU(0) preconditioners and No PC on the lens problems.

Problem	CTF						CNF					
	No PC		4PBDP		ILU(0)		No PC		4PBDP		ILU(0)	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
L1	205	162	*		105	61	368	292	140	102	44	26
L2	278	939	*		152	442	†	–	333	1,115	87	257
L3	276	1,853	*		227	1,525	†	–	406	2,734	87	589
L4	321	4,458	*		229	3,165	MLE		MLE		117	1,627
	MNMF						JMCFIE					
	No PC		4PBDP		ILU(0)		No PC		4PBDP		ILU(0)	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
L1	78	51	52	34	32	19	138	110	77	57	40	23
L2	114	386	86	288	43	124	227	786	114	391	67	198
L3	146	970	131	871	48	328	276	1,850	128	860	71	501
L4	166	2,282	166	2,284	52	720	310	4,310	135	1,872	88	1,224

Notes: “iter” and “time” denote the number of iterations and total solution time in minutes. An asterisk “*” denotes that the problem is not solved with that particular preconditioner. A dagger “†” denotes nonconvergence. “MLE” denotes that memory limitation is exceeded.

- JMCFIE benefits more from 4PBDP than MNMF, and iteration counts for these formulations become close to each other with 4PBDP. For the largest problem L4, JMCFIE converges even faster than MNMF. Superiority of JMCFIE over MNMF for large problems has also been demonstrated in [114].
- ILU(0) performs significantly better than 4PBDP on the lens problem. All of the formulations can be solved faster with ILU(0), but second-kind formulations are accelerated more than CTF since they have more diagonally dominant matrices than CTF does.

Approximate Schur Complement Preconditioners

In Table 6.12, we present solutions of the lens problems with the Schur complement preconditioners. Solutions of L1 and L2 show that ASP performs significantly better than other Schur complement preconditioners, hence, we perform

Table 6.12: Performances of the Approximate Schur complement preconditioners on the lens problems.

Problem	CTF						CNF					
	UTASP		LTASP		ASP		UTASP		LTASP		ASP	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
L1	129	75	93	55	57	34	101	59	70	41	45	27
L2	189	550	135	394	85	249	227	655	160	461	92	266
L3	*		*		99	669	*		*		94	635
L4	*		*		114	1,592	*		*		128	1,785
	MNMF						JMCFIE					
	UTASP		LTASP		ASP		UTASP		LTASP		ASP	
	iter	time	iter	time	iter	time	iter	time	iter	time	iter	time
L1	90	53	41	25	35	21	56	34	48	29	31	19
L2	133	386	54	159	48	142	100	292	80	235	52	154
L3	*		*		54	369	*		*		54	368
L4	*		*		60	846	*		*		64	901

Notes: “iter” and “time” denote the number of iterations and total solution time in minutes. An asterisk “*” denotes that the problem is not solved with that particular preconditioner.

solutions of larger L3 and L4 problems only with ASP. We summarize our comments on the results as follows:

- With ASP, all of the formulations can be solved much faster than with No PC or 4PBDP, and solution times are reduced two-fold to five-fold, depending on the type of formulation. The number of iterations for JMCFIE and MNMF are close to each other, and are approximately half of the number of iterations for CTF and CNF.
- For CTF and JMCFIE, ASP performs significantly better than ILU(0). But for CNF and MNMF, ILU(0) performs slightly better than ASP.

In Table 6.13, we present solutions of the first two lens problems obtained with NF-LU. When we compare these iteration counts with the ones in Table 6.12, we observe that iteration counts obtained with ASP are already close to those of NF-LU, except CNF. Therefore, we need not to use iterative Schur complement preconditioners for this problem.

Table 6.13: Number of iterations obtained with NF-LU for the lens problems.

Problem	CTF	CNF	MNMF	JMCFIE
L1	58	32	39	29
L2	90	62	52	49

Memory Comparisons

We compare the memory consumptions of preconditioned solutions in Table 6.14. In addition to providing more accurate results, with CTF and JMCFIE problems are solved using less memory. For MLFMA setup, CTF requires less memory than JMCFIE does, but this advantage disappears considering the total memory because of the higher memory use of GMRES for CTF solutions.

6.7.3 Periodic Slabs (PS)

Periodic dielectric slabs can be used as filters in microwave circuits and antenna systems [139]. As shown in Fig. 6.13, this structure is transparent to electromagnetic waves at 250 MHz and 350 MHz, whereas at 300 MHz, the structure becomes opaque and a shadowing occurs. The filtering capability of the device increases when the wall sizes or the number of walls increase. We analyze this problem at its resonance frequency, i.e., 300 MHz, where it becomes transparent to electromagnetic fields. The inner dielectric constant of the device is 4.8. In Table 6.15, we present information about the cases investigated in this study.

For this problem, compared to CTF and JMCFIE, it is much more difficult to obtain convergence with CNF and MNMF, hence, we omit those solutions. For Schur complement preconditioners, we include solutions obtained with ASP or ISP, which perform much better than other Schur complement preconditioners for all cases.

Table 6.14: Memory requirements of the Schur complement preconditioners, MLFMA, and solutions with the no-restart GMRES for the lens problems.

Problem	CTF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		No PC	ISP	No PC	ASP	ISP	No PC	ASP	ISP
L1	50	13	33	60	17	35	110	132	118
L2	117	52	133	336	103	210	453	485	460
L3	470	113	289	745	267	545	1,215	1,315	1,304
L4	1,931	201	517	1,548	550	1,109	3,479	3,514	3,557
	CNF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
L1	66	13	33	41	13	52	120	145	151
L2	152	52	133	402	111	432	606	528	717
L3	607	113	289	1,095	254	1,209	1,815	1,439	2,105
L4	2,483	201	517	3,376	617	3,222	6,060	4,133	6,221
	JMCFIE								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
L1	66	13	33	23	9	18	101	141	117
L2	152	52	133	138	63	121	341	480	405
L3	607	113	289	345	146	275	1,065	1,331	1,171
L4	2,483	201	517	651	309	550	3,335	3,825	3,549
	MNMF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
L1	71	13	65	15	10	21	99	147	158
L2	164	52	265	104	58	121	320	487	550
L3	656	113	578	353	146	302	1,122	1,380	1,536
L4	2,680	201	1,033	801	289	608	3,681	4,002	4,321

Notes: All values are in MB. "PC Memory" stands for the memory requirement of the preconditioner. For MNMF, the memory requirements of ASP and ISP are the same. For others, the memory requirement of ASP is twice that of ISP.

ILU-Type and Simple Preconditioners

The iteration counts and solution times obtained with 4PBDP and ILU-type preconditioners are presented in Table 6.16. For CTF, No PC fails to converge in 1,000 iterations, even though we use the robust GMRES solver. ILU(0) results in modest iteration counts for small problems, but it causes false convergence in some instances, such as the CTF solution of PS4. For this problem, the *condest* value, which provides an estimate of the condition number of the incomplete factors, turns out to be $1.2 \cdot 10^6$, indicating the instability of the incomplete factors.

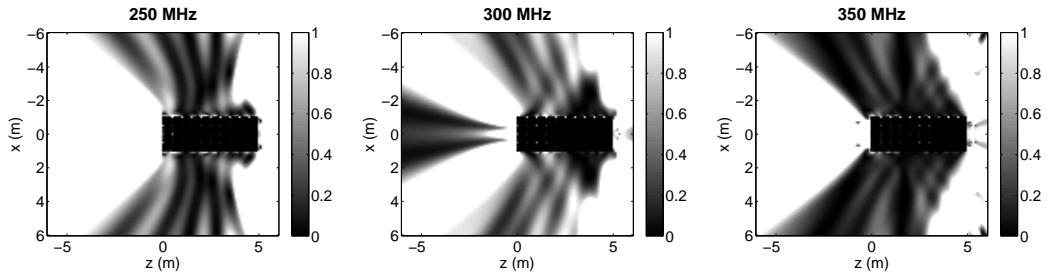
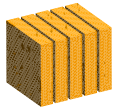


Figure 6.13: Illustration of the filtering capability of the periodic slabs problem. At 250 MHz and 350 MHz, the power transmission is unity in the transmission region on the left-hand side of the structure. On the other hand, a shadowing occurs at 300 MHz and the device becomes opaque.

Table 6.15: Salient features of the periodic-slab problems investigated in this study.

Problem		Frequency (MHz)	Slab Size (m)	Number of Walls	MLFMA Levels	Number of Unknowns
	PS1	300	$0.41 \times 2 \times 2$	5	5	38,700
	PS2		$0.41 \times 2 \times 2$	10	6	77,400
	PS3		$0.41 \times 4 \times 4$	5	6	131,460
	PS4		$0.41 \times 4 \times 4$	10	6	262,920

Even though ILUTP produces stable factors, this preconditioner also fails to provide convergence for large problems. JMCFFIE solutions can be obtained faster than CTF solutions. On the other hand, the number of iterations of 4PBDFP for the largest problem PS4 is quite large. For JMCFFIE, ILU(0) is preferred over ILUTP because of its better performance, negligible setup time, and lower memory requirement due to in-place factorization [2]. On the other hand, the memory requirement of ILU(0) is still significant; it consumes 2.15 GB of memory for PS4 problem in a single-precision implementation.

Schur Complement Preconditioners

In Table 6.17, we compare the three preconditioners obtained from Schur complement reduction. In ISP1(5), we use $\overline{\mathbf{M}}_{11}$ as a preconditioner for the solution of both reduced systems, and set the maximum number of inner iterations to five. In ISP2, we construct $\overline{\mathbf{M}}_S$ for the inner solution of the system involving $\overline{\mathbf{S}}$, in

Table 6.16: Comparison of ILU and simple preconditioners for the periodic slab problems.

CTF	No PC		ILU(0)		ILUTP		
	iter	time	iter	time	setup	iter	time
PS1	473	1.6	186	0.6	0.8	409	1.4
PS2	†	–	335	2.7	2.7	702	5.7
PS3	†	–	353	5.1	11.2	†	–
PS4	†	–	237*	7.0	–		

JMCFIE	4PBDP		ILU(0)		ILUTP		
	iter	time	iter	time	setup	iter	time
PS1	149	0.5	87	0.3	0.8	124	0.4
PS2	107	0.9	175	1.4	2.6	205	1.7
PS3	276	4.0	197	2.9	-	-	-
PS4	698	20.7	408	12.2	–		

Notes: “iter” denotes number of iterations. “setup” and “time” denotes setup and solution times, in hours. A dagger “†” denotes nonconvergence and * denotes false convergence. 4PBDP and ILU(0) have negligible setup times.

addition to $\overline{\mathbf{M}}_{11}$ for the inner solution of the system involving $\overline{\mathbf{Z}}_{11}^{NF}$. Therefore, the memory use and setup time of ISP2 is twice as that of ISP1. The setup times of $\overline{\mathbf{M}}_{11}$ and $\overline{\mathbf{M}}_S$ seem to be identical, because the time spent during incomplete matrix-matrix multiplications is negligible. For CTF solutions with ISP2, we set the maximum number of iterations to three, since increasing this parameter beyond three does not decrease the iteration counts, but results in higher solution times due to increased application cost.

Our comments on the results are as follows:

- Compared to the sphere, which has a similar dielectric constant, convergence rates of both CTF and JMCFIE decrease remarkably for this problem. In particular, CTF solutions for PS2, PS3, and PS4 cannot be obtained without an effective preconditioner. Note that CTF solutions of PS4 converges only with ISP1 or ISP2.
- When we compare JMCFIE solutions in Table 6.16 and Table 6.17, we found that for the first three problems ISP1 results in significantly smaller

Table 6.17: Comparison of the Schur complement preconditioners for the PS problems.

CTF	$\overline{S}^{-1} \approx \overline{M}_{11}$			$\overline{S}^{-1} \approx \overline{M}_S$					NF-LU
	\overline{M}_{11}	ISP1(5)		\overline{M}_S	ASP		ISP2(3)		
	setup	iter	time	setup	iter	time	iter	time	iter
PS1	0.1	107	0.4	0.1	104	0.4	109	0.4	157
PS2	0.2	210	2.0	0.2	216	1.8	206	1.9	358
PS3	0.3	358	6.0	0.3	490	7.1	386	6.1	MLE
PS4	0.7	933	31.9	0.7	†	–	964	32.2	MLE

JMCFIE	\overline{M}_{11}	ISP1(5)		\overline{M}_S	ASP		ISP2(5)		NF-LU
	setup	iter	time	setup	iter	time	iter	time	iter
PS1	0.1	53	0.2	0.1	106	0.4	54	0.2	43
PS2	0.2	47	0.5	0.2	207	1.7	45	0.5	80
PS3	0.3	112	1.9	0.3	233	3.4	116	1.9	MLE
PS4	0.7	388	13.4	0.7	558	16.8	344	11.3	MLE

Notes: “iter” denotes number of iterations. “setup” and “time” denotes setup and solution times, in hours. A dagger “†” denotes nonconvergence. ‘MLE’ denotes that memory limitation is exceeded.

solution times than ILU(0) does. When we consider the solution times including the setup of the preconditioner, ILU(0) produces smallest solution times for the largest PS4 problem, but the improvement is minor compared to ISP1 or ISP2.

- For CTF, ISP and ASP yield similar results for PS1 and PS2 problems. For the largest problem (PS3), however, ISP solves the problem much faster than ASP does. On the other hand, the memory requirement of ILU(0) is around 1.5 GB more than ISP1 and 1 GB more than ISP2.
- The benefit of inner solves is more evident for JMCFIE solutions. For PS1, PS3, and PS4, ASP fails to provide a significant improvement compared to 4PBDP, and even slows down the convergence for PS2. However, compared to 4PBDP, ISP2 reduces total solution times by 50%.

- Interestingly, ISP leads to fewer iteration counts for PS solutions compared to NF-LU. As explained in [3], this can be related to the indefiniteness of the dense coefficient matrices.
- Due to the difficulty of this problem and its low dielectric constant, JMC-FIE solves the matrix systems much faster compared to CTF. However, it is still possible to obtain CTF solutions.

Memory Comparisons

Information about memory consumption is presented in Table 6.18. We first note that the memory use of ASP with respect to MLFMA is larger compared to the sphere because of the dense structure of this problem. For instance, memory requirement of ISP is about 20% of that of MLFMA for the CTF solution of PS3. The high iteration counts also cause a significant increase in the memory cost of GMRES for CTF. ASP and ISP have similar total memory costs, because the extra memory cost of ISP due to the use of FGMRES has been compensated for by a decrease in the memory of the preconditioner. For this problem, there is a significant gap between the iteration counts of CTF and JMC-FIE. Hence, the GMRES memory cost is much smaller for JMC-FIE, for which the solution of PS3 can be obtained with 10% less memory compared to CTF, even though JMC-FIE uses significantly more memory for MLFMA.

6.7.4 The Perforated-Waveguide Problem

We conclude this section with a comparative investigation of the performance of Schur complement preconditioners on a complicated structure, namely, a photonic crystal waveguide, which is composed of a dielectric slab etched with a waveguiding pattern of holes [138]. An example of the problem and its near-field pattern are shown in Fig. 6.7.4. We increase the problem size by enlarging the

Table 6.18: Memory requirements of the Schur complement preconditioners, MLFMA, and solutions with the no-restart GMRES for the periodic-slab problems.

Problem	CTF								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		No PC	ISP	No PC	ASP	ISP	No PC	ASP	ISP
PS1	386	–	75	140	31	63	526	566	524
PS2	813	–	161	–	128	248	–	1,263	1,222
PS3	1,319	–	260	–	491	718	–	2,330	2,297
PS4	2,672	–	538	–	–	3,743	–	–	6,953
	JMCFIE								
	ML-FMA	PC Memory		GMRES Memory			Total Memory		
		4PBDP	ISP	4PBDP	ASP	ISP	4PBDP	ASP	ISP
PS1	471	16	75	44	31	31	531	651	577
PS2	984	32	161	63	122	56	1,079	1,428	1,201
PS3	1,608	50	260	277	234	225	1,935	2,361	2,092
PS4	3,249	99	538	1,400	1,119	1,557	4,749	5,444	5,343

Notes: All values are in MB. “PC Memory” stands for the memory requirement of the preconditioner. Note that the memory requirement of ASP is twice that of ISP.

size of the structure and including more holes, as shown in Table 6.19. We investigate PW1 and PW2 at 8.3 GHz, and PW3 and PW4 at 7.6 GHz, at the frequencies for the most efficient transmission. Diameters of the holes are on the order of 0.1λ , hence, this problem requires a fine meshing of about 0.05λ in order to model these small details. As a result, the lowest-level clusters of MLFMA contain more basis functions and the resulting near-field matrices become denser compared to previous problems.

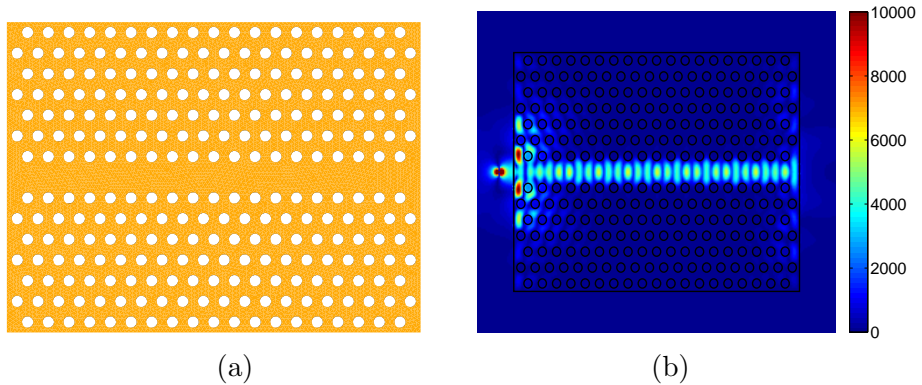
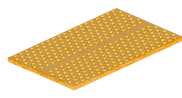


Figure 6.14: (a) A perforated photonic crystal waveguide. (b) Near-zone magnetic fields of the problem when illuminated by a Hertzian dipole.

Table 6.19: Salient features of the perforated waveguide (PW) investigated in this study.

Problem		Frequency (GHz)	Number of Holes	MLFMA Levels	Number of Unknowns
	PW1	7.6	$0.6 \times 5 \times 10$	38	27,798
	PW2		$0.6 \times 15 \times 20$	272	162,420
	PW3	8.3	$0.6 \times 26 \times 34$	828	475,782
	PW4		$0.6 \times 29 \times 38$	1,042	597,462

We first list the solutions of PW1 and PW2 problems in Table 6.20. Because of the high contrast of the device, we use $\overline{\mathbf{M}}_S$ as an approximate inverse (ASP) or as an inner preconditioner (ISP2) for $\overline{\mathbf{S}}$. Even both CTF and JMCFIE solutions converge with ISP1, number of iterations and solution times are significantly higher than those of ASP or ISP2. The CTF solution of PW2 do not converge with No PC, and the JMCFIE solution requires around 600 iterations with 4PBDP. The ineffectiveness of the 4PBDP on JMCFIE is related to a lack of diagonal dominance for high contrasts [12]. Similar to the periodic-slabs problems, ILU(0) performs better than ILUTP for CTF and JMCFIE. On the other hand, in terms of solution times, ILU(0) performs poorer than Schur complement preconditioners for PW1, and does not fit in the 16GB memory for PW2 because of the $\lambda/20$ mesh size and denser near-field matrices. We present the solutions with ASP for CTF and with ISP2(5) for JMCFIE, which lead to smallest solution times. Note that the solution times of PW1 and PW2 with CTF and JMCFIE are close to each other, since the per iteration times with ASP are significantly smaller than per iteration times with ISP2.

The solutions of the larger problems PW3 and PW4 have been performed using another server with 32 GB memory. We solved the problems with only Schur complement preconditioners, because of the excessive memory requirement of ILU(0) for these problems. The results are presented in Table 6.21. We note that obtaining CTF solutions especially for PW4 is challenging. ISP1 do not converge, and the memory requirement of FGMRES for ISP2 exceeds the available 32GB memory after 500th iteration. We are able to obtain CTF solutions

Table 6.20: Comparison of the preconditioners for the PW1 and PW2 problems.

CTF	No PC		ILU(0)			ASP		
	iter	time	setup	iter	time	setup	iter	time
PW1	695	43	5	141	10	12	58	4
PW2	†	–	MLE	–	–	110	217	104
JMCFIE	4PBDP		ILU(0)			ISP2(5)		
	iter	time	setup	iter	time	setup	iter	time
PW1	183	12	5	32	5	12	27	4
PW2	593	255	MLE	–	–	110	79	78

Notes: “iter” denotes number of iterations. “time” denotes total solution time in hours. A dagger “†” denotes nonconvergence. “MLE” denotes that memory limitation is exceeded.

Table 6.21: Comparison of the Schur complement preconditioners for PW3 and PW4 problems.

CTF			$\overline{S}^{-1} \approx \overline{M}_S$			
	\overline{M}_{11}	\overline{M}_S	ASP		ISP2(3)	
	setup	setup	iter	time	iter	time
PW3	4.4	4.6	697	28	587	37
PW4	5.7	6.1	829	110	MLE	–
JMCFIE			$\overline{S}^{-1} \approx \overline{M}_{11}$		$\overline{S}^{-1} \approx \overline{M}_S$	
	\overline{M}_{11}	\overline{M}_S	ISP1(5)		ISP2(5)	
	setup	setup	iter	time	iter	time
PW3	4	5	274	21	110	15
PW4	6	6	301	28	139	22

Notes: “iter” denotes number of iterations. “setup” and “time” denotes setup and total solution times in hours. “MLE” denotes that memory limitation is exceeded.

of PW4 only with ASP, which requires 829 GMRES iterations to converge. The JMC-FIE solution of the same problem, on the other hand, has been obtained in merely 301 iterations with ISP1 and in 139 iterations with ISP2.

6.7.5 Accuracy of the Solutions of the Photonic Crystal Waveguide

In Fig. 6.15, we present near-zone magnetic fields for a perforated waveguide problem (PW3 in Table 6.19). The total magnetic field is calculated point-wise inside and outside the structure in order to demonstrate the transmission of electromagnetic waves from the left-hand side to the bottom. For this problem, we observe that results obtained by using CTF and JMC-FIE are significantly different. This is due to the deteriorating accuracy of JMC-FIE in the case of complicated structures and relatively high contrasts.

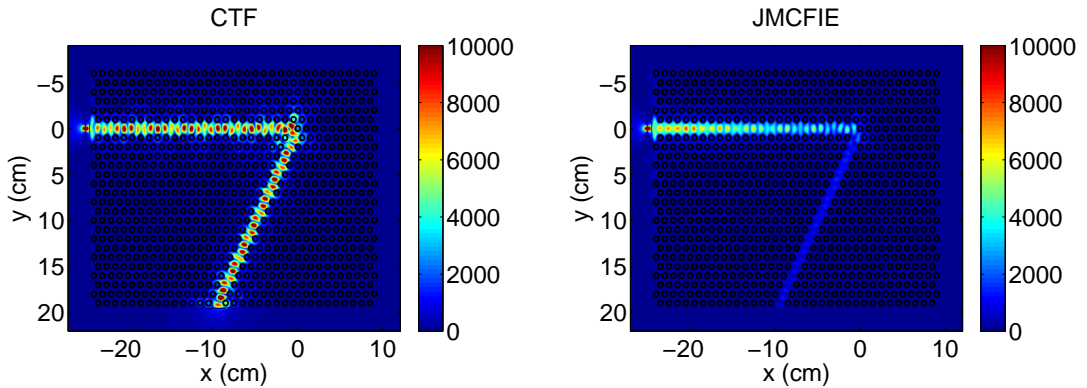


Figure 6.15: Near-zone magnetic fields for a perforated waveguide (PW3 in Table 6.19) illuminated by a Hertzian dipole.

Finally, in order to show that the inconsistency between CTF and JMC-FIE results is due to the inaccuracy of JMC-FIE, we consider the solution of an electromagnetics problem involving a $0.6 \text{ cm} \times 7 \text{ cm} \times 10 \text{ cm}$ perforated PhC waveguide. The problem is formulated with CTF and JMC-FIE discretized by using $\lambda/20$ and $\lambda/40$ triangles. Fig. 6.16 presents the magnetic field at 8.25 GHz.

We observe that results obtained by JMCFIE change drastically when the discretization is refined. Specifically, JMCFIE results become consistent with CTF results for the dense discretization.

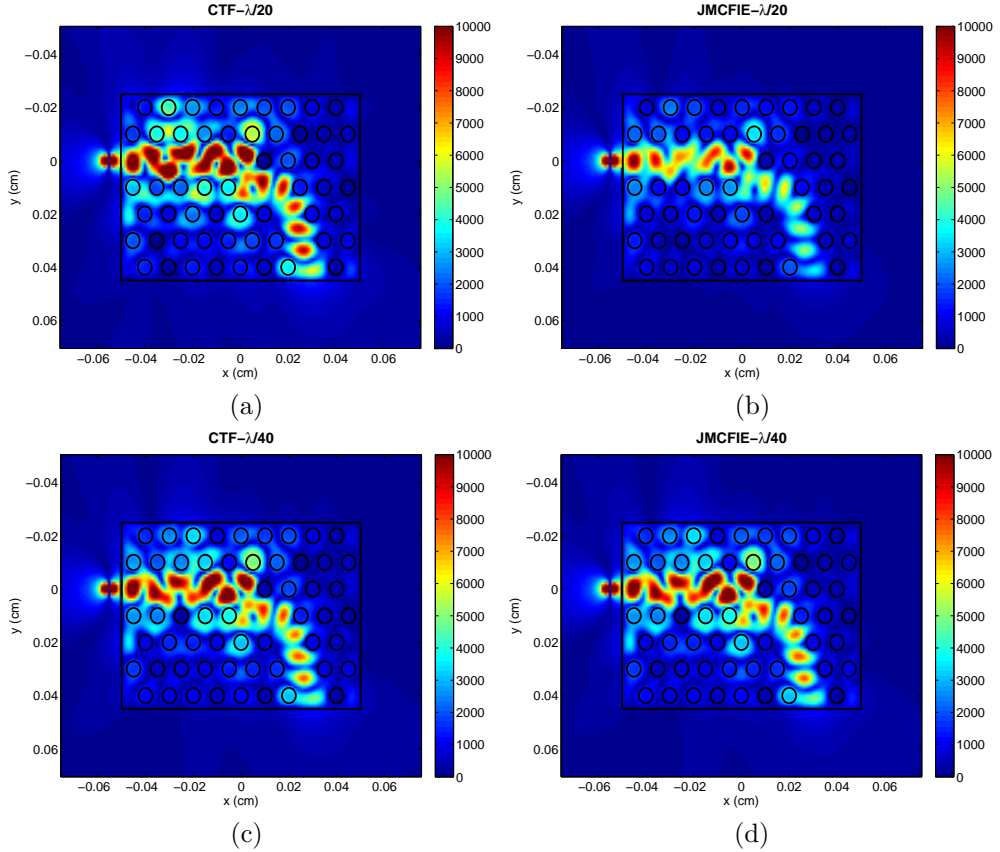


Figure 6.16: Near-zone magnetic fields for a perforated PhC waveguide involving 7×10 holes illuminated by a Hertzian dipole. Solutions are obtained with (a) CTF and $\lambda/20$ triangulation, (b) JMCFIE and $\lambda/20$ triangulation, (c) CTF and $\lambda/40$ triangulation, and (d) JMCFIE and $\lambda/40$ triangulation.

6.8 Conclusions

In the context of surface integral-equation methods for dielectric problems, simultaneous discretization of the surface currents and integral equations leads to matrix equations with 2×2 partitions. These partitions show some resemblance to the matrices that are obtained in PEC problems. Based on our prior

experience with the preconditioning of PEC problems, we have developed robust Schur complement preconditioners for dielectric problems using the 2×2 partitioned structure of matrices. Inspired by its success in PEC problems [44], the SAI preconditioner is applied to the (1,1) partition. For the Schur complement, we discuss several approximation strategies and show that obtaining an approximation via sparse matrix-matrix multiplications yields the best results.

Using those approximate inverses, we propose both approximate (direct) and iterative versions of Schur complement preconditioners. The direct approach requires no inner solves, hence the linear systems can be solved with non-optimal solvers or regular GMRES, which requires half of the GMRES memory. For iterative Schur complement preconditioners, we compute solutions of the systems involving the (1,1) partition and the Schur complement by SAI-preconditioned iterative solvers. It has been shown that [120] for Schur complement preconditioners, similar approximation levels should be targeted for the solutions of these two systems since it would be wasteful to solve one system significantly more accurate than the other one. With the iterative preconditioning scheme, the requirement in [120] can be better satisfied. Hence, it has been possible to obtain effective preconditioners for accelerated iterative solutions of all dielectric problems considered here, even those problems are formulated with the accurate but difficult-to-solve first-kind integral equation CTF. Moreover, we have shown that for a low dielectric constant (e.g., $\epsilon_r \leq 4$), the same approximate inverse of the (1,1) partition can be used as an effective preconditioner for both the (1,1) partition and the Schur complement, reducing the setup and memory cost of the preconditioner. As a result, sphere and periodic-slab problems have been solved faster than with ASP, with the memory cost reduced by one half.

To the best of our knowledge, the following conclusions drawn from the numerical experiments are novel and have the potential to change the common

wisdom regarding the solutions of surface integral equations for dielectric problems:

- The no-restart GMRES solver is much more robust and efficient for preconditioned and unpreconditioned matrix systems than other non-optimal solvers.
- When high accuracy is a concern, CTF solutions can be obtained without difficulty by using the Schur complement preconditioners. The lack of diagonal dominance in CTF prevents the success of block-diagonal-type (i.e., 4PBDDP [114]) or ILU(0) preconditioners. Although they are known as the most general and effective preconditioners for non-symmetric and indefinite systems [2], ILUT and ILUTP also have discouraging performances on CTF.
- Normal formulations and JMCIE are second-kind integral equations that are expected to yield well-conditioned linear systems. Particularly for large problem sizes, however, effective preconditioning becomes indispensable for these formulations when the problem involves a high dielectric constant.
- Furthermore, the photonic crystal problem shows that the complexity of the geometry and the high dielectric constant may render linear systems obtained from normal formulations unsolvable even with effective preconditioners. Linear systems obtained from JMCIE can be solved with simple preconditioners, but they require many iterations. When ASP or ISP is used, on the other hand, even accurate CTF solutions can be attained with modest iteration counts.

Chapter 7

Conclusions and Future Work

Nothing will be more central to computational science in the next century than the art of transforming a problem that appears intractable into another whose solution can be approximated rapidly. For Krylov subspace matrix iterations, this is preconditioning.

L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*. SIAM Publications, 1997.

In this dissertation, we have explained our efforts in designing effective preconditioners to provide the robustness of direct methods to CEM problems that employ MLFMA. The importance of preconditioning on surface-integral-equation methods is two folds:

- First-kind integral equations, such as EFIE and CTF, results in the most accurate results for PEC and dielectric problems, respectively. The convergence of resulting matrices, however, can only be guaranteed through effective preconditioners. It has been shown that for second-kind and better-conditioned formulations, such as CFIE and JMCIE, one should use basis functions that are higher order than RWG is or use much denser meshes to achieve a similar accuracy [12, 18].

- Even though CFIE and JMCIE yield better-conditioned linear systems for PEC and dielectric problems, there are many effects that cause an increase in iteration counts. In that cases, solution times can be significantly lowered using preconditioners.

We summarize our conclusions related to PEC problems in Fig. 7.1. If a sequential solution is targeted, ILU-class preconditioners resulted in close to optimal results. Note that tested implementations of ILU-class preconditioners can be found in many solver packages, such as PETSc [74] and ILUPACK [78]. We remind that the *condest* value, which presents valuable information about the stability of the ILU factors, can be computed before the iterations begin. Depending on this value, ILUT or ILUTP can be used safely for EFIE matrices. For CFIE, however, we suggest the use of ILU(0) since it has a lower memory and setup cost.

When the size of the problem grows and solutions cannot be obtained using a single processor, there is a need for parallel preconditioners. We have shown that SAI and INF preconditioners can be safely used up to millions of unknowns. However, for larger problems, we suggest to use the AMLFMA preconditioner, which is a more effective approach that uses a dense matrix for preconditioning.

For dielectric problems, which give rise to partitioned matrices, general-purpose algebraic preconditioners are known to be ineffective. This is also valid for CEM problems, particularly for those formulated with CTF. Even though ILU(0) seems to work with JMCIE, it has a very high memory requirement. We have shown that it is possible to obtain stronger preconditioners with reduced memory requirement using Schur complement preconditioners that exploit partitioned structure of the near-field matrix. Using these preconditioners, we have been able to solve a perforated photonic crystal problem for both CTF and JMCIE formulations. However, the comparisons of the near-field patterns of the

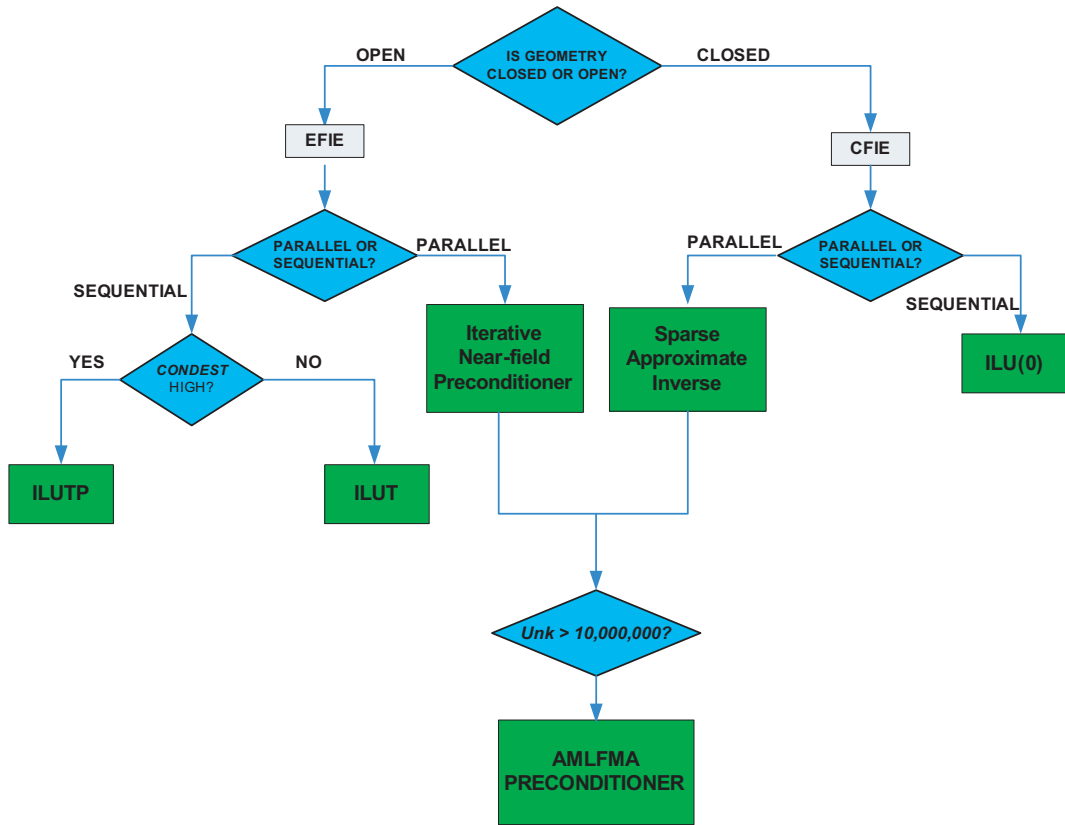


Figure 7.1: Decision chart for the selection of preconditioners for PEC problems.

device revealed that the accuracy of JMCFFIE is much worse than that of CTF, and the results obtained with the JMCFFIE can be severely misleading [67].

Iterative solvers and preconditioning are diverse areas that experience daily improvements in scientific computing. Hence, there is a bunch of future work that deserves attention. We list some of those that we can foresee:

Adaptive preconditioning for integral-equation problems: The GMRES solver produces valuable information during the iterations, and previous researchers used this information in multi-right-hand-side solutions [140, 141]. In AMLFMA preconditioner that makes use of an inner-outer solution scheme, this information can be extracted in the first inner solution. Then, it is possible to improve existing SAI preconditioner and improve convergence of subsequent inner solutions. In this way, the total cost of inner iterations can be substantially alleviated.

Schur complement preconditioners for cavity resonances: Physical cavity resonances originate very small eigenvalues, which in turn increase the iteration counts tremendously at resonant frequencies. When the coefficient matrix is reordered so that the cavity interactions form a block of the matrix, our preliminary results show that those small eigenvalues are associated with the cavity block, which has a much smaller size compared to the whole matrix size. Using the advantage of Schur complement preconditioning, it is possible to construct a very powerful local preconditioner for this cavity block with modest computational requirements. This example shows the benefit of using physical information from the problem for preconditioning.

Hierarchical matrices: Another topic of interest is the emerging hierarchical-matrix techniques [142]. These methods have two important advantages: First, they allow some matrix operations that cannot be performed with MLFMA, such as matrix-matrix addition and multiplication, almost in linear complexity. This property can be used to approximate the inverse of dense matrices with data-sparse matrices, which can be used as a fast solver or a very effective preconditioner. The second advantage of hierarchical matrices is that those methods are kernel free; hence, they can be used to develop fast solvers for layered or non-homogenous media. These methods can also be used for broadband MLFMA, for which the near-field matrix becomes even sparser than the regular MLFMA. The adaption of hierarchical matrices to CEM problems is an open area.

Symmetric CG-like solvers for EFIE: It is possible to use a slightly modified form of the CG method for symmetric-complex matrices, such as those obtained from EFIE. Even though there is a risk for breakdown in this case [143, 144], this topic deserves future research since the memory cost of GMRES can be circumvented. We note that a symmetric preconditioner should be used with the CG solver. Hence, symmetrization procedures, such as

the ones described in [84] should be applied during the construction of the preconditioner.

Bibliography

- [1] L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*. Philadelphia, USA: SIAM, 1997.
- [2] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, second ed., 2003.
- [3] J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. van der Vorst, *Numerical Linear Algebra for High Performance Computers*. Philadelphia, PA, USA: SIAM, 1998.
- [4] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, vol. 13 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge, UK: Cambridge University Press, 2003.
- [5] Ö. S. Ergül, *Accurate and efficient solutions of electromagnetic problems with the multilevel fast multipole algorithm*. PhD thesis, Bilkent University, Ankara, Turkey, 2009.
- [6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. The MIT press, 2001.
- [7] S. M. Rao, D. R. Wilton, and A. W. Glisson, “Electromagnetic scattering by surfaces of arbitrary shape,” *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 409–418, 1982.

- [8] S. M. Rao and D. R. Wilton, “E-field, H-field, and combined-field solution for arbitrarily shaped three-dimensional dielectric bodies,” *Electromag.*, vol. 10, pp. 407–421, 1990.
- [9] X. Q. Sheng, J.-M. Jin, J. Song, W. C. Chew, and C.-C. Lu, “Solution of combined-field integral equation using multilevel fast multipole algorithm for scattering by homogeneous bodies,” *IEEE Trans. Antennas Propagat.*, vol. 46, no. 11, pp. 1718–1726, 1998.
- [10] P. Ylä-Oijala, M. Taskinen, and S. Järvenpää, “Surface integral equation formulations for solving electromagnetic scattering problems with iterative methods,” *Radio Science*, vol. 40, RS6002, doi:10.1029/2004RS003169,, no. 6, 2005.
- [11] K. Chen, *Matrix Preconditioning Techniques and Applications, 2nd ed.*, vol. 19 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge, UK: Cambridge University Press, 2005.
- [12] P. Ylä-Oijala, M. Taskinen, and S. Järvenpää, “Analysis of surface integral equations in electromagnetic scattering and radiation problems,” *Eng. Anal. Boundary Elem.*, vol. 32, no. 3, pp. 196–209, 2008.
- [13] L. Gürel and Ö. Ergül, “Comparisons of FMM implementations employing different formulations and iterative solvers,” in *IEEE Antennas Propagat. Society Int. Symp.*, vol. 1, pp. 19–22, June 2003.
- [14] L. Gürel and Ö. Ergül, “Singularity of the magnetic-field integral equation and its extraction,” *IEEE Antennas Wirel. Propag. Lett.*, vol. 4, 2005.
- [15] Ö. Ergül and L. Gürel, “The use of curl-conforming basis functions for the magnetic-field integral equation,” *IEEE Trans. Antennas Propagat.*, vol. 54, pp. 1917–1926, July 2006.

- [16] Ö. Ergül and L. Gürel, “Improved testing of the magnetic-field integral equation,” *IEEE Microwave Wireless Comp. Lett.*, vol. 15, pp. 615–617, Oct. 2005.
- [17] Ö. Ergül and L. Gürel, “Discretization error due to the identity operator in surface integral equations,” *Comput. Phys. Comm.*, vol. 180, no. 10, pp. 1746–1752, 2009.
- [18] Ö. Ergül and L. Gürel, “Improving the accuracy of the magnetic-field integral equation with the linear-linear basis functions,” *Radio Sci.*, vol. 41, RS4004, doi:10.1029/2005RS003307, July 2006.
- [19] D. R. Wilton and J. E. Wheeler III, “Comparison of convergence rates of the conjugate gradient method applied to various integral equation formulations,” *Prog. Electromagn. Res. (PIER)*, vol. 5, pp. 131–158, 1991.
- [20] Ö. Ergül and L. Gürel, “A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm,” *IEEE Trans. Antennas Propagat.*, vol. 57, no. 6, 2009.
- [21] D. A. Dunavant, “High degree efficient symmetrical gaussian quadrature rules for the triangle,” *Int. J. Numer. Methods Eng.*, vol. 21, no. 6, 1985.
- [22] R. D. Graglia, “On the numerical integration of the linear shape functions times the 3-d green’s function or its gradient on a plane triangle,” *IEEE Trans. Antennas Propagat.*, vol. 41, no. 10, pp. 1448–1455, 1993.
- [23] R. E. Hodges and Y. Rahmat-Samii, “The evaluation of MFIE integrals with the use of vector triangle basis functions,” *Microwave Opt. Technol. Lett.*, vol. 14, no. 1, 1997.
- [24] P. Ylä-Oijala and M. Taskinen, “Calculation of cfe impedance matrix elements with rwg and $\hat{\mathbf{n}} \times \text{rwg}$ functions,” *IEEE Trans. Antennas Propagat.*, vol. 51, no. 8, pp. 1837–1846, 2003.

- [25] L. Greengard and V. Rokhlin, “A fast algorithm for particle simulations,” *J. Comput. Phys.*, vol. 73, no. 2, pp. 325–348, 1987.
- [26] W. C. Chew, J.-M. Jin, E. Michielssen, and J. Song, eds., *Fast and Efficient Algorithms in Computational Electromagnetics*. Norwood, MA, USA: Artech House, Inc., 2001.
- [27] V. Rokhlin, “Rapid solution of integral equations of scattering theory in two dimensions,” *J. Comput. Phys.*, vol. 86, no. 2, pp. 414–439, 1990.
- [28] R. Coifman, V. Rokhlin, and S. Wandzura, “The fast multipole method for the wave equation: A pedestrian prescription,” *IEEE Antennas Propag. Mag.*, vol. 35, no. 3, pp. 7–12, 1993.
- [29] S. Koc, J. Song, and W. C. Chew, “Error analysis for the numerical evaluation of the diagonal forms of the scalar spherical addition theorem,” *SIAM J. Numer. Anal.*, pp. 906–921, 1999.
- [30] M. L. Hastriter, S. Ohnuki, and W. C. Chew, “Error control of the translation operator in 3D MLFMA,” *Microwave Opt. Technol. Lett.*, vol. 37, no. 3, pp. 184–188, 2003.
- [31] Ö. Ergül and L. Gürel, “Enhancing the accuracy of the interpolations and anterpolations in MLFMA,” *IEEE Trans. Antennas Propagat. Lett.*, vol. 5, pp. 467–470, 2006.
- [32] Ö. Ergül and L. Gürel, “Optimal interpolation of translation operator in multilevel fast multipole algorithm,” *IEEE Trans. Antennas Propagat.*, vol. 54, pp. 3822–3826, December 2006.
- [33] A. Brandt, “Multilevel computations of integral transforms and particle interactions with oscillatory kernels,” *Comput. Phys. Comm.*, vol. 65, no. 1-3, pp. 24–38, 1991.

- [34] I. C. F. Ipsen, “A note on preconditioning nonsymmetric matrices,” *SIAM J. Sci. Comput.*, vol. 23, no. 3, pp. 1050–1051, 2002.
- [35] G. H. Golub and C. F. van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.
- [36] R. W. Freund, G. H. Golub, and N. sM. Nachtigal, “Iterative solution of linear systems,” *Acta Numerica*, vol. 1, pp. 57–100, 2008.
- [37] L. N. Trefethen, “Computation of pseudospectra,” *Acta Numerica*, vol. 8, no. 1, pp. 247–295, 1999.
- [38] M. Benzi, “Preconditioning techniques for large linear systems: A survey,” *J. Comput. Phys.*, vol. 182, no. 2, pp. 418–477, 2002.
- [39] I. C. F. Ipsen and C. D. Meyer, “The idea behind Krylov methods,” *Am. Math. Mon.*, vol. 105, no. 10, pp. 889–899, 1998.
- [40] T. Malas and L. Gürel, “Incomplete LU preconditioning with the multi-level fast multipole algorithm for electromagnetic scattering,” *SIAM J. Sci. Comput.*, vol. 29, no. 4, pp. 1476–1494, 2007.
- [41] T. Malas and L. Gürel, “Incomplete LU preconditioning strategies for MLFMA,” in *2006 IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting and AMEREM Meeting*, (Albuquerque, New Mexico, USA), July 2006.
- [42] T. Malas and L. Gürel, “Incomplete LU preconditioning for the electric-field integral equation,” in *1st European Conference on Antennas and Propagation (EuCAP)*, (Nice, France), Nov. 2006.
- [43] T. Malas and L. Gürel, “Effective preconditioning techniques for iterative solutions of integral-equation methods,” in *IV. International Workshop on Electromagnetic Wave Scattering*, (Gebze, Turkey), Sept. 2006.

- [44] T. Malas and L. Gürel, “Accelerating the multilevel fast multipole algorithm with the sparse-approximate-inverse (SAI) preconditioning,” *SIAM J. Sci. Comput.*, vol. 31, no. 3, pp. 1968–1984, 2009.
- [45] T. Malas and L. Gürel, “Effective parallelization of the sparse-approximate-inverse preconditioner for the solution of large-scale integral-equation problems,” in *IEEE International Symposium on Antennas and Propagation*, (Charleston, South Carolina, USA), June 2009.
- [46] T. Malas and L. Gürel, “A parallel sparse approximate inverse preconditioner for the multilevel fast multipole algorithm,” in *URSI-Turkey 2006 Scientific Symposium*, (Ankara, Turkey), Sept. 2006.
- [47] L. Gürel, Ö. Ergül, and T. Malas, “Parallel MLFMA solution of large integral-equation problems,” in *URSI-Turkey 2006 Scientific Symposium*, (Ankara, Turkey), Sept. 2006.
- [48] T. Malas, Ö. Ergül, and L. Gürel, “Sequential and parallel preconditioners for large-scale integral-equation problems,” in *2007 Computational Electromagnetics Workshop (CEM’07)*, (İzmir, Turkey), pp. 35–43, August 2007.
- [49] Ö. Ergül, T. Malas, Ç. Yavuz, A. Ünal, and L. Gürel, “Computational analysis of complicated metamaterial structures using MLFMA and nested preconditioners,” in *European Conference on Antennas and Propagation (EuCAP)*, (Edinburgh, UK), Nov. 2007.
- [50] Ö. Ergül, A. Ünal, T. Malas, and L. Gürel, “Integral-equation-based approach for the accurate analysis of metamaterials,” in *NanoTR-III Nanoscience and Nanotechnology Conference*, (Ankara, Turkey), June 2007.
- [51] L. Gürel, Ö. Ergül, A. Ünal, and T. Malas, “Fast and accurate analysis of large metamaterial structures using the multilevel fast multipole algorithm,” *Prog. Electromagn. Res., PIER 95*, pp. 179–198, 2009.

- [52] T. Malas and L. Gürel, “Solution of large EFIE problems via preconditioned multilevel fast multipole algorithm,” in *2nd European Conference on Antennas and Propagation (EuCAP)*, (Edinburgh, UK), Nov. 2007.
- [53] Ö. Ergül, T. Malas, A. Ünal, and L. Gürel, “Solutions of large integral-equation problems with preconditioned MLFMA,” in *37th European Microwave Conference (EuMC)*, (Munich, Germany), Oct. 2007.
- [54] T. Malas, Ö. Ergül, and L. Gürel, “Parallel preconditioners for solutions of dense linear systems with tens of millions of unknowns,” in *22nd International Symposium on Computer and Information Sciences (ISCIS07)*, (Ankara, Turkey), Nov. 2007.
- [55] L. Gürel, T. Malas, and Ö. Ergül, “Efficient preconditioning strategies for the multilevel fast multipole algorithm,” in *PIERS 2007 Progress in Electromagnetics Research Symposium*, (Beijing, China), pp. 1620–1624, March 2007.
- [56] T. Malas and L. Gürel, “Iterative near-field (INF) preconditioner for the multilevel fast multipole algorithm,” *SIAM J. Sci. Comput.*, vol. submitted, Feb. 2009.
- [57] T. Malas, Ö. Ergül, and L. Gürel, “Approximate MLFMA as an efficient preconditioner,” in *2007 IEEE International Symposium on Antennas and Propagation*, (Honolulu, Hawaii, USA), June 2007.
- [58] T. Malas, Ö. Ergül, and L. Gürel, “Iterative solutions of large-scale electromagnetics problems using an inner-outer scheme with ordinary and approximate multilevel fast multipole algorithms,” *J. Comput. Phys.*, Feb. 2009, submitted.

- [59] T. Malas, Ö. Ergül, and L. Gürel, “Variable preconditioning with nested iterations employing MLFMA,” in *2006 IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting and AMEREM Meeting*, (Albuquerque, New Mexico, USA), July 2006.
- [60] T. Malas, Ö. Ergül, and L. Gürel, “Effective preconditioners for large integral-equation problems,” in *2nd European Conference on Antennas and Propagation (EuCAP)*, (Edinburgh, UK), Nov. 2007.
- [61] T. Malas and L. Gürel, “Preconditioning large integral-equation problems involving complex targets,” in *2008 IEEE International Symposium on Antennas and Propagation and the 2008 USNC/URSI National Radio Science Meeting*, (San Diego, California, USA), July 2008.
- [62] T. Malas and L. Gürel, “Iterative block near-field preconditioners for surface integral-equation formulations of dielectric problems,” in *Vth International Workshop on Electromagnetic Wave Scattering (EWS)*, (Antalya, Turkey), Oct. 2008.
- [63] L. Gürel, Ö. Ergül, and T. Malas, “Solutions of extremely large electromagnetics problems involving tens of millions of unknowns using parallel MLFMA and preconditioners,” in *XXIX General Assembly of the International Union of Radio Science*, (Chicago, Illinois, USA), Aug. 2008.
- [64] T. Malas and L. Gürel, “Schur complement preconditioners for surface integral-equation formulations of dielectric problems solved with the multilevel fast multipole algorithm,” *SIAM J. Sci. Comput.*, submitted, Jan. 2010.
- [65] T. Malas and L. Gürel, “Iterative Schur complement preconditioners for the simulation of dielectric problems in computational electromagnetics,” *SIAM J. Sci. Comput.*, submitted, Feb. 2010.

- [66] T. Malas and L. Gürel, “An effective preconditioner based on the Schur complement reduction for integral-equation formulations of dielectric problems,” in *IEEE International Symposium on Antennas and Propagation*, (Charleston, South Carolina, USA), June 2009.
- [67] Ö. Ergül, T. Malas, S. Kılınc, S. Sarıtaş, and L. Gürel, “Analysis of photonic-crystal problems with mlfma and approximate Schur preconditioners,” in *2009 Computational Electromagnetics International Workshop (CEM’09)*, (İzmir, Turkey), July 2009.
- [68] T. Malas and L. Gürel, “Increasing robustness and efficiency of surface-integral-equation solutions of dielectric problems with approximate Schur preconditioners,” in *2009 Computational Electromagnetics International Workshop (CEM’09)*, (İzmir, Turkey), July 2009.
- [69] “Intel MKL Web page.” <http://software.intel.com/en-us/intel-mkl/>.
- [70] “Intel MPI Web page.” <http://software.intel.com/en-us/intel-mpi-library/>.
- [71] “OPEN MPI Web page.” <http://www.open-mpi.org/>.
- [72] “MVAPICH Web page.” <http://mvapich.cse.ohio-state.edu/>.
- [73] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith, “PETSc users manual,” Tech. Rep. ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.
- [74] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, “PETSc Web page.” <http://www.mcs.anl.gov/petsc>.
- [75] G. Meurant, *Computer Solution of Large Linear Systems*, vol. 28 of *Studies In Mathematics and Its Applications*. North Holland, Amsterdam: North-Holland, 1999.

- [76] E. Chow and Y. Saad, “Experimental study of ILU preconditioners for indefinite matrices,” *J. Comput. Appl. Math.*, vol. 86, no. 2, pp. 387–414, 1997.
- [77] M. A. Heroux and J. M. Willenbring, “Trilinos users guide,” Tech. Rep. SAND2003-2952, Sandia National Laboratories, 2003.
- [78] M. Bollhöfer and Y. Saad, “ILUPACK - preconditioning software package,” 2004. Available online at <http://www.math.tu-berlin.de/ilupack/>.
- [79] 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.
- [80] K. Sertel and J. L. Volakis, “Incomplete LU preconditioner for FMM implementation,” *Microw. Opt. Tech. Lett.*, vol. 28, pp. 265–267, 2000.
- [81] 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.*, vol. 185, no. 1, pp. 158–175, 2003.
- [82] B. Carpentieri, I. S. Duff, and L. Giraud, “Robust preconditioning of dense problems from electromagnetics,” in *Int. J. Comput. Numer. Anal. Appl.* (L. Vulkov, J. Waśniewski, and P. Yalamov, eds.), pp. 170–178, Springer, 2000.
- [83] B. Carpentieri, I. S. Duff, and L. Giraud, “Sparse pattern selection strategies for robust frobenius-norm minimization preconditioners in electromagnetism,” *Numer. Linear Algebra Appl.*, vol. 7, no. 7-8, pp. 667–685, 2000.
- [84] B. Carpentieri, I. S. Duff, L. Giraud, and M. M. m. Made, “Sparse symmetric preconditioners for dense linear systems in electromagnetism.,” Tech. Rep. TR/PA/01/35, CERFACS, Toulouse, France, 2001.

- [85] 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 Propagat.*, vol. 52, no. 9, pp. 158–175, 2004.
- [86] G. Alléon, M. Benzi, and L. Giraud, “Sparse approximate inverse preconditioning for dense linear systems arising in computational electromagnetics,” *Numer. Algorithms*, vol. 16, pp. 1–15, 1997.
- [87] M. Benzi, D. B. Szyld, and A. van Duin, “Orderings for incomplete factorization preconditioning of nonsymmetric problems,” *SIAM J. Sci. Comput.*, vol. 20, no. 5, pp. 1652–1670, 1999.
- [88] M. Bollhöfer, “A robust and efficient *ILU* that incorporates the growth of the inverse triangular factors,” *SIAM J. Sci. Comput.*, vol. 25, no. 1, pp. 86–103, 2003.
- [89] Y. Saad, “ILUT: a dual threshold incomplete LU factorization,” *Numer. Linear Algebra Appl.*, vol. 1, no. 4, pp. 387–402, 1994.
- [90] B. Carpentieri, *Sparse preconditioners for dense complex linear systems in electromagnetic applications*. Ph.D. dissertation, INPT, April 2002. TH/PA/02/48.
- [91] L. Gürel, 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, pp. 1046–1058, 2003.
- [92] B. Carpentieri, 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, pp. 774–792, 2005.

- [93] M. Benzi and M. Tuma, “A comparative study of sparse approximate inverse preconditioners,” *Appl. Numer. Math.*, vol. 30, no. 2–3, pp. 305–340, 1999.
- [94] L. Y. Kolotilina and A. Y. Yeremin, “Factorized sparse approximate inverse preconditioning I. Theory,” *SIAM J. Matrix Anal. Appl.*, vol. 14, no. 9, pp. 45–58, 1993.
- [95] M. Benzi and M. Tuma, “A sparse approximate inverse preconditioner for nonsymmetric linear systems,” *SIAM J. Sci. Comput.*, vol. 19, no. 3, pp. 968–994, 1998.
- [96] E. Chow and Y. Saad, “Approximate inverse preconditioners via sparse-sparse iterations,” *SIAM J. Sci. Comput.*, vol. 19, no. 3, pp. 995–1023, 1998.
- [97] M. J. Grote and T. Huckle, “Parallel preconditioning with sparse approximate inverses,” *SIAM J. Sci. Comput.*, vol. 18, no. 3, pp. 838–853, 1997.
- [98] E. Chow, “Parallel implementation and practical use of sparse approximate inverse preconditioners with a priori sparsity patterns,” *Int. J. High Perform. Comput. Appl.*, vol. 15, no. 1, pp. 56–74, 2001.
- [99] L. Gürel and Ö. Ergül, “Fast and accurate solutions of integral-equation formulations discretised with tens of millions of unknowns,” *Electronics Lett.*, vol. 43, pp. 499–500, 2007.
- [100] A. Grbic and G. V. Eleftheriades, “Overcoming the diffraction limit with a planar left-handed transmission-line lens,” *Phys. Rev. Lett.*, vol. 92, pp. 117403–1–117403–4, Mar. 2004.
- [101] K. Aydın, I. Bulu, and E. Özbay, “Subwavelength resolution with a negative-index metamaterial superlens,” *Appl. Phys. Lett.*, vol. 90, pp. 254102–1–254102–3, June 2007.

- [102] D. Schurig, J. J. Mock, B. J. Justice, S. A. Cummer, J. B. Pendry, A. F. Starr, and D. R. Smith, “Metamaterial electromagnetic cloak at microwave frequencies,” *Science*, vol. 314, pp. 977–980, Nov. 2006.
- [103] Z. Weng, N. Wang, Y. Jiao, and F. Zhang, “A directive patch antenna with metamaterial structure,” *Microwave Opt. Technol. Lett.*, vol. 49, pp. 456–459, Feb. 2007.
- [104] M. Gokkavas, K. Güven, I. Bulu, K. Aydın, R. S. Penciu, M. Kafesaki, C. M. Soukoulis, and E. Özbay, “Experimental demonstration of a left-handed metamaterial operating at 100 ghz,” *Phys. Rev. B.*, vol. 73, pp. 193103–1–193103–4, May 2006.
- [105] K. Abe and S.-L. Zhang, “A variable preconditioning using the SOR method for GCR-like methods,” *Int. J. Numer. Anal. Model.*, vol. 2, no. 2, pp. 147–161, 2005.
- [106] Ö. Ergül and L. Gürel, “Efficient parallelization of the multilevel fast multipole algorithm for the solution of large-scale scattering problems,” *IEEE Trans. Antennas Propagat.*, vol. 56, pp. 2335–2345, Aug. 2008.
- [107] A. Bouras and V. Frayssé, “Inexact matrix-vector products in Krylov methods for solving linear systems: A relaxation strategy,” *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 3, pp. 660–678, 2005.
- [108] V. Simoncini and D. B. Szyld, “Flexible inner-outer krylov subspace methods,” *SIAM J. Numer. Anal.*, vol. 40, no. 6, pp. 2219–2239, 2002.
- [109] J. van den Eshof, G. L. G. Sleijpen, and M. B. van Gijzen, “Relaxation strategies for nested Krylov methods,” *J. Comput. Appl. Math.*, vol. 177, no. 2, pp. 347–365, 2005.
- [110] H. van der Vorst and C. Vuik, “GMRESR: a family of nested GMRES methods,” *Numer. Linear Algebra Appl.*, vol. 1, no. 4, pp. 369 – 386, 1994.

- [111] A. P. Pavacic, D. L. del Río, J. R. Mosig, and G. V. Eleftheriades, “Three-dimensional ray-tracing to model internal reflections in off-axis lens antennas,” *IEEE Trans. Antennas Propagat.*, vol. 54, pp. 604–612, 2006.
- [112] J. D. Joannopoulos, R. D. Meade, and J. N. Winn, *Photonic Crystals: Molding the Flow of Light*. Princeton University Press, 2008.
- [113] T. W. Lloyd, J. M. Song, and M. Yang, “Numerical study of surface integral formulations for low-contrast objects,” *IEEE Antennas Wirel. Propag. Lett.*, vol. 4, pp. 482–485, 2005.
- [114] Ö. Ergül and L. Gürel, “Comparison of integral-equation formulations for the fast and accurate solution of scattering problems involving dielectric objects with multilevel fast multipole algorithm,” *IEEE Trans. Antennas Propagat.*, vol. 57, pp. 176–187, Jan. 2009.
- [115] P. Ylä-Oijala and M. Taskinen, “Well-conditioned muller formulation for electromagnetic scattering by dielectric objects,” *IEEE Trans. Antennas Propagat.*, vol. 53, no. 10, pp. 3316–3323, 2005.
- [116] P. Ylä-Oijala and M. Taskinen, “Application of combined field integral equation for electromagnetic scattering by composite metallic and dielectric objects,” *IEEE Trans. Antennas Propagat.*, vol. 53, no. 3, pp. 1168–1173, 2005.
- [117] C. Müller, *Foundations of the Mathematical Theory of Electromagnetic Waves*. Springer Verlag, 1969.
- [118] M. Benzi, G. H. Golub, and J. Liesen, “Numerical solution of saddle point problems,” *Acta Numer.*, vol. 14, pp. 1–137, 2005.
- [119] E. Chow and Y. Saad, “Approximate inverse techniques for block-partitioned matrices,” *SIAM J. Sci. Comput.*, vol. 18, no. 6, pp. 1657–1675, 1997.

- [120] C. Siefert and E. de Sturler, “Preconditioners for generalized saddle-point problems,” *SIAM J. Numer. Anal.*, vol. 44, no. 3, pp. 1275–1296, 2006.
- [121] M. Benzi and G. H. Golub, “A preconditioner for generalized saddle point problems,” *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 1, pp. 20–41, 2005.
- [122] J. Zhang, “On preconditioning Schur complement and Schur complement preconditioning,” *Electron. Trans. Numer. Anal.*, vol. 10, pp. 115–130, 2000.
- [123] J. M. Ford, K. Chen, and D. Evans, “On a recursive Schur preconditioner for iterative solution of a class of dense matrix problems,” *Int. J. Comput. Math.*, vol. 80, no. 1, pp. 105–122, 2003.
- [124] M. A. Olshanskii and Y. V. Vassilevski, “Pressure Schur complement preconditioners for the discrete oseen problem,” *SIAM J. Sci. Comput.*, vol. 29, no. 6, pp. 2686–2704, 2007.
- [125] E. de Sturler and J. Liesen, “Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. part i: Theory,” *SIAM J. Sci. Comput.*, vol. 26, no. 5, p. 1598, 2005.
- [126] Z.-Z. Bai, “Structured preconditioners for nonsingular matrices of block two-by-two structures,” *Math. Comp.*, vol. 75, no. 254, p. 791, 2006.
- [127] G. Schmidlin, U. Fischer, Z. Andjelić, and C. Schwab, “Preconditioning of the second-kind boundary integral equations for 3d eddy current problems,” *Int. J. Numer. Methods Eng.*, vol. 51, no. 9, 2001.
- [128] W. Bomhof and H. A. van der Vorst, “A parallel linear system solver for circuit simulation problems,” *Numer. Linear Algebra Appl.*, vol. 7, no. 7-8, pp. 649–665, 2000.

- [129] A. J. Poggio and E. K. Miller, “Integral equation solutions of three-dimensional scattering problems,” in *Computer Techniques for Electromagnetics* (R. Mittra, ed.), Oxford: Pergamon Press, 1973, Chap. 4.
- [130] G. C. Hsiao and R. E. Kleinman, “Mathematical foundations for error estimation in numerical solutions of integral equations in electromagnetics,” *IEEE Trans. Antennas Propagat.*, vol. 45, no. 3, pp. 316–328, 1997.
- [131] T.-K. Wu and L. L. Tsai, “Scattering from arbitrarily-shaped lossy dielectric bodies of revolution,” *Radio Science*, vol. 12, no. 5, pp. 709–718, 1977.
- [132] Y. Chang and R. F. Harrington, “A surface formulation for characteristic modes of material bodies,” *IEEE Trans. Antennas Propagat.*, vol. 25, no. 6, pp. 789–795, 1977.
- [133] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. Oxford University Press, USA, 2005.
- [134] L. Little, Y. Saad, and L. Smoch, “Block LU preconditioners for symmetric and nonsymmetric saddle point problems,” *SIAM J. Sci. Comput.*, vol. 25, no. 2, pp. 729–748, 2003.
- [135] W. W. Hager, “Updating the inverse of a matrix,” *SIAM review*, pp. 221–239, 1989.
- [136] S. S. Haykin, *Kalman Filtering and Neural Networks*. New York, NY, USA: John Wiley & Sons, Inc., 2001.
- [137] V. Simoncini and D. B. Szyld, “The effect of non-optimal bases on the convergence of krylov subspace methods,” *Numerische Mathematik*, vol. 100, no. 4, pp. 711–733, 2005.

- [138] S. G. Johnson, S. Fan, P. R. Villeneuve, and J. D. Joannopoulos, “Guided modes in photonic crystal slabs,” *Physical Review B*, vol. 60, no. 8, pp. 5751–5758, 1999.
- [139] P. Loschialpo, D. W. Forester, and J. Schelleng, “Anomalous transmission through near unit index contrast dielectric photonic crystals,” *J. Appl. Phys.*, vol. 86, no. 10, pp. 5342–5347, 1999.
- [140] I. S. Duff, L. Giraud, J. Langou, and E. Martin, “Using spectral low rank preconditioners for large electromagnetic calculations,” *Internat. J. Numer. Methods Engrg.*, vol. 62, no. 3, pp. 416–434, 2005.
- [141] P.-L. Rui, R.-S. Chen, D.-X. Wang, and E. K.-N. Yung, “Spectral two-step preconditioning of multilevel fast multipole algorithm for the fast monostatic RCS calculation,” *IEEE Trans. Antennas Propagat.*, vol. 55, pp. 2268–2275, Aug. 2007.
- [142] L. Banjai and W. Hackbusch, “Hierarchical matrix techniques for low- and high-frequency Helmholtz problems,” *IMA J. Numer. Anal.*, vol. 28, pp. 46–79, 2008.
- [143] V. E. Howle and S. A. Vavasis, “An iterative method for solving complex-symmetric systems arising in electrical power modeling,” *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 4, pp. 1150–1178, 2005.
- [144] R. Natarajan, “An iterative scheme for dense, complex-symmetric, linear systems in acoustics boundary-element computations,” *SIAM J. Sci. Comput.*, vol. 19, pp. 1450–1470, 1998.