Combining fast multipole techniques and an approximate inverse preconditioner for large electromagnetism calculations[1]

Bruno Carpentieri[2,3], Iain S. Duff[4], Luc Giraud[2,5], and Guillaume Sylvand[6]

ABSTRACT

The boundary element method has become a popular tool for the solution of Maxwell's equations in electromagnetism. From a linear algebra point of view, this leads to the solution of large dense complex linear systems where the unknowns are associated with the edges of the mesh defined on the surface of the illuminated object. In this paper, we address the iterative solution of these linear systems via preconditioned Krylov solvers. Our primary focus is on the design of an efficient parallelizable preconditioner. In that respect, we consider an approximate inverse method based on the Frobenius-norm minimization. The preconditioner is constructed from a sparse approximation of the dense coefficient matrix, and the patterns both for the preconditioner and for the coefficient matrix are computed *a priori* using geometric information from the mesh. We describe how such a preconditioner can be naturally implemented in a parallel code that implements the multipole technique for the matrix-vector product calculation. We investigate the numerical scalability of our preconditioner on realistic industrial test problems and show that it exhibits some limitations on very large problems of size close to one million unknowns. To improve its robustness on those large problems we propose an embedded iterative scheme that combines nested GMRES solvers with different fast multipole computations. We show through extensive numerical experiments that this new scheme is extremely robust at affordable memory and CPU costs for the solution of very large and challenging problems.

Keywords: electromagnetic scattering problems, Frobenius–norm minimization preconditioner, large dense complex linear systems, spectral low rank update preconditioner.

AMS(MOS) subject classifications: 65F05, 65F50.

---

[1]Also appeared as CERFACS Report TR/PA/03/77. Current RAL reports available by anonymous ftp to `ftp.numerical.rl.ac.uk` in directory pub/reports. This report is in file cdgsRAL2003024.ps.gz. The report also available through the URL `www.numerical.rl.ac.uk/reports/reports.html`.

[2]CERFACS, 42 Ave G. Coriolis, 31057 Toulouse Cedex, France.

[3]carpenti@cerfacs.fr

[4]i.s.duff@rl.ac.uk. This work was supported by the EPSRC Grant GR/R46441.

[5]giraud@cerfacs.fr.

[6]gsylvand@sophia.inria.fr. CERMICS-INRIA, Sophia Antipolis, France.

Computational Science and Engineering Department
Atlas Centre
Rutherford Appleton Laboratory
Oxon OX11 0QX

November 28, 2003

Contents

1 Introduction

The analysis of wave propagation phenomena is gaining an increasing interest in recent years in the simulation of many challenging industrial processes, ranging from the prediction of the Radar Cross Section (RCS) of arbitrarily shaped 3D objects like aircraft, the study of electromagnetic compatibility of electrical devices with their environment, the design of antennae and absorbing materials, and many others. All these simulations are very demanding in terms of computer resources, and require fast and efficient numerical methods to compute an approximate solution of Maxwell's equations. Using the equivalence principle, Maxwell's equations can be recast in the form of four integral equations that relate the electric and magnetic fields to the equivalent electric and magnetic currents on the surface of the object.

Amongst integral formulations, the electric-field integral equation (EFIE) is the most general for electromagnetic scattering problems as it can handle fairly general geometries, and thus is widely used in industrial simulations. The EFIE provides a first-kind integral equation which is well known to be ill-conditioned and gives rise to linear systems that are challenging to solve by iterative methods. The discretization is performed on the surface of the object and gives rise to a linear system

$$Ax = b, \tag{1.1}$$

where the matrix A is a $n \times n$, dense, complex, symmetric, non-Hermitian matrix. Direct solution methods have been for years the method of choice for solving systems (1.1) because they are reliable and predictable both in terms of accuracy and computing costs. However, for the solution of large-scale problems, direct methods are infeasible even on large parallel platforms because they require the unaffordable storage of n^2 single or double precision complex entries of the coefficient matrix and $\mathcal{O}(n^3)$ floating-point operations to compute the factorization. The use of preconditioned Krylov solvers can be an alternative to direct solution methods, provided we have fast matrix-free matrix-vector products and robust preconditioners. Research efforts have recently concentrated on fast methods for performing matrix-vector products with $\mathcal{O}(n \log(n))$ computational complexity, including strategies for parallel distributed memory implementations. These methods, generally referred to as *hierarchical methods*, were introduced originally in the context of the study of particle simulations and can be effectively used in boundary element applications.

In this paper, we focus on the design of a parallelizable preconditioner to be used in conjunction with a parallel distributed fast multipole technique that implements the matrix-vector calculation. We consider an approximate inverse preconditioner based on Frobenius-norm minimization with a pattern prescribed in advance. In Section 2, we describe the implementation of the preconditioner within an out-of-core parallel code that implements the Fast Multipole Method (FMM) for the matrix-vector product. We investigate the numerical scalability of the preconditioner on a

set of industrial problems of increasing size and show that it becomes less effective when the problem size becomes very large. To overcome this weakness, we propose in Section 3 an embedded iterative scheme based on the GMRES method that aims at improving the robustness of the preconditioner on large applications. We illustrate the numerical efficiency and the cost effectiveness of the proposed scheme on systems of up to one million unknowns arising from challenging problems from the electromagnetism community.

2 Preconditioning Boundary Integral Equations

For large meshes with many surface details, the density of the discretization mesh is non-uniform, and the matrix generated by the Method of Moments can become ill-conditioned. The convergence of Krylov methods depends to a large extent on the eigenvalue distribution of the coefficient matrix. In Figure 2.1, we plot the eigenvalue distribution in the complex plane of the matrix associated with a satellite geometry. This distribution is representative of the general trend; it can be seen that the eigenvalues of the system are very scattered, many of them have a large negative real part and no clustering appears. Such a distribution is not at all favourable for the rapid convergence of Krylov methods. One goal of preconditioning is to improve this distribution by grouping the eigenvalues into a few small clusters.

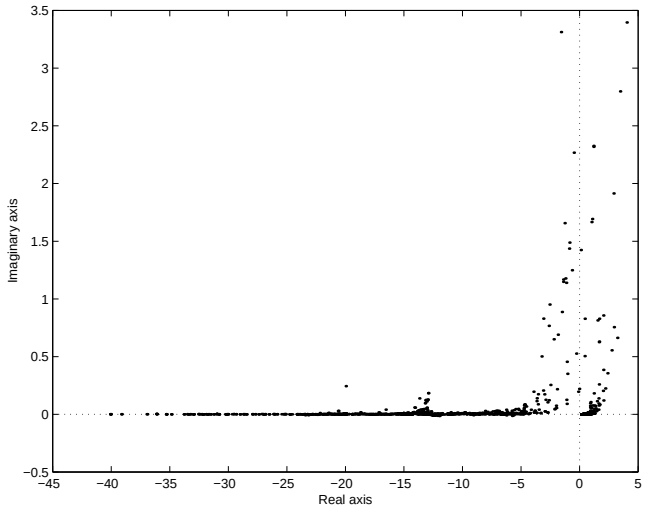


Figure 2.1: Eigenvalue distribution in the complex plane of the coefficient matrix for a scattering problem from a satellite that is representative of the general trend.

The design of robust preconditioners for boundary integral equations can be

challenging. Simple preconditioners like the diagonal of A, diagonal blocks, or a band can be effective only when the coefficient matrix has some degree of diagonal dominance arising from the integral formulation (Song, Lu and Chew 1997). Block diagonal preconditioners are generally more robust than their pointwise counterparts, but may require matrix permutations or renumbering of the grid points to cluster the large entries close to the diagonal. Incomplete factorizations have been successfully used on nonsymmetric dense systems by Sertel and Volakis (2000) and hybrid integral formulations by Lee, Lu and Zhang (2003), but on the EFIE the triangular factors computed by the factorization are often very ill-conditioned due to the indefiniteness of A. This makes the triangular solves highly unstable and the preconditioner useless. In Carpentieri, Duff, Giraud and Magolu monga Made (2002), we used a small diagonal shift applied to A before computing the factorization with the intention of moving the eigenvalues along the imaginary axis. This might avoid a possible eigenvalue cluster close to zero and can help the computation of more stable factors in some cases. However, this shift is not easy to tune and there is no way to predict its effect.

Approximate inverse methods are generally less prone to instabilities on indefinite systems, and several preconditioners of this type have been proposed in electromagnetism (see for instance Alléon, Benzi and Giraud, 1997, Carpentieri, 2002, Carpentieri, Duff and Giraud, 2000, Chen, 2001, Lee, Lu and Zhang, 2002, Samant, Michielssen and Saylor, 1996, Vavasis, 1992). Owing to the rapid decay of the discrete Green's function, the location of the large entries in the inverse matrix exhibit some structure. In addition, only a very small number of its entries have relatively large magnitude. This means that a very sparse matrix is likely to retain the most relevant contributions to the exact inverse. This remarkable property can be effectively exploited in the design of robust approximate inverses as preconditioners for electromagnetism applications.

2.1 Frobenius-norm minimization preconditioner

In this section, we describe an approximate inverse preconditioner based on Frobenius-norm minimization. The original idea, due to Benson (1973) and Frederickson (1975), is to compute the sparse approximate inverse as the matrix M which minimizes $\|I - MA\|_F$ (or $\|I - AM\|_F$ for right preconditioning) subject to certain sparsity constraints. The Frobenius norm is usually chosen since it allows the decoupling of the minimization problems into n independent linear least-squares problems, one for each column of M, when preconditioning from the right (or row of M, when preconditioning from the left). The independence of these least-squares problems follows immediately from the identity:

$$\|I - AM\|_F^2 = \sum_{j=1}^{n} \|e_j - Am_{\bullet j}\|_2^2 \tag{2.1}$$

where e_j is the jth canonical unit vector, $m_{\bullet j}$ is the column vector representing the jth column of M and n the dimension of the square matrices. In the case of left preconditioning, because A is symmetric, the analogous relation

$$\|I - MA\|_F^2 = \|I - AM^T\|_F^2 = \sum_{j=1}^{n} \|e_j - Am_{j\bullet}\|_2^2 \tag{2.2}$$

holds, where $m_{j\bullet}$ is the column vector representing the jth row of M. Clearly, there is considerable scope for parallelism in this approach. The main issue is the selection of the sparsity pattern of M, that is the set of indices

$$S = \{ (i,j) \subseteq [1,n]^2 \; s.t. \; m_{ij} = 0 \}. \tag{2.3}$$

The idea is to keep M reasonably sparse while trying to capture the "large" entries of the inverse, which are expected to contribute the most to the quality of the preconditioner. Two different approaches can be followed for this purpose: an adaptive technique that dynamically tries to identify the best structure for M, and a static technique, where the pattern of M is prescribed *a priori* based on some heuristics. Adaptive methods usually start with a simple initial guess, like a diagonal matrix, and then improve the pattern until a criterion of the form $\|Am_{\bullet j} - e_j\|_2 < \varepsilon$ (for each j) is satisfied for a given $\varepsilon > 0$ or until a maximum number of nonzeros in $m_{\bullet j}$ is reached. If the norm is larger than ε and the number of nonzeros used is less than a fixed maximum, the pattern is enlarged according to some heuristics and the jth column of the approximate inverse is recomputed. The process is repeated until the required accuracy or storage limit is met (see Chow and Saad, 1998, Grote and Huckle, 1997).

Adaptive strategies can solve fairly general and hard problems but tend to be very expensive. The use of effective static pattern selection strategies can greatly reduce the amount of work in terms of CPU-time, and can substantially improve the overall setup process. When the coefficient matrix has a special structure or special properties, efforts have been made to find a pattern that can retain the entries of A^{-1} having large modulus (de Boor 1980, Demko 1977, Demko, Moss and Smith 1984, Tang 1987). If A is row diagonally dominant, then the entries in the inverse decay columnwise and vice versa (Tang 1987). On boundary integral equations the discrete Green's function decays rapidly far from the diagonal, and the inverse of A may have a very similar structure to that of A. The discrete Green's function can be considered as a row or as a column of the exact inverse depicted on the physical computational grid. In this case a good pattern for the preconditioner can be computed in advance using graph information from $\tilde{A}$, a sparse approximation of the coefficient matrix constructed by dropping all the entries lower than a prescribed global threshold (Alléon et al. 1997, Carpentieri et al. 2000, Kolotilina 1988). When fast methods are used for the matrix-vector products, all the entries of A are not available and the pattern can be formed by exploiting the near-field part of the matrix that is explicitly computed and available in the FMM (Lee et al. 2002).

Since we work in an integral equation context, relevant information for the construction of the pattern of M can be extracted from the mesh. When the object geometries are smooth, only the neighbouring edges (in the mesh topology sense) in the mesh can have a strong interaction with each other, while far-away connections are generally much weaker. Thus an effective pattern for the jth column of the approximate inverse can be computed by selecting in the mesh edge j and its qth level nearest-neighbours. Three levels generally provide a good pattern for constructing an effective sparse approximate inverse. Using more levels increases the computational cost but does not improve the quality of the preconditioner substantially (Carpentieri et al. 2000). When the object geometries are not smooth or have disconnected parts, far-away edges in the mesh can have a strong interaction and be strongly coupled in the inverse matrix. In this case, a more robust pattern for the preconditioner can be computed using geometrical information, that is selecting for each edge all those edges within a sufficiently large geometric neighbourhood. In Carpentieri et al. (2000) we compared pattern selection strategies based both on algebraic and mesh information on a large set of problems, and found that those exploiting geometric information are the most effective in capturing the large entries of the inverse.

In order to preserve sparsity, $\mathcal{O}(1)$ nonzero locations are computed in the pattern of the column $m_{\bullet j}$ of M. This makes each QR least-squares solution for each column of M from (2.1) cost $\mathcal{O}(n)$, and the overall construction of M costs approximately $\mathcal{O}(n^2)$ arithmetic operations. This cost can be significantly reduced if the preconditioner is computed using as input a sparse approximation $\tilde{A}$ of the dense coefficient matrix A. On general problems, this approach can cause a severe deterioration of the quality of the preconditioner. In an integral equation context, it is likely to be more effective because the boundary element method generally introduces a very localized strong coupling among the edges in the underlying mesh. It means that a very sparse matrix can still retain the most relevant contributions from the singular integrals that give rise to dense matrices. If the sparsity pattern S of M is known in advance, the nonzero structure for the jth column of M is automatically determined and defined as

$$J = \{i \in [1,n] \; s.t. \; (i,j) \in S\}.$$

The least-squares solution involves only the columns of $\tilde{A}$ indexed by J; we indicate this subset by $\tilde{A}(:,J)$. When $\tilde{A}$ is sparse, many rows in $\tilde{A}(:,J)$ are usually null, not affecting the solution of the least-squares problems (2.1). Thus if I is the set of indices corresponding to the nonzero rows in $\tilde{A}(:,J)$, and if we define $\hat{A} = \tilde{A}(I,J)$, $\hat{m}_j = m_j(J)$, and $\hat{e}_j = e_j(J)$, the actual "reduced" least-squares problems are

$$min \|\hat{e}_j - \hat{A}\hat{m}_j\|_2, \; j = 1, ..., n. \tag{2.4}$$

Usually problems (2.4) have much smaller size than problems (2.1) and can be efficiently solved by a dense QR factorization. In Alléon et al. (1997) the same

nonzero sparsity pattern is selected both for $\tilde{A}$ and M; in that case, especially when the pattern is very sparse, the computed preconditioner may be poor on some geometries. Selecting more entries in $\tilde{A}$ than in M can provide a more robust preconditioner, and the additional cost in terms of CPU time is negligible because of the complexity of the QR factorization (Carpentieri et al. 2000). Increasing the number of rows q_I, that is the number of entries of $\tilde{A}$, affects the CPU time far less than increasing the density of the preconditioner, that is the number of columns p_J in the least-squares problems. This is because each least-squares solution costs $\mathcal{O}(q_I p_J^2)$.

2.2 Implementation of the preconditioner in the FMM context

The Fast Multipole Method, introduced by Greengard and Rokhlin (1987), provides an algorithm for computing approximate matrix-vector products for electromagnetic scattering problems. The method is fast in the sense that the computation of one matrix-vector product costs $\mathcal{O}(n \log n)$ arithmetic operations instead of the usual $\mathcal{O}(n^2)$ operations. It requires storage only of the near-field part of the matrix, that is of $\mathcal{O}(n \log n)$ instead of $\mathcal{O}(n^2)$ entries. These properties mean that using iterative solvers for the solution of large problems becomes feasible.

The basic idea of the algorithm is to compute interactions amongst degrees of freedom in the mesh at different levels of accuracy depending on their physical distance. Single and multilevel variants of the FMM exist. The 3D object is first entirely enclosed in a large cube that is subdivided into eight cubes. Each cube is recursively divided until the size is small compared with the wavelength. In the hierarchical multilevel algorithm, the box-wise partitioning of the obstacle is carried out until the size of the smallest box is generally half of a wavelength, and a tree-structured data is used at all levels. In particular, only non-empty cubes are indexed and recorded in the data structure. The resulting tree is called the *oct-tree* (see Figure 2.2) and its leaves are referred to as *leaf boxes*. The oct-tree provides a hierarchical representation of the computational domain partitioned by boxes: each cube has up to eight children and one parent in the oct-tree, except for the largest cube which encloses the whole domain. Obviously, the leaf boxes have no children. Multipole coefficients are computed for all cubes starting from the lowest level of the oct-tree, that is from the leaf boxes, and then recursively for each parent cube by summing together multipole coefficients of its children. For each observation cube, an interaction list is defined which consists of those cubes that are not neighbours of the cube itself but whose parent is a neighbour of the cube's parent (see Figure 2.3 for a 2D representation). The interactions of degrees of freedom within neighbouring boxes are computed exactly, while the interactions between cubes that are in the interaction list are computed using the FMM. All the other interactions are computed hierarchically on a coarser level by traversing

the oct-tree. Both the computational cost and the memory requirement of the algorithm are of order $\mathcal{O}(n \log n)$. Further information on the algorithmic steps and recent theoretical investigations of the FMM can be found in Darve (2000*a*), Darve (2000*b*), and Sylvand (2002). Grama, Kumar and Sameh (1995), Greengard and Gropp (1990), and Zhao and Johnsson (1991) also discuss parallel implementation issues.

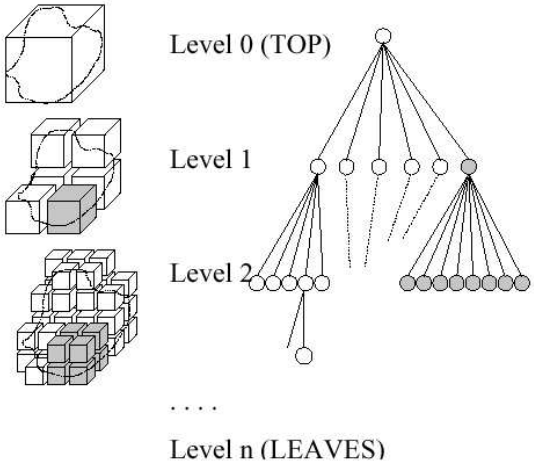


Figure 2.2: The oct-tree in the FMM algorithm. The maximum number of children is eight. The actual number corresponds to the subset that intersect the object.

The box-wise decomposition of the domain required by the FMM naturally leads to an *a priori* pattern selection strategy for M and $\tilde{A}$ using geometric information, that is on the spatial distribution of its degrees of freedom. We will adopt the following criterion: the nonzero structure of the column of the preconditioner associated with a given edge is defined by retaining all the edges within its leaf box and those in one level of neighbouring boxes, and the structure for the sparse approximation of the dense coefficient matrix is defined by retaining the entries associated with edges included in the given leaf box as well as those belonging to the two levels of neighbours. The approximate inverse has a sparse block structure; each block is dense and is associated with one leaf box. Indeed the least-squares problems corresponding to edges within the same box are identical because they are defined using the same nonzero structure and the same set of entries of A. It means that we only have to compute one QR factorization per leaf box. In our implementation we use two different oct-trees, and thus two different partitionings, to assemble the approximate inverse and for the approximate multipole coefficient matrix. The size of the smallest boxes in the partitioning associated with the preconditioner is a user-defined parameter that can be tuned to control the number of nonzeros computed per column, that is the density of the preconditioner. According to our criterion,

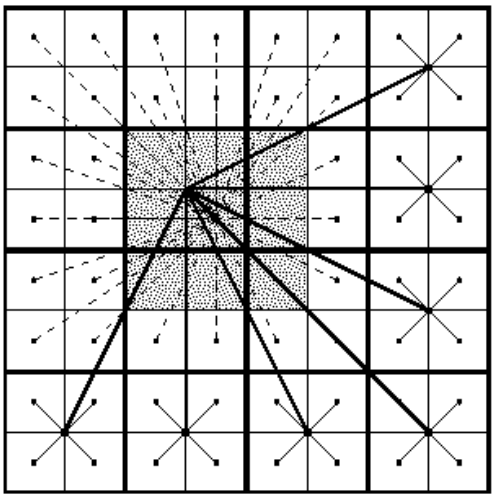


Figure 2.3: Interactions in the multilevel FMM. The interactions for the gray boxes are computed directly. We denote by dashed lines the interaction list for the observation box, that consists of those cubes that are not neighbours of the cube itself but whose parent is a neighbour of the cube's parent. The interactions of the cubes in the list are computed using the FMM. All the other interactions are computed hierarchically on a coarser level, denoted by solid lines.

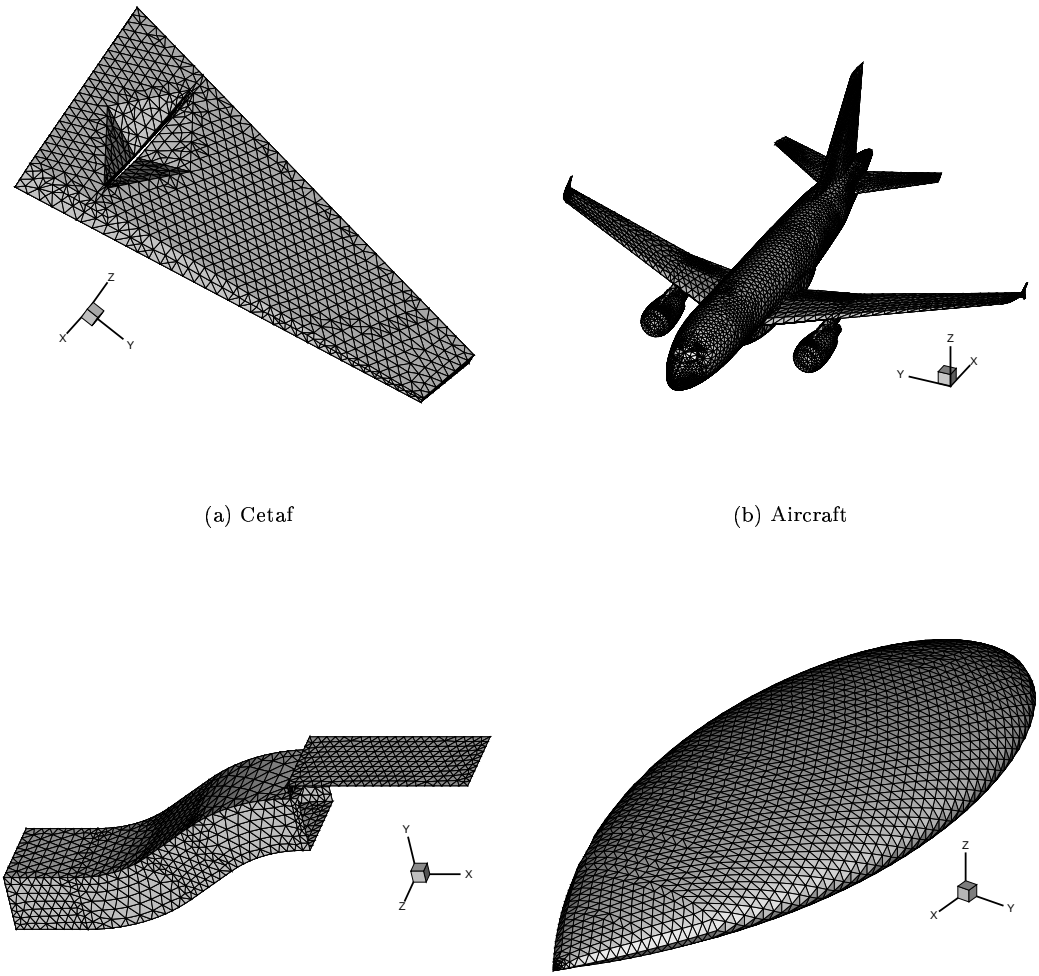
the larger the size of the leaf boxes, the larger the geometric neighbourhood that determines the sparsity structure of the columns of the preconditioner. Parallelism can be exploited by assigning disjoint subsets of leaf boxes to different processors and performing the least-squares solutions independently on each processor. We refer to Sylvand (2002) for a complete description of the parallel code that we use.

2.3 Numerical scalability of the preconditioner

In this section, we study the numerical scalability of the Frobenius-norm minimization preconditioner. The optimal behaviour would be to get frequency independent numerical behaviour, that would result in convergence independent of the problem size. We show results on the following geometries:

- a Cetaf (see Figure 2.4(a)), a typical test case in electromagnetic simulations,
- an Airbus aircraft (see Figure 2.4(b)), a real life model problem in an industrial context,
- a Cobra (see Figure 2.4(c)), a complex geometry that represents an air intake,

- an Almond (see Figure 2.4(d)) that is also a typical test case in electromagnetic simulations.



(a) Cetaf (b) Aircraft

(c) Cobra (d) Almond

Figure 2.4: Mesh associated with test examples

In all the numerical experiments, the surface of the object is always discretized using ten points per wavelength; larger discretizations are obtained by increasing the frequency of the illuminating wave. In all the experiments, we consider a right preconditioned GMRES method (Saad and Schultz 1986) and the threshold for the stopping criterion is set to 10^{-3} on the normwise backward error $\frac{||r||}{||b||}$, where r denotes the residual and b the right-hand side of the linear system. This tolerance is

accurate for engineering purposes, as it enables the correct construction of the radar cross section of the object. The initial guess is the zero vector. All the runs have been performed in single precision on eight processors of a Compaq Alpha server. The Compaq Alpha server is a cluster of Symmetric Multi-Processors. Each node consists of four DEC Alpha processors (EV 6, 1.3 GFlops peak) that share 512 MB of memory. On that computer, the temporary disk space that can be used by the out-of-core solver is around 189 GB. Among all the possible right-hand sides for each geometry, we have selected those that are the most difficult to solve in order to better illustrate the robustness and the efficiency of our preconditioner.

Cetaf						
Size	Density FROB	Time FROB	GMRES(∞)		GMRES(120)	
			Iter	Time	Iter	Time
86256	0.18	4m	656	1h 25m	1546	1h 44m
134775	0.11	6m	618	1h 45m	1125	1h 55m
264156	0.06	13m	710	9h	1373	4h 46m
531900	0.03	20m	844	1d 18m	1717	14h 8m
1056636	0.01	37m	+750	+9h$^{(32)}$	+2000	> 1d
Aircraft						
Size	Density FROB	Time FROB	GMRES(∞)		GMRES(120)	
			Iter	Time	Iter	Time
94704	0.28	11m	746	2h 9m	1956	3h 13m
213084	0.13	31m	973	7h 19m	+2000	7h 56m
591900	0.09	1h 30m	1461	16h 42m$^{(64)}$	+2000	1d 57m
1160124	0.02	3h 24m	M.L.E.$^{(64)}$	N.A.	+2000	> 4d
Cobra						
Size	Density FROB	Time FROB	GMRES(∞)		GMRES(120)	
			Iter	Time	Iter	Time
60695	0.24	2m	369	26m	516	23m
179460	0.09	7m	353	1h 11h	406	1h 2m
Almond						
Size	Density FROB	Time FROB	GMRES(∞)		GMRES(120)	
			Iter	Time	Iter	Time
104793	0.19	6m	234	20m	253	17m
419172	0.05	21m	413	2h 44m	571	2h 26m
943137	0.02	49m	454	3h 35m$^{(32)}$	589	5h 55m

Table 2.1: Number of matrix-vector products and elapsed time required to converge on the four problems on 8 processors of the Compaq machine, except those marked with $^{(k)}$, that were run on k processors. Tolerance for the iterative solution was 10^{-3}. Acronyms: N.A. $\equiv$ not available. M.L.E. $\equiv$ memory limits exceeded

In Table 2.1, we show the numerical behaviour of the preconditioner observed on the four geometries when the size of the problems is increased. In this table

"d" means day, "h" hour and "m" minute. We see that the numerical behaviour of the preconditioner does not scale well with the size of the problems, especially GMRES(120) on the Cetaf, and there is no convergence on the largest aircraft problems. For GMRES(∞) the increase in the iteration count is less significant, even though on the Cetaf and the aircraft convergence cannot be obtained because we either exceed the memory limits of our computer or the time limit allocated to a single run. From a timing point of view, we see that the solution time of full GMRES is strongly affected by the orthogonalisation involved in the Arnoldi procedure. On the Cetaf problem discretized with 531900 points, the number of iterations of GMRES(120) is twice as large with respect to full GMRES, but GMRES(120) is about twice as cheap. Provided we get convergence, the use of a large restart often reduces the solution time even though it significantly deteriorates the convergence. On the Cetaf geometry, the solution time for the GMRES method increases superlinearly for small and medium problems, but nearly quadratically for large problems. On the largest test case, discretized with one million unknowns, unrestarted GMRES does not converge after 750 iterations requiring more than nine hours of computation on 32 processors. The Airbus aircraft is very difficult to solve because the mesh has many surface details and the discretization matrices become ill-conditioned. On small and medium problems, the number of GMRES iterations increases with the problem size, and the solution time increases superlinearly. On the largest test case, discretized with one million unknowns, full GMRES exceeds the memory limit on 64 processors. In this case, the use of large restarts (120 in this table) does not enable convergence within 2000 iterations except on a small mesh of size 94704.

Radius	Density	# mat-vec in GMRES(120)	Construction time	Solution time	Overall time
0.08	0.039	430	218	3562	3780
0.10	0.062	383	294	3008	3302
0.12	0.091	406	434	2954	3388
0.14	0.120	380	640	2873	3513
0.18	0.204	351	1863	3122	4985
0.24	0.358	280	4947	2583	7531

Table 2.2: Number of matrix-vector products and elapsed time to build the preconditioner and to solve the problem using GMRES(120) on a Cobra problem of size 179460, varying the parameters controlling the density of the preconditioner. The runs were performed on 8 processors of the Compaq machine.

We see in Table 2.1 that our strategy for adjusting the leaf box dimension for increasing frequencies causes the density of the sparse approximate inverse to decrease for increasing problem size. The number of unknowns per box remains

constant, but the number of boxes increases leading to a decrease in density and to poorer preconditioners. In Table 2.2, we investigate the influence of the density on the quality of the preconditioner on the Cobra using GMRES with a large restart (120). We adopt the same criterion described in Section 2 to define the sparsity patterns, but we increase the size of the leaf boxes in the oct-tree associated with the preconditioner. The best trade-off between cost and performance is obtained for a radius of around 0.12 wavelengths, the default value set in the code. If the preconditioner is used to solve systems with the same coefficient matrix and multiple right-hand sides, it might be worth computing more nonzeros if we have sufficient disk space, because the construction cost can be quickly amortized. However, significantly enlarging the density of the preconditioner is not feasible on the largest problems because we would exceed the memory and disk capacity of our computer.

Finally, in Table 2.3, we show the parallel scalability of the implementation of the FMM code (Sylvand 2002). We solve problems of increasing size on a larger number of processors, keeping the number of unknowns per processor constant. It can be seen that the construction of the preconditioner scales perfectly. This is a benefit from keeping the size of the leaf-box constant. Its use requires some communication but it still scales reasonably well. The scalability of the matrix-vector product is also satisfactory as the increase of the elapsed time is not only due to the amount of data exchanges but also to the $log(n)$ effect of its computational complexity.

Problem size	Nb procs	Construction time (sec)	Elapsed time precond (sec)	Elapsed time mat-vec (sec)
112908	8	513	0.39	1.77
161472	12	488	0.40	1.95
221952	16	497	0.43	2.15
288300	20	520	0.45	2.28
342732	24	523	0.47	3.10
393132	28	514	0.47	3.30
451632	32	509	0.48	2.80
674028	48	504	0.54	3.70
900912	64	514	0.60	3.80

Table 2.3: Tests on the parallel scalability of the code with respect to the construction and application of the preconditioner and to the matrix-vector product operation on problems of increasing size. The test example is the Airbus aircraft.

3 Improving the preconditioner robustness using embedded iterations

The numerical results shown in the previous section indicate that the Frobenius-norm minimization preconditioner tends to become less effective when the problem size increases, especially on difficult problems. By its nature the sparse approximate inverse is inherently local because each degree of freedom is coupled to only a very few neighbours. Because the exact inverse is dense the compact support used to define the preconditioner may not allow an exchange of global information and on large problems the lack of a global approximation may have a severe impact on the convergence. In our implementation, the overall number of computed nonzeros decreases for increasing values of the frequency. When the preconditioner becomes very sparse, information related to the far-field is completely lost. In this case, some suitable mechanism has to be introduced to recover global information for the numerical behaviour of the discrete Green's function.

Outer solver $\longrightarrow$ FGMRES, GMRES$\star$

Do k=1,2, ...

- M-V product: FMM with high accuracy
- Preconditioning : **Inner solver** (GMRES, TFQMR, ...)

  Do i=1,2, ...

  - M-V product: FMM with low accuracy
  - Preconditioning : M_{Frob}

  End Do

End Do

Figure 3.1: Inner-outer solution schemes in the FMM context. Sketch of the algorithm.

In this section, we describe an embedded iterative scheme, combined with multipole techniques, that is designed to meet the goals of robustness, scalability and parallelism of the iterative solver. The basic idea is to carry out a few steps of an inner Krylov method for the preconditioning operation. The overall algorithm results in the inner-outer scheme depicted in Figure 3.1. The outer solver must be able to work with variable preconditioners. Amongst various possibilities, we mention FGMRES (Saad 1993) and GMRES$\star$ (page 176 of Dongarra, Duff, Sorensen and van der Vorst, 1998 and page 91 of van der Vorst, 2003); this latter reduces to GMRESR (van der Vorst and Vuik 1994) when the inner solver is GMRES.

The efficiency of the proposed algorithm relies on two main factors: the inner solver has to be preconditioned so that the residual in the inner iterations can be significantly reduced in a few steps, and the matrix-vector products within the inner and the outer solvers can be carried out with a different accuracy. The motivation that naturally leads us to consider inner-outer schemes is to try to balance the locality of the preconditioner with the use of the multipole matrix. Experiments conducted by Grama, Kumar and Sameh (1999) with inner-outer schemes combined with multipole techniques on the potential equation were not entirely successful. In that case, no preconditioner was used in the inner solver. The desirable feature of using different accuracies for the matrix-vector products is enabled by the use of the FMM. In our scheme, a highly accurate FMM is used within the outer solver as it governs the final accuracy of the computed solution. A less accurate FMM is used within the inner solver as it is a preconditioner for the outer scheme; in this respect the inner iterations only attempt to give a rough approximation of the solution and consequently do not require an accurate matrix-vector calculation. In fact, we solve a nearby system for the preconditioning operation, that enables us to save considerable computational effort during the iterative process as the less accurate matrix-vector calculation requires about half of the computing time of the accurate one. In Table 3.1, we show the average elapsed time in seconds observed on 8 processors for a matrix-vector product using the FMM with different accuracy levels.

Geometry	inner FMM	outer FMM
Airbus 94704	3.5	4.8
Airbus 213084	6.9	11.8
Airbus 591900	17.4	30.2
Airbus 1160124	34.5	66.5
Cetaf 86256	1.9	3.7
Cetaf 134775	3.2	5.2
Cetaf 264159	5.6	11.4
Cetaf 539100	11.5	19.3
Cetaf 1056636	20.6	38.2
Almond 104793	1.8	3.0
Almond 419172	6.2	11.2
Almond 943137	14.3	25.5
Cobra 60695	1.1	2.0
Cobra 179460	3.2	5.6

Table 3.1: Average elapsed time observed on 8 processors in seconds for a matrix-vector product using the FMM with different levels of accuracy.

3.1 Numerical results

In this section, we conduct experiments using the FGMRES method as the outer solver with an inner GMRES iteration preconditioned with the Frobenius-norm minimization method described in Section 2. It is possible to vary the accuracy of the inner solve as the solution converges. However, in this paper, we present preconditioning results for a constant number of inner GMRES iterations. For the inner scheme we do not restart and only perform a prescribed number of full GMRES iterations or equivalently one step of restarted GMRES with the same prescribed restart. For the GMRES and FGMRES methods, we consider the implementations described by Frayssé, Giraud, Gratton and Langou (2003) and Frayssé, Giraud and Gratton (1998), respectively. The convergence history of GMRES depicted in Figure 3.2 for different values of the restart gives us some clues as to the numerical behaviour of the proposed scheme. The residual of GMRES tends to decrease very rapidly in the first few iterations independently of the restarts, then decreases much more slowly, and finally tends to stagnate to a value that depends on the restart; the larger the restart, the lower the stagnation value. It suggests that a few steps in the inner solver could be very effective for obtaining a significant reduction of the initial residual. A different convergence behaviour was observed when using other Krylov methods as an inner solver, for example the TFQMR solver (Carpentieri 2002). In this case, the residual at the beginning of the convergence is nearly constant or decreases very slowly. The use of TFQMR as an inner solver is ineffective.

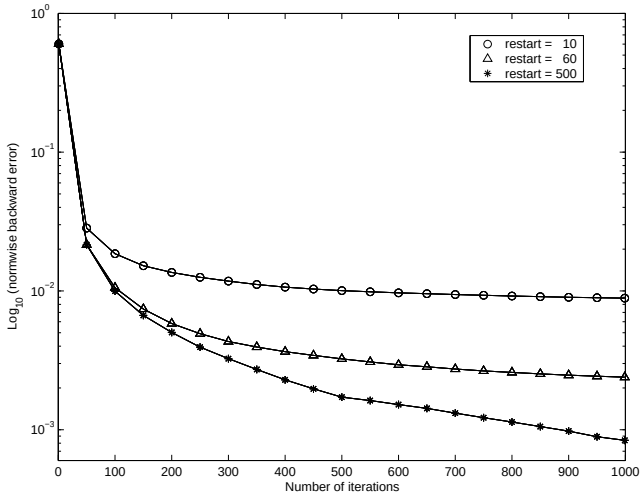


Figure 3.2: Convergence history of GMRES for different values of restarts on an Airbus aircraft discretized with 94704 points.

In Table 3.2, we describe the results of experiments on an Airbus aircraft with 213084 points using different combinations of restarts for the inner and outer solvers.

If the number of inner steps is too small, the preconditioning operation is poor and the convergence slows down, while too large restarts of GMRES tend to increase the overall computational cost but do not cause further reduction of the normwise backward error at the beginning of convergence. The choice of the restart for the outer solver depends to a large extent on the available memory of the target machine and the difficulty of the problem at hand, an issue that is also related to the illuminating direction of the incident wave that defines the right-hand side. Amongst the various possibilities, we select FGMRES(30) and GMRES(60) on the Airbus aircraft and the Cetaf problem, and FGMRES(15) and GMRES(30) on the Cobra and the Almond. These combinations are a good trade-off.

restart FGMRES	restart GMRES	# total inner mat-vec	# total outer mat-vec	times
30	40	1960	51	4h 58m
30	50	1900	40	4h 53m
30	60	1920	34	4h 55m
30	70	2030	30	5h 13m
30	80	2240	29	5h 50m

Table 3.2: Global elapsed time and total number of matrix-vector products required to converge on an Airbus aircraft with 213084 points varying the inner restart parameter. The runs have been performed on 8 processors of the Compaq machine.

In Table 3.3 we show the results of experiments on the same geometries that we considered in Section 2. We show the number of inner and outer matrix-vector products and the elapsed time needed to achieve convergence using a tolerance of 10^{-3} on eight processors of the Compaq machine. For comparison purposes, we show again in this table the results obtained with the restarted GMRES method. The comparison between the two solvers made in the tables is fair because GMRES has exactly the same storage requirements as the combination FGMRES/GMRES. In fact, for the same restart value, the storage requirements for the FGMRES algorithm are twice that for the standard GMRES algorithm, because it also stores the preconditioned vectors of the Krylov basis. It can be seen that the combination FGMRES/GMRES remarkably enhances the robustness of the preconditioner especially on large problems. The increase in the number of outer iterations is fairly modest except on the largest aircraft test cases. Nevertheless, the scheme is the only one that enables us to get the solution of this challenging problem since classical restarted GMRES does not converge and full GMRES exceeds the memory of our computer. Similarly, on the Cetaf discretized with one million points, the embedded scheme enables us to get convergence in 22 outer iterations, whereas GMRES(120) does not converge in 2000 iterations. The saving in time is also noticeable. The gain ranges from two to four depending on the geometry and

tends to become larger when the problem size increases. On the Cobra and the Almond test case, the embedded solver not only reduces the solution time but also the memory used as FGMRES(15)/GMRES(30) is faster than GMRES(120) (see Table 2.1).

Cetaf						
Size	Density FROB	Time FROB	GMRES(120)		FGMRES(30)/GMRES(60)	
			Iter	Time	Iter	Time
86256	0.18	4m	1546	1h 43m	17+ 960	55m
134775	0.11	6m	1125	1h 55m	15+ 840	1h 19m
264156	0.06	13m	1373	4h 46m	17+ 960	2h 22m
531900	0.03	20m	1717	14h 8m	19+1080	6h
1056636	0.01	37m	+2000	> 1d	22+1260	14h
Aircraft						
Size	Density FROB	Time FROB	GMRES(120)		FGMRES(30)/GMRES(60)	
			Iter	Time	Iter	Time
94704	0.28	11m	1956	3h 13m	27+1560	2h 14m
213084	0.13	31m	+2000	7h 56m	34+1920	5h
591900	0.09	1h 30m	+2000	1d 57m	57+3300	1d 9h 45m
1160124	0.02	3h 24m	+2000	> 4d	51+2940	16h 41m$^{(64)}$
Cobra						
Size	Density FROB	Time FROB	GMRES(60)		FGMRES(15)/GMRES(30)	
			Iter	Time	Iter	Time
60695	0.24	2m	708	29m	24+660	18m
179460	0.09	7m	433	48m	20+540	42m
Almond						
Size	Density FROB	Time FROB	GMRES(60)		FGMRES(15)/GMRES(30)	
			Iter	Time	Iter	Time
104793	0.19	6m	302	19m	11+300	14m
419172	0.05	21m	+2000	> 10h	20+540	1h 24m
943137	0.02	49m	1633	6h 36m$^{(16)}$	22+600	3h 32m

Table 3.3: Number of matrix-vector products and elapsed time required to converge on 8 processors of the Compaq machine. The tests were run on 8 processors of the Compaq machine, except those marked with $^{(k)}$, that were run on k processors.

Similarly to what has been observed with GMRES, using a full FGMRES in the outer loop enables us to reduce the number of outer iterations but often results in a longer solution time. In Table 3.4 we show results obtained with full FGMRES and compare this with the full GMRES method. It can also be seen that the saving in time due to the use of the inner-outer scheme with respect to GMRES is also strongly related to the reduction in the number of dot product calculations since there are far fewer basis vectors to be orthogonalized. This gain is particularly visible on the Airbus test example with 213084 unknowns. The time spent in the

FMM is longer for the FGMRES/GMRES solver, about 3 hours and 47 minutes (i.e. $33 \times 11.8 + 1920 \times 6.9$ see Table 3.4 and 3.1), and it is about 3 hours and 11 minutes for GMRES. On that example, the embedded scheme is about 2 hours faster than regular GMRES. In this latter case, we have to orthogonalize up to 973 basis vectors while in the inner-outer scheme we do not have to orthogonalize a basis larger than 60. This is another illustration that the orthogonalisation procedure is extremely time consuming.

Aircraft				
Size	GMRES(∞)		FGMRES(∞,60)	
	Iter	Time	Iter	Time
94704	746	2h 9m	23+1320	2h 30m
213084	973	7h 19m	30+1740	6h 11m
591900	1461	16h 42m$^{(64)}$	43+2520	12h*
1160124	M.L.E.$^{(64)}$	$>$ 1d	43+2520	14 h 28m**
Cobra				
Size	GMRES(∞)		FGMRES(∞,60)	
	Iter	Time	Iter	Time
60695	369	26m	21+600	17m
179460	353	1h 11m	18+510	38m

Table 3.4: Number of matrix-vector products and elapsed time required to converge on 8 processors of the Compaq machine. The tests were run on 8 processors of the Compaq machine, except those marked with $^{(k)}$, that were run on k processors.

Finally, in Figure 3.3 we show the typical convergence history of the FGMRES/GMRES and GMRES solvers on a large Cetaf problem discretized with 264156 points. We depict the convergence curve as a function of the elapsed time. It can be seen that the embedded solver clearly outperforms the single GMRES scheme. The FGMRES/GMRES solver continues to succeed in reducing the residual norm while GMRES converges slowly. Similar behaviour is also reported for simpler model examples by Simoncini and Szyld (2003).

4 Conclusions

In this paper we have described an effective and inherently parallel approximate inverse preconditioner based on a Frobenius-norm minimization preconditioner, that can easily be implemented in a FMM code. We have studied the numerical scalability of our method for the solution of very large and dense linear systems of

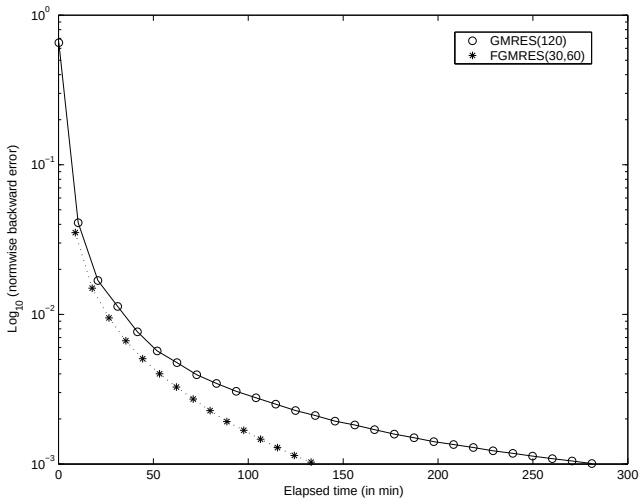


Figure 3.3: Convergence history as a function of elapsed time of GMRES versus FGMRES on the Cetaf problem discretized with 264156 points.

equations arising from real life applications in electromagnetism. The locality of the preconditioner on large problems can be significantly improved by embedding it within inner-outer solvers. We have described an embedded iterative scheme based on the GMRES method, implemented in a multipole context with different levels of accuracy for the matrix-vector products in the inner and outer loops. We have shown that the combination FGMRES/GMRES can effectively enhance the robustness of the preconditioner and reduce significantly the computational cost and the storage requirements for the solution of large problems. One could apply this idea recursively and embed several FGMRES schemes with decreasing FMM accuracy down to the lowest accuracy in the innermost GMRES. However, in our work, we only consider a two-level scheme. Most of the experiments shown in this paper require a huge amount of computation and storage, and they often reach the limits of our target machine in terms of memory and disk storage. To give an idea of how large these simulations are, the solution of systems with one million unknowns using a direct method would require 8 Tbytes of storage and 37 years of computation on one processor of the target computer (assuming the computation runs at peak performance). Such simulations are nowadays feasible thanks to the use of iterative methods and can be integrated in the design processes where the bottleneck moves from the simulation to the pre and post-processing of the results as the tools are not yet available to easily manipulate meshes with millions of degrees of freedom.

Acknowledgements

We would like to thank Guillaume Alléon, from EADS-CCR, for providing us some test examples considered in this paper.

References

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

Benson, M. W. (1973), Iterative solution of large scale linear systems., Master's thesis, Lakehead University, Thunder Bay, Canada.

Carpentieri, B. (2002), Sparse preconditioners for dense complex linear systems in electromagnetic applications, Ph.D. dissertation, INPT. TH/PA/02/48.

Carpentieri, B., Duff, I. S. and Giraud, L. (2000), 'Sparse pattern selection strategies for robust Frobenius-norm minimization preconditioners in electromagnetism', *Numerical Linear Algebra with Applications* **7**(7-8), 667–685.

Carpentieri, B., Duff, I. S., Giraud, L. and Magolu monga Made, M. (2002), Sparse symmetric preconditioners for dense linear systems in electromagnetism, Technical Report RAL-TR-2002-016, Rutherford Appleton Laboratory, Oxfordshire, England. To appear in *Numerical Linear Algebra with Applications*.

Chen, K. (2001), 'An analysis of sparse approximate inverse preconditioners for boundary integral equations', *SIAM J. Matrix Analysis and Applications* **22**(4), 1058–1078.

Chow, E. and Saad, Y. (1998), 'Approximate inverse preconditioners via sparse-sparse iterations', *SIAM J. Scientific Computing* **19**(3), 995–1023.

Darve, E. (2000*a*), 'The fast multipole method (i) : Error analysis and asymptotic complexity', *SIAM J. Numerical Analysis* **38**(1), 98–128.

Darve, E. (2000*b*), 'The fast multipole method: Numerical implementation', *J. Comp. Phys.* **160**(1), 195–240.

de Boor, C. (1980), 'Dichotomies for band matrices', *SIAM J. Numerical Analysis* **17**, 894–907.

Demko, S. (1977), 'Inverses of band matrices and local convergence of spline projections', *SIAM J. Numerical Analysis* **14**, 616–619.

Demko, S., Moss, W. and Smith, P. (1984), 'Decay rates for inverses of band matrices.', *Mathematics of Computation* **43**, 491–499.

Dongarra, J. J., Duff, I. S., Sorensen, D. C. and van der Vorst, H. A. (1998), *Numerical Linear Algebra for High-Performance Computers*, SIAM Press, Philadelphia.

Frayssé, V., Giraud, L. and Gratton, S. (1998), A set of Flexible-GMRES routines for real and complex arithmetics, Technical Report TR/PA/98/20, CERFACS, Toulouse, France.

Frayssé, V., Giraud, L., Gratton, S. and Langou, J. (2003), A set of GMRES routines for real and complex arithmetics on high performance computers, Technical Report TR/PA/03/3, CERFACS, Toulouse, France.

Frederickson, P. O. (1975), Fast approximate inversion of large sparse linear systems., Math. Report 7, Lakehead University, Thunder Bay, Canada.

Grama, A., Kumar, V. and Sameh, A. (1995), Parallel matrix-vector product using approximate hierarchical methods, *in* S. Karin, ed., 'Proceedings of the 1995 ACM/IEEE Supercomputing Conference, December 3–8, 1995, San Diego Convention Center, San Diego, CA, USA', ACM Press and IEEE Computer Society Press, New York, NY, USA.

Grama, A., Kumar, V. and Sameh, A. (1999), 'Parallel hierarchical solvers and preconditioners for boundary element methods', *SIAM J. Scientific Computing* **20**(1), 337–358.

Greengard, L. and Gropp, W. (1990), 'A parallel version of the fast multipole method', *Comput. Math. Appl.* **20**, 63–71.

Greengard, L. and Rokhlin, V. (1987), 'A fast algorithm for particle simulations', *Journal of Computational Physics* **73**, 325–348.

Grote, M. J. and Huckle, T. (1997), 'Parallel preconditioning with sparse approximate inverses', *SIAM J. Scientific Computing* **18**(3), 838–853.

Kolotilina, L. Y. (1988), 'Explicit preconditioning of systems of linear algebraic equations with dense matrices.', *J. Sov. Math.* **43**, 2566–2573. English translation of a paper first published in Zapisli Nauchnykh Seminarov Leningradskogo Otdeleniya Matematicheskogo im. V.A. Steklova AN SSSR 154 (1986) 90-100.

Lee, J., Lu, C.-C. and Zhang, J. (2002), Sparse inverse preconditioning of multilevel fast multipole algorithm for hybrid integral equations in electromagnetics, Technical Report 363-02, Department of Computer Science, University of Kentucky, KY.

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

Saad, Y. (1993), 'A flexible inner-outer preconditioned GMRES algorithm', *SIAM J. Scientific and Statistical Computing* **14**, 461–469.

Saad, Y. and Schultz, M. H. (1986), 'GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems.', *SIAM J. Scientific and Statistical Computing* **7**, 856–869.

Samant, A., Michielssen, E. and Saylor, P. (1996), Approximate inverse based preconditioners for 2d dense matrix problems, Technical Report CCEM-11-96, University of Illinois.

Sertel, K. and Volakis, J. L. (2000), 'Incomplete LU preconditioner for FMM implementation', *Microwave and Optical Technology Letters* **26(7)**, 265–267.

Simoncini, V. and Szyld, D. B. (2003), 'Flexible inner-outer Krylov subspace methods', *SIAM J. Matrix Analysis and Applications* **40**(6), 2219–2239.

Song, J., Lu, C.-C. and Chew, W. C. (1997), 'Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects', *IEEE Transactions on Antennas and Propagation* **45**(10), 1488–1493.

Sylvand, G. (2002), Résolution Itérative de Formulation Intégrale pour Helmholtz 3D : Applications de la Méthode Multipôle à des Problèmes de Grande Taille, PhD thesis, Ecole Nationale des Ponts et Chaussées.

Tang, W.-P. (1987), Schwartz splitting and template operators, PhD thesis, Computer Science Dept., Stanford University, Stanford, CA.

van der Vorst, H. A. (2003), *Iterative Krylov Methods for Large Linear systems*, Cambridge University Press.

van der Vorst, H. A. and Vuik, C. (1994), 'GMRESR: A family of nested GMRES methods', *Numerical Linear Algebra with Applications* **1**, 369–386.

Vavasis, S. A. (1992), 'Preconditioning for boundary integral equations', *SIAM J. Matrix Analysis and Applications* **13**, 905–925.

Zhao, F. and Johnsson, S. L. (1991), 'The parallel multipole method on the connection machine', *SIAM J. Scientific and Statistical Computing* **12**, 1420–1437.