Multilevel Adaptive Cross Approximation (MLACA)

José M. Tamayo, Alexander Heldring, and Juan M. Rius

***Abstract*—The Multilevel Adaptive Cross Approximation (MLACA) is proposed as a fast method to accelerate the matrix-vector products in the iterative solution of the linear system that results from the discretization of electromagnetic Integral Equations (IE) with the Method of Moments (MoM). The single level ACA, already described in the literature, is extended with a multilevel recursive algorithm in order to improve the asymptotical complexity of both the computational cost and the memory requirements. The main qualities of ACA are maintained: it is purely algebraic and independent of the integral equation kernel Green's function as long as it produces pseudo-rank-deficient matrix blocks. The algorithm is presented in such a way that it can be easily implemented on top of an existing MoM code with most commonly used Green's functions.**

***Index Terms*—Adaptive cross approximation, fast integral equation methods, impedance matrix compression, method of moments, numerical simulation.**

I. Introduction

In recent years, a wide range of fast methods [1] have been developed for accelerating the iterative solution of the electromagnetic integral equations [2] discretized by method of moments (MoM) [3]. Most of them are based on multilevel subdomain decomposition and require a computational cost per iteration of order $N \log N$ or $N \log^2 N$. One of these methods is the multilevel fast multipole algorithm (MLFMA) [4]. The MLFMA has been widely used in the last years to solve very large electromagnetic problems [5], [6] due to its excellent computational efficiency.

The main drawback of the MLFMA is the dependence of its formulation on the problem Green's function. Notwithstanding, other general purpose methods have been developed. For instance, the multilevel matrix decomposition algorithm (MLMDA) [7]–[9] or its renewed version, the matrix decomposition algorithm - singular value decomposition (MDA-SVD) [10] exploit the compressibility of MoM submatrices corresponding to well separated subscatterers by using equivalent surface basis/testing functions cleverly distributed that radiate and receive the same fields as the original basis/testing functions. Another fast solver, the adaptive integral method (AIM) [11] replaces the actual basis and testing functions with a regular volumetrical grid, that again radiates and receives the same fields as the original discretization, in order to efficiently compute the integral equations convolution using the FFT algorithm.

However, all the aforementioned methods rely on the appropriate selection of elements with an equivalent physical behavior, either multipoles or equivalent surface or volume basis/testing functions. Hence the interest of purely algebraic methods, whose formulation is still independent of the problem Green's function and operate solely with some of the impedance matrix elements, such as the IE-QR Algorithm [12], the IES3 [13], [14] and the adaptive cross approximation (ACA) [15]. Unfortunately, these algebraic methods present an asymptotic computational time and memory requirement not as good as that of the above-mentioned methods.

The ACA, which is the basis of the multilevel algorithm proposed here, was developed in 2000 by Bebendorf [15] and since then has been widely used to solve large magnetostatic problems [16]. Recently, the ACA has been applied also to antenna radiation and RCS computation [17], [18]. For very large problems, the compression of well separated subscatterers matrix blocks with ACA has an asymptotical computational complexity of $O(\bar{N}^3)$ for matrix compression and $O(\bar{N}^2)$ for storage, as proved in [19], where $\bar{N}$ is the number of unknowns of each subscatterer. In practice, these asymptotical behaviors are not reached unless $\bar{N}$ is of the order of several millions, and therefore for moderately large values of $\bar{N}$ one usually observes a much lower computational complexity.

The aim of this paper is to present a multilevel version of the ACA method, first introduced by the authors in [20] at a preliminary state, using physical concepts similar to those presented in [7] for the MLMDA. Maintaining the purely algebraic formulation and in comparison with the single level ACA, the computational cost and memory requirements are reduced to $O(\bar{N}^2)$ and $O(\bar{N} \log \bar{N})$, respectively. The memory, and therefore the matrix-vector product time, are comparable to that of the MLFMA.

II. Description of the Algorithm

A. Tree Domain Decomposition Based on the Solid Angle

Hereupon and up to Section V we address the interaction between two boxes or sets of samples contained within two spheres that do not intersect each other. These boxes come from a domain decomposition of the object that will be introduced in Section V. We shall call the two separated boxes, source and observation boxes, respectively.

The degrees of freedom (DoF) of this far-interaction basically depend on the solid angle, or the angular portion, that one of the

Manuscript received July 01, 2010; revised March 09, 2011; accepted June 02, 2011. Date of publication August 18, 2011; date of current version December 02, 2011. This work was supported in part by the Spanish Interministerial Commission on Science and Technology (CICYT) under projects TEC2009-13897-C03-01, TEC2009-13897-C03-02 and TEC2010-20841-C04-02 and CONSOLIDER CSD2008-00068 and in part by the Ministerio de Educación y Ciencia through the FPU fellowship program.

The authors are with the AntennaLab, Department of Signal Processing and Telecommunications, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain (e-mail: jose.maria.tamayo@tsc.upc.edu; heldring@tsc.upc.edu; rius@tsc.upc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAP.2011.2165476

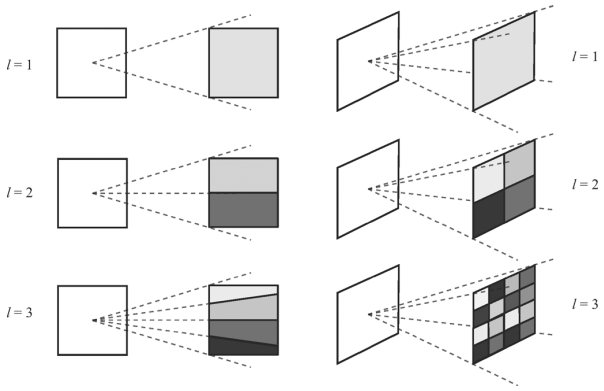

Fig. 1. Representation of the subdivision of the blocks of samples based on the solid angle seen from the other block. Coplanar on the left and perpendicular on the right.

boxes occupies from the view point of the other box, i.e., the solid angle when the coordinates origin is placed at the center of the other box. One box is subdivided into children boxes having all of them the same solid angle and therefore approximately the same number of DoF in their interaction with the other box. Two examples of this sort of subdivision are shown in Fig. 1. This subdivision must be applied to both the source and observation boxes placing the origin at the center of the other.

The use of a decomposition based on the solid angle rather than just on the electrical size of the boxes is crucial to have a real gain in computational complexity and memory already for moderately electrically large problems.

In the matrix representations throughout the article, the basis and testing functions will be reordered following the tree decomposition described, in order to obtain block-like representation. It means that functions in the same box of the tree are represented by contiguous rows/columns of the matrix. Furthermore, although the tree decomposition is not necessarily binary (see Fig. 1 on the right), it has been considered binary without loss of generality in the description of the algorithm.

B. Adaptive Cross Approximation-Singular Value Decomposition (ACA-SVD)

The ACA [15] is a purely algebraic technique for matrix compression. It accurately decomposes a low τ-rank matrix arising from an asymptotically smooth function into the product of two new matrices in a compressed form

$$M \approx UV \tag{1}$$

being M a $m \times n$ matrix, U a $m \times k$ matrix and V a $k \times n$ matrix with $k \ll m, n$, where k is the τ-rank or number of DoF. The error in this approximation is controlled by a threshold τ which determines when to stop looking for more columns and rows of U and V, respectively.

Most of the open Green's functions in electromagnetism lead to an integral equation function (EFIE, MFIE, CFIE, ...) which, when dealing with the interaction between two well-separated blocks in space, lead in turn to an interaction matrix with a deficient pseudo-rank that can be efficiently compressed with the ACA technique.

A SVD recompression of the obtained ACA decomposition can be performed utilizing two QR factorizations [21], so that either the U or V matrices can be done orthonormal, with an extra saving in memory. We have chosen the ACA threshold ten times lower than the SVD recompression threshold τ.

The ACA-SVD has a computational cost or number of operations scaling with $k^2(m+n)$ whereas the final amount of necessary memory scales with $k(m+n)$. The last quantity coincides also with the number of matrix elements evaluations in M to decompose $M \approx UV$ allowing us not to compute the whole matrix. Although this number of evaluations is asymptotically smaller than the computational cost of the ACA, for medium size problems it can dominate the total computational time as Green's function evaluations can be highly time consuming.

C. Degrees of Freedom (DoF)

As stated in [7], a far interaction not only have a reduced number of DoF, fact exploited by the ACA, but also the DoF accomplish the following two properties, necessary for the MLACA. The number of DoF remains asymptotically constant for large boxes when the two interacting boxes are inversely changed in size, fixing everything else. The DoF of the interaction between a box at the finest level in one object, source or observation, and the whole other object remains also constant if the finest level is kept at a certain fixed electrical size, easily done varying the actual number of levels of the tree subdivision.

D. Multilevel Adaptive Cross Approximation (MLACA)

Applying the method of moments (MoM) it is possible to express the two boxes far-interaction with an impedance matrix Z_{mn} with dimensions $m \times n$, where m and n are the number of basis functions in which the observation and source boxes have been discretized, respectively. In the first place, we must subdivide each box recursively into smaller domains utilizing the tree decomposition described in Section II-A. If L is the number of levels, at the finest level we have in each subset or subdomain approximately $M = \bar{N}/2^L$ elements, considering $\bar{N} = n = m$.

In step 0 [Fig. 2(a)] Z_{mn} is transformed into the product of two new matrices $A_0^{(1)}$ and $B_0^{(1)}$. The procedure to obtain them is to split Z_{mn} into strips, each corresponding to the interaction of one subset of basis functions at the finest level in the source box with the whole observation box (see first row of pictures in Fig. 3). Each of those strips only has k DoF. They can be compressed with the ACA-SVD algorithm and regrouped [see Fig. 2(a)] concatenating horizontally the left-hand side matrices forming the new matrix $A_0^{(1)}$ and diagonally the right-hand side matrices forming the new block-diagonal matrix $B_0^{(1)}$. In that way, the product of the two new matrices gives us an approximation of the original Z_{mn} matrix. Then only the "B" matrices need to be permanently stored and the "A" matrices will be subdivided and recompressed in the next steps of the algorithm.

At step 1 ($i = 1$), the matrix $A_0^{(1)}$ from level 0 is transformed into four new matrices $A_0^{(2)}$, $A_1^{(2)}$, $B_0^{(2)}$, $B_1^{(2)}$ [Fig. 2(b)] as follows. Due to the proper reordering of columns according to the tree decomposition of the source box, consecutive couples of strips in matrix $A_0^{(1)}$ correspond to the children of each box

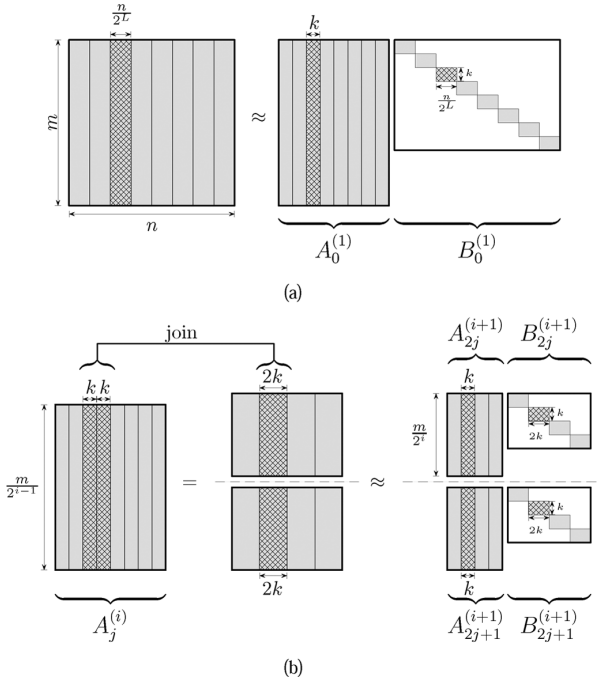

Fig. 2. Graphical representation of the matrix transformations in the MLACA algorithm described in Section II. First step or step 0 (a) and step i (b) with $i \geq 1$.

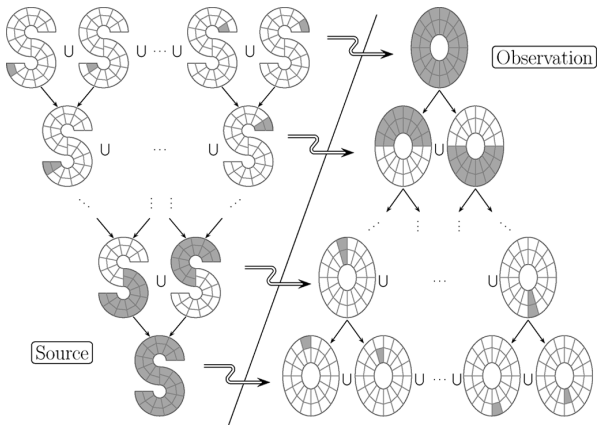

Fig. 3. Schematic representation of the described multilevel procedure. Each row of figures represents each step on the algorithm. Gray blocks are the ones considered at each step. It starts with the interaction between all the finest level boxes of the source box "S" and the total observation box "O." In the next steps, the source blocks are sequentially grouped whilst the observation boxes are split.

in the next decomposition level. Similarly, due to the proper reordering of rows according to the tree decomposition of the observation box, the matrix $A_0^{(1)}$ can be horizontally split obtaining that each subblock corresponds to each one of the two halves of the observation box (second row of pictures in Fig. 3). According to Section II-C, the new strips have again the same k DoF. As the actual number of columns of the strips is $2k$, they can be recompressed with the ACA-SVD and regrouped for each matrix in the same manner as in step 0. The obtained new matrices $A_0^{(2)}$ and $A_1^{(2)}$ are sent to the next step in order to equally proceed recursively.

Generalizing this procedure in a recursive manner, at step i, the set of "A" matrices from step $i-1$, $A_j^{(i)}$ $j = 0, \ldots, 2^{i-1}-1$,

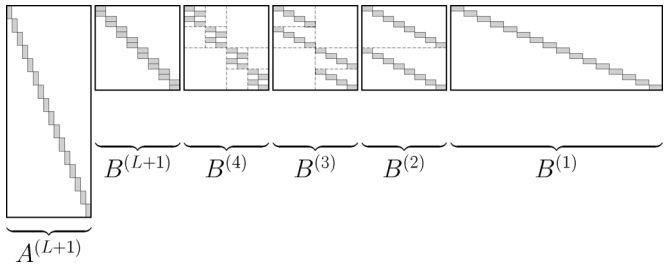

Fig. 4. Multilevel decomposition matrices for $L = 4$ levels. White parts are zero elements whereas gray blocks are filled.

are transformed into four new matrices $A_{2j}^{(i+1)}$, $A_{2j+1}^{(i+1)}$, $B_{2j}^{(i+1)}$, $B_{2j+1}^{(i+1)}$ [Fig. 2(b)], where we join the contributions of the source box in the next level of the tree decomposition while splitting the observation object in the boxes from the previous level (each row of Fig. 3 represents a step of the algorithm). Each interaction has again k DoF, and therefore is recompressible. After the step $i = L$ we have the whole set of matrices which correctly combined represent a compressed version of the matrix Z_{mn}. Only the "B" matrices and the "$A^{(L+1)}$" from the last step need to be permanently stored.

In this procedure, all the "B" matrices are orthonormal to maintain the information (singular values) of each interaction in the "A" matrices, which are the ones to recompress at each step.

In order to have our original matrix Z_{mn} decomposed as a product of very sparse matrices (see Fig. 4) we need to regroup the obtained matrices as follows:

$$B^{(1)} = B_0^{(1)}. \tag{2}$$

For each $i = 2, \ldots, L+1$:

$$B^{(i)} = \begin{bmatrix} B_0^{(i)} & & & & & \\ B_1^{(i)} & & & & & \\ & \ddots & & & & \\ & & B_{2j}^{(i)} & & & \\ & & B_{2j+1}^{(i)} & & & \\ & & & \ddots & & \\ & & & & B_{2^{i-1}-2}^{(i)} & \\ & & & & B_{2^{i-1}-1}^{(i)} & \end{bmatrix} \tag{3}$$

$$A^{(L+1)} = \begin{bmatrix} A_0^{(L+1)} & & \\ & \ddots & \\ & & A_{2^L-1}^{(L+1)} \end{bmatrix}. \tag{4}$$

The previous procedure has a computational cost of $O(\bar{N}^2)$ and a final storage of $O(\bar{N}\log\bar{N})$ (see Section III). However, during the algorithm the amount of necessary memory at some steps can be temporarily of $O(\bar{N}^2)$ (for instance, at step 0, the matrix $A_0^{(1)}$ has $O(\bar{N}^2)$ elements). Nevertheless, this temporary storage can be reduced by simply reordering the operations. It means that we do exactly the same operations and the result is exactly the same but the order of the operations is different.

A representation of that reordering is shown in Fig. 5 for two levels. Each row in the figure represents all the matrix blocks

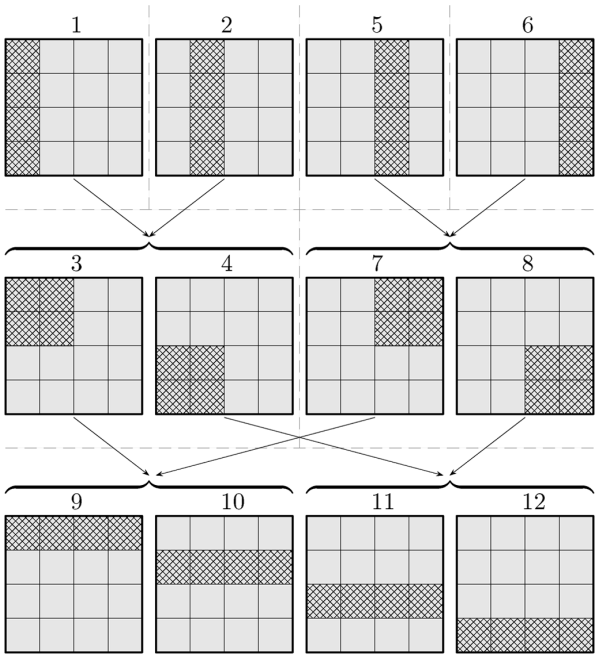

Fig. 5. Representation of the order to follow (numbers) showing schematically which are the elements of the matrix involved. The arrows display which steps are necessary from the previous row in order to compute the matrices below. In this case there are $L = 2$ levels.

involved at each step of the algorithm. The numbers above each matrix determine the order in which the operations are computed. It can be understood as an under demand algorithm, in the sense that each computation is only done when it is necessary, starting from the final goal, i.e., the interaction of the complete source box with each finest level observation box.

III. Computational Cost and Storage

We remind that this section deals with time and storage of the MLACA algorithm applied to the interaction between two separated sets of samples and not to the global algorithm applied to the entire problem under study.

Here k is the constant number of DoF of each submatrix interaction inside the multilevel procedure defined in Section II-D.

The final storage, which coincides with the computational cost of a matrix-vector product, corresponds with the whole set of B matrices and the $A^{(L+1)}$ matrices (see Fig. 4). The matrix $B^{(1)}$ contains 2^L small block-matrices each one with dimensions $k \times (n)/(2^L)$. The L matrices $B^{(2)} \dots B^{(L+1)}$ contain each one 2^L submatrices with dimensions $k \times 2k$. Finally, the matrix $A^{(L+1)}$ contains 2^L submatrices with dimensions $(m)/(2^L) \times k$. Therefore, the final storage is:

$$S_{\text{END}} = 2^L k \frac{n}{2^L} + L 2^L 2k^2 + 2^L \frac{m}{2^L} k \tag{5}$$

$$= 2k\bar{N} + 2k^2 \frac{\bar{N}}{M} \log \frac{\bar{N}}{M} = O(\bar{N} \log \bar{N}) \tag{6}$$

where it has been used that $\bar{N} = m = n$ and $L = \log(\bar{N}/M)$ and that k and M are independent of $\bar{N}$. So finally, the total necessary amount of memory to store the matrix and the computational cost of a matrix-vector product is $O(\bar{N} \log \bar{N})$.

Similarly, the total temporary necessary memory at the worst part of the algorithm can be proved to equal the following expression, development omitted for the sake of brevity and readability

$$S_{\text{TOTAL}} = 3k\bar{N} + \left(k + 2\frac{k^2}{M}\right) \bar{N} \log \frac{\bar{N}}{M} = O(\bar{N} \log \bar{N}). \tag{7}$$

Only remains to obtain the computational cost of the construction of the compressed matrix. At step 0 of the algorithm we need to perform 2^L ACA decompositions of matrices with k DoF and dimensions $m \times (n)/(2^L)$. Furthermore, at each step i with $i > 0$ we need to compute 2^L ACA decompositions of matrices with again k DoF but with dimensions $(m)/(2^i) \times 2k$. Therefore, the total computational cost becomes

$$CC_{\text{TOTAL}} = 2^L k^2 \left(m + \frac{n}{2^L}\right) + \sum_{i=1}^{L} 2^L k^2 \left(\frac{m}{2^i} + 2k\right) \tag{8}$$

$$= \frac{2k^2}{M} \bar{N}^2 + 2k^3 \frac{\bar{N}}{M} \log \frac{\bar{N}}{M} = O(\bar{N}^2). \tag{9}$$

Summarizing, the computational cost of computing the matrix is $O(\bar{N}^2)$, and the computational cost of a matrix-vector product as well as the storage is $O(\bar{N} \log \bar{N})$.

IV. SVD Threshold

Similarly to what happens with real numbers, if we have a product of matrices and we want to assure a certain relative error in the resulting matrix, we have to consider the worst case where all the relative errors of each matrix are summed. So, to have a better error than with the direct application of the ACA-SVD (equivalent to the MLACA with $L = 0$), we need to use a threshold $\tau/(L+2)$ instead of τ because the matrix is decomposed into the product of $L+2$ matrices. The influence of that change of threshold in the computational costs and storage is analyzed below.

The only thing which is affected by the threshold is the number of DoF k at each level that now depends on L and therefore on $\bar{N}$. Let us consider that the singular values decay exponentially up to some point

$$\sigma_t = s_0 10^{-\beta t}, \quad \beta > 0, t \geq t_0. \tag{10}$$

Then we must accomplish

$$\frac{\sigma_{k(\frac{\tau}{L+2})}}{\sigma_0} \leq \frac{\tau}{L+2} \tag{11}$$

and substituting (10) for $t = k((\tau)/(L+2))$ and isolating k we finally obtain

$$k\left(\frac{\tau}{L+2}\right) = \left\lceil \frac{1}{\beta} \left[\log_{10}(L+2) + \log_{10}\left(\frac{s_0}{\tau \sigma_0}\right) \right] \right\rceil \tag{12}$$

$$= O\left(\log_{10}\left(\log \frac{\bar{N}}{M}\right)\right). \tag{13}$$

As the coefficients of the order in the last section were proportional to k^2, what we need to add is a factor $\log^2(\log(\bar{N}))$ to the $O(\bar{N}^2)$ computational cost and $O(\bar{N}\log\bar{N})$ storage. In practice the new factor is negligible against the others.

V. Global Algorithm (Putting It Together)

In this section, we are going to consider a complete radiation or scattering problem. It is necessary to subdivide the whole set of interactions in near and far interactions, so that we can apply the multilevel MLACA algorithm described above to the far parts. Commonly in the literature, a hierarchical octal tree is used to split the object or the related matrix and this approach can also be applied in our case if desired. Notwithstanding, we propose the utilization of a binary tree for this hierarchical subdivision. The differences in the results should not be too high in the vast majority of the cases. The only gain comes when the object discretization is very nonuniformly distributed or the object is much larger in one dimension than the rest, because in the proposed decomposition the boxes are adapted to the object at each subdivision.

Once the matrix is divided in far and near interactions, the near parts are directly computed whereas the far parts are computed with the presented MLACA algorithm. Then we have a routine to perform a matrix-vector product which can be introduced in any existing iterative method, like a GMRES for instance. This procedure adds a $\log N$ factor to the aforementioned asymptotical costs, obtaining finally a computational cost growing with $O(N^2\log N)$ and a storage and matrix-vector product with $O(N\log^2 N)$.

VI. Numerical Results

All the numerical experiments reported in this section have been performed on a PC with 64 GB of RAM and a Dual Intel Xeon X5460 processor at 3.16 GHz (eight cores). The computations were done in MATLAB© 7.8.0, always forcing to use only one CPU.

We present two different experiments where we consider only the interaction between two sets of samples separated in space (Section VI-A). They are the most important in order to evaluate the computational cost and the necessary memory of the algorithm. They could well be a part of a major object or electromagnetic problem. Then, three complete object examples are presented (Section VI-B).

In all the presented numerical experiments, when reference is made to a relative error in a computed (vector) parameter X, this error has been calculated according to

$$\frac{||X - X_{\text{ref}}||}{||X_{\text{ref}}||}, \tag{14}$$

where $||\cdot||$ is the 2-norm and X_{ref} is an independently created reference solution. The proper reference solution is usually hard to obtain when dealing with extremely large-scale problems due to the limitations in the available resources. Therefore, the obtained error is usually an estimated error which has only into account the error introduced by the accelerated solver rather than the actual error, which would include the effect of the finite discretization among others.

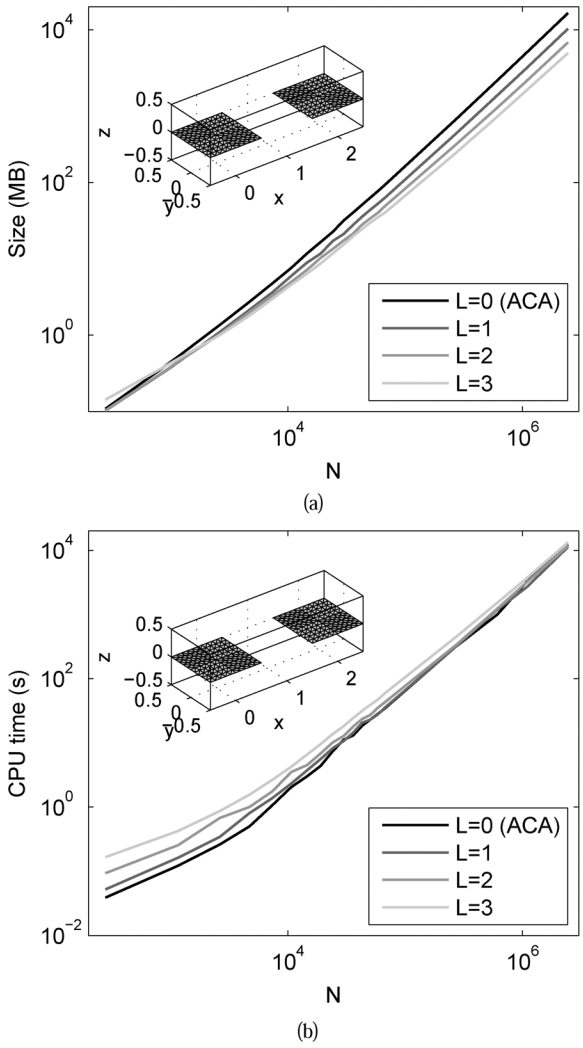

Fig. 6. Storage requirement and CPU time versus the number of unknowns, with a fixed electrical discretization size, in the interaction between two separated parallel square plates for different number of levels in the MLACA. The threshold τ equals 10^{-3}. (a) Storage. (b) CPU time.

A. Space Separated Blocks

1) Parallel Square Plates: In this experiment, a couple of 1 m $\times$ 1 m square plates are placed in the same plane, separated 2m. Varying the frequency, and accordingly the number of unknowns $\bar{N}$ of each square plate to have a fixed discretization of $\lambda/10$, we perform the computation of the interaction MoM-EFIE matrix between the two plates with the presented MLACA, for different number of used levels.

Fig. 6 shows the required memory and the computational time of this simulation when a threshold $\tau = 10^{-3}$ is used. Clearly, the memory requirement decreases significantly with the number of levels, mainly when the electrical size of the plates grows.

With respect to the CPU time, the optimum for small electrical sizes is the single level ACA ($L = 0$). Nonetheless, there is a certain point, approximately for $\bar{N} = 29800$ or a plate side of 10λ, where the time for $L = 1$ levels starts to be equal or even better when the plates size increase. The same happens for $L = 2$ at a further point, approximately for $\bar{N} = 270000$ or

TABLE I
STORAGE REQUIREMENT AND CPU TIME IN THE INTERACTION BETWEEN TWO SEPARATED PARALLEL SQUARE PLATES WITH $\bar{N} = 2428200$ SAMPLES PER PLATE FOR DIFFERENT NUMBER OF LEVELS IN THE MLACA. THE THRESHOLD τ EQUALS 10^{-3}

	$L = 0$ (ACA)	$L = 1$	$L = 2$	L=3
Time	3h 24m	3h 8m	3h 16m	3h 42m
Size (GB)	16.47	10.27	6.79	4.9

a plate side of 30λ. Exactly the same will happen for $L = 3$, not represented in the figure as it is beyond the maximum computed value, and so on and so forth. Therefore, an appropriate utilization to have a good trade-off between necessary memory and CPU time is the following: using the maximum number of levels but assuring that the CPU time is at least equal to the CPU time for $L = 0$. The number of levels to use is then chosen depending on the solid angles between the two blocks of samples. For larger solid angles, larger number of levels.

To stress the gain that we are actually obtaining, we have extracted the results for the largest computed parallel square plates ($\bar{N} = 2428200$ or square side 90λ) and included them in Table I. A good trade-off in this case would be to use $L = 2$, whose computation takes 8 minutes less than with the single level ACA with 41% of the memory (from 16.47 GB to 6.79 GB).

2) Opposite Square Plates: As will be shown in the present section, the gain is much greater when the two square plates are placed face to face instead of in the same plane. The two 1 m $\times$ 1 m square plates are oppositely placed and separated 2m. Again, the computation of the MoM-EFIE compressed interaction matrix is computed for different frequencies and number of MLACA levels. The used threshold is $\tau = 10^{-3}$ and the discretized elements have an average length of $\lambda/10$.

Fig. 7 shows the required memory and the computational time of this simulation. For electrically large blocks, the memory requirement is much smaller when the number of levels grows. For small blocks, the size with more levels is larger because the threshold is reduced proportionally to the number of levels (see Section IV).

As for the CPU time, the single level ACA ($L = 0$) is faster until blocks with about $\bar{N} = 7400$ samples or a side size of 5λ. From this point, $L = 1$ becomes the reference until blocks with $\bar{N} = 67200$ or a side size of 15λ. Afterwards, $L = 2$ is the fastest and it follows successively for larger electrical sizes and number of MLACA levels. In this case, the better behavior of the multilevel, even in terms of CPU time is much more evident.

Similarly to the parallel square plates case, we have included the results for the electrically largest simulated plates into Table II. It corresponds to square plates with $\bar{N} = 606600$ samples or a side size of 45λ. If we use $L = 2$ levels, it only takes a 37% of the time with respect to the ACA and it only needs a 15% of the memory. Furthermore, with $L = 3$, although it takes a 46% of the ACA time which is a bit slower than with $L = 2$, it only needs a 7.2% of the memory. Therefore, the number of MLACA levels to use will depend on our resources. If we are more restricted by memory, we can use a larger number of levels. If CPU time is our main constraint there is always an optimum number of levels in terms of time, with still an excellent compression.

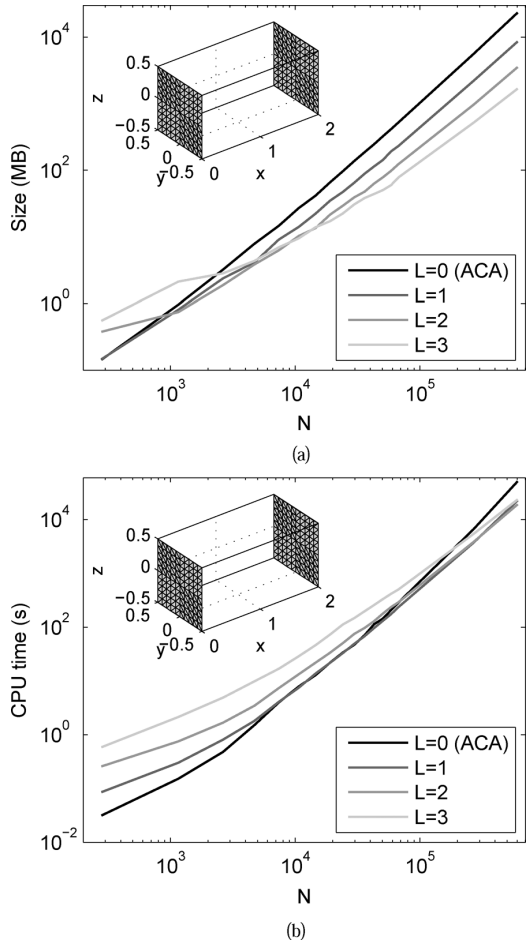

Fig. 7. Storage requirement and CPU time versus the number of unknowns, with a fixed electrical discretization size, in the interaction between two separated opposite square plates for different number of levels in the MLACA. The threshold τ equals 10^{-3}. (a) Storage, (b) CPU time.

TABLE II
STORAGE REQUIREMENT AND CPU TIME IN THE INTERACTION BETWEEN TWO SEPARATED OPPOSITE SQUARE PLATES WITH $\bar{N} = 606600$ SAMPLES PER PLATE FOR DIFFERENT NUMBER OF LEVELS IN THE MLACA. THE THRESHOLD τ EQUALS 10^{-3}

	$L = 0$ (ACA)	$L = 1$	$L = 2$	$L = 3$
Time	14h 15m	6h 16m	5h 19m	6h 33m
Size (GB)	23.31	8.57	3.54	1.68

B. Complete Problem Solutions

1) Large PEC Square Plate: The first object under study is a PEC square plate with a length side of 59.7λ. The threshold τ for the SVD recompressions is 10^{-3} and the size of the finest level in the subdivision of the object to split in far and near interactions is 0.5λ. The square plate has $N = 1385840$ RWG unknowns with an average edge length of the discretized elements of $\lambda/10$. In this case we have used the EFIE formulation and as iterative method we have used a GMRES with a residual threshold to stop of 10^{-5}. Note that in this case all the far interactions are of the kind where there are square plates in the same plane, which as proved before, is the case where the MLACA needs electrically larger blocks to be worth using.

TABLE III
MLACA PERFORMANCE ON CURRENTS COMPUTATION OF 59.7λ SIDE PEC SQUARE PLATE UTILIZING EFIE FORMULATION. $N = 1385840$ UNKNOWNS AND THRESHOLD $\tau = 10^{-3}$

	ACA	MLACA
Build time	5h 57m	6h 11m
Size Near (GB)	9.30	9.30
Size Far (GB)	38.22	32.75
GMRES threshold	10^{-5}	10^{-5}
# iterations	141	142
Iterations time	4h 44m	4h 30m
Total Time	10h 41m	10h 41m
Total Size (GB)	47.52	42.05
Error in $\mathbf{J}$(%)	0.29	0.19

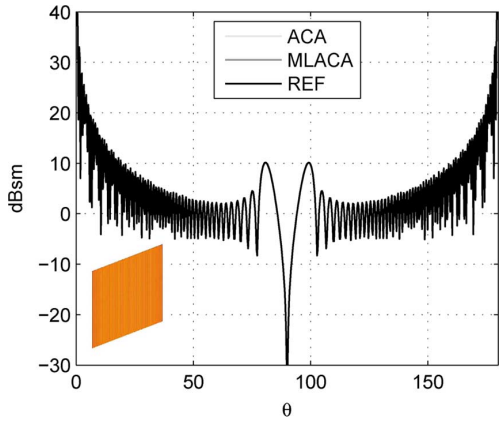

Fig. 8. Bistatic RCS of a 59.7λ side PEC square plate with $N = 1385840$ unknowns.

The reference current vector to compute the error has been obtained solving the problem with the MLACA with a threshold $\tau = 10^{-4}$. We tried to obtain it with the ACA to be fairer but it was then beyond the memory limit with the correspondent slowing down of the simulation when the swapping to disk started, making it unfeasible in a reasonable time.

Table III shows the performance of this simulation with the single level ACA and the MLACA. With the MLACA method we obtain a further compression of 5.5 GB with respect to the ACA with the same total solution time. It represents a reduction of an 11% of the total necessary memory. The estimated error is also lower for the MLACA than for the ACA, and in particular is of the same order as the selected threshold τ.

Something worth mentioning, which will appear in all the simulation from here on, is that the iterations time is not improved in the same amount as the used memory. This is because the GMRES has an additional cost apart from a matrix-vector product which increases with each iteration.

Fig. 8 shows the bistatic RCS of this simulation for the different methods. As expected from the errors in the surface currents, an excellent agreement can be observed for the RCS.

Note that we only use multiple levels for the far interaction of electrically large blocks. Therefore, in this case only a few interactions are improved (78 interactions with $L = 2$ and 1006 with $L = 1$), which do not represent a considerable percentage of the total matrix size. However, this compression will increase considerably when the object is larger. This behavior will be better analyzed for the PEC missile in Section VI-B.3.

TABLE IV
MLACA PERFORMANCE ON CURRENTS COMPUTATION OF 26.4λ DIAMETER PEC SPHERE UTILIZING CFIE FORMULATION. $N = 786432$ UNKNOWNS AND THRESHOLD $\tau = 10^{-1}$

	ACA	MLACA
Build time	4h 17m	4h 40m
Size Near (GB)	7.19	7.19
Size Far (GB)	29.12	22.35
GMRES threshold	10^{-5}	10^{-5}
# iterations	77	77
Iterations time	1h 19m	1h 17m
Total Time	5h 36m	5h 57m
Total Size (GB)	36.31	29.54
Error in $\mathbf{J}$(%)	7.97	7.80

2) Large PEC Sphere: In order to avoid conditioning problems, which are not the issue of this manuscript, and having a good convergence, in the next two examples we will only analyze closed geometries using the CFIE formulation. A drawback of this approach is that the matrix is not symmetric. Therefore, all the interactions need to be computed.

The next object under study is a PEC sphere with a diameter of 26.4λ with $N = 786432$ unknowns. The threshold τ for the SVD recompressions is 10^{-1}, the size of the finest level in the subdivision of the object to split in far and near interactions is again 0.5λ and the threshold for the GMRES residual to stop is 10^{-5}. The discretization has an average length of the elements of $\lambda/10$.

The reference current vector to compute the error has been obtained solving the problem with the ACA with a threshold $\tau = 10^{-3}$. Therefore, in any case it should benefit the ACA simulation, but as expected (see Section IV), the error will be slightly smaller for the MLACA.

Table IV shows the results of this simulation. In this case, the MLACA has only been used at the first level (largest blocks) of the object subdivision where there are far interactions (1946 interactions with $L = 1$), even though it should not be applied yet if we wanted to optimize the CPU time. The memory requirement is 6.77 GB lower for the MLACA than for the ACA with CPU time 21 minutes higher. It represents a gain in memory of a 18.6% with an increasing of 6.6% of CPU time. When the object becomes electrically larger, the gain in memory will be much more important together with a time improvement. The estimated relative error is slightly smaller for the MLACA and of the same order as the used threshold τ.

Fig. 9 shows the bistatic RCS of the sphere computed from the surface currents obtained with the different methods. We have also included the reference available for the sphere from the Mie series development. There are slight differences which are normal considering the errors in the surface currents. These errors can be easily reduced by decreasing the threshold τ, which has been set very high for this case ($\tau = 10^{-1}$).

3) PEC Missile: Finally, we analyze a PEC missile presented in [22] which has a length of 75λ. The shape and dimensions of

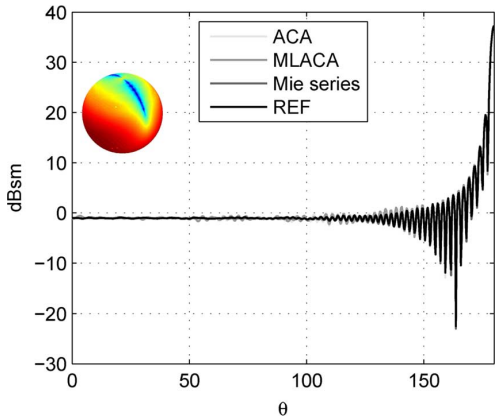

Fig. 9. Bistatic RCS of a PEC sphere with diameter 26.4λ and $N = 786432$ unknowns.

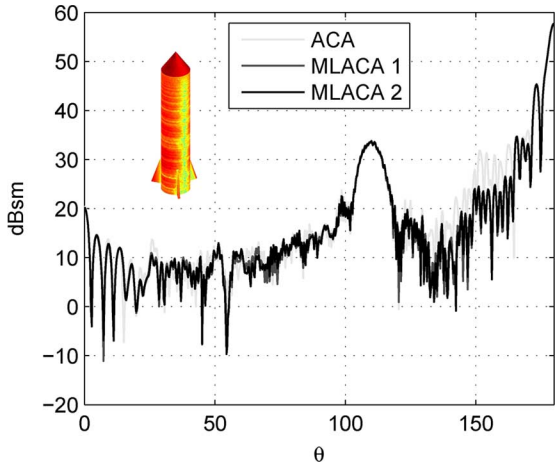

Fig. 10. Bistatic RCS of a 75λ length PEC missile with $N = 1222419$ unknowns.

the missile can be extracted from the referenced paper. The missile has been discretized with $N = 1222419$ unknowns and an average length of the elements of $\lambda/10$ as in the previous cases. As in the sphere, we use the CFIE formulation with a threshold $\tau = 10^{-1}$ and the size of the finest level in the subdivision of the object to split in far and near interactions of 0.5λ. However, the residual tolerance to stop the GMRES algorithm is set to 10^{-3} due to the worse convergence rate.

We were not able to obtain a good reference solution as we are very close to the RAM memory constraint of our computer. However, a comparison of the bistatic RCS with our results (see Fig. 10) and the one included in [22], calculated with a very different approach, shows a good agreement.

Table V shows the results for this simulation. In this case, there are two columns for the MLACA. We use MLACA 1 when multiple levels are only applied to the block interactions at the subdivision levels 2 and 3 of the object which correspond with the largest far-interaction blocks (16 interactions with $L = 2$ and 38 with $L = 1$). In the MLACA 2, also the forth level far-interactions are calculated with multiple levels (2110 interactions with $L = 1$). With the inclusion of the forth level, the build time is a bit higher but the gain in compression is considerable. In this case, the number of iterations is higher for the ACA than

TABLE V
MLACA PERFORMANCE ON CURRENTS COMPUTATION OF 75λ PEC MISSILE UTILIZING CFIE FORMULATION. $N = 1222419$ UNKNOWNS AND THRESHOLD $\tau = 10^{-1}$

	ACA	MLACA 1	MLACA 2
Build time	9h 45m	9h 52m	10h 9m
Size Near (GB)	8.55	8.55	8.55
Size Far (GB)	49.59	43.78	39.18
GMRES threshold	10^{-3}	10^{-3}	10^{-3}
# iterations	235	197	195
Iterations time	9h 40m	7h 28m	7h 23m
Total Time	19h 25m	17h 20m	17h 32m
Total Size (GB)	58.14	52.33	47.73

TABLE VI
MLACA PERFORMANCE ON THE MATRIX COMPUTATION OF 75λ PEC MISSILE UTILIZING CFIE FORMULATION. THE CONTRIBUTION TO SUBBLOCKS OF THE ORIGINAL OBJECT OF DIFFERENT SIZES IS CONTEMPLATED. $N = 1222419$ UNKNOWNS AND THRESHOLD $\tau = 10^{-1}$

	level 2		level 3		level 4	
	ACA	MLACA	ACA	MLACA	ACA	MLACA
# interactions	6	6	48	48	6892	6892
Size (GB)	5.50	1.36	3.05	1.38	22.11	17.50
Time	53m	56m	20m	24m	2h 1m	2h 18m

the others. Often the one which takes longer to converge is the one with a higher relative error in the matrix elements but this cannot be concluded in the absence of a good reference. In any case, the total time of the MLACA is smaller than the ACA and even the build time, without considering the iterations is very close. The gain in memory is 10.4 GB with respect to the ACA which corresponds with a 18% of the total.

To evaluate the contribution of the MLACA to the different far-interaction blocks, we have included Table VI. When the MLACA is applied to the relatively electrically small blocks at level 4 of the object subdivision we save a 21% of the memory but with an increase of a 14% of CPU time. In the next larger blocks level 3, the gain in memory is already a 55% with only 4 minutes more of time. And finally, the gain at the largest level 2, with only 6 far-interactions, is a 75% with just 3 extra minutes. Therefore, if the object is electrically larger, the compression of the larger block interactions will be much better, and also the contribution of the largest blocks interactions will represent a good proportion of the total memory.

Fig. 10 shows the bistatic RCS computed with the different methods. There are some small differences which are normal considering the high threshold we are using. The main difference is in the single level ACA.

VII. CONCLUSION

The MLACA has been presented as a novel multilevel version of the well-known ACA algorithm, for the compression of matrices appearing in the Method of Moments discretization of radiation and scattering electromagnetic problems.

Equally to ACA, MLACA applies when the source and field domains corresponding to the impedance matrix block being compressed are well-separated in space, which needs a previous hierarchical domain decomposition of the object under study

and a different treatment of compressible far field blocks from noncompressible near field blocks.

The storage and the matrix-vector product (or iteration) time scale asymptotically as $O(N\log^2 N)$ and the matrix compression time scales as $O(N^2\log N)$. The accuracy is expected to improve by reducing a threshold τ.

Several numerical experiments have corroborated the gain in terms of memory requirement and CPU time with respect to the single level version ACA, already for objects of around one million of unknowns.

It has been proved that the MLACA will be much superior for problems that are one or two orders of magnitude larger. Therefore, the author expects that the MLACA will find its place in the framework of supercomputers and parallelization, which are becoming a reality nowadays even at very low prices and will allow the solution of much larger objects.

REFERENCES

[1] W. C. Chew, J.-M. Jin, C.-C. Lu, E. Michielssen, and J. Song, "Fast solution methods in electromagnetics," *IEEE Trans. Antennas Propag.*, vol. 45, no. 3, pp. 533–543, Mar. 1997.

[2] N. Morita, N. Kumagai, and J. Mautz, *Integral Equation Methods for Electromagnetics*. Boston, MA: Artech House, 1990.

[3] R. Harrington, *Field Computation by Moment Methods*. New York: MacMillan, 1968.

[4] J. Song, C.-C. Lu, and W. C. Chew, "Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects," *IEEE Trans. Antennas Propag.*, vol. 45, no. 10, pp. 1488–1493, Oct. 1997.

[5] S. Velamparambil and W. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 53, no. 8, pp. 2719–2727, Aug. 2005.

[6] L. Gurel and O. Ergul, "Fast and accurate solutions of extremely large integral-equation problems discretised with tens of millions of unknowns," *Electron. Lett.*, vol. 43, no. 9, pp. 499–500, Apr. 2007.

[7] E. Michielssen and A. Boag, "A multilevel matrix decomposition algorithm for analyzing scattering from large structures," *IEEE Trans. Antennas Propag.*, vol. 44, no. 8, pp. 1086–1093, Aug. 1996.

[8] J. M. Rius, J. Parrón, E. Úbeda, and J. R. Mosig, "Multilevel matrix decomposition algorithm for analysis of electrically large electromagnetic problems in 3-d," *Microw. Opt. Technol. Lett.*, vol. 22, no. 3, pp. 177–182, Aug. 1999.

[9] J. Parron, J. Rius, and J. Mosig, "Application of the multilevel matrix decomposition algorithm to the frequency analysis of large microstrip antenna arrays," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 721–724, Mar. 2002.

[10] J. M. Rius, J. Parrón, A. Heldring, J. M. Tamayo, and E. Úbeda, "Fast iterative solution of integral equations with method of moments and matrix decomposition algorithm - singular value decomposition," *IEEE Trans. Antennas Propag.*, vol. 56, no. 8, pp. 2314–2324, Aug. 1998.

[11] E. Bleszynski, M. Bleszynski, and T. Jaroszewicz, "AIM: Adaptive integral method for solving large-scale electromagnetic scattering and radiation problems," *Radio Sci.*, vol. 31, no. 5, pp. 1225–1251, Sep./Oct. 1996.

[12] S. M. Seo and J.-F. Lee, "A single-level low rank IE-QR algorithm for PEC scattering problems using EFIE formulation," *IEEE Trans. Antennas Propag.*, vol. 52, no. 8, pp. 2141–2146, Aug. 2004.

[13] S. Kapur and D. Long, "IES3: A fast integral equation solver for efficient 3-dimensional extraction," in *Proc. ICCAD*, 1997, pp. 448–455.

[14] S. Kapur and D. Long, "IES3: Efficient electrostatic and electromagnetic simulation," *IEEE Comput. Sci. Eng.*, pp. 60–66, Oct.-Dec. 1998.

[15] M. Bebendorf, "Approximation of boundary element matrices," *Numer. Math.*, vol. 86, no. 4, pp. 565–589, 2000.

[16] S. Kurz, O. Rain, and S. Rjasanow, "The adaptive cross-approximation technique for the 3D boundary-element method," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 421–424, Mar. 2002.

[17] K. Zhao, M. Vouvakis, and J.-F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Trans. Electromagn. Compat.*, vol. 47, no. 4, pp. 763–773, Nov. 2005.

[18] J. Shaeffer, "Lu factorization and solve of low rank electrically large MOM problems for monostatic scattering using the adaptive cross approximation for problem sizes to 1 025 101 unknowns on a PC workstation," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, Sep. 9–15, 2007, pp. 1273–1276.

[19] A. Heldring, J. Tamayo, and J. Rius, "On the degrees of freedom in the interaction between sets of elementary scatterers," in *Proc. 3rd Eur. Conf. Antennas Propag.*, 23–27, 2009, pp. 2511–2514.

[20] J. M. Tamayo, A. Heldring, and J. M. Rius, "Multilevel adaptive cross approximation (MLACA)," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 2009, pp. 1–4.

[21] M. Bebendorf and S. Kunis, "Recompression techniques for adaptive cross approximation," *J. Integ. Equat. Appl.*, vol. 21, no. 3, pp. 331–357, 2009.

[22] W.-D. Li, W. Hong, and H.-X. Zhou, "An IE-ODDM-MLFMA scheme with DILU preconditioner for analysis of electromagnetic scattering from large complex objects," *IEEE Trans. Antennas Propag.*, vol. 56, no. 5, pp. 1368–1380, May 2008.

José M. Tamayo was born in Barcelona, Spain, on October 23, 1982. He received the degree in mathematics and the degree in telecommunications engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, both in 2006, and the Ph.D. degree in telecommunications engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, in 2011.

In 2004, he joined the Telecommunications Department, Universitat Politècnica de Catalunya (UPC), Barcelona. His current research interests include accelerated numerical methods for solving electromagnetic problems.

Alex Heldring was born in Amsterdam, The Netherlands, on December 12, 1966. He received the M.S. degree in applied physics and the Ph.D. degree in electrical engineering from the Delft University of Technology, Delft, The Netherlands, in 1993 and 2002, respectively.

He is presently an Assistant Professor with the Telecommunications Department, Universitat Politècnica de Catalunya, Barcelona, Spain. His research interests include integral equation methods for electromagnetic problems and wire antenna analysis.

Juan M. Rius received the "Ingeniero de Telecomunicación" degree in 1987 and the "Doctor Ingeniero" degree in 1991, both from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.

In 1985 he joined the Electromagnetic and Photonic Engineering group at UPC, Department of Signal Theory and Communications (TSC), where he currently holds a position of "Catedrático" (equivalent to Full Professor). From 1985 to 1988 he developed a new inverse scattering algorithm for microwave tomography in cylindrical geometry systems. Since 1989 he has been engaged in the research for new and efficient methods for numerical computation of electromagnetic scattering and radiation. He is the developer of the Graphical Electromagnetic Computation (GRECO) approach for high-frequency RCS computation, the Integral Equation formulation of the Measured Equation of Invariance (IE-MEI) and the Multilevel Matrix Decomposition Algorithm (MLMDA) in 3D. Current interests are the numerical simulation of electrically large antennas and scatterers. He has held positions of "Visiting Professor" at EPFL (Lausanne) from May 1, 1996 to October 31, 1996; "Visiting Fellow" at City University of Hong Kong from January 3, 1997 to February 4, 1997; "CLUSTER Chair" at EPFL from December 1, 1997 to January 31, 1998; and "Visiting Professor" at EPFL from April 1, 2001 to June 30, 2001. He has more than 50 papers published or accepted in refereed international journals (28 in IEEE TRANS.) and more than 150 in international conference proceedings.