ISSN: 2277-9655
Impact Factor: 3.785

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

IMPLEMENTATION OF FAST ITERATIVE SOLVER USING ADAPTIVE CROSS APPROXIMATION TECHNIQUE

Shruthi D*, Gourav Chatterjee, Dr.Radhika K.R, Member IEEE

DOI: 10.5281/zenodo.51542

ABSTRACT

The Multilevel Adaptive cross Approximation (MLACA), an algorithm to solve MoM electromagnetic problems with computational cost O(N2) and a storage scaling with O(NlogN), is presented here and for the first time applied to a whole electromagnetic problem and not only to the interaction between blocks whose containing spheres do not intersect each other. For compressing an off-diagonal submatrix of the method of moments (MoM) impedance matrix with a binary tree, the *L*-level MLACA includes $L + 1$ steps, and each step includes $2L$ ACA-SVD decompositions.

KEYWORDS: Finite element analysis (FEM), Method of Moments (MoM), Direct solver, Iterative solver, The Multilevel Adaptive cross Approximation (MLACA),Fast iterative solver.

INTRODUCTION

Electromagnetic analysis of the interaction of electromagnetic waves with dielectric bodies plays a pivotal role in a variety of highly complex areas like radio-wave propagation, scattering by and detection of airborne particulates, medical diagnostics, radome design, non-linear eddy current analysis, surface tomography, analysis of scattering by anisotropic and high contrast complex media like magnetic photonic crystals, for these areas and in particular, due to the continued increase of integrated circuit problem complexity, there is an ever increasing demand of reducing both memory and timing requirements.

Fast computation of the electromagnetic characteristic of large complex object has been a challenging problem in computing electromagnetism. It plays an important role in the research of radar target identification, the stealth and anti-stealth technology, and the design of modern antenna systems. As the state-of-art method in computing electromagnetism, Adaptive cross approximation (ACA) and its recursive variant, the multilevel Adaptive cross approximation (MLACA) have been widely used to accelerate the matrix-vector product when solving electromagnetic surface integral equations by iterative method. When MLACA is used, the computational and memory requirements are both alleviated form $O(N2)$ to $O(N\log N)$. With the need of solving larger and more complex problems, parallelizing MLACA is a good choice. There are several challenges in parallel MLACA, one is to achieve scalability comparing with the number of processors and another is to achieve load balance and data communication efficiency independent of irregular problem shape. In order to overcome these problems, we use a parallel compressed octree based domain decomposition method, which can make every processor auto load-balanced in depend of the irregular problem shape.

When solving scattering problem with iterative method, the matrix vector product that is required by an iterative solver is computed as follows: the matrix is decomposed into a nearfield part and a far field part. The nearfield matrix is computed once as common moment method matrix and used in every iteration. The product of the far field matrix with the vector is computed using MLACA in every iteration. To implement MLACA, an octree data structure should be built first using a recursive decomposition of domain with each node in the tree represents a cubical box. For each node in the octree, two list must be constructed: the near-list and the far-list.

ISSN: 2277-9655
Impact Factor: 3.785

THEORY

Finite Element Method

Finite element method (FEM) is a numerical method for solving a differential or integral equation. It has been applied to a number of physical problems, where the governing differential equations are available. The method essentially consists of assuming the piecewise continuous function for the solution and obtaining the parameters of the functions in a manner that reduces the error in the solution. In this article, a brief introduction to finite element method is provided. The method is illustrated with the help of the plane stress and plane strain formulation.

FEM formulation for a linear differential equation

A linear differential equation can be of the following form:

$Lu+q=0$, (1)

where u is the vector of primary variables of the problem, which are functions of the coordinates, L is the differential operator and q is the vector of known functions. This differential equation will be subjected to boundary conditions, which are usually of two types- (i) the essential boundary conditions (ii) the natural boundary conditions. The essential boundary conditions are the set of boundary conditions that are sufficient for solving the differential equations completely. The natural boundary conditions are the boundary conditions involving higher order derivative terms and are not sufficient for solving the differential equation completely, requiring atleast one essential boundary condition. For example, consider the differential equation:

$$\frac{d}{dx}(EA\frac{du}{dx})+q=0. \quad (2)$$

This problem can be solved completely under one of the following two conditions:
(i) u is prescribed at both ends.
(ii) u is prescribed at one end and du/dx is prescribed at the same or other end.

However, the problem cannot be solved if only du/dx is prescribed at both ends. Thus, we surely require one boundary condition prescribing u. Therefore, for this problem u= u* is an essential boundary condition and du/dx= (du/dx)* is a natural boundary condition, where * indicates the prescribed value. Now consider the differential equation

$$\frac{d^2}{dx^2}(EI\frac{d^2w}{dx^2})-q=0 \quad (3)$$

This differential equation can be solved completely by specifying w and dw/dx at both ends. One can also specify d^2w/dx^2 and/or d^3w/dx^3 as boundary conditions, however out of total four boundary conditions, two must be of one of the following forms:
(i) w prescribed at both ends.
(ii) w prescribed at one end and dw/dx prescribed at the other end.

Thus, the prescribed values of w and dw/dx form the part of essential boundary conditions and prescribed values of d^2w/dx^2 and d^3w/dx^3 form the part of natural boundary conditions.
Two popular FEM formulations are Galerkin formulation and Ritz formulation. In Galerkin formulation, the primary variable is approximated by a continuous function inside the element. When the approximate primary variable ue is substituted in Eq. (1), we shall get residue depending on the approximating function, i.e.,

$LU^e+q=R$ (4)

Ideally, the residue should be zero everywhere. In that case, approximation becomes equal to true value. As it is very difficult to make the residue 0 at all points, we make the weighted residual equal to zero, i.e.,

$$\int WRdA=0 \quad (5)$$

where w is the weight function. In order to weaken the requirement on the differentiability of the approximating function, we integrate Eq. (5) by parts to redistribute the order of derivative in w and R. In Galerkin method, the weight function is chosen of the same form as the approximating function. The approximating function is some algebraic function. It is common to replace the unknown coefficients of the function by unknown nodal degrees of freedom. Thus, typically,

$$u^e = [N]\{u^{ne}\} \quad (6)$$

where [N] is the matrix of shape functions and $\{u^{ne}\}$ is the nodal degrees of freedom.

In Ritz formulation, the differential equation Eq. (1) is converted into an integral form using calculus of variation. (Sometimes the integral form itself may be easily derivable from the physics of the problem.) The approximation (Eq. (6)) is substituted in the integral form and the form is extremized by partially differentiating with respect to $\{u^{ne}\}$.

After obtaining the elemental equations, the assembly is performed. A simple way of assembly is to write equations for each element in global form and then add each similar equations of all the elements, i.e., we add the equation number 1 from each element to obtain the first global equation, all equation number 2 are added together to give second equation, and so on. The boundary conditions are applied to assembled equation and then are solved by a suitable solver. Then, post-processing is carried out to obtain the derivatives.

Integral Equation Method (IE)

In mathematics, an integral equation is an equation in which an unknown function appears under an integral sign. There is a close connection between differential and integral equations, and some problems may be formulated either way. See, for example, Green's function, Fredholm theory, and Maxwell's equations.

Method of Moments

Iterative techniques are typically used for the solution of a large system of linear equations arising from the method of moments (MOM) formulation, often in conjunction with efficient algorithms for matrix-vector multiplication, as for instance the fast multipole method (FMM) and the multilevel fast multipole algorithm (MLFMA).

To deal with electromagnetic problems scattering and radiation of unbounded space, solving of method of moments(MOM) is only about the electric filed on the surface of scattering target rather then the whole space, thus the region to be solved reduced a dimension, greatly reduce the number of unknown variables. The moment's method has the incomparable advantage with other methods in efficient and distinctive.

The coefficient matrix of linear equation in the classic method of moments is dense. Its storage is $O(N^2)$,where N is the number of unknowns ,the computational complexity reach $O(N^3)$ with the direct method to solve. Increasing along with the electrical size of scattering target, the corresponding coefficient matrix is also getting bigger and bigger, such a complexity has restricted the scale of the problem for solving, and also the applied range of the moment's method. When N gets bigger, iterative method of linear equation must be used. The total computation load is decided by the order of complexity in each iteration and total times of iteration, the zero elements of coefficient matrix are more, the coefficient matrix is sparser, and the calculation complexity then is lower. The total times of iteration depends on the behavior of matrix, the lower the condition number of coefficient matrix, the less iteration is needed to convergent.

Direct Solver

Most of examiners for large-scale parallel structural analysis using finite element method have focused on iterative solution methods since direct solution methods normally have many difficulties and disadvantages for large-scale problems. However due to the numerical robustness of direct methods that guarantees the solution to be obtained within estimated time, direct methods are much more advantageous for general application of large-scale structural analysis, as well as due to ease of use which makes most of structural engineers prefer direct methods.

Direct solvers perform direct factorization of a global stiffness matrix with the non-zero pattern of the matrix considered and then solve the equation by substitution procedure, so the solution always can be found after required computational operations are completed unless the rank of the stiffness matrix is deficient.

Fast Iterative Solver

Iterative solvers use a completely altered approach for solving large systems of linear equations than direct solvers. The iterative solver method is based on the preconditioned conjugate gradient method. It starts out with an initial guess which is, for instance, the zero vector and experiences an iterative procedure to update the arrangement vector in each iteration utilizing the system matrix and a preconditioner matrix to focalize to the arrangement. A merging basis is utilized to figure out if the exactness of the arrangement is worthy or more iteration is expected to enhance the accuracy. On the off chance that a pre-decided most extreme number of iterations is come to without getting the desired accuracy of the solution, the solver exits with the message that it couldn't unite to the right solution.

Convergence of the solver and the convergence rate are dependent on the preconditioner used. For a simple preconditioner, work done and memory required is less for every iteration, however, the solvers take many iterations to convergence or will not converge at all. More refined preconditioners require more memory and more work in every iterations, but they may converge quickly to the correct solution. Iterative solver has been better and tuned continuously with regard to applicability, convergence rate, memory usage, performance and spill logic. The decision of the direct versus the iterative solution technique relies upon the sort of components in the model, the shape of the model, the computer assets and the kind of examination. The user ought to ask whether the principle objective is to run a job as quick as would be prudent or to have the capacity to run a job at all given specific computer assets. For models worked from 2D components it is constantly prescribed to utilize the direct solver while for models of 3D components the iterative solver should to be preferred.

There are many methods to implement a fast iterative solver depending upon the models used one of the method used is a Adaptive Cross Approximation(ACA) and advanced one is Multilevel Adaptive Cross Approximation(MLACA).

Multilevel Adaptive Cross Approximation(MLACA)

Imagine we have two objects contained within two spheres that do not intersect each other. We shall call them source and observation objects respectively. Applying the Method of Moments (MoM) it is possible to express the interaction between them with a matrix Zmn with dimensions m× n, where m and n are the number of basis functions in which the observation and source object have been discretized, respectively. In the first place, we must subdivide each object recursively into smaller domains in a binary tree manner, so in each subdivision approximately half of the basis functions goes to each side. The number of levels is L meaning that at the finest level we have in each subset approximately $M = N/2^L$ elements considering N = n = m. For asymptotical analysis L is going to be chosen to yield a fixed value of M or equivalently, a fixed minimum box electrical size.

A scheme of the algorithm is shown in figure 1. The idea is to express the initial matrix Zmn as a product of L+2 matrices $Z_{mn}=A^{(L+!)}\ B^{(L+1)}B^{(L)}\ldots\ldots B^{(1)}$

In the step 0 (figure 1(a)) Z_{mn} is transformed into two new matrices $A_0^{(1)}$ and $B_0^{(1)}$. The procedure to obtain them is to split Z_{mn} into strips, corresponding to the interaction of each subset of basis functions at the finest level in the source object with the whole observation object. Now each of those strips only has k degrees of freedom, therefore can be compressed with the ACA-SVD algorithm and regrouped as is shown in figure 1(a). Note that the matrix to store is $B_0^{(1)}$ which has 2L blocks in the diagonal of size $k \times n/2^L$ and is orthogonal.

For each i = 1, . . . ,L and each j = 0, . . . , $2^{i-1}-1$ the matrices from the step i−1, $A_j^{(i)}$, are transformed into four new matrices $A_{2j}^{i+1}, A_{2j+1}^{i+1}, B_{2j}^{i+1}, B_{2j+1}^{i+1}$ (figure2.1(b)). The initial matrix A_j^i is split into two sets of rows corresponding to the next subdivision in the observation object tree. On the other hand, the strips are grouped in pairs corresponding to the previous level in the source object tree. As the source is doubled in size and the observation is divided by two, the number of degrees of freedom of the new substrips is again k [3], and therefore can be recompressed using the ACA-SVD algorithm to obtain the new matrices. After the step i = L we have the whole set of matrices which correctly combined represent a compressed version of the matrix Z_{mn}. It can be proved that the memory required to

ISSN: 2277-9655
Impact Factor: 3.785

store those matrices at the end of the algorithm is proportional to kNlogN and as the size of the finest level is constant with respect to N, we have that k is constant, and therefore the memory scales with O(NlogN). Consequently, the computational cost of a product of the matrix Zmn with a vector scales also with O(NlogN). The computational cost to build the compressed matrix can be proved to be O(N2), which is asymptotically better than for single-level ACA.

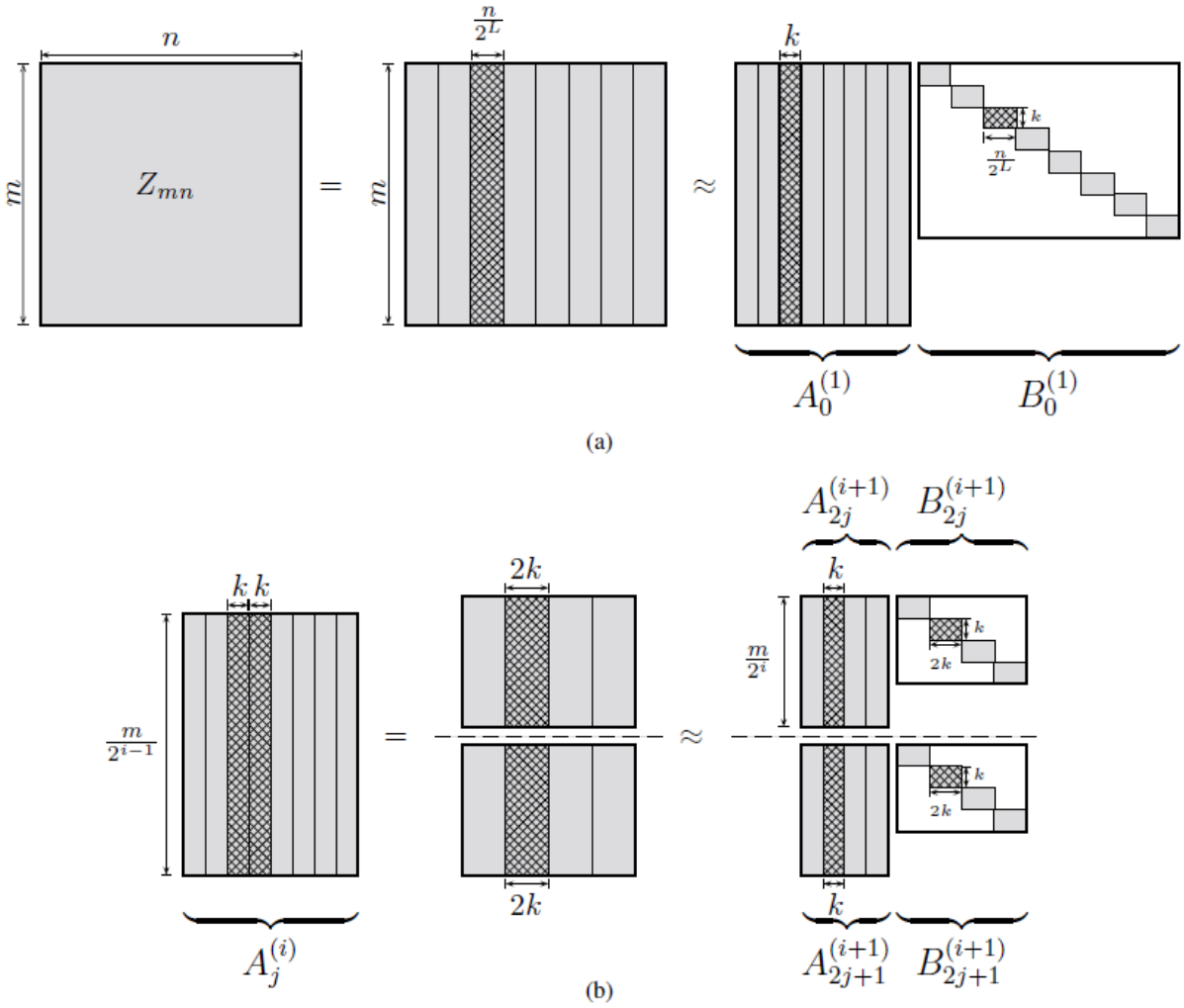

Figure 1:Graphical representation of the matrix transformations in the MLACA algorithm described in section"Description of the algorithm". First step or step 0 (a) and step i (b) with i>=1.

NUMERICAL RESULTS

In this section we evaluate the performance and stability of the proposed technique i.e. for the electro-magnetic analysis of different components.

ISSN: 2277-9655
Impact Factor: 3.785

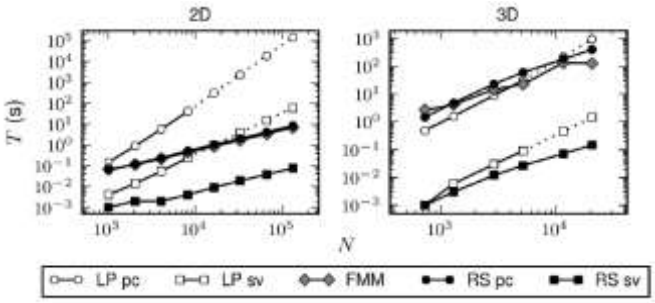

Fig 3. Graph representation

Computational Complexities

In the above, computational complexities of the algorithms are expressed in terms of P, which is the number of harmonics required to express the scattered field of a larger scatterer made up of N subscatterers. Clearly then, P depends on N. The exact dependence of P on N is determined by the geometry. Table 1 summarizes this dependence for some interesting cases.

Geometry	Dependence of P	$O(NP^2)$ becomes ...	$O(NP)$ becomes ...
1D	$P \propto N^0$	$O(N)$	$O(N)$
Planar 2D	$P \propto \log N$	$O(N \log^2 N)$	$O(N \log N)$
2D	$P \propto \sqrt{N}$	$O(N^2)$	$O(N^{1.5})$
Planar 3D	$P \propto \sqrt{N} \log N$	$O(N^2 \log^2 N)$	$O(N^{1.5} \log N)$
3D	$P \propto N^{2/3}$	$O(N^{7/3})$	$O(N^{5/3})$

Table 1: Dependence of P on N and consequently the expressions for the computational complexities are determined by the geometry.

REFERENCES

[1] Jeong Ho Kim, Chang Sung Lee, Seung Jo Kim, "Development of a High-performance Domain-wise parallel Direct Solver for Large-scale Structural Analysis," in 0-7695-2138-X/04.
[2] Wenwen Chai and Dan Jiao, "Fast H-Matrix-Based Direct Integral equation solver with reduced computational cost for Large-Scale interconnect extraction", in IEEE transactions on components, packing and manufacturing technology, February 2013.
[3] Bnagda Zhou and Dan Jiao, "Direct finite-element solver of linear complexity for large-scale 3D electromagnetic analysis and circuit extraction," in IEEE transactions on microwave theory and techniques, vol. 63, NO. 10, October 2015.
[4] P.G.Martinsson, V.Rokhlin, "A fast direct solver for boundary integral equations in two dimensions", in journal of computational physics 205(2005).
[5] Levent Gurel, Weng Cho Chew, "Fast direct (Non iterative) solvers for integral-equation formulations of scattering problems".
[6] Kai He, Sheldon X.-D.Tan, Hai Wang and Guoyong Shi, "GPU-Accelerated parallel sparse LU factorization method foe fast circuit analysis", in IEEE transactions on very large scale integration systems, vol.24,NO.3,March 2016.