Rapid Acoustic Boundary Element Method for Solution of 3D Problems using Hierarchical Adaptive Cross Approximation GMRES Approach

A. Brancati[1], M. H. Aliabadi[2] and I. Benedetti[3]

[1] Department of Aeronautics, Imperial College London, South Kensington Campus, SW7 2AZ London, UK, a.brancati@imperial.ac.uk

[2] Department of Aeronautics, Imperial College London, South Kensington Campus, SW7 2AZ London, UK, m.h.aliabadi@imperial.ac.uk

[3] On leave from DISAG Dip. Ing. Strutturale Aerospaziale Geotecnica, Università degli Studi di Palermo, Palermo, Italy

Keywords: Rapid Acoustic Boundary Element Method, Adaptive Cross Approximation, Hierarchical Approach, GMRES, speed-up ratio, large scale problems, high frequency range.

Abstract. This paper presents a new solver for 3D acoustic problems called RABEM (Rapid Acoustic Boundary Element Method). The Adaptive Cross Approximation and a Hierarchical GMRES solver are used to generate both the system matrix and the right hand side vector by saving storage requirement, and to solve the system solution. The potential and the particle velocity values at selected internal points are evaluated using again the Adaptive Cross Approximation (ACA). A GMRES without preconditioner and with a block diagonal preconditioner are developed and tested for low and high frequency problems. Different boundary conditions (i.e. Dirichlet, Neumann and mixed Robin) are also implemented. Herein the problem of engine noise emanating from the Falcon aircraft is presented. The tests demonstrated that the new solver can achieve CPU times of almost O(N) for low frequency and O($NlogN$) for high frequency problems.

Introduction

The need of faster solver for three dimension acoustic simulations is encouraged by required solution in higher frequency range, for large scale and more accurate geometries.
One of the most general and efficient numerical technique for solving acoustic problems is the Boundary Element Method (BEM) [1]. The boundary element discretisation of the surface of the problem leads to a non-symmetric and fully populated system matrix. Both the memory storage and the setting up of the system matrices for a standard BEM formulation are of O(N^2), where N denotes the degree of freedom. Moreover, direct solvers require O(N^3) operations while iterative solvers O(kN^2), where k is the number of iterations.
The ACA is an effective method for solving non-symmetric and fully populated matrices and decreases the CPU time significantly [2] and has been applied to the Helmholtz equation by, for example, [3]. The solution of linear system of equations is accelerated by calculating only few entries of the original matrix. The whole matrix is divided into two rank (low and full rank) blocks based on size and distance between a group of collocation points and a group of boundary elements. The ACA algorithm has been applied to the low rank blocks achieving approximately O(N) for both storage and matrix-vector multiplication [4].
Herein a solver for 3D boundary element solution of Helmholtz problems that uses a new hierarchical adaptive cross approximation technique coupled with GMRES is presented. Implementing different types of boundary conditions (i.e. Dirichlet, Neumann and mixed Robin) into the ACA solution algorithm is widely considered. The constant elements are utilized to discretize the problem. The test carried out show that the new assembly and solution technique can achieve CPU times of almost O(N) for low frequency and O($N\backslash log\ N$) for high frequency problems.

Hierarchical BEM for Acoustics

The BEM system of equations was represented using the hierarchical matrices in conjunction with Krylov subspace methods by, for example [5, 6], and herein is extended to BEM acoustic problems with different boundary conditions. This technique speeds up the computation, whilst maintaining the required accuracy and saving on the memory storage.

Matrix Assembly using the Collocation Method. In a hierarchical representation the boundary element matrix is subdivided by a collection of two groups of blocks called *low rank blocks*, that have a compressed representation, and *full rank blocks*, that are represented in their entirety. The classification of the blocks is achieved subdividing the whole mesh into cluster of closed elements. A block populated by integrating over a cluster of elements whose distance, suitably defined, from the cluster of collocation nodes is above a certain threshold is called admissible and it can be represented in the low rank format. The remaining *full rank blocks* are generated and stored in their entirety.

A preliminary hierarchical partition of the matrix index set is the basis of the process for the subdivision and classification of the blocks. Contiguous elements are detected on the basis of some computationally efficient geometrical criterion. The process starts from the complete set of indices *I={1,2...n}* where *n* denotes the number of collocation points. This initial set constitutes the root of the tree node. Each cluster in the tree is split into two subsets, called *sons*, on the basis of the longest extended dimension of the whole geometry or of the geometry of each cluster *son* (see Fig.1).

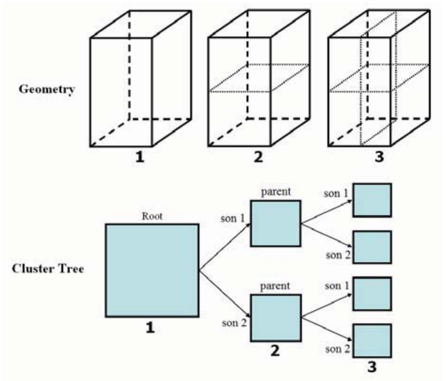

Fig. 1: A schematic of the first two iterations of the cluster tree creation: 1) the whole geometry is divided 2) into two parts, each of which is then subdivided again 3) into two parts.

The tree node that generates two sets of tree nodes is called the *parent*. The tree nodes that cannot be further split are called *leaves* of the tree. A leave of the tree is a tree node that cannot be further split because it contains a number of indices equal to or less than a minimum number n_{min}, called *cardinality* of the tree. This is a previously fixed value. This partition creates a binary tree of index subsets, or cluster tree, that constitutes the basis for the subsequent construction of the hierarchical block subdivision that will be stored in a quaternary *block tree*.

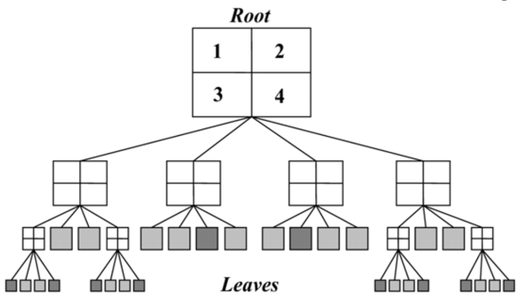

Fig. 2: A schematic presentation of the first three iterations that form the block tree. The light grey stands for low rank blocks, while the dark grey for full rank blocks.

The generation of the block tree is based on the previous found cluster tree and is created starting from the complete index $I \times I$ (both rows and columns) of the collocation matrix. The goal of this process is to split hierarchically the matrix into sub-blocks until that are classified as *leaves* of the tree (see Fig. 2). A *low-rank block* and a *full-rank block* are leaves that satisfy and not satisfy, respectively, an admissible criterion, based on a geometrical consideration related directly to the boundary mesh (see for example [7, 8]).
The admissible criterion can be written as

$$\min(diam\{\Omega_{x_0}\}, diam\{\Omega_x\} \le \eta \cdot dist(\Omega_{x_0}, \Omega_x) \tag{1}$$

where Ω_{x_0} and Ω_x denote the cluster of the row and the column elements, respectively, and η is a positive parameter influencing the number of admissible blocks and the convergence speed of the adaptive approximation of low rank blocks.
Let C be an $m \times n$ admissible block. It admits the low rank representation

$$C \approx C_k = A \cdot B^T = \sum_{i=1}^{k} a_i \cdot b_i^T \tag{2}$$

where *A* is of order $m \times k$ and *B* is of order $n \times k$, with *k* being the *rank* of the new representation. Sometimes it is useful to represent the matrix using the alternative sum representation, where a_i and b_i are the *i-th* columns of *A* and *B*, respectively. The approximate representation allows storage savings with respect to the full rank representation and speeds up the matrix-vector product [9].

Boundary Conditions and Right Hand Side Setting The actual setting of the final system for a problem where different boundary conditions are specified requires some additional considerations when ACA is applied. Different types of boundary conditions are here considered in such a way that rigid, soft and absorbing surfaces can all be studied in order to simulate a real situation and to perform an eventual parametric analysis.
The ACA algorithm is applied to one or both of the matrices **G** and **H** depending upon the boundary conditions that are predominant for each block matrix. Particular care must be taken when not pure Dirichlet, Neumann or mixed Robin conditions are applied. There may be four different cases that require different approaches [10]:

- BCs mainly in terms of flux (or potential) except for the kth value expressed in terms of potential (or flux);
- BCs mainly in terms of flux except for the kth value expressed in terms of impedance;
- BCs mainly in terms of potential except for the kth value expressed in terms of impedance;
- BCs mainly in terms of impedance except for the kth value expressed in terms of potential (or flux).

Additional considerations for the setting up of the right hand side vector are now explained. When the ACA algorithm is applied, the routine that calculates the ith row of one of the block matrices **G** or **H** also calculates the ith row of the other block matrix. Thus, the right hand side contribution of that row for the block matrix analyzed is directly calculated. Now, there are two main cases to analyze.

1. BCs mainly in terms of flux (or potential).
   The ACA algorithm is applied again on the matrix **G** (or **H**). A frequent occurrence is when all the boundary conditions are zero. In this case there is no need to calculate the contribution of the block matrix to the final right hand side and another block matrix can be analyzed. Moreover, owing to the fact that the number of entries needed for the ACA is, in most the cases, equal for both the block matrix **G** or **H**, may be convenient to calculate the contribution to the right hand side with a standard procedure.

2. BCs mainly in terms of admittance.
   Once the ACA as been applied to the block matrix **H**, the possible contribution to the right hand side vector **F**, due to the presence of the kth value of the boundary conditions expressed in terms of potential, is easily calculated by multiplying each column of the ACA to the kth value of each row with opposite sign. Finally, if the ACA algorithm applied to the block matrix **G** is also successful reached, the contribution to the right hand side of the eventual presence of the boundary condition expressed in terms of flux is clearly calculated.

System Solution The solution of the system can be computed through iterative solvers with or without preconditioners, once the system matrix has been represented in the hierarchical form.
In this study the value of the parameter η is 10 and the cardinality is set to 22. These values are chosen because they give the best performances. Being more specific, the value of the cardinality is quite restricted. As this value decreases, many blocks that satisfy the admissible criterion need, in fact, a bigger storage memory than if the same block is considered to be full rank. A direct consequence of it is a higher value of the CPU time. Nevertheless, the value of the parameter η can be modified between 1 and 1000 without loss of accuracy and resulting in a 5% CPU time acceleration. Furthermore, the optimum value of this parameter depends upon the geometry and the elements of the mesh.
Fig (3) shows the block-wise storage requirements of the collocation matrix generated by the ACA algorithm for four different values of η (1, 2, 4 and 10). The tone of grey is proportional to the ratio between the memory required for the low rank representation and the memory required for a standard format. Hence black blocks stand for the full rank block matrix, while almost white blocks are those for which the ACA compression works better.

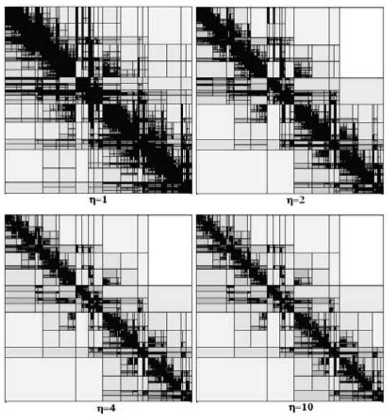

Fig. 3: Block-wise representation of the ACA generated matrix for η=1, η=2, η=4 and η=10.

Results

The simulation of a 53,074 elements mesh (see Fig. 4) of a model representing the Dassault Falcon airplane is presented.

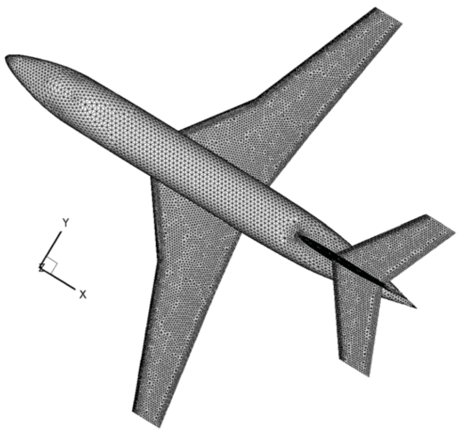

Fig. 3: Block-wise representation of the ACA generated matrix for η=1, η=2, η=4 and η=10.

The total length of the aircraft is 18.5m and the wing extension is 22.46m. The highest frequency applied was 100 Hz. The sizes of the smallest and the largest triangles are 8.73E-03 and 0.19 millimeter, respectively, so the maximum frequency that can be applied is around 250 Hz. In the simulation performed all the surfaces has been set as hard and two monopole sources with unit complex potential amplitude have been inserted where the two engines are generally located in this airplane. The CPU time ratio, obtained by dividing the CPU time for different frequencies and preconditioners with the frequency of 25Hz for the unpreconditioned GMRES, is shown in Fig. 5 for three frequencies (25, 50 and 100 Hz). Finally, Fig. 6 shows the Sound Pressure Level (SPL) at 100Hz. It is important to point out that a standard BEM code cannot solve such a simulation because the storage required is around 45 Gb whereas the limit for a medium desktop is 2Gb. Moreover, the speed up ratio, defined as the ratio between the CPU time of the standard code and the CPU time of the RABEM code, would be more than 350!

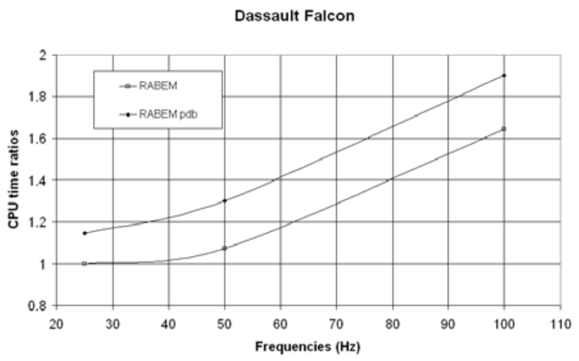

Fig. 4: Comparison between the unpreconditioned and block diagonal preconditioned GMRES for a Dassault Falcon meshed with 53,074 triangular elements.

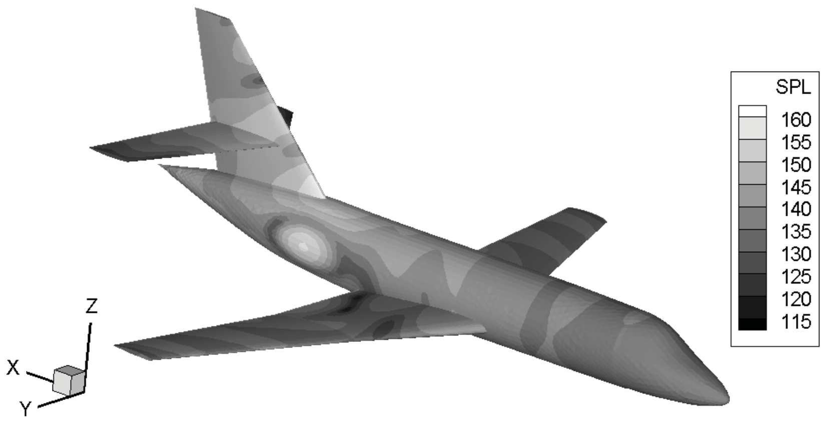

Fig. 4: Sound Pressure Level for a Falcon geometry with two monopoles at 100 Hz.

Conclusion

A new Rapid Acoustic BEM solver (RABEM) for 3D numerical simulation using the ACA algorithm in conjunction with GMRES has been herein presented. This study demonstrates that the new approach reduces significantly the storage and the solution time. Moreover, the simulation shown that the new solver can achieve CPU times of almost O(N) for low frequency and O($N \cdot \log N$) for high frequency problems.

Acknowledgment

This work was carried out with the support of European research project (SEAT: Smart tEchnologies for stress free Air Travel) AST5-CT-2006-030958. The authors are especially grateful to Dr. Vincenzo Mallardo for our many and always fruitful conversations. We wish to thank Dr. Joaquim Peiro for the Falcon mesh.

References

[1] L.C.Wrobel and M.H Aliabadi *The Boundary Element Method, Vol1 :Applications in Thermo-Fluids and Acoustics, Vol2: Applications in Solids and Structures,* Wiley (2002).

[2] M. Bebendorf and S. Rjasanow *Adaptive low-rank approximation of collocation matrices. Computing*, **70** (1), 1-24 (2003).

[3] O. Von Estorf, S. Rjasanow, M. Stolper and O. Zalesk *Two efficient methods for a multifrequency solution of the Helmholtz equation: Computing and Visualization in Science*,

8, 159-167 (2005).

[4] I. Benedetti, M. H. Aliabadi and G. Daví *A fast 3D dual boundary element method based on hierarchical matrices: International Journal of Solids and Structures*, **45 (**7-8), 2355-2376 (2007).

[5] C. Y. Leung and S. P. Walker *Iterative solution of large three dimensional BEM elastostatic analyses using the GMRES technique: International Journal for Numerical Methods in Engineering*, **40**, 2227-2236 (1997).

[6] M. Merkel, V. Bulgakov, R. Bialecki and G. Kuhn *Iterative solution of large-scale 3D-BEM industrial problems: Engineering Analysis with Boundary Elements*, **2**, 183-197(1998).

[7] M. Bebendorf and S. Rjasanow *Adaptive low-rank approximation of collocation matrices: Computing*, **70 (**1), 1-24 (2003).

[8] M. Bebendorf *Approximation of boundary element matrices: Numerische Mathematik*, **86**, 565-589 (2000).

[9] L. Grasedyck and W. Hackbusch *Construction and arithmetics of H-matrices: Computing*, vol **70**, 295-334 (2003).

[10] A. Brancati, M. H. Aliabadi and I. Benedetti. *Hierarchical Adaptive Cross Approximation GMRES Technique for Solution of Acoustic Problems Using the Boundary Element Method: CMES: Computer Modeling in Engineering & Sciences*. Accepted for publication in 2009.