A high performance 3D simulation framework for acoustics : from mesh to visualization

G. Alléon * D. Barbier † G. Sylvand*

June 3, 2005

The development of a complete chain of numerical simulation tools to predict the behaviour of complex devices is crucial in an industrial framework. Nowadays the numerical simulation is fully integrated in the design processes of the aeronautics industry. During the last decades, robust models and numerical schemes have emerged and have been implemented in simulation codes to study the various physical phenomena involved in the design of aeronautic applications. The simulation tools enable the engineers to perform virtual prototyping of new products and to study their behaviours in various contexts before a first real prototype is built. Such a tool permits to reduce the time to market and the cost to design a new product. A typical simulation platform is composed by three main components that are

1. the preprocessing facilities that include the CAD systems and the mesh generators,
2. the numerical simulation software, that implement the selected mathematical models and the associated numerical solution techniques,
3. the post-processing facilities, that enable to visualize the results of the numerical simulations. This latter step is often performed using virtual reality capabilities.

Each of these main components operate with each other because it is necessary to go back and forth between them when the characteristics of the studied devices change. This occurs for instance when numerical optimization is performed to maximize some governing parameters of the new product. In this presentation, we concentrate on computational electromagnetics/acoustics. This application requires the solution of partial differential equations (PDE) defined in large domains and discretized using fine meshes. Because those calculations are one step in a larger production process, they should be performed on high performance computers so that they do not slow-down the overall process. In the recent years, those high performance computers are mainly clusters of symmetric multiprocessors (SMP) that have demonstrated to be cost effective.

1 The pre-processing phase

During the last decade, the numerical methods have progressed in such a way that the pre and post-processing tools are seen as bottlenecks. The new airplane generation faces new environmental requirement that lead to mesh size out of the scope of classical mesh generation software. As an example, it may be quite common in the coming years to have problems requiring a few millions mesh cells. The major difficulty with boundary element methods is that their mesh rely directly on the CAD definition.

An out-of-core distributed mesh generator has been developed to be able to generate large triangle meshes. This mesher is basically divided into three steps:

*EADS CCR, Centre de Toulouse, Centreda 1, 4, Avenue Didier Daurat, 31700 Blagnac, France
†IMACS, X-Tec, École Polytechnique, Route de Saclay 91120 Palaiseau

- 1D mesh generation
- 2D mesh generation
- 3D mesh assembly

The first step is basically a sequential phase which aims at discretizing CAD edges according to some criterion (mesh size or maximum deflection). The second phase can be distributed over several resources; it is doing the actual meshing based on a Riemannian metrics in the parameter space. This metric allows a close control of the 3D mesh based on edge size, absolute or relative deflection. The last step is finally to gather all the 2D meshes through a 3D projection operator. At this stage, a limitation of the mesher is that it is missing a remeshing step allowing triangles to cross patch frontiers. Nevertheless, this is not really a problem for our application since only fine meshes are considered.

The mesher is now used to generate mesh up to 100 millions cells, the sequential generation time is approximately 13 hours on a single CPU 2 GHz Athlon workstation. The distribution of the step 2 has been tested either using a classical job scheduler like OpenPBS/Torque or a Condor direct acyclic graph. Both of these methods lead to drastically reduced run time, the limiting factor being the maximum meshing time for a single patch assuming that we have more computing resources than patches in the geometry. The meshing time for the phase 2 of a pylon is requiring an aggregated time of 8106 seconds while the maximum time for meshing the biggest patch is 490 seconds.

2 Solving the physical problem: the Helmholtz equation

2.1 The integral equations and the FMM

In this section, we introduce the integral equations used in acoustic in our study. We are interested in the solution of the Helmholtz equations in the frequency domain using an integral equation formulation. The integral approach used here to solve the Helmholtz equation has several advantages over a classic surfacic formulation. First, it requires to mesh only the surface of the objects, not the propagation media inside and around it. In an industrial context, where the mesh preparation is a very time consuming step, this is crucial. Second, the radiation condition at infinity is treated naturally by the formulation in an exact manner. At last, the surfacic mesh allows to have a very accurate description of the object's shape, which is not the case with finite difference methods for instance.

The main drawback of integral formulation is that it leads to solve dense linear systems difficult to tackle with iterative solvers. Even though the spectral condition number is usually not very high, the eigenvalues distribution of these matrices is not favorable to fast convergence of unpreconditioned Krylov solvers [1]. Furthermore for large objects and/or large frequencies, the number of unknowns can easily reach several millions. In that case, the classic iterative and direct solvers are unable to solve our problem, because they become very expensive in CPU time and storage requirements.

The technique that we used to to solve this acoustic problem is very similar to the one used to solve electromagnetics problems (see [14]).

The Fast Multipole Method (FMM) is a new way to compute fast but approximate matrix-vector products. In conjonction with any iterative solver, it is an efficient way to overcome these limitations ([13, 12, 10]). It is fast in the sense that CPU time is $\mathcal{O}(n.\log(n))$ instead of $\mathcal{O}(n^2)$ for standard matrix-vector products, and approximate in the sense that there is a relative error between the "old" and the "new" matrix-vector products $\varepsilon \approx 10^{-3}$. Nevertheless, this error is not a problem since it is usually below the target accuracy requested to the iterative solver or below the error introduced by the surfacic triangle approximation.

In order to widely exploit the available parallel architectures, we have developped a parallel distributed memory implementation of the FMM algorithm based on the message passing paradigm and the MPI library.

2.2 The parallel iterative linear solvers

We focus now on the solution of the linear system

$$Ax = b$$

associated with the discretization of the wave propagation problem under consideration. As mentioned earlier, direct dense methods based on Gaussian elimination quickly becomes unpractical when the size of the problem increases. Iterative Krylov methods are a promising alternative in particular if we have fast matrix-vector multiplications and robust preconditioners. For the solution of large linear systems arising in wave propagation, we found that GMRES [8] was fairly efficient [4, 14].

Most of the linear algebra kernels involved in the algorithms of this section (sum of vectors, dot products calculation) are straightforward to implement in a parallel distributed memory environment. The only two kernels that require to pay attention are the matrix-vector product and the preconditioning. The matrix vector product is performed with the parallel FMM implementation described in Section 2.1. The preconditioner is described in the next section, its design was constrained by the objectives that its construction and its application must be easily parallelizable.

2.3 The preconditioner

The design of robust preconditioners for boundary integral equations can be challenging. Simple parallel 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 depending on the integral formulation [11]. Incomplete factorizations have been successfully used on nonsymmetric dense systems [9] and hybrid integral formulations [7], 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 uneffective.

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 [1, 2, 3, 5, 15]). Owing to the rapid decay of the discrete Green's function, the location of the large entries in the inverse matrix exhibit some structure [1]. 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 of the exact inverse. This remarkable property can be effectively exploited in the design of a robust approximate inverse.

The original idea of an approximate inverse preconditioner based on Frobenius-norm minimization is to compute the sparse approximate inverse as the matrix M which minimizes $\|I - AM\|_F$ subject to certain sparsity constraints. The Frobenius norm is chosen since it allows the decoupling of the constrained minimization problems into n_{dof} independent linear least-squares problems, one for each column of M, when preconditioning from the right. Hence, there is considerable scope for parallelism in this approach.

For choosing the sparsity pattern, 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. 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 [1]. The discrete Green's function can be considered as a column of the exact inverse defined 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 [1, 3, 6]. 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. In that context, relevant information for the construction of the pattern of M can be extracted from the octree.

2.4 The parallel performance

We show here an example of computation realised with a realistic objet (a rigid wing portion corrying almost 6 millions of unknowns) and a simplified illumination (a plane wave at 3150 Hz). Basically, we would obtain the same computationnal performance no matter what the right hand side of the system is. These computations were made on a cluster using opteron processors and a Gigabit ethernet interconnection.

The solution of the initial system was obtained in 9 hours on 32 processors, using CFIE formulation plus SPAI preconditionner. The CFIE formulation has the ability to converge much faster than the usual EFIE formulation, but works only on rigid bodies. The matrix vector products were done in 78 second each with the FMM, with only 10 seconds for MPI communication. A final residual of 0.006 was obtained after 100 iterations.

Then, using the surfacic potentials obtained by the solver, we ran a near field computation. That is to say, using the Helmholtz representation theorem , with a simple matrix-vector product (accelerated by the FMM) we compute the diffracted and total pressure around the wing. This was done on a grid of 700 x 1400 points in a vertical plane with x varying from -35 m to +25 m and z ranging from -15 m to +15 m (x being the axis of the plane).

Both the resolution itself and the near field computation would have been impossible to treat without the fast multipole acceleration. Here, we are able to compute accurately the pressure in a domain of $60m \times 30m = 560\lambda \times 280\lambda$.

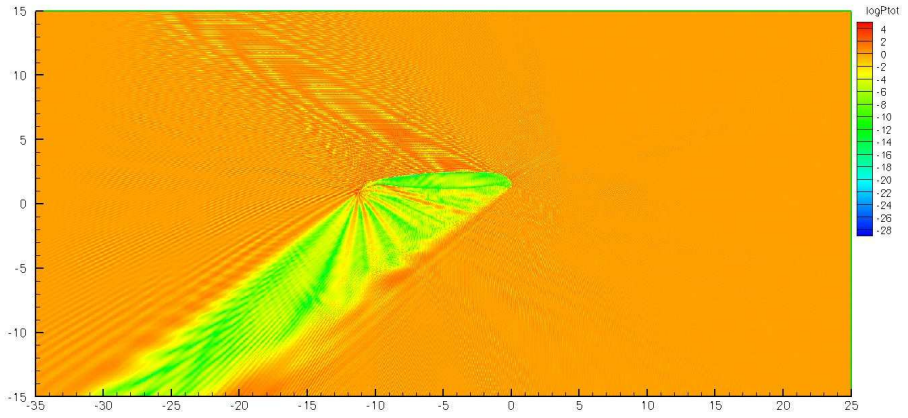

Figure 1: Total pressure aroung the wing computed here on a 700 x 1400 points grid using the near field FMM

3 Acknowledgments

The authors want to thanks the Algo team from CERFACS and especially Luc Giraud for their valuable contributions especially in the design of the preconditioners.

References

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

[2] B. Carpentieri. *Sparse preconditioners for dense linear systems from electromagnetic applications*. PhD thesis, CERFACS, Toulouse, France, April 2002.

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

[4] B. Carpentieri, I. S. Duff, L. Giraud, and G. Sylvand. Combining fast multipole techniques and an approximate inverse preconditioner for large parallel electromagnetism calculations. *SIAM J. Scientific Computing*, to appear.

[5] K. Chen. An analysis of sparse approximate inverse preconditioners for boundary integral equations. *SIAM J. Matrix Analysis and Applications*, 22(3):1058–1078, 2001.

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

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

[8] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986.

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

[10] X. Q. Sheng, J. M. Jin, J. Song W. C. Chew, and C. C. Lu. Solution of combined field integral equation using multilevel fast multipole algorithm for scattering by homogeneous bodies. *IEEE Ant. Propag*, 46(11):1718–1726, November 1998.

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

[12] J. M. Song and W. C. Chew. The fast illinois solver code : Requirements and scaling properties. *IEEE Computationnal Science and Engineering*, 5(3):19–23, July-September 1998.

[13] J. M. Song, C. C. Lu, W. C. Chew, and S. W. Lee. Fast illinois solver code. *IEEE Antennas and Propagation Magazine*, 40(3), June 1998.

[14] G. Sylvand. *La mthode multipôle rapide en lectromagntisme : performances, paralllisation, applications*. PhD thesis, Ecole Nationale des Ponts et Chausses, Juin 2002.

[15] S. A. Vavasis. Preconditioning for boundary integral equations. *SIAM J. Matrix Analysis and Applications*, 13:905–925, 1992.