ADVANCED METHODOLOGY FOR SIMULATION OF COMPLEX FLOWS USING STRUCTURED GRID SYSTEMS

Erlendur Steinthorsson and David Modiano
Institute for Computational Mechanics in Propulsion,
NASA Lewis Research Center, Cleveland, Ohio

ABSTRACT

Detailed simulations of viscous flows in complicated geometries pose a significant challenge to current capabilities of Computational Fluid Dynamics (CFD). To enable routine application of CFD to this class of problems, advanced methodologies are required that employ (a) automated grid generation, (b) adaptivity, (c) accurate discretizations and efficient solvers, and (d) advanced software techniques. Each of these ingredients contributes to increased accuracy, efficiency (in terms of human effort and computer time), and/or reliability of CFD software. In the long run, methodologies employing structured grid systems will remain a viable choice for routine simulation of flows in complex geometries only if genuinely automatic grid generation techniques for structured grids can be developed and if adaptivity is employed more routinely. More research in both these areas is urgently needed.

INTRODUCTION

Computer technology and algorithms for Computational Fluid Dynamics (CFD) have developed to a point where CFD is now used routinely in the study of engineering and prototype flows. The purpose of using CFD is to give concrete predictions of specific data, such as surface pressure distributions or heat transfer, and to enhance understanding of the characteristics of the flows. Unfortunately, current methodologies and software lack the desired accuracy and efficiency for routine application to complicated viscous flows in complex geometries such as internal cooling passages of turbine blades. For these flows, large human effort and computing time are typically required to arrive at acceptable solutions. When structured grids are used in simulations, the process of arriving at a final solution often involves first, a considerable effort in generating an initial grid system and, thereafter, an iterative process involving obtaining a solution on the given grid, evaluating the solution, and refining or otherwise modifying the grid in preparation for computing another solution to the same problem.

Because of the large human effort needed in the past to generate high quality structured grids, many, whose interest is in the simulation of flows in complex geometries, have looked towards unstructured grids (mainly triangles in two dimensions, prisms and tetrahedra in three dimensions) to model the complicated geometries. Unstructured grids offer all the flexibility needed to model complicated geometries, and the grid generation process for unstructured grids is far more automated than that for structured grids. Also, algorithms for adapting unstructured grids to computed solutions, for example through local mesh refinement, are well developed (see e.g., Ref 1). These advantages have led many to believe that unstructured grids are the best for use in simulations of flows in complex geometries. However, unstructured grids also have disadvantages, such as large memory requirements and/or large computing time (CPU time) requirements. Additionally, generating high-quality unstructured grids for high Reynolds number flows has proven difficult although recent progress has been made (see e.g., Ref. 2-4).

Due to difficulties like those mentioned above, high-fidelity simulations of viscous flows in complex geometries still pose a considerable challenge whether structured grids or unstructured are used. If the difficulties can be overcome through improved software and solution methodology, one can use existing computing resources to tackle bigger and more complicated problems than can now be handled, and make more routine the simulation of flows now analyzed only with great difficulty. For organizations employing CFD, this translates into economic savings through better use of manpower and computing equipment. In this paper, an advanced CFD methodology is outlined that will overcome at least some of the difficulties related to the use of structured grids. The ideas behind the methodology are not necessarily new but they are rarely put forward in a systematic way as done here. In the next section, the ingredients in methodology are discussed in general terms that apply independent of grid structure. Following the general description, the role of grid structure is discussed and a specific strategy for structured grids is laid out. The paper ends with a short summary.

METHODOLOGY

The thesis of this paper is that in order to optimize the use of human and computing resources, any methodology for simulating flows in complex geometries needs certain "ingredients." These ingredients are

(a) automated grid generation,

(b) adaptivity,

(c) accurate discretizations of the governing equations and efficient solvers,

(d) advanced software techniques.

All these ingredients or components contribute to the effectiveness and reliability of the software used for flow simulation. The importance of each is discussed below. A diagram showing the impact of integrating the components in a single methodology is shown in Fig. 1.

Automated grid generation is now recognized as an important step towards an effective simulation methodology. It reduces the human effort needed to generate grids for complicated geometries and can significantly cut the total time needed to arrive at acceptable solutions. In the past, the time needed to generate a structured grid for a complicated geometry was often an order of magnitude greater than the time needed to compute a solution. Today, this may still be true although new grid generation methods and improved graphical user interface with the grid generation software have remedied this problem somewhat. Grid generation methods for unstructured grids are typically much more automated than for structured grids. For instance, advancing front techniques for generation of triangular and tetrahedra meshes allow grids for very complicated geometries to be generated quickly (in a matter of hours or even minutes) with little human intervention.[5-7] Generation of unstructured Cartesian grids is even more automated and can be completed in minutes.[8]

Adaptivity—i.e., automatically adjusting mesh resolution (h-refinement) or order of accuracy of the discretization (p-refinement) in response to the computed solution—can also bring important improvements to the overall solution process. The improvements include enhanced accuracy of the computed solutions, reduced computational effort, and reduced human effort. Accuracy is enhanced through improved resolution of important flow features, even when it is not known beforehand where these flow features are located. Computational effort is reduced as adaptivity allows appropriate resolution to be used everywhere in the flow field, e.g., adaptive local mesh refinement produces fine grids only where they are needed leaving appropriately coarser grids elsewhere. Possibly the greatest benefit of adaptivity is the reduction or even elimination of the iterative process that typifies the present use of CFD for complex flows, where a user, iteratively, computes a solution on a given grid, evaluates the solution with respect to accuracy, and then modifies the grid "manually" in preparation for computing a new solution to the same problem. Adaptivity takes on added importance with increasing demands for accurate solutions and for concrete error estimation for the computed solutions. The increased demands for accuracy are evident, for example, in new editorial policies by professional journals such as the Journal of Fluids Engineering.[9]

The need for good discretizations and efficient solvers for flows in complex geometries is self explanatory in the light of the large number of grid points needed for high fidelity simulations of such flows. However, it is important to note that memory requirements of the flow solvers is also an issue for problems involving a large number of grid points. The memory required by a flow solver is often what limits the size of the problems that can realistically be tackled using that particular flow solver on a particular computer.

The need for advanced software techniques arises inevitably as the increased automation of the solution procedures and increased use of advanced solution algorithms makes the CFD software more and more complicated. Increased generality of the software and improved user interface only add to the complexity. Consequently, the software must be carefully and systematically developed to ensure bug-free codes and reliable operation. Advanced software techniques, such as object oriented programming is one way to achieve this. Modern programming languages that support object oriented programming include C++, which supports object oriented programming without sacrificing efficiency.[10] The new FORTRAN90 programming language also has some support for object oriented program design. Object oriented programming techniques enhance the modularity, reliability and maintainability of computer codes, and contribute to the development of bug-free modules. They are already in use for development of CFD codes (see e.g., Ref. 11, 12).

When all are combined, the ingredients discussed above can significantly enhance the capabilities of CFD software and widen the range of applications that can be tackled on a given computing platform. The results that are obtained will also be more reliable and the power of CFD can more easily be exploited by users who are not experts in CFD or fluid mechanics.

STRUCTURED GRIDS FOR COMPLICATED GEOMETRIES

To a varying degree of sophistication, current CFD software makes use of some or all of the ingredients of the ideal methodology described in the previous section. With respect to automation of the grid generation and adaptivity, CFD methodologies employing unstructured grids come closest to the ideal. However, a persistent drawback of CFD codes employing unstructured grids is a large memory requirement. Due to lack of structure in the grids, information about connectivity between cells or nodes in the grid needs to be stored. Also coefficients related to reconstruction of the solution based on cell average values are stored in many codes. A total storage of about 180 double precision variables (over 1400 bytes) per solution element (e.g., cells in cell-centered schemes) appears to be typical for flow solvers capable of simulating three-dimensional viscous flows (see e.g., Ref. 13, 14). In comparison, codes employing structured grids typically require between 45 and 70 double precision variables per solution element. Thus simply from the perspective of storage requirements, structured grids are preferable to unstructured grids for problems requiring high resolution of the flow field since they allow larger problems to be tackled on a given computer.

In addition to requiring less memory, structured grids offer other advantages. First, due to the regularity of the grid structure, various efficient schemes based on dimensional splitting can be applied on structured grids. For instance, ADI schemes and line-Gauss-Seidel methods can be used effectively for implicit discretizations. Also due to the regularity of the grid structure, multigrid schemes can be applied in a straightforward manner on structured grids to accelerate convergence to steady state or to speed up the resolution of systems of equations arising from implicit discretizations. Finally, structured body fitted grids are well suited for simulations of viscous flows due to good resolution of boundary layer regions.

Based on the considerations touched upon above, structured grids appear at the present time to be better suited than unstructured grids for simulation of complex flows requiring high resolution of the flow field. However, three questions arise immediately: First, can efficient structured grids be generated for very complicated geometries? Second, can the generation of structured grids be automated to the extent needed? Finally, can adaptivity be implemented in an effective manner? If the answer to any of these questions is negative then structured grids will in the long run only be useful for a very limited class of flow problems. The first two questions will be addressed in the remainder of this section. The last question will addressed in the next section.

Efficient Structured Grids for Complicated geometries

The first question to be answered regarding the use of structured grids is whether *efficient* structured grids can be generated for very complicated geometries. This question is best addressed by posing it as two questions, namely for how complicated a geometry can a high quality structured grid be generated and when are the grids efficient?

A number of strategies have been devised to generate structured grids for complicated geometries. Two main approaches are to use "multiblock" grids (continuous block-structured body-fitted grid systems) and "Chimera" grids.[15] Multiblock grids are generated by "carving up" the flow domain into multiple zones or blocks of reasonable size and shape such that a continuous grid can be generated for the entire domain. In contrast, Chimera grids are created by independently generating simple body-fitted grids around different components of a geometry, allowing the component grids to overlap in an arbitrary manner. Multiblock grids have the greatest degree of continuity of all block structured grids while Chimera grids have the least. A number of alternative structures have been devised that, in terms of continuity across blocks boundaries, fall in between the two main structures.

The advantage of using Chimera grids is that such grids can be generated for almost any geometry. In fact, the flexibility of Chimera grids closely approaches that of unstructured grids. Furthermore, grid generation effort is greatly simplified since the geometry can be broken up into components for which simple, high quality grids can be generated. A disadvantage is that the flow solver must deal with the arbitrary overlap between grids. In a sense, the difficulty in dealing with the complicated geometry has been postponed from the grid generation stage to the flow-simulation stage. Often, the interpolation needed to transfer data between component grids leads to noticeable glitches in the computed solutions. Recently, new approaches have been proposed that promise to to improve the communication between grids. One is to replace the region of overlap with an unstructured grid (DRAGON grids, see Ref. 16). Another is to eliminate the overlap region from one of two overlapping grids and to enforce flux conservation at the surface (curve in two dimensions) of intersection between the two blocks.[17] Commercial grid generators such as ICEM-CFD[18] and Gridgen[19] are well suited for generating Chimera grids. A number of flow solvers are designed to use Chimera grids, including OVERFLOW (developed at NASA Ames Research Center) and CFL3D (developed at NASA Langley Research Center).

The advantage of multiblock grid systems is the high degree of continuity in the entire grid system. In high quality multiblock grids, grid lines are at least C^1 continuous across block boundaries (except at certain singular points in the grids). Consequently, implementation of schemes for multiblock grids in flow solvers is relatively straightforward and block-boundaries become transparent to the solver. For instance, higher-order accurate finite volume discretizations can be implemented without loss in accuracy at block boundaries. The disadvantage of multiblock grids is that it has traditionally been considerably harder to generate high-quality multiblock grids than almost any other type of grid system. Recently, however, new methods[20-22] and commercial software[23] have become available that go a long way towards making multiblock grids competitive with Chimera grids, both in terms of human effort in grid generation and in terms of capability to handle complicated geometries. A sample grid multiblock grid system generated by the commercially available GridPro[23] is shown in Fig. 2 and 3. The strength of this particular software package is that the only input required, in addition to the geometry itself, is the topological structure of the grid system. Exact specification of block interfaces is not needed. Flow solvers designed to use multiblock grids include TLNS-MB[24] and TRAF3D.MB.[25]

Whether a structured grid for a complicated geometry will be efficient must be answered on a case by case basis since the answer depends not only on the geometry but also on the resolution that is required. To be efficient, each grid block should have at least few hundred nodes or cells (in three-dimensions). Otherwise, too much memory and computer-time overhead will be involved in communicating data between blocks. In general, if high-fidelity simulations of viscous flows are planned, the grid spacing required for accuracy of the solution will be orders of magnitude smaller than the physical size of the blocks used in a multiblock or Chimera grid system to resolve the features of the geometry. In most such cases then, efficient structured grids can be generated.

Automation of Grid Generation

The second question to be answered regarding the use of structured grids was whether the grid generation can be automated. Unfortunately, up to this point little effort has gone into developing truly automatic algorithms for generation of structured grids. It is therefore premature to judge whether effective algorithms can be developed. However, some recent work has been published on automatic generation of multiblock grids for two-dimensional geometries. In particular, Shoenfeld and Weinerfelt[26] devised an algorithm based on an advancing front technique. This same technique can be used to generate an initial topology of a grid system while other techniques can be used to optimize the grid for that particular block structure. It therefore appears likely that automatic techniques can be developed that work for two-dimensional geometries. For three-dimensional geometries, other techniques based for instance on feature recognition may need to be used.

MESH REFINEMENT IN STRUCTURED GRIDS

The third and last question related to the use of structured grids in simulations of complex flows was whether adaptivity could be effectively implemented for structured grids. As discussed earlier, the objective of using adaptivity in flow simulations is twofold. First, to ensure that without human intervention all important features of the flow are captured to the desired level of accuracy, and second, to minimize the computational effort required to achieve the desired accuracy. Two approaches have been used to implement adaptivity in finite volume schemes. The first is to move grid points in the grid system towards regions where higher resolution is needed, leaving coarser grids elsewhere. The second approach is to locally refine the mesh where higher resolution is needed by adding points (local Adaptive Mesh Refinement or AMR). The moving mesh approach can be used to maximize the accuracy of a solution computed on a grid with a fixed number of nodes; in a sense, maximizing accuracy for a given cost. When used in structured grids it has the limitation that grid points can only move a certain amount before grid quality (smoothness and orthogonality) degrades and begins to counteract the advantages of increased resolution. Adaptive mesh refinement has no such limitation and multiple levels of refinement can be used to achieve the desired accuracy (within the limits of applied models such as turbulence models, etc.). Thus, AMR offers the needed flexibility to automate adaptivity in the solution process.

To be compatible with the use of structured grids, an AMR algorithm should allow the flow solver to take full advantage of the strengths of structured grid systems, including efficient solution algorithms for implicit discretizations, various schemes based on dimensional splitting, and multigrid schemes. In essence, this means that the algorithm should use a block structure for the refined grid. To date, only a few methodologies of this nature have been proposed. One of the earliest was the method of Berger and Oliger.[27] In their method, the refined grids are allowed to overlay the underlying coarse grids in an arbitrary manner. The blocks of the refined grids are constructed in physical space based on the shape and size of the region to be refined. The resulting grids tend to align with discontinuities and other features that determine the shape and size of the region.

Building on the work of Berger and Oliger, Berger and Colella[28] devised a methodology in which the refined grids conform with the coarse grids, i.e., the boundaries of the fine-grid blocks are made to coincide with grid lines of the coarse grid. In this method, the block structure for the refined grids is created using a special algorithm that fits topologically rectangular patches over the regions of the coarse grids where the estimated error in the solution is above a specified threshold value. The blocks of the refined grids are then created by subdividing the coarse grid cells within each of the patches. Advantages of the new approach include greatly simplified prolongation and restriction operators for transferring data between a coarse grid and a refined grid, and that conservation at interfaces between coarse and fine grids can be enforced in a rigorous manner. This approach has been used extensively for Cartesian grids (see e.g., Ref 29, 30). Efforts to implement the algorithm in structured grids, including Chimera grids, are ongoing.[31,32]

A third AMR algorithm, proposed by Davis and Dannenhoffer[33], also involves creating the fine grids by subdividing cells on the coarser grid. In this approach, however, the entire structured grid system is divided up into *a priori* defined sub-blocks of uniform size and dimensions. During adaptation, each sub-block is refined either in its entirety or not at all. In a recursive manner, a sub-block that has been refined is itself divided into sub-blocks, each of which can be refined. In the algorithm of Davis and Dannenhoffer, directional refinement is used, i.e., the grid can be refined in one, two or three directions at a time as desired.

To demonstrate the feasibility of using AMR in structured grids, results obtained using AMR in simulation of a two-dimensional inviscid transonic flow over a NACA0012 airfoil are shown in Fig. 4. The AMR algorithm used is that of Berger and Colella.[27] The adaptation of the algorithm to structured grids is described in Ref. 31. The discretization used for the governing equations is also described in Ref. 31. The solution shown in Fig. 4 was computed using three levels of refinement with a refinement ratio of two. The dimensions of the original coarse grid around the NACA airfoil were 32×96. Only about 20% were refined three times (to the finest level). According to available data[1] the error in the solution is minimal, less than 2% as measured by the location of the normal shock behind the airfoil.

Based on the above review of AMR methods for structured grids, it is fair to say that the ability to use solution-adaptive mesh refinement to implement adaptivity in structured grids has been demonstrated. Nonetheless, further demonstrations may be needed, particularly for three-dimensional flows and for multiblock grid systems. Here it is appropriate to point out that implementing AMR in a general purpose flow solver requires careful programming and the use of a modern programming language. In fact, due to limitations of FORTRAN77, related to dynamic memory allocation and advanced data structures, it may be impossible to write in that language a practical general purpose flow solver that makes use of AMR. One can speculate that it is precisely because of the limitation of that otherwise effective programming language for scientific computing that use of AMR for structured grids is not already more wide spread.

SUMMARY

In order to optimize the use of human and computing resources, any methodology for simulating flows in complex geometries needs the following ingredients: (a) automated grid generation,(b) adaptivity, (c) accurate discretizations and efficient solvers, and (d) advanced software techniques. For simulations of viscous flows in complicated geometries or other flows requiring high resolution of the flow field, structured grid systems offer clear advantages over unstructured grids due to less memory usage and efficient flow-solver technology. However, in terms of automation of grid generation and adaptivity, methodologies employing structured grids trail those using unstructured grids.

In the long run, structured grids will remain a viable choice for routine simulations of flows in complicated geometries only if sufficiently efficient automatic grid generation techniques for structured grids are developed and only if adaptivity is employed in the solution methodology. At the present time, automatic grid generation methods for two-dimensional geometries may be on the horizon and adaptivity based on local adaptive mesh refinement schemes that are compatible with the use of structured grids have been demonstrated or are under development. More research is urgently needed in both areas.

REFERENCES

1 Warren, G. P., Anderson, W. K., Thomas, J. L., and Krist, S. L.; "Grid Convergence for Adaptive Methods," AIAA 91-1592, June 1991.

2 Pirzadeh, S.; "Unstructured Viscous Grid Generation by the Advancing Layers Method," AIAA 93-3453, August, 1993.

3 Pirzadeh, S.; "Viscous Unstructured Three-Dimensional Grids by the Advancing Layers Method," AIAA 94-0417, August, 1993.

4 Ward, S. and Kallinderis, Y.; "Hybrid Prismatic/Tetrahedral Grid Generation for Complex 3-D Geometries," AIAA 93-00669, January, 1993.

5 Peraire, J.V., Morgan, M., and Zienkiewicz, O.C.; "Adaptive Remeshing for Compressible Flow Computations," *J. Comp. Phys.*, Vol. 72, October, 1987, pp. 449–466.

6 Gumbert, C., Lohner, R., Parikh, P. and Pirzadeh, S.; "A Package for Unstructured Grid Generation and Finite Element Flow Solvers," AIAA 89-2175, June, 1989.

7 Mavriplis, D. J.; "An Advancing Front Delaunay Triangulation Algorithm Designed for Robustness," ICASE Report, 92-49, October, 1992.

8 Coirier, W. J.; *An Adaptively-Refined Cartesian, Cell-Based Scheme for the Euler and Navier Stokes Equations*, PhD thesis, University of Michigan, Department of Aerospace Engineering, 1994. Also, NASA TM 106754.

9 "Journal of Fluids Engineering Editorial Policy Statement on the Control of Numerical Accuracy," *Transactions of the ASME, Journal of Fluids Engineering*, Vol. 115, Sept. 1993, p. 340.

10 Stroustrup, B.; "*The C++ Programming Language*," 2nd edition, Addison Wesley, 1991.

11 Crutchfield, W. Y. and Welcome, M. L.; "Object Oriented Implementation of Adaptive Mesh Refinement Algorithm," *Scientific Programming*, Vol. 2, number 4, winter 1993.

12 Tchon, K.-F.; "Object-Oriented Programming for a Temporal Adaptive Navier Stokes Solver," AIAA 95-0574

13 Frink, N. T.; "Recent Progress Toward a Three-Dimensional Unstructured Navier-Stokes Flow Solver," AIAA-94-0061.

14 Mavriplis, D.J., and Venkatakrishnan, V.; "A 3-D Agglomeration Multigrid Solver for the Reynolds-Averaged Navier Stokes Equations on Unstructured Meshes," AIAA 95-0345, January, 1995.

15 Steger, J.L., Dougherty, F.C. and Benec, J.A.; "A Chimera Grid Scheme," Advances in Grid Gener-

ation, Ed. by K.N. Ghia, and U. Ghia, 1983, pp. 59-70.

16 Liou, M.-S., and Kao, K.-H.; "Progress in Grid Generation: From Chimera to DRAGON Grids," *Frontiers of Computational Fluid Dynamics 1994*, (D.A. Caughey and M. M. Hafez, ed's), Chapter 20, John Wiley and Sons, Nov. 1994.

17 Wang, Z.J., Buning, P., and Benek, J.; "Critical Evaluation of Conservative and Non-Conservative Interface Treatment for Chimera Grids," AIAA 95-0077, January, 1995.

18 Akdag, D. and Wulf, A.; "Integrated Geometry and Grid Generation System for Complex Configurations," NASA CP 3143, April, 1992, pp. 161-171.

19 Steinbrenner, J.P. and Chawner, J.R.; "The Gridgen Version 9, Multiple-Block Grid Generation Software," MDA Engineering Report 94-01, June 1994.

20 Allwright, S. E.; "Techniques in Multiblock Domain Decomposition and Surface Grid Generation," in *Numerical Grid Generation for Computational Fluid Mechanics '88*; Sengupta, S., Häuser, J., Eiseman, P. R., and Thompson, J. F., (ed), pp. 559-568, Pineridge Press, 1988.

21 Amdahl, D. J.; "Interactive Multi-Block Grid Generation," in *Numerical Grid Generation for Computational Fluid Mechanics '88*; Sengupta, S., Häuser, J., Eiseman, P. R., and Thompson, J. F., (ed), pp. 579-578, Pineridge Press, 1988.

22 Stewart, M.; "A General Decomposition Algorithm Applied to Multi-Element Airfoil Grids," AIAA 90-1606, 1990.

23 "GRIDPRO,^{TM} /AZ3000, Users Guide and Reference Manual", Program Development Corporation, White Plains, NY, 1993.

24 Vatsa, V. N., Sanetrik, M. D. and Parlette, E.B.; "Development of of a Flexible and Efficient Multigrid-Based Multiblock Flow Solver," AIAA 93-0677, January, 1993.

25 Steinthorsson, E., Liou, M.-S., and Povinelli, L.A.; "Development of an Explicit Multiblock / Multigrid Flow Solver for Viscous Flows in Complex Geometries," AIAA-93-2380, 1993. Also, NASA TM 106356 and ICOMP 93-34.

26 Schönfeld, T. and Weinerfelt, P.; "Automatic Generation of Quadrilateral Multi-Block Grids by the Advancing Front Technique," Proceedings of the *Third International Conference on Numerical Grid Generation*, pp. 743-754, S.-Arcilla, A., Häuser, J., Eisemann, P.R., and Thompson, J.F. (Eds.), Elsevier Science Publishers, North Holland, June 1991.

27 Berger, M. J. and Oliger, J.; "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," *J. of Comp. Physics*, Vol. 53, pp. 484-512, 1984.

28 Berger, M. J. and Colella, P.; "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *J. of Comp. Physics*, Vol. 82, pp. 64-84, 1989.

29 Bell, J. B., Berger, M. J., Saltzman, J. S., and Welcome, M. L.; "Three-Dimensional Adaptive Mesh Refinement of Hyperbolic Conservation Laws," UCRL-JC-108794, Lawrence Livermore Natioanl Laboratory, December, 1991.

30 Pember, R. B., Bell, J. B., Colella, P., Crutchfield, W. Y., and Welcome, M. L.; "Adaptive Cartesian Grid Methods for Representing Geometry in Inviscid Compressible Flow," UCRL-JC-113851, Lawrence Livermore Natioanl Laboratory, June, 1993.

31 Steinthorsson, E., Modiano, D. and Colella, P.; "Computations of Unsteady Viscous Flows Using Solution Adaptive Mesh Refinement in Curvilinear Body-Fitted Grid Systems" AIAA-94-2330, June, 1994. Also, ICOMP-94-17, NASA TM 106704.

32 Brislawn, K. D., Brown, D. L., Chesshire, G. S., and Saltzman, J. S.; "Adaptively Refined Overlapping Grids for the Numerical Solutions of Systems of Hyperbolic Conservation Laws," LA-UR-95-257, Los Alamos Natioanl Laboratory.

33 Davis, R. L. and Dannenhoffer, J. F., "Three-Dimensional Adaptive Grid Embedding Euler Technique," AIAA 93-0330, January 1993.

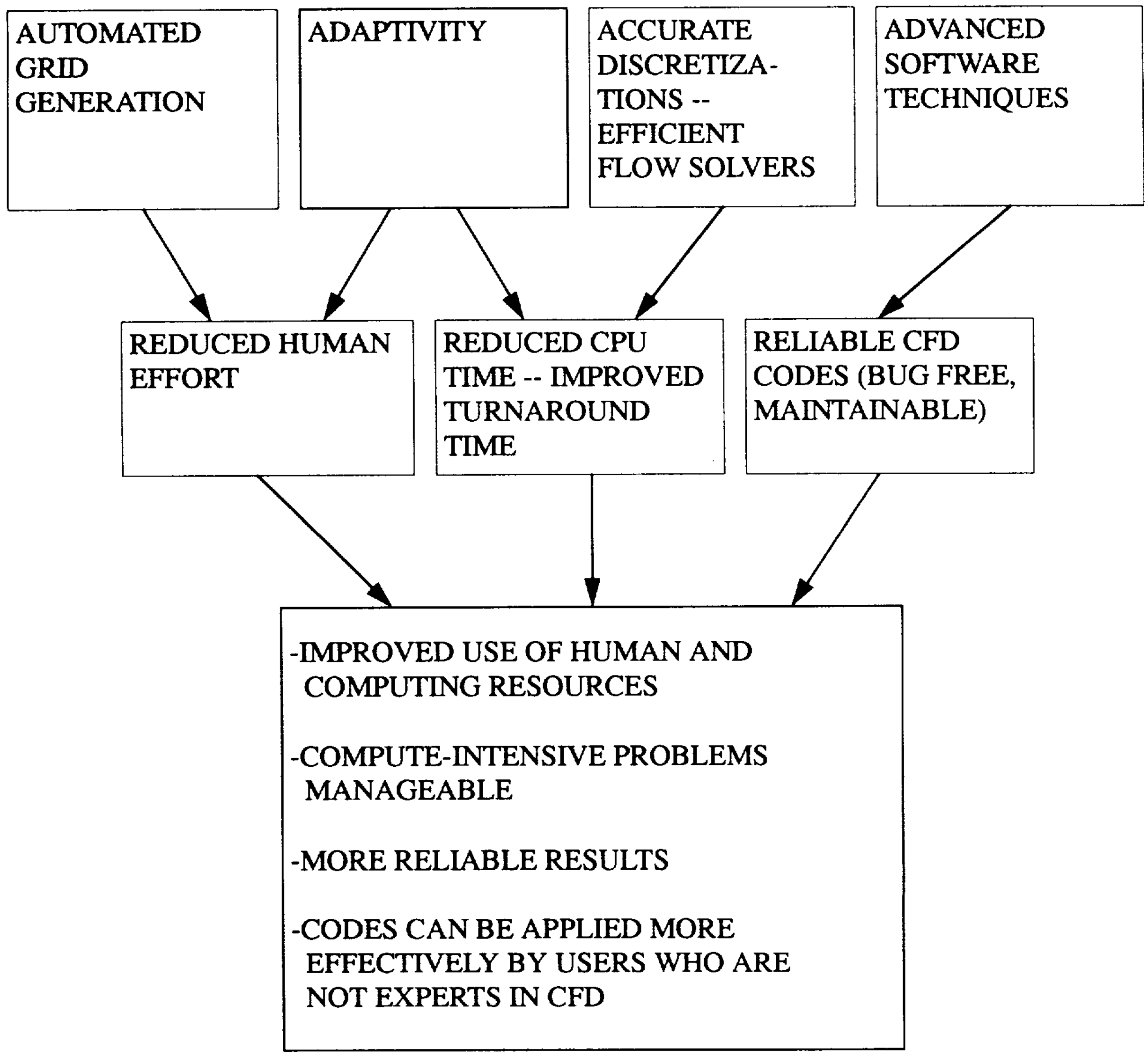
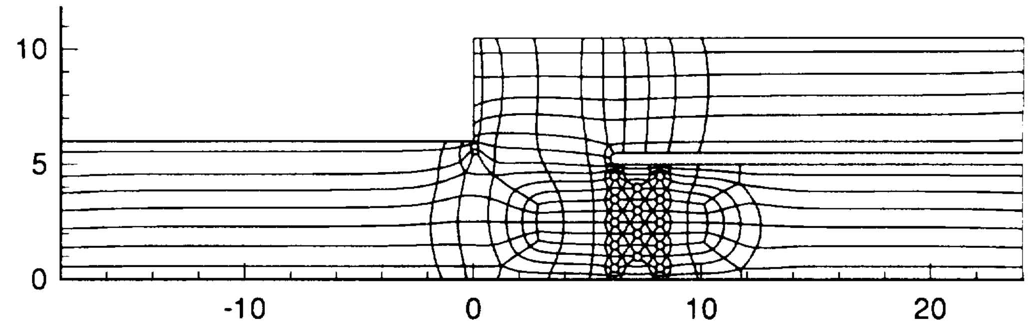
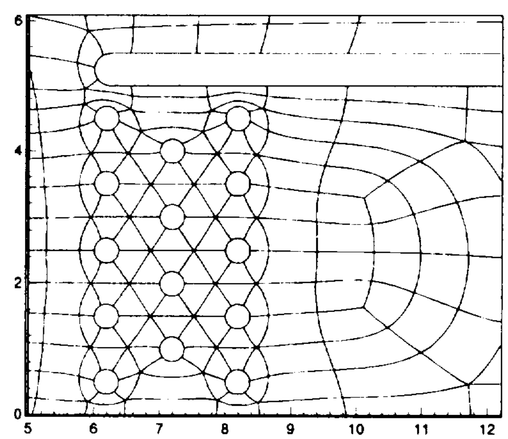


Figure 1 --- Ingredients and impact of an advanced methodology for simulation of viscous flows in complicated geometries



(a)



(b)

Figure 2. Topology of multiblock grid system (a) overall grid, (b) around pin array and partition

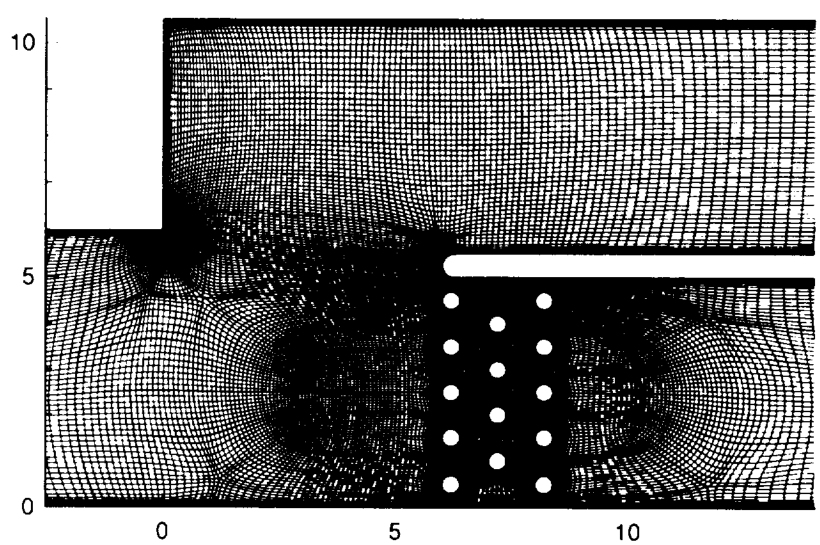


Figure 3. Grid system for the center region of the branched duct.

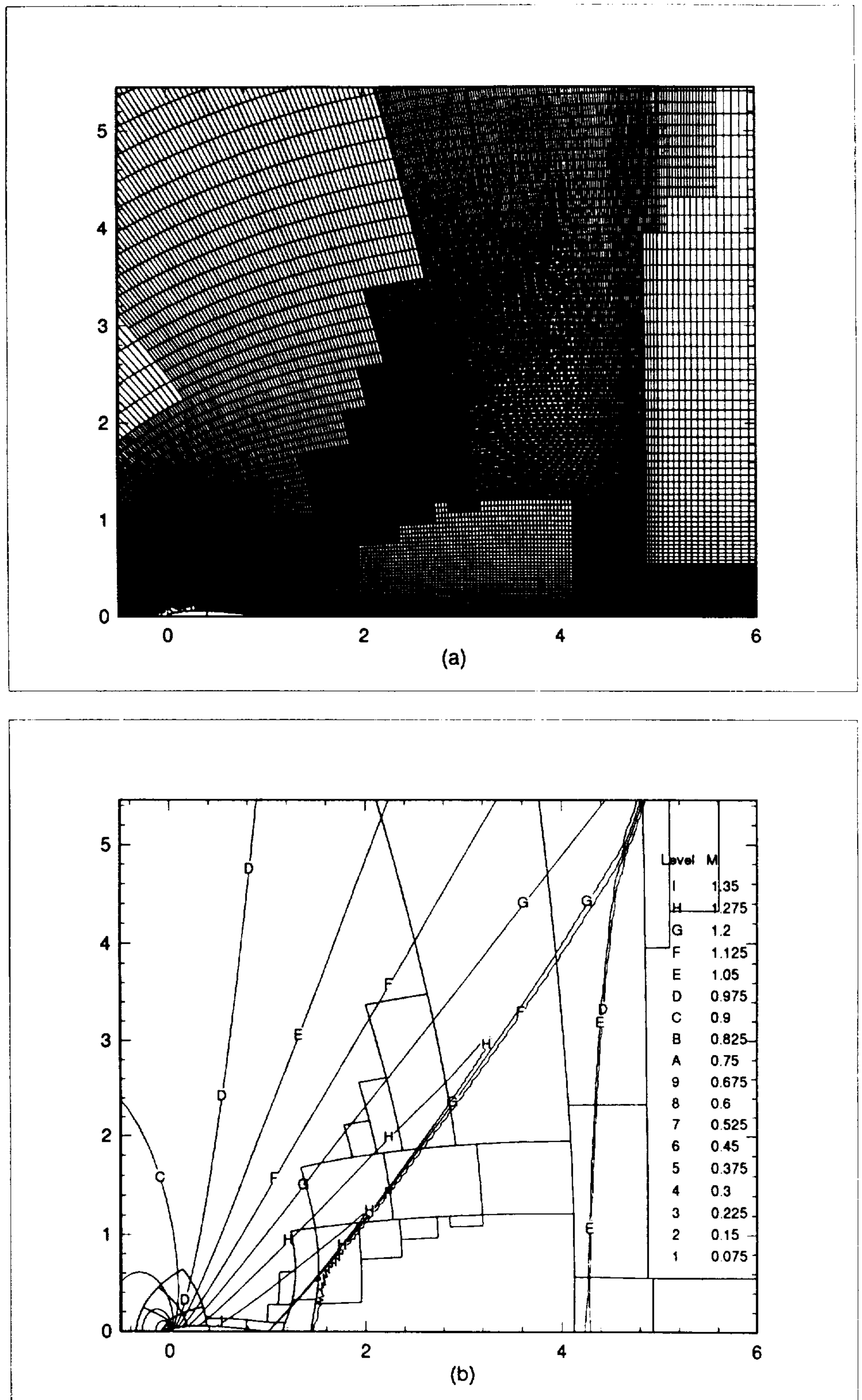


Figure 4. Transonic flow over a NACA0012 airfoil ($M = 0.95$); (a) adapted structured grid system, (b) contours of Mach number.