Conclusion

A numerical model solving the full N–S equation with the Baldwin–Barth one-equation model of turbulence is satisfactorily developed. This numerical method is based on a finite volume, TVD spatial discretization and is integrated by an implicit unfactored method with preconditioning Bi-CGSTAB algorithm matrix solvers with multigrid acceleration. The test cases demonstrate good lift/drag predictive capability, as well as the efficiency and robustness of the convergence for transonic turbulent flow past a RAE2822 subcritical airfoil and a Cast 7 supercritical airfoil.

Acknowledgment

The authors thank the National Science Council of the Republic of China for support under Contract NCHC85-04-014.

References

[1]Hestenes, M. R., and Stiefel, E., "Methods of Conjugate Gradients for Solving Linear Systems," *Journal of Research of the National Bureau of Standards*, Vol. 49, No. 6, 1952, pp. 409–436.

[2]Van der Vorst, H. A., "Bi-CGSTAB: A Fast and Smoothly Convergent Varient of Bi-CG for the Solution of Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing*, Vol. 13, No. 2, 1992, pp. 631–644.

[3]Meijerink, T. A., and van der Vorst, H. A., "Guidelines for the Usage of Incomplete Decompositions in Solving Sets of Linear Equations as They Occur in Practical Problems," *Journal of Computational Physics*, Vol. 44, No. 1, 1981, pp. 134–155.

[4]Lin, H., Yang, D. Y., and Chieng, C. C., "Variants of Biconjugate Gradient Method for Compressible Navier–Stokes Solver," *AIAA Journal*, Vol. 33, No. 7, 1995, pp. 1177–1184.

[5]Thomas, J. L., Taylor, S. L., and Anderson, W. K., "Navier–Stokes Computations of Vortical Flows over Low Aspect Ratio Wings," AIAA Paper 87-0207, Jan. 1987.

[6]Anderson, W. K., Rausch, R. D., and Bonhaus, D. L., "Implicit/Multigrid Algorithms for Incompressible Turbulent Flows on Unstructured Grids," *Journal of Computational Physics*, Vol. 128, No. 2, 1996, pp. 391–408.

[7]Jameson, A., and Baker, T. J., "Multigrid Solution of the Euler Equations for Aircraft Configurations," AIAA Paper 84-0093, Jan. 1984.

[8]Jameson, A., and Yoon, S., "Multigrid Solution of the Euler Equations Using Implicit Schemes," *AIAA Journal*, Vol. 24, No. 11, 1986, pp. 1737–1743.

[9]Goldberg, U. C., and Ramakrishnan, S. V., "A Pointwise Version of Baldwin–Barth Turbulence Model," *Computational Fluid Dynamics*, Vol. 1, 1993, pp. 321–328.

[10]Yee, H. C., and Harten, A., "Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinate," AIAA Paper 85-1513, July 1985.

[11]Sahu, J., and Danberg, J. E., "Navier–Stokes Computations of Transonic Flows with a Two-Equation Turbulence Model," *AIAA Journal*, Vol. 24, No. 11, 1986, pp. 1744–1751.

[12]Siikonen, T., "An Application of Roe's Flux-Difference Splitting for k–ϵ Turbulence Model," *International Journal for Numerical Methods in Fluids*, Vol. 21, No. 11, 1995, pp. 1017–1039.

[13]Yokota, J., "A Diagonally Inverted LU Implicit Multigrid Scheme for the 3-D Navier–Stokes Equations and a Two Equation Model of Turbulence," AIAA Paper 89-0467, Jan. 1989.

[14]Dick, E., and Steelant, J., "Multigrid Solution of the Steady Compressible Navier–Stokes Equations Coupled to the k–ϵ Turbulence Equations," AIAA Paper 95-1669, June 1995.

[15]Cook, P. H., McDonald, M. A., and Firmin, M. C. P., "Aerofoil RAE2822-Pressure Distributions, and Boundary Layer and Wake Measurement," AGARD-AR-138, Rept. A6, May 1979.

[16]Shuen, J. S., and Yoon, S., "Numerical Study of Chemically Reacting Flows Using Lower–Upper Symmetric Successive Overrelaxation Scheme," *AIAA Journal*, Vol. 27, No. 12, 1989, pp. 1752–1760.

[17]Schroder, W., and Hartman, G., "Implicit Solutions of Three-Dimensional Viscous Hypersonic Flows," *Computers and Fluids Journal*, Vol. 21, No. 1, 1992, pp. 109–132.

[18]Shih, T. I.-P., "Approximate Factorization with Source Terms," *AIAA Journal*, Vol. 29, No. 10, 1991, pp. 1759, 1760.

[19]Stanewsky, E., Puffert, W., Muller, R., and Bateman, T. E. B., "Supercritical Airfoil Cast 7—Surface Pressure, Wake and Boundary Layer Measurements," Aircraft Research Association, Bedford, England, UK, AGARD-AR-138, Rept. A3, May 1979.

J. Kallinderis
Associate Editor

Blockwise Adaptive Grids with Multigrid Acceleration for Compressible Flow

Lars Ferm*
Uppsala University, S-75104 Uppsala, Sweden
and
Per Lötstedt†
Saab AB, S-58188 Linköping, Sweden

Introduction

In most programs in practical use for flow calculation, e.g., around an airplane, the grid is partitioned into a number of blocks with a structured grid in each block. The data structure of a multiblock grid allows for a finer grid in certain blocks and a coarser grid in other blocks. With refinement and coarsening of the grid, it is possible to concentrate the cells in areas where they are needed to obtain sufficient accuracy in the final solution.

Adaptive grid procedures for structured grids are developed in Refs. 1–4. Either new cells are introduced into the grid or the available cells are moved to the region where the resolution is poor. The decision of where to change the grid density is based on estimates of the solution error or the truncation error or on a sensor that detects a feature of the flow requiring a finer grid, such as a shock. With a better distribution of the cells, the same solution accuracy is obtained with fewer cells. Memory is saved, and the CPU time is shorter.

The convergence rate to steady state is often improved dramatically by using a multigrid method.[5,6] In practice, the number of levels in complicated multiblock grids is limited to three or four.

Here we combine adaptive grid refinement and coarsening with the multigrid algorithm for efficient calculation of the steady-state solution and apply the method to the Euler equations. All of the cells in a block are refined or coarsened. In this way, no new data structure is introduced, but the total number of cells will sometimes be greater than necessary.

Numerical Solution and Adaptation

The Euler equations are discretized on a structured grid with a cell-centered finite volume method according to Jameson.[7] The grid is partitioned into blocks. Grid refinement in a block is always made by halving h, and coarsening always means doubling h in all three directions of the grid. At the block faces, two extra layers of ghost cells are added to simplify the evaluation of the difference stencil in the cells adjacent to the face, as is usual in multiblock codes. The values in the cells in the overlapping region are determined by the boundary conditions, or there is a neighboring block with cells corresponding to those in the extra layers. In the latter case, the values in the ghost cells are given by the values in the adjacent block. For the solution to remain reasonably accurate and stable also at the block boundary, a ghost cell in the coarser grid occupies the same volume as eight cells in three dimensions in the finer grid. This implies that the grid size is at most doubled between two blocks.

There are two cases at a block interface. The cell face at the boundary in one block coincides with a cell face in the other block. Then the solution values in the ghost cells are copied from the values in the corresponding inner cells in the other block. If a coarse grid is adjacent to a fine grid, then the values in the ghost cells in the coarse grid are computed by a volume-weighted average. In the fine grid, the conservative variables are calculated by trilinear interpolation.

These transfers of the solution between the grids will not make the scheme conservative, and problems may occur if a shock crosses

Received Jan. 16, 1996; revision received Aug. 5, 1998; accepted for publication Aug. 21, 1998.

*Research Scientist, Department of Scientific Computing, P.O. Box 120.
†Specialist, Aerodynamics Department.

a block face with a grid refinement. Our choice of grid coarsening and refinement in the following sections will avoid doubling the grid size at a block face penetrated by a shock. With the same cell size on both sides, the discretization is conservative at the block boundaries, as well as in the interior.

The smoothing iterations are of the Runge–Kutta type as in Ref. 7, with local time steps and with coefficients selected for rapid convergence to the steady-state solution.

The grid is refined and coarsened in the original blocks only, and no extra data structure has to be introduced. We estimate the discretization error in the same way as in Richardson extrapolation (cf. Ref. 1). Let the solution W_h on a grid with a typical grid size h have the numerical error ch^q, where c is independent of h. Then the discretization error is given by ε in

$$\varepsilon = \frac{W_{2h} - W_h}{2^q - 1} \tag{1}$$

The order of the discretization q is not known except for Cartesian grids with constant h. For general grids, the order of accuracy is often lower. At shocks relation (1) is no longer valid, but ε will be large there and acts as a trigger for refinement in a block with a shock. In particular, if a shock crosses a block boundary, then both blocks will have the same fine grid, and conservation is assured.

Multigrid Iteration

The multigrid acceleration is performed with a V-cycle of the FAS algorithm.[5] The restriction from a fine to a coarse grid is computed by volume-weighted averaging, and the prolongation from a coarse to a fine grid is implemented as trilinear interpolation, as in the data transfer between two blocks.

It is important for the efficiency of multigrid iterations that smooth error waves can be propagated through the grid at all levels. The variables in the ghost cells at the block interfaces should be updated as often as possible. This calls for global grids including all blocks at all multigrid levels. The grids at coarser levels are not necessarily uniform; i.e., there may be jumps in the grid size at the block boundaries.

The strategy chosen here to generate the grids at coarser levels is to reduce the number of cells as much as possible when going from a fine to a coarse level by doubling the grid sizes in all blocks. The coarsening continues until the coarsest grid size has been reached in a block.

Some blocks will have the same grid size at the fine and coarse levels. Restriction and prolongation are then the identity operators. Furthermore, the residuals defining the right-hand sides are also identical at the two levels, with one exception. Ghost cell values may be changed by the restriction in a neighboring block. Hence, residuals close to such a boundary must be recalculated.

Three-Grid Coarsening Algorithm

The grid around, e.g., a complex aircraft geometry is determined by special-purpose grid generators. The solid surfaces are defined by a CAD program. Automatic refinement of an existing grid requires access to the CAD surfaces and the capability of generating a smooth grid inside the solver. It is a complicated programming task to accomplish this.

It is much easier to coarsen a grid in a block by removing every second grid point in the three index directions. We assume that a user has generated a grid that is sufficiently fine everywhere. This grid is the basis for all coarser grids. The error estimate determines where to remove grid points.

For estimation of the discretization error and the multigrid iteration, a fine grid with at least three grid levels is needed. Then we have a fine grid, an intermediate grid, and a coarse grid. The final nonuniform grid consists of blocks with grids from the fine, intermediate, and coarse grids.

The three-grid algorithm for solving the equations on the final, adapted grid is as follows:

1) Generate a fine grid with three grid levels.

2) Solve the equations on the coarse grid using single-grid iterations.

3) Prolongate the solution to the intermediate grid.

4) Solve the equations on the intermediate grid using two-grid iterations.

5) Estimate the error in the solution on the intermediate grid.

6) Based on the error estimate, generate the final grid and its connectivity conditions at the block interfaces.

7) Transfer the solution to the final, adapted grid.

8) Solve the equations on the final grid with multigrid iterations.

In the fifth step, the maximum numerical error in the solution in each block on the intermediate grid is estimated as in Eq. (1). To be conservative, we take $q = 1$. Suppose that this error is $\varepsilon_{\max}$ in a block. Choose ε_0, and let $\varepsilon_1 = 0.1\varepsilon_0$. If $\varepsilon_{\max} > \varepsilon_0$, then the fine grid is necessary for an acceptable solution. If $\varepsilon_{\max} < \varepsilon_1$, then the coarse grid is sufficient in the block. If we use the coarse grid and h is doubled, then the error estimate increases by about 2^q, but still $\varepsilon_{\max} < \varepsilon_0$. Otherwise, we keep the intermediate grid in the block.

If a neighbor of a fine grid block has a coarse grid, then the coarse grid is replaced by the intermediate grid so that the quotient between the grid sizes is at most two. The next coarser multigrid level is generated by coarsening blocks with fine or intermediate grids. The coarse level will consist of coarse grids in all blocks. There will be at most three multigrid levels.

Euler Solution Around a Wing

The adaptation algorithm is applied to the solution of the Euler equations around the standard ONERA M6 wing. The fine grid around the wing consists of 0.922×10^6 cells partitioned into 225 blocks of equal size. A small block size is advantageous for the computational speed on processors with limited cache memories. Compared to a single-block grid of the same size, the memory requirements grow by 42% due to the ghost cells. The Mach number is 0.84, and the angle of attack is 3.06 deg. The parameter to control the coarsening and refinement of the intermediate grid is chosen to be $\varepsilon_0 = 3 \times 10^{-2}$. The adapted grid has 0.108×10^6 cells, a reduction of 88% . Even if the blocked grid needs more memory than a single grid, the savings after adaption are substantial.

The solution on the final, adapted grid at 44% of the span is shown in Fig. 1. The c_p values on the adapted grid (solid line) and the fine grid (○) are compared to experimental data (∗).

The solution over the wing in the adapted grid is found in Fig. 2.

The times to reach convergence on one R8000 processor of a Silicon Graphics Power Challenge are compared in Fig. 3 for different iterative algorithms. In the multigrid algorithms, we apply one presmoothing and one postsmoothing iteration and two iterations at the coarse level. The logarithm of the rms differences in ρ per time step is plotted vs the CPU time in seconds.

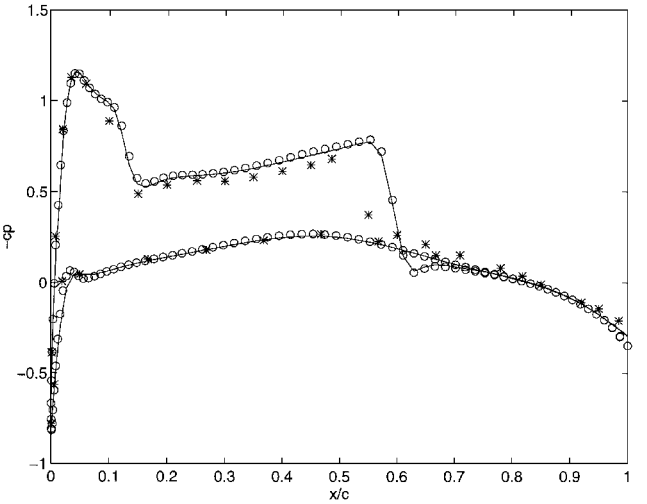

Fig. 1 Comparison of computed and experimental data.

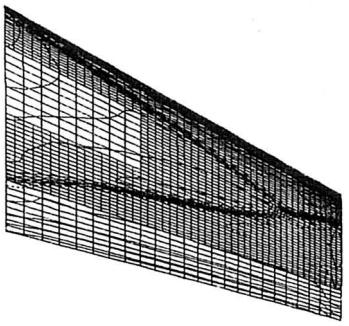

Fig. 2 Adapted grid and computed c_p isobars on the upper side of the wing.

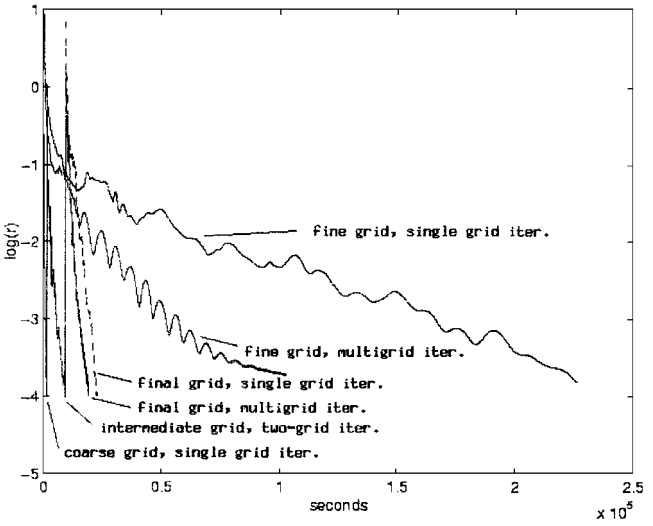

Fig. 3 Iteration on the fine grid compared to the adaptive algorithm.

Conclusion

By adapting the grid with the three-grid algorithm, we reduce not only the number of cells by almost 9/10 but also the CPU time by the same amount compared to single, fine-grid iteration and obtain a solution close to the fine-grid solution. The reduction in CPU time is due to both fewer iterations to convergence and fewer cells. There is a gain in CPU time on the adapted grid with multigrid iteration compared to single-grid iteration. The CPU time is about 35% longer for the single grid.

Acknowledgment

The work of the first author was supported by Swedish Research Council for Engineering Sciences Grant 95-560.

References

[1]Berger, M. J., and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *Journal of Computational Physics*, Vol. 82, No. 1, 1989, pp. 64–84.

[2]Davis, R. L., and Dannenhoffer, J. F., III, "Three-Dimensional Adaptive Grid-Embedding Euler Technique," *AIAA Journal*, Vol. 32, No. 6, 1994, pp. 1167–1174.

[3]Pantelelis, N. G., and Kanarachos, A. E., "The Parallel Block Adaptive Multigrid Method for the Implicit Solution of the Euler Equations," *International Journal for Numerical Methods in Fluids*, Vol. 22, No. 5, 1996, pp. 411–428.

[4]Powell, K. G., "A Tree-Based Adaptive Scheme for Solution of the Equations of Gas Dynamics and Magnetohydrodynamics," *Applied Numerical Mathematics*, Vol. 14, Nos. 1–3, 1994, pp. 327–352.

[5]Brandt, A., "Multi-Level Adaptive Solutions to Boundary-Value Problems," *Mathematics of Computation*, Vol. 31, No. 138, 1977, pp. 333–390.

[6]Jameson, A., "Solution of the Euler Equations for Two Dimensional Transonic Flow by a Multigrid Method," *Applied Mathematics and Computation*, Vol. 13, Nos. 3, 4, 1983, pp. 327–355.

[7]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time-Stepping Schemes," AIAA Paper 81-1259, June 1981.

J. Kallinderis
Associate Editor

Smoothness Improvements of Algebraic Surface Grid Generation

Yih Nen Jeng* and Wei Chung Hsu†
National Cheng Kung University, 70101 Tainan, Taiwan, Republic of China

Introduction

DURING the past decade, the development of computer hardware has significantly enhanced engineering applications of computational fluid dynamics (CFD).[1] As a consequence, the need to generate structured and unstructured volume grids in complicated regions has become more acute. The first step in obtaining a CFD solution is the generation of surface grids from a background grid system.

To the authors' knowledge, the most convenient and fastest method to generate surface grids seems to be the parameter space method developed by Samareh-Abolhassani and Stewart.[2] Assume that a structured quadrilateral background grid system is known, and then define the normalized arc length of the background grid lines and consider the arc lengths along each computational direction to be independent variables. By employing the scaled transfinite interpolation method of Soni,[3] followed by a backward projection to the background grids, a surface grid system can be generated rapidly.

The authors' experience has shown that the method proposed by Samareh-Abolhassani and Stewart[2] works very well for many engineering applications provided that the range of the arc length normalization just covers the target region of surface grid generation. However, if the total arc lengths between two adjacent lines of the background grids differ significantly from each other, numerical tests show that grid line oscillation occurs. These extreme cases correspond to a fin extruding from a smooth surface, where the exact geometry of the fin is not too important in CFD simulation.

To remove the described grid oscillation, three modifications were proposed in Ref. 4. The method that adds a uniform smoothing term to the arc-length parameter space (APS) method proposed in Ref. 2 effectively improves grid smoothness provided that the smoothing factor is large enough. However, as the smoothing factor increases, it approaches the uniform parameter space (UPS) method of Ref. 2, which cannot handle the nonsmoothness of the background grids. To avoid the requirement of a large smoothing factor, four different modifications have been examined. However, only the best modification is presented here due to length limitations.

Generally speaking, a surface definition is frequently obtained from the CAD software that provides the patched grid system. If the patched grids are not of a structured quadrilateral system, the user must reorganize them into a structured quadrilateral background grid system. If the background grid system is not smooth, numerical tests show that neither the methods in Refs. 2–4 nor the modification mentioned earlier can always eliminate this oscillation. Further, numerical tests show that, if the background grid lines do not seriously overlap, it is helpful to smooth the background grids point by point before the surface grid generation is done. Note that such a smoothing may include three stages: projecting neighboring points to a tangent plane passing through a point, finding the new point via local grid smoothing,[5] and projecting the new point backward onto the original surface. Otherwise, reorganization of the background grids is necessary.

Presented as Paper 97-0203 at the AIAA 35th Aerospace Sciences Meeting, Reno, NV, Jan. 6–9, 1997; received April 28, 1997; revision received Aug. 5, 1998; accepted for publication Aug. 21, 1998.

*Professor, Department and Institute of Aeronautics and Astronautics. Member AIAA.

†Graduate Student, Department and Institute of Aeronautics and Astronautics.