MCAT Institute
Final Report
94-004

DEVELOPMENT OF ADVANCED NAVIER-STOKES SOLVER

Seokkwan Yoon

June 1994

NCC2-505

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

I. Introduction

Scope

The objective of research was to develop and validate new computational algorithms for solving the steady and unsteady Euler and Navier-Stokes equations. The end-products are new three-dimensional Euler and Navier-Stokes codes that are faster, more reliable, more accurate, and easier to use.

Approach

The three-dimensional Euler and full/thin-layer Reynolds-averaged Navier-Stokes equations for compressible/incompressible flows are solved on structured hexahedral grids. The Baldwin-Lomax algebraic turbulence model is used for closure. The space discretization is based on a cell-centered finite-volume method augmented by a variety of numerical dissipation models with optional total variation diminishing limiters. The governing equations are integrated in time by an implicit method based on lower-upper factorization and symmetric Gauss-Seidel relaxation. The algorithm is vectorized on diagonal planes of sweep using two-dimensional indices in three dimensions. Convergence rates and the robustness of the codes are enhanced by the use of an implicit full approximation storage multigrid method.

II. Previous Results

Previous results obtained under this cooperative agreement were reported in MCAT Progress Reports submitted in Feb. '93, Oct. '92, Apr. '92, and Oct. '89.

III. Current Work

A new computer program named *CENS3D-MG* has been developed for compressible flows with discontinuities. Details of the code are described in Appendix A and Appendix B.

IV. Summary and Conclusions

The improved computational efficiency of the Navier-Stokes solvers by the use of a highly vectorizable and unconditionally stable algorithm makes fine grid calculations affordable. The resulting code is fast enough to be used as an aerodynamic design tool for advanced subsonic and supersonic transport aircrafts.

APPENDIX - A

Multigrid Convergence of an Implicit Symmetric Relaxation Scheme

Seokkwan Yoon
and
Dochan Kwak
MCAT Institute
Moffett Field
California

AIAA JOURNAL
Vol. 32, No. 5, May 1994

Multigrid Convergence of an Implicit Symmetric Relaxation Scheme

Seokkwan Yoon* and Dochan Kwak†
NASA Ames Research Center, Moffett Field, California 94035

The multigrid method has been applied to an existing three-dimensional compressible Euler solver to accelerate the convergence of the implicit symmetric relaxation scheme. This lower-upper symmetric Gauss-Seidel implicit scheme is shown to be an effective multigrid driver in three dimensions. A grid refinement study is performed including the effects of large cell aspect ratio meshes. Performance figures of the present multigrid code on Cray computers including the new C90 are presented. A reduction of three orders of magnitude in the residual for a three-dimensional transonic inviscid flow using 920 k grid points is obtained in less than 4 min on a Cray C90.

I. Introduction

ALTHOUGH unstructured grid methods have been used successfully in solving the Euler equations for complex geometries, structured grid solvers still remain useful for the Navier-Stokes equations because of their natural advantages in dealing with the highly clustered meshes in the viscous boundary layers. Structured grid methods not only handle reasonably complex geometries using multiple blocks but also offer a hybrid grid scheme to alleviate difficulties which unstructured grid methods have often encountered. Recent developments in structured grid solvers have been focused on efficiency, as well as accuracy, since most existing three-dimensional Navier-Stokes codes are still not efficient enough to be used routinely for aerodynamic design.

Multigrid methods have been useful for accelerating the convergence of iterative schemes. Efficient Euler codes have been developed by Jameson[1] using a full approximation storage method of Brandt[2] in conjunction with the explicit Runge-Kutta scheme. The explicit multigrid method has demonstrated impressive convergence rates by taking large time steps and propagating waves fast on coarse meshes. The explicit multigrid method has been extended to the Navier-Stokes equations by Martinelli.[3] Several explicit multigrid codes for the three-dimensional Navier-Stokes equations have been developed successfully by Vatsa and Wedan,[4] Radespiel et al.,[5] and Turkel et al.[6]

It does not seem to be profitable to consider an unfactored implicit scheme for a multigrid driver since the implicit scheme can take large time steps which are limited by the physics rather than the grid. However, the multigrid method can improve the convergence rates of factored implicit schemes in two dimensions as demonstrated by Yoon[7] and in Refs. 8–10. The implicit multigrid method has been implemented by Caughey[12] to the diagonalized alternating direction scheme,[11] and by Anderson et al.[13] to the three-dimensional alternating direction scheme. Yoon[7] has introduced an implicit algorithm based on a lower-upper (LU) factorization and symmetric Gauss-Seidel (SGS) relaxation. The scheme has been used successfully in computing chemically reacting flows due in part to the algorithm's property which reduces the size of the left-hand side matrix for nonequilibrium flows with finite rate chemistry.[14–16] A recent study[17] shows that the three-dimensional extension of the method using a single grid requires less computational time than most existing codes on a Cray YMP computer. One of the objectives of the present work is to accelerate the convergence of the lower-upper symmetric Gauss-Seidel relaxation scheme in three dimensions by introducing a multigrid technique. The performance of the code is demonstrated for an inviscid transonic flow past an ONERA M6 wing on highly clustered grids.

II. Governing Equations

Let t be time; $\hat{Q}$ the vector of conserved variables; $\hat{E}$, $\hat{F}$, and $\hat{G}$ the convective flux vectors; and $\hat{E}_v$, $\hat{F}_v$, and $\hat{G}_v$ the flux vectors for the viscous terms. Then the three-dimensional Navier-Stokes equations in generalized curvilinear coordinates (ξ, η, ζ) can be written as

$$\partial_t \hat{Q} + \partial_\xi(\hat{E} - \hat{E}_v) + \partial_\eta(\hat{F} - \hat{F}_v) + \partial_\zeta(\hat{G} - \hat{G}_v) = 0 \qquad (1)$$

where the flux vectors are defined in Ref. 17. The Euler equations are obtained by neglecting the viscous terms.

III. Lower-Upper Symmetric Gauss-Seidel Implicit Scheme

An unfactored implicit scheme can be obtained from a nonlinear implicit scheme by linearizing the flux vectors about the previous time step and dropping terms of the second and higher order.

$$[I + \alpha\Delta t(D_\xi \hat{A} + D_\eta \hat{B} + D_\zeta \hat{C})]\delta\hat{Q} = -\Delta t\hat{R} \qquad (2)$$

where the residual $\hat{R}$ is

$$\hat{R} = D_\xi \hat{E} + D_\eta \hat{F} + D_\zeta \hat{G} \qquad (3)$$

and I is the identity matrix. The correction $\hat{Q}^{n+1} - \hat{Q}^n$ is $\delta\hat{Q}$, where n denotes the time level. D_ξ, D_η, and D_ζ are difference operators that approximate ∂_ξ, ∂_η, and ∂_ζ, respectively. $\hat{A}$, $\hat{B}$, and $\hat{C}$ are the Jacobian matrices of the convective flux vectors. For $\alpha = \frac{1}{2}$, the scheme is second-order accurate in time. For other values of α, the time accuracy drops to first order.

An efficient implicit scheme can be derived by combining the advantages of LU factorization and Gauss-Seidel relaxation.

$$LD^{-1}U\delta\hat{Q} = -\Delta t\hat{R} \qquad (4)$$

Received May 17, 1993; presented as Paper 93-3357 at the AIAA Computational Fluid Dynamics Conference, Orlando, FL, July 6–9, 1993; revision received Oct. 4, 1993; accepted for publication Oct. 13, 1993.

*Research Scientist, Computational Algorithms and Applications Branch.
†Chief, Computational Algorithms and Applications Branch.

Here,

$$L = I + \alpha\Delta t(D_\xi^- \hat{A}^+ + D_\eta^- \hat{B}^+ + D_\zeta^- \hat{C}^+ - \hat{A}^- - \hat{B}^- - \hat{C}^-)$$

$$D = I + \alpha\Delta t(\hat{A}^+ - \hat{A}^- + \hat{B}^+ - \hat{B}^- + \hat{C}^+ - \hat{C}^-) \tag{5}$$

$$U = I + \alpha\Delta t(D_\xi^+ \hat{A}^- + D_\eta^+ \hat{B}^- + D_\zeta^+ \hat{C}^- + \hat{A}^+ + \hat{B}^+ + \hat{C}^+)$$

where D_ξ^-, D_η^-, and D_ζ^- are backward difference operators, while D_ξ^+, D_η^+, and D_ζ^+ are forward difference operators.

In the framework of the LU-SGS algorithm, a variety of schemes can be developed by different choices of numerical dissipation models and Jacobian matrices of the flux vectors. Jacobian matrices leading to diagonal dominance are constructed so that + matrices have non-negative eigenvalues whereas − matrices have nonpositive eigenvalues. For example,

$$\hat{A}^\pm = \hat{T}_\xi \Lambda_\xi^\pm \hat{T}_\xi^{-1}$$

$$\hat{B}^\pm = \hat{T}_\eta \Lambda_\eta^\pm \hat{T}_\eta^{-1} \tag{6}$$

$$\hat{C}^\pm = \hat{T}_\zeta \Lambda_\zeta^\pm \hat{T}_\zeta^{-1}$$

where typically $\hat{T}_\xi$ and $\hat{T}_\xi^{-1}$ are similarity transformation matrices of the eigenvectors of $\hat{A}$. Another possibility is to construct the Jacobian matrices of the flux vectors approximately which yield diagonal dominance.

$$\hat{A}^\pm = \tfrac{1}{2}[\hat{A} \pm \tilde{\rho}(\hat{A})I]$$

$$\hat{B}^\pm = \tfrac{1}{2}[\hat{B} \pm \tilde{\rho}(\hat{B})I] \tag{7}$$

$$\hat{C}^\pm = \tfrac{1}{2}[\hat{C} \pm \tilde{\rho}(\hat{C})I]$$

where

$$\tilde{\rho}(\hat{A}) = \kappa \max[|\lambda(\hat{A})|] \tag{8}$$

Here $\lambda(\hat{A})$ represents eigenvalues of the Jacobian matrix $\hat{A}$, and κ is a constant that is greater than or equal to 1. A typical value of κ is 1. However, stability and convergence can be controlled by adjusting κ as the flowfield develops.

IV. Vectorization

The algorithm is completely vectorizable on $i+j+k=$ const diagonal planes of sweep. This is achieved by reordering the arrays.

$$\hat{Q}(i\text{point}, i\text{plane}) = \hat{Q}(i, j, k) \tag{9}$$

where $i$plane is the serial number of the diagonal plane, and $i$point is the address on that plane. The number of diagonal planes is given by

$$n\text{plane} = i\max + j\max + k\max - 5 \tag{10}$$

with the maximum vector length of

$$n\text{point} = (j\max - 1)*(k\max - 1) \tag{11}$$

V. Numerical Dissipation

The cell-centered finite-volume method is augmented by a numerical dissipation model based on blended first- and third-order terms.[18-21] The finite volume method is based on the local flux balance of each mesh cell. For example,

$$\partial_\xi \hat{E} + \partial_\eta \hat{F} + \partial_\zeta \hat{G} = \hat{E}_{i+\frac{1}{2},j,k} - \hat{E}_{i-\frac{1}{2},j,k} + \hat{F}_{i,j+\frac{1}{2},k} - \hat{F}_{i,j-\frac{1}{2},k} + \hat{G}_{i,j,k+\frac{1}{2}} - \hat{G}_{i,j,k-\frac{1}{2}} \tag{12}$$

Here, the conserved variables are averaged at the cell faces to evaluate the fluxes. The dissipative flux d is added to the convective flux in a conservative manner.

$$(\hat{E}_{i+\frac{1}{2},j,k} - \hat{E}_{i-\frac{1}{2},j,k} + \hat{F}_{i,j+\frac{1}{2},k} - \hat{F}_{i,j-\frac{1}{2},k} + \hat{G}_{i,j,k+\frac{1}{2}} - \hat{G}_{i,j,k-\frac{1}{2}}) - (d_{i+\frac{1}{2},j,k} - d_{i-\frac{1}{2},j,k} + d_{i,j+\frac{1}{2},k} - d_{i,j-\frac{1}{2},k} + d_{i,j,k+\frac{1}{2}} - d_{i,j,k-\frac{1}{2}}) \tag{13}$$

For simplicity, $d_{i+\frac{1}{2},j,k}$ is denoted by $d_{i+\frac{1}{2}}$ hereafter.

$$d_{i+\frac{1}{2}} = \epsilon^{(2)}_{i+\frac{1}{2}}(\hat{Q}_{i+1} - \hat{Q}_i) - \epsilon^{(4)}_{i+\frac{1}{2}}(\hat{Q}_{i+2} - 3\hat{Q}_{i+1} + 3\hat{Q}_i - \hat{Q}_{i-1}) \tag{14}$$

The coefficients of the dissipative terms are the directionally scaled spectral radii of the Jacobian matrices. The use of directional scaling provides anisotropic dissipation to each direction, resulting in improved performance on meshes with high aspect ratio cells. Third-order terms formed from fourth differences provide the background damping. First-order terms are added by second differences near shock waves under the control of a sensor $\tilde{\nu}$.

$$\tilde{\nu}_{i+\frac{1}{2}} = \max(\nu_{i+1}, \nu_i) \tag{15}$$

where

$$\nu_i = \max(\nu_i^p, \nu_i^T) \tag{16}$$

$$\nu_i^p = |p_{i+1} - 2p_i + p_{i-1}|/(p_{i+1} + 2p_i + p_{i-1}) \tag{17}$$

$$\nu_i^T = |T_{i+1} - 2T_i + T_{i-1}|/(T_{i+1} + 2T_i + T_{i-1}) \tag{18}$$

Here p and T are the pressure and the temperature. The low-order dissipative coefficient is proportional to the sensor $\tilde{\nu}$ as

$$\epsilon^{(2)}_{i+\frac{1}{2}} = \kappa^{(2)} r(\hat{A})_{i+\frac{1}{2}} \tilde{\nu}_{i+\frac{1}{2}} \tag{19}$$

where $r(\hat{A})$ denotes the spectral radius of the Jacobian matrix $\hat{A}$. The high-order dissipative coefficient is controlled by the sensor.

$$\epsilon^{(4)}_{i+\frac{1}{2}} = \max[0, \kappa^{(4)} r(\hat{A})_{i+\frac{1}{2}} - \epsilon^{(2)}_{i+\frac{1}{2}}] \tag{20}$$

Here $\kappa^{(2)}$ and $\kappa^{(4)}$ are constants which are different from other κ in Eq. (8).

Dissipative terms for the coarse grids in the multigrid levels are formed from second differences with constant coefficients.

Since the constant total enthalpy is not preserved in general except for the Euler equations, the dissipation for the energy equation is based on the total energy rather than the total enthalpy. To reduce the amount of dissipation in the direction normal to the body surface inside boundary layers, Swanson and Turkel[20] provided additional scaling by multiplying a spectral radius in the normal direction by a function of the local Mach number. Although the Mach number scaling technique may improve the accuracy of the Navier-Stokes solutions, it has not been used for the Euler solutions. It has been shown that the convergence rate on high cell aspect ratio meshes can be enhanced by multiplying Martinelli's[3] scaling factor based on local cell aspect ratio to the dissipative coefficients. However, this technique has not been used here because the aspect ratio based scaling factor seems to compromise the accuracy of the solution.[22]

VI. Multigrid Method

In the present multigrid method, part of the task of tracking the evolution of the solution is transferred through a sequence of successively coarser meshes. The use of larger control volumes on the coarser meshes tracks the large-scale evolution,

with the consequence that global equilibrium can be more rapidly attained. This evolution on the coarse grid is driven by the solution of the fine grid equations. The solution vector on a coarse grid is initialized as

$$Q_{2h}^{(0)} = \Sigma S_h Q_h / S_{2h} \tag{21}$$

where the subscripts denote values of the mesh spacing parameter h, S is the cell volume, and the sum is over the eight cells of the fine grid which compose each cell of the coarse grid. After updating the fine grid solution, the values of the conserved variables are transferred to the coarse grid using Eq. (21). The pressure is calculated on the coarse grid using the transferred variables. Then a forcing function is defined as

$$P_{2h} = \Sigma R_h(Q_h) - R_{2h}(Q_{2h}^{(0)}) \tag{22}$$

The residual on the coarse grid is given by

$$R_{2h}^{*} = R_{2h}(Q_{2h}) + P_{2h} \tag{23}$$

For the next coarser grid, the residual is calculated as

$$R_{4h}^{*} = R_{4h}(Q_{4h}) + P_{4h} \tag{24}$$

where

$$P_{4h} = \Sigma R_{2h}^{*} - R_{4h}(Q_{4h}^{(0)}) \tag{25}$$

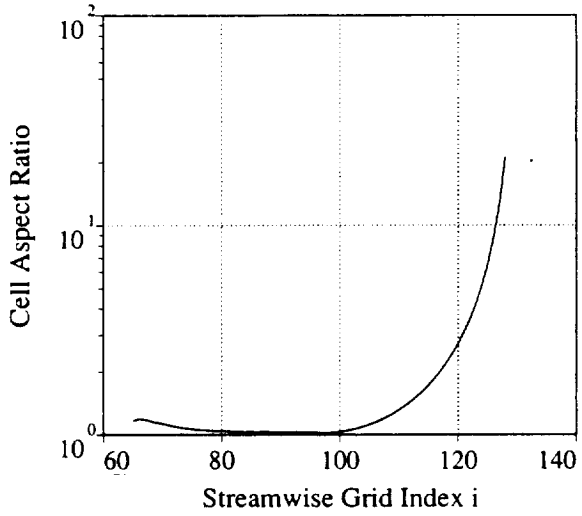


Fig. 1 Distribution of cell aspect ratios (CAR) for the low CAR grid (140 k grid points).

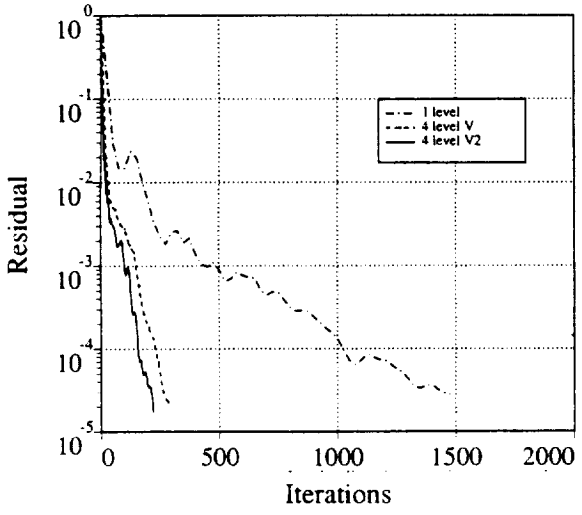


Fig. 2 Convergence histories on the low CAR grid.

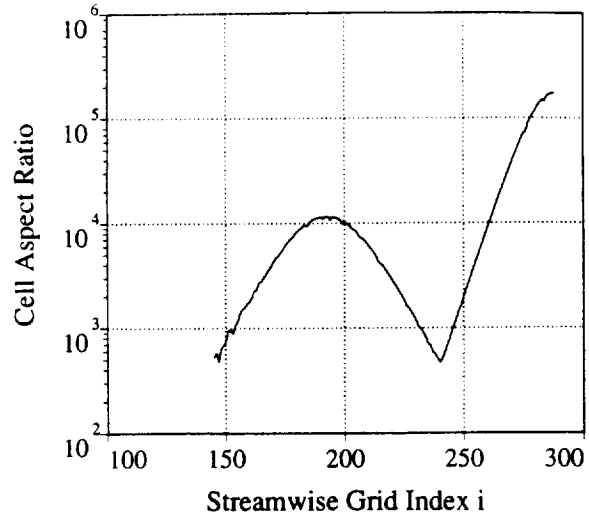


Fig. 3 Distribution of cell aspect ratios for the high CAR grid (920 k grid points).

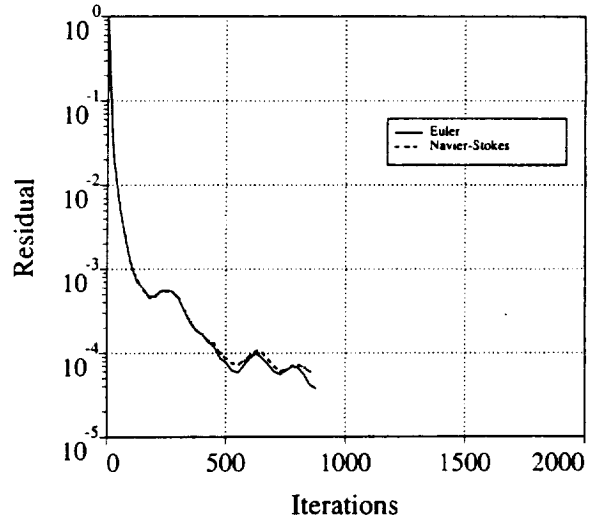


Fig. 4 Convergence histories for the Euler and Navier-Stokes equations.

The process is repeated on successively coarser grids. Multiple iterations can be done on each coarse grid. Finally, the correction calculated on each grid is interpolated back to the next finer grid. Let $\tilde{Q}_{2h}$ be the final value of Q_{2h} resulting from both the correction calculated on grid $2h$ and the correction transferred from the grid $4h$. Then

$$\tilde{Q}_h = Q_h + I_{2h}^{h}(\tilde{Q}_{2h} - Q_{2h}^{(0)}) \tag{26}$$

where Q_h is the solution on grid h before the transfer to the grid $2h$, and I is a trilinear interpolation operator. Since the evolution on a coarse grid is driven by residuals collected from the next finer grid, the final solution on the fine grid is independent of the choice of boundary conditions on the coarse grids.

VII. Results

To investigate the effectiveness of the full approximation storage multigrid method in conjunction with the lower-upper symmetric Gauss-Seidel relaxation scheme, transonic flow calculations have been carried out for an ONERA M6 wing. The freestream conditions are at a Mach number of 0.8395, and a 3.06-deg angle of attack. Since this is an unseparated flow case, only the solution of the three-dimensional Euler equations is considered.

All of the calculations are performed without the aid of a couple of conventional techniques which have been crucial to ensure the robustness of the multigrid method. The aspect ratio based scaling factor for the numerical dissipation is not employed because of the reason stated in Sec. V. The enthalpy damping technique, which is not valid for the Navier-Stokes equations, is not used here.

The treatment of the far-field boundary condition is based on the Riemann invariants for a one-dimensional flow normal to the boundary.

To study the effects of mesh cell aspect ratios on the convergence rates, a low cell aspect ratio $129 \times 33 \times 33$ C-H grid (140,481 points) is used first. Figure 1 shows the distribution of geometric cell aspect ratios of the first normal mesh cells at the body from the leading edge to the downstream boundary. Although geometric aspect ratios are less accurate than spectral radius based aspect ratios, they seem to be useful. The cell aspect ratios are of the order of 1 at the wing surface. Figure 2 shows convergence histories of the root-mean-squared residuals which correspond to the density corrections. The residuals are normalized by their initial values. The chained line indicates the single grid convergence history. The dashed line indicates the multigrid convergence history using a four-level V cycle with one iteration on each coarse grid. The solid line indicates the multigrid convergence history using a four-level V cycle with two iterations on each coarse grid. Clearly, the two iteration strategy converges faster than the single iteration cycle. Although not shown here, the three-iteration cycle does not improve the performance due to a slight increase in work per cycle. Hence, all of the following multigrid calculations have been performed using the V cycle with two coarse grid iterations. Four orders of residual drop requires 1022 and 156 iterations for single and multigrid, respectively. The convergence has been accelerated by a factor of 6.5 by the use of the multigrid method on this low cell aspect ratio grid.

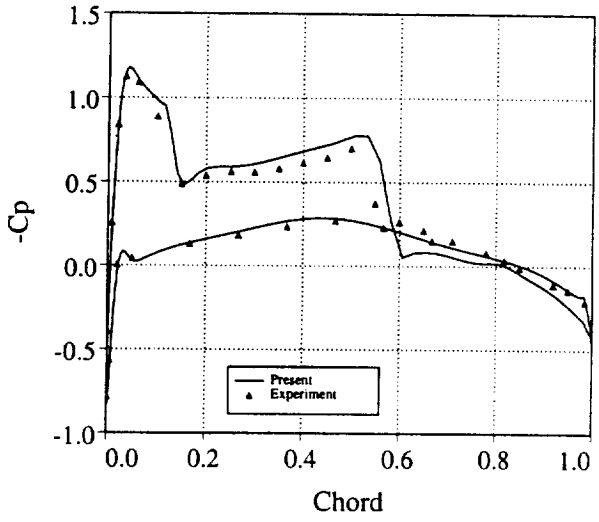


Fig. 5 C_p at 44% span for ONERA M6 wing.

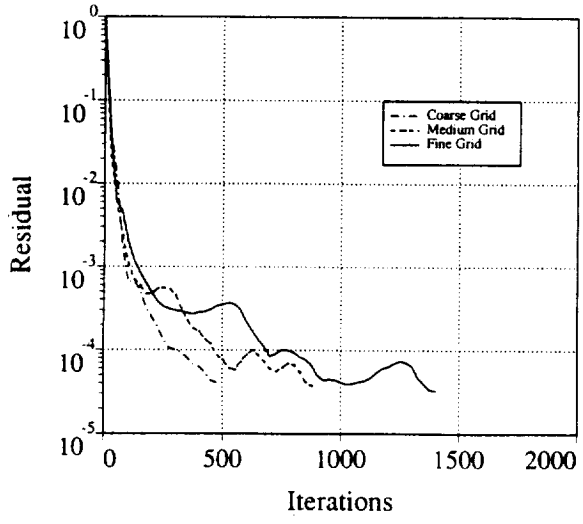


Fig. 6 Single grid convergence histories on the high CAR grids.

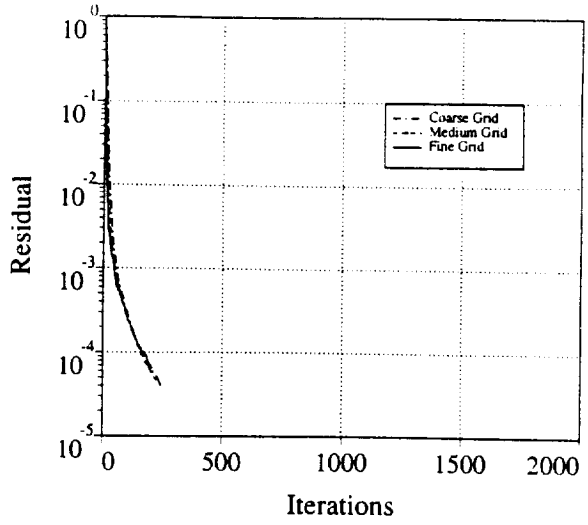


Fig. 7 Multigrid convergence histories on the high CAR grids.

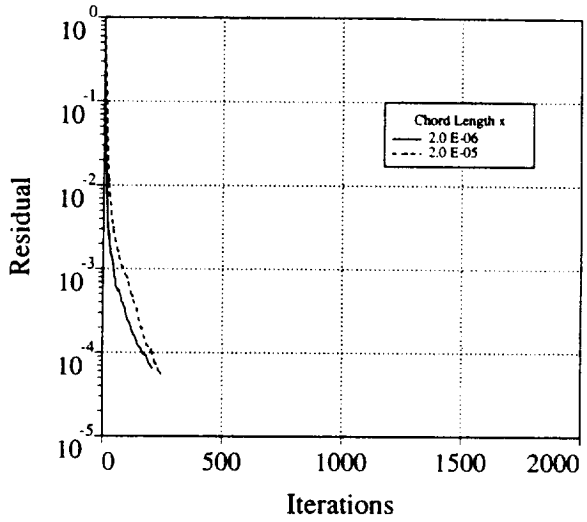


Fig. 8 Effect of the first grid distance on the multigrid convergence.

A high cell aspect ratio $289 \times 65 \times 49$ C-H grid (920,465 points) gives a severe test to the code. Here 65 grid points are used in the normal direction. The distance of the first normal grid point from the wing surface is 2.0×10^{-6} times the chord length. Figure 3 shows that the cell aspect ratios on the wing reach as high as 10,000. This grid will be known as the fine grid. Two more grids are prepared for a grid refinement study. A $145 \times 33 \times 25$ medium grid (119,625 points) is generated by eliminating every other grid point from the fine grid. A $73 \times 17 \times 13$ coarse grid (16,133 points) is generated from the medium grid using the same process. Figure 4 shows single grid convergence histories of both the Euler and the Navier-Stokes equations on the medium grid. For the present attached flow case, there is not much difference between the two. This suggests that the convergence rate is controlled not by the equations but by the grid.

Figure 5 shows good pressure coefficient agreement between experimental data[23] and the multigrid computations on the

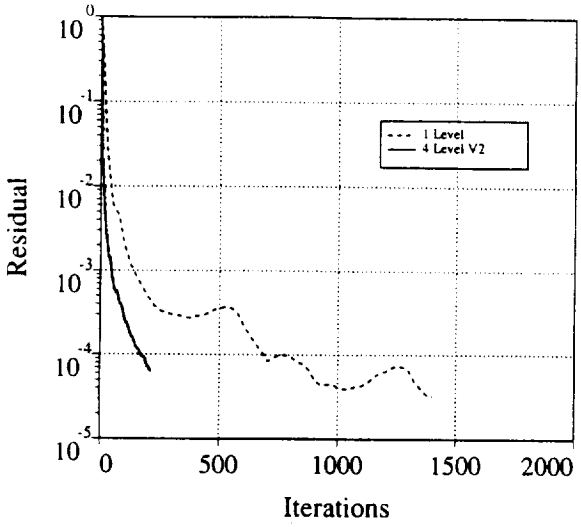


Fig. 9 Convergence histories on the fine grid (iterations).

Table 1 Performance on Cray computers (single processor)

Computer	Grid	CPU[a]	Mflops	Note
YMP	Single	7.4	170	Euler
YMP	Multiple	11.1	160	Euler
C90	Single	3.1	410	Euler
C90	Multiple	4.7	390	Euler

[a]CPU μs per grid point per iteration.

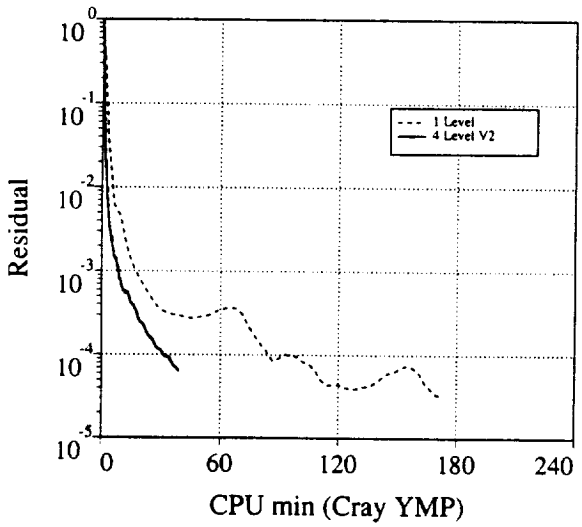


Fig. 10 Convergence histories on the fine grid (CPU time on a Cray YMP).

fine grid at the 44% semispan station. Single grid convergence histories are compared in Fig. 6 and show that the convergence rate slows down significantly as the number of grid points increases. Figure 7 shows that multigrid convergence histories of the three grids are almost identical. Here, the coarse, medium, and fine grids use two-, three-, and four-level V cycles, respectively. The results demonstrate grid independent convergence rates which are achieved by the multigrid method.

To study the effect of grid stretching on the convergence, another fine grid whose first grid distance is 2.0×10^{-5} is generated. Figure 8 shows that the present numerical method is insensitive to grid clustering despite the fact that cell aspect ratios differ by an order of magnitude. The convergence rate on the highly stretched grid appears to be slightly better than the less stretched one for this case. Although not shown here, the rate of convergence slows down significantly after the

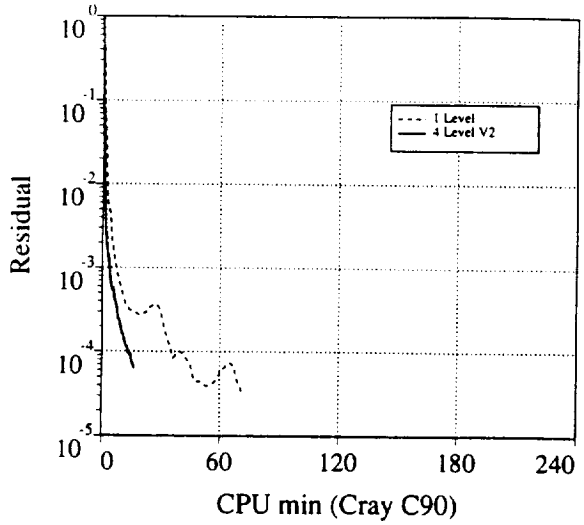


Fig. 11 Convergence histories on the fine grid (CPU time on a Cray C90).

residual drops about five orders of magnitude on the highly stretched grid.

Figure 9 shows that the convergence on the multigrid is about 6.5 times faster than on the single grid in terms of iterations. However, a multigrid cycle requires more time per iteration than a single grid cycle not only because of additional operations for transfer and interpolation but because of the short vector lengths on the coarse grids. As presented in Table 1, the overhead for the multigrid cycle is approximately 50%. Nevertheless, the present CENS3D code requires less than 5 μs per grid point per iteration on a Cray C90 computer. With the single grid mode, the code needs only 3 μs at the sustained rate of 410 Mflops. The CPU times on a Cray YMP and a Cray C90 are compared in Figs. 10 and 11, respectively. The CPU time is reduced by a factor of four by the use of the multigrid method. Figure 11 shows that the residuals for the multigrid drop three orders of magnitude in 50 iterations or 4 CPU min and four orders in 160 iterations or 13 CPU min on a Cray C90.

Conclusions

An efficient three-dimensional multigrid code has been developed for inviscid compressible flows. The lower-upper symmetric Gauss-Seidel implicit scheme is shown to be an effective multigrid driver in three dimensions. The present numerical method appears to be insensitive to grid refinement or to highly clustered grids with high mesh cell aspect ratios. Grid independent convergence rates are achieved by the multigrid method. A reduction of three orders of magnitude in the residual for a three-dimensional transonic inviscid flow using 920 k grid points is obtained in less than 4 min on a Cray C90.

References

[1]Jameson, A., "Solution of the Euler Equations for Two-Dimensional Transonic Flow by a Multigrid Method," *Applied Mathematics and Computation*, Vol. 13, Nov. 1983, pp. 327–356.

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

[3]Martinelli, L., "Calculation of Viscous Flows with a Multigrid Method," Princeton Univ., MAE Rept. 1754T, Princeton, NJ, June 1987.

[4]Vatsa, V. N., and Wedan, B. W., "Development of an Efficient Multigrid Code for 3-D Navier-Stokes Equations," AIAA Paper 89-1791, June 1989.

[5]Radespiel, R., Rossow, C., and Swanson, R. C., "An Efficient Cell-Vertex Multigrid Scheme for the Three-Dimensional Navier-Stokes Equations," AIAA Paper 89-1953, June 1989.

[6]Turkel, E., Swanson, R. C., Vatsa, V. N., and White, J. A.,

"Multigrid for Hypersonic Viscous Two- and Three-Dimensional Flows," AIAA Paper 91-1572, June 1991.

[7]Yoon, S., "Numerical Solution of the Euler Equations by Implicit Methods with Multiple Grids," Princeton Univ., MAE Rept. 1720T, Princeton, NJ, Sept. 1985.

[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]Jameson, A., and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal,* Vol. 25, No. 7, 1987, pp. 929-935.

[10]Yoon, S., and Jameson, A., "Lower-Upper Symmetric-Gauss-Seidel Method for the Euler and Navier-Stokes Equations," *AIAA Journal,* Vol. 26, No. 9, 1988, pp. 1025-1026.

[11]Pulliam, T. H., and Chaussee, D. S., "A Diagonal Form of an Implicit Approximate Factorization Algorithm," *Journal of Computational Physics,* Vol. 39, No. 2, 1981, pp. 347-363.

[12]Caughey, D., "Diagonal Implicit Multigrid Algorithm for the Euler Equations," *AIAA Journal,* Vol. 26, No. 7, 1988, pp. 841-851.

[13]Anderson, W. K., Thomas, J. L., and Rumsey, C. L., "Extension and Applications of Flux Vector Splitting to Unsteady Calculations on Dynamic Meshes," AIAA Paper 87-1152, June 1987.

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

[15]Park, C., and Yoon, S., "Calculation of Real-Gas Effects on Blunt-Body Trim Angles," *AIAA Journal,* Vol. 30, No. 4, 1992, pp. 999-1007.

[16]Park, C., and Yoon, S., "A Fully Coupled Implicit Method for Thermo-Chemical Nonequilibrium Air at Suborbital Flight Speeds," *Journal of Spacecraft and Rockets,* Vol. 28, No. 1, 1991, pp. 31-39.

[17]Yoon, S., and Kwak, D., "Implicit Navier-Stokes Solver for Three-Dimensional Compressible Flows," *AIAA Journal,* Vol. 30, No. 11, 1992, pp. 2653-2659.

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

[19]Pulliam, T. H., "Artificial Dissipation Models for the Euler Equations," *AIAA Journal,* Vol. 24, No. 12, 1986, pp. 1931-1940.

[20]Swanson, R. C., and Turkel, E., "Artificial Dissipation and Central Difference Schemes for the Euler and Navier-Stokes Equations," AIAA Paper 87-1107, June 1987.

[21]Yoon, S., and Kwak, D., "Artificial Dissipation Models for Hypersonic Internal Flow," AIAA Paper 88-3277, July 1988.

[22]Jou, W. H., Wigton, L. B., Allmaras, S. R., Spalart, P. R., and Yu, N. J., "Towards Industrial-Strength Navier-Stokes Codes," Fifth Symposium on Numerical and Physical Aspects of Aerodynamic Flows, Long Beach, CA, Jan. 1992.

[23]Schmitt, V., and Charpin, F., "Pressure Distributions on the ONERA M6 Wing at Transonic Mach Numbers," AGARD AR-138-B1, 1979.

APPENDIX - B

AIAA 93-2965

Multi-Zonal Compressible Navier-Stokes Code With the LU-SGS Scheme

Seokkwan Yoon
and
G.H. Klopfer
MCAT Institute
Moffett Field
California

MULTI-ZONAL NAVIER-STOKES CODE
with the LU-SGS SCHEME

G.H. Klopfer* and S. Yoon*
NASA Ames Research Center
Moffett Field, CA 94035-1000

Abstract

The LU-SGS (lower upper symmetric Gauss Seidel) algorithm has been implemented into the Compressible Navier-Stokes, Finite Volume (CNSFV) code and validated with a multizonal Navier-Stokes simulation of a transonic turbulent flow around an Onera M6 transport wing. The convergence rate and robustness of the code have been improved and the computational cost has been reduced by at least a factor of 2 over the diagonal Beam-Warming scheme.

Introduction

The numerical simulation of the Navier-Stokes equations about complex and realistic aerodynamic configurations with a structured grid requires the use of zonal methods. A popular and fairly efficient numerical scheme used to solve the Navier-Stokes equations is the diagonal implicit Beam-Warming algorithm [1,2]. A finite volume multi-zonal Navier-Stokes code, CNSFV, was developed using this scheme. However the Beam-Warming scheme uses approximate factorization to simplify the matrix inversion process. As a result the convergence properties of the scheme are dependent on the time step and time step scaling chosen, and much user input is required to determine the optimum time step and time step scaling. For a numerical simulation of a realistic aerodynamic configuration, several dozen zones may be required. Choosing the optimum time step and scaling for each of these zones becomes a tedious and time consuming process. For reasons not understood yet, the finite volume formulation of the diagonal Beam-Warming scheme suffers a severe time step restriction. Typically for constant time steps, the largest CFL number that could be exercised were less than ten and much less than that for the initial transients. With a judicious time step scaling, the maximum CFL can be as high as 50 to 75. The time step restriction is not so severe with the finite difference formulation of the diagonal Beam-Warming scheme, where the maximum CFL numbers can be as high as 500. The time step restriction for the finite volume form of the diagonal Beam-Warming scheme is severe enough that the time step is too small to be practical for an unsteady Navier-Stokes code.

A numerical scheme without the above mentioned time step restrictions is the LU-SGS (lower upper symmetric Gauss Seidel) algorithm [3,4]. This scheme has been implemented into the CNSFV code and validated with a multizonal Navier-Stokes simulation of a transonic flow around an Onera M6 transport wing.

Navier-Stokes Equations

The three-dimensional thin-layer Navier-Stokes equations in strong conservation law form in curvilinear coordinates are

$$\partial_\tau \hat{V} Q + \partial_\xi F + \partial_\eta G + \partial_\zeta H$$

$$= Re^{-1}(\partial_\xi F_v + \partial_\eta G_v + \partial_\zeta H_v) \qquad (1)$$

where

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{pmatrix}, F = \begin{pmatrix} \rho U \\ \rho u U + \xi_x \hat{V} p \\ \rho v U + \xi_y \hat{V} p \\ \rho w U + \xi_z \hat{V} p \\ (e+p)U - \xi_t \hat{V} p \end{pmatrix},$$

$$G = \begin{pmatrix} \rho V \\ \rho u V + \eta_x \hat{V} p \\ \rho v V + \eta_y \hat{V} p \\ \rho w V + \eta_z \hat{V} p \\ (e+p)V - \eta_t \hat{V} p \end{pmatrix}, H = \begin{pmatrix} \rho W \\ \rho u W + \zeta_x \hat{V} p \\ \rho v W + \zeta_y \hat{V} p \\ \rho w W + \zeta_z \hat{V} p \\ (e+p)W - \zeta_t \hat{V} p \end{pmatrix} \qquad (2)$$

The contravariant velocity components are defined as

$$\begin{aligned} U &= \hat{V}(\xi_t + \xi_x u + \xi_y v + \xi_z w), \\ V &= \hat{V}(\eta_t + \eta_x u + \eta_y v + \eta_z w), \\ W &= \hat{V}(\zeta_t + \zeta_x u + \zeta_y v + \zeta_z w) \end{aligned} \qquad (3)$$

and the viscous fluxes are given by

$$F_v = \mu \hat{V} \begin{pmatrix} 0 \\ m_1 u_\xi + m_2 \xi_x \\ m_1 v_\xi + m_2 \xi_y \\ m_1 w_\xi + m_2 \xi_z \\ m_1 m_{0\xi} + m_2(\xi_x u + \xi_y v + \xi_z w) \end{pmatrix},$$

$$G_v = \mu \hat{V} \begin{pmatrix} 0 \\ m_3 u_\eta + m_4 \eta_x \\ m_3 v_\eta + m_4 \eta_y \\ m_3 w_\eta + m_4 \eta_z \\ m_3 m_{0\eta} + m_4(\eta_x u + \eta_y v + \eta_z w) \end{pmatrix},$$

* Senior Research Scientist, MCAT Institute, San Jose, CA 95127

$$H_v = \mu \hat{V} \begin{pmatrix} 0 \\ m_5 u_\zeta + m_6 \zeta_x \\ m_5 v_\zeta + m_6 \zeta_y \\ m_5 w_\zeta + m_6 \zeta_z \\ m_5 m_{0\zeta} + m_6(\zeta_x u + \zeta_y v + \zeta_z w) \end{pmatrix} \quad (4)$$

with

$$\begin{aligned} m_0 &= (u^2+v^2+w^2)/2 + (Pr(\gamma-1))^{-1}(a^2), \\ m_1 &= \xi_x^2 + \xi_y^2 + \xi_z^2, \\ m_2 &= (\xi_x u_\xi + \xi_y v_\xi + \xi_z w_\xi)/3, \\ m_3 &= \eta_x^2 + \eta_y^2 + \eta_z^2, \\ m_4 &= (\eta_x u_\eta + \eta_y v_\eta + \eta_z w_\eta)/3, \\ m_5 &= \zeta_x^2 + \zeta_y^2 + \zeta_z^2, \\ m_6 &= (\zeta_x u_\zeta + \zeta_y v_\zeta + \zeta_z w_\zeta)/3 \end{aligned} \quad (5)$$

The pressure is given by the equation of state

$$p = (\gamma - 1)(e - \rho(u^2+v^2+w^2)/2) \quad (6)$$

where γ is the ratio of specific heats. The sound speed is denoted by a. The nondimensional parameters are the Reynolds number Re and the Prandtl number Pr. The coefficient of viscosity μ and thermal conductivity k are decomposed into laminar and turbulent contributions as follows:

$$\mu = \mu_l + \mu_t$$

$$\frac{k}{Pr} = \frac{\mu}{Pr} = \left(\frac{\mu_l}{Pr_l} + \frac{\mu_t}{Pr_t} \right)$$

where Pr_l and Pr_t are the laminar and turbulent Prandtl numbers and $k = \mu$ with the nondimensionalization used in this paper. The standard Baldwin-Lomax turbulent eddy viscosity model [5] is chosen for this study.

The nondimensional parameters chosen for this code are the same as those in the CNS code [6]. Denoting dimensional quantities with a $\tilde{()}$, the normalizing parameters are the freestream density $\tilde{\rho}_\infty$, the freestream sound speed $\tilde{a}_\infty$, the freestream viscosity coefficient $\tilde{\mu}_\infty$, and a characteristic length $\tilde{l}$.

The metrics used above have a different meaning for a finite volume formulation compared to the finite difference formulation of [6]. Referring to a typical finite volume cell as shown in figure 1, the finite volume metrics are defined (see, for example, Vinokur [7]) as

$$\mathbf{s}_{j+\frac{1}{2}} = s_{x,j+\frac{1}{2}}\mathbf{i} + s_{y,j+\frac{1}{2}}\mathbf{j} + s_{z,j+\frac{1}{2}}\mathbf{k}$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_4) \times (\mathbf{r}_8 - \mathbf{r}_4) + (\mathbf{r}_3 - \mathbf{r}_4) \times (\mathbf{r}_7 - \mathbf{r}_4)]$$

$$\mathbf{s}_{k+\frac{1}{2}} = s_{x,k+\frac{1}{2}}\mathbf{i} + s_{y,k+\frac{1}{2}}\mathbf{j} + s_{z,k+\frac{1}{2}}\mathbf{k}$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_2) \times (\mathbf{r}_3 - \mathbf{r}_2) + (\mathbf{r}_6 - \mathbf{r}_2) \times (\mathbf{r}_7 - \mathbf{r}_2)]$$

$$\mathbf{s}_{l+\frac{1}{2}} = s_{x,l+\frac{1}{2}}\mathbf{i} + s_{y,l+\frac{1}{2}}\mathbf{j} + s_{z,l+\frac{1}{2}}\mathbf{k} \quad (7)$$

$$= \frac{1}{2}[(\mathbf{r}_7 - \mathbf{r}_5) \times (\mathbf{r}_6 - \mathbf{r}_5) + (\mathbf{r}_8 - \mathbf{r}_5) \times (\mathbf{r}_7 - \mathbf{r}_5)]$$

The finite volume metrics represent the cell face area normals in each of the curvilinear coordinates (ξ, η, ζ). They are related to the metrics introduced in equations (1 - 5) as follows

$$\xi_{x_i}\hat{V} = s_{x_i,j+\frac{1}{2}}$$

$$\eta_{x_i}\hat{V} = s_{x_i,k+\frac{1}{2}} \quad (8)$$

$$\zeta_{x_i}\hat{V} = s_{x_i,l+\frac{1}{2}}$$

where $x_i = x, y, z$ for $i = 1, 2, 3$. The volume of the computational cell is given by

$$\begin{aligned} \hat{V} = \frac{1}{6}[&(\mathbf{r}_4 - \mathbf{r}_2) \times (\mathbf{r}_3 - \mathbf{r}_1) \cdot (\mathbf{r}_7 - \mathbf{r}_3) \\ &+(\mathbf{r}_2 - \mathbf{r}_5) \times (\mathbf{r}_6 - \mathbf{r}_1) \cdot (\mathbf{r}_7 - \mathbf{r}_6) \\ &+(\mathbf{r}_5 - \mathbf{r}_4) \times (\mathbf{r}_8 - \mathbf{r}_1) \cdot (\mathbf{r}_7 - \mathbf{r}_8)] \end{aligned} \quad (9)$$

and is the finite volume equivalent of the inverse Jacobian of the coordinate transformation in the finite difference formulation of [6].

Numerical Method

The governing equations are integrated in time for both steady and time accurate calculations. The unfactored linear implicit scheme is obtained by linearizing the flux vectors about the previous time and dropping second and higher order terms. The resulting scheme in finite volume form is given by

$$[I + \hat{V}^{-1}\hat{h}(\delta_\xi A^n + \delta_\eta B^n + \delta_\zeta C^n - Re^{-1}(\delta_\xi L^n + \delta_\eta M^n + \delta_\zeta N^n))]$$

$$\cdot \Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \quad (10)$$

where the residual R^n is

$$R^n = [\delta_\xi F^n + \delta_\eta G^n + \delta_\zeta H^n - Re^{-1}(\delta_\xi F_v^n + \delta_\eta G_v^n + \delta_\zeta H_v^n)] \quad (11)$$

The convective flux jacobians A, B, C and the viscous flux jacobians L, M and N are defined in the appendix of [1].

Solving the above equation set by a direct matrix inversion is still not practical for three dimensional problems. However there are several indirect or approximate methods available, including the diagonal Beam-Warming scheme and the lower-upper (LU) factorized scheme of Yoon and Jameson. In a previous study [8,9], the diagonal Beam-Warming scheme was the basic flow solver algorithm for the multi-zonal compressible Navier- Stokes finite volume code, CNSFV. While the scheme worked well for finite difference codes, the performance deteriorated for the finite volume code. The cause of the deterioration is not known, but it is not isolated to the CNSFV code; several other finite volume codes using ADI schemes seem to suffer from the same type of deterioration. Because of this problem, a LU scheme which does not seem to suffer the same problem as the ADI schemes is incorporated into the CNSFV code. A brief description of both the diagonal Beam-Warming and the LU scheme is presented.

Diagonal Beam-Warming Algorithm

With the use of approximate factorization and diagonalization of the flux jacobian matrices, a scalar pentadiagonal algorithm [2] can be derived from eqn (10) as

$$T_\xi[I + \hat{V}^{-1}\hat{h}\delta_\xi\Lambda_\xi]N[I + \hat{V}^{-1}\hat{h}\delta_\eta\Lambda_\eta]P[I + \hat{V}^{-1}\hat{h}\delta_\zeta\Lambda_\zeta]$$

$$\cdot T_\zeta^{-1}\Delta Q^n = R^n \quad (12)$$

where δ_ξ is a central difference operator and $\Delta Q^n = Q^{n+1} - Q^n$ with $Q^{n+1} = Q(t^n + \hat{h})$. The viscous terms are not included in the implicit side. The artificial dissipation is included in both sides and is derived below.

The inviscid flux jacobians are diagonalized as follows:

$$\partial_Q F = A = T_\xi \Lambda_\xi T_\xi^{-1}$$

$$\partial_Q G = B = T_\eta \Lambda_\eta T_\eta^{-1}$$

$$\partial_Q H = C = T_\zeta \Lambda_\zeta T_\zeta^{-1}$$

The T_ξ, T_η, T_ζ are the eigenvector matrices of A, B, C, respectively with $\Lambda_\xi, \Lambda_\eta, \Lambda_\zeta$ as the respective eigenvalues.

We also have

$$N = T_\xi^{-1} T_\eta$$

$$P = T_\eta^{-1} T_\zeta$$

Each of the factors of the implicit operator of equation (12) has an artificial dissipation term added to stabilize the central difference operator. The added term is based on Jameson's nonlinear second and fourth order dissipation and for the ξ-operator takes the form

$$\hat{h}\hat{V}^{-1}\nabla_\xi[\bar{\sigma}_{j+\frac{1}{2}}(\epsilon^{(2)}\Delta_\xi \cdot - \epsilon^{(4)}\Delta_\xi\nabla_\xi\Delta_\xi \cdot)]\Delta Q^n \quad (13)$$

with

$$\epsilon^{(2)} = \kappa_2 max(\gamma_{j+1}, \gamma_j, \gamma_{j-1}) \quad (14)$$

$$\gamma_j = \frac{|p_{j+1} - 2p_j + p_{j-1}|}{|p_{j+1} + 2p_j + p_{j-1}|} \quad (15)$$

$$\epsilon^{(4)} = max(0, \kappa_4 - \epsilon^{(2)}) \quad (16)$$

where κ_2, κ_4 are constants of o(1), and Δ_ξ, ∇_ξ are the forward and backward difference operators. $\bar{\sigma}_{j+\frac{1}{2}}$ is a modified spectral radius defined as

$$\bar{\sigma}_{j+\frac{1}{2}} = \bar{\sigma}_j + \bar{\sigma}_{j+1}$$

$$\bar{\sigma}_j = \sigma_j(1 + \sqrt{max(\sigma_k/\sigma_j, \sigma_l/\sigma_j)}), \quad (17)$$

$$\sigma_j = [|U| + a\sqrt{s_x^2 + s_y^2 + s_z^2}]_j \quad (18)$$

and where cell centered surface areas are used, e.g.

$$s_{x,j} = \frac{1}{2}(s_{x,j+\frac{1}{2}} + s_{x,j-\frac{1}{2}}) \quad (19)$$

The modified form of the spectral radius, equation (17), is suggested by Turkel [10] to account for large aspect ratio computational cells as for example in a viscous layer. Similar dissipation terms are obtained for the $\eta-$ and ζ-operators. The dissipation terms added to the right hand side of equation (10) are identical to those given above except that ΔQ^n is replaced by Q^n.

LU-SGS Scheme

The unfactored scheme of eqn (10) is given in terms of central differences for the implicit operator. It can also be represented in terms of upwinded differences. Dropping

the viscous terms in the implicit operator, equation (10) is in terms of upwind differences as follows:

$$[I + \hat{V}^{-1}\hat{h}(\delta_\xi^- A^- + \delta_\xi^- A^- + \delta_\eta^- B^+ + \delta_\eta^+ B^- + \delta_\zeta^- C^+ + \delta_\zeta^+ C^-)]$$

$$\cdot \Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \tag{20}$$

where δ_ξ^+ and δ_ξ^- are the forward and backward difference operators. Similarly, A^+ and A^- are the Jacobian matrices which contain non-negative and non-positive eigenvalues, respectively.

The Yoon and Jameson version of the LU scheme can be obtained from the above equation by a simple reordering of the matrix elements and approximately factoring into two matrices. Define D, $\tilde{L}$, and $\tilde{U}$ to be matrices which contain the diagonal, sub-diagonal and super-diagonal elements of the implicit operator of equation (20), respectively.

$$[D + \tilde{L} + \tilde{U}]\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n$$

which can be factored into

$$[D + \tilde{L}]D^{-1}[D + \tilde{U}]\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n$$

Redefine $L = D + \tilde{L}$ and $U = D + \tilde{U}$ to yield the final form of the LU-SGS scheme.

$$LD^{-1}U\Delta Q^n = -\hat{V}^{-1}\hat{h}R^n \tag{21}$$

where

$$L = I + \hat{V}^{-1}\hat{h}(\delta_\xi^- A^+ + \delta_\eta^- B^+ + \delta_\zeta^- C^+ - A^- - B^- - C^-)$$

$$D = I + \hat{V}^{-1}\hat{h}(A^+ - A^- + B^+ - B^- + C^+ - C^-) \tag{22}$$

$$U = I + \hat{V}^{-1}\hat{h}(\delta_\xi^+ A^- + \delta_\eta^+ B^- + \delta_\zeta^+ C^- + A^+ + B^+ + C^+)$$

A variety of LU-SGS schemes can be obtained by different choices of the numerical dissipation models and Jacobian matrices of the flux vectors. In this study we use the same artificial dissipation described for the diagonal Beam-Warming scheme. For robustness and to ensure that the scheme converges to a steady state, the matrices should be diagonally dominant. To ensure diagonal dominance, the Jacobian matrices can be constructed so that + matrices have nonnegative eigenvalues and − matrices have nonpositive eigenvalues. For example, the diagonalization used for the Beam-Warming scheme can be used to obtain the ± matrices.

$$A^\pm = T_\xi \Lambda_\xi^\pm T_\xi^{-1}$$

$$B^\pm = T_\eta \Lambda_\eta^\pm T_\eta^{-1} \tag{23}$$

$$C^\pm = T_\zeta \Lambda_\zeta^\pm T_\zeta^{-1}$$

Another method to obtain diagonal dominance is to construct approximate Jacobian matrices:

$$A^\pm = [A \pm \tilde{\rho}(A)I]/2$$

$$B^\pm = [B \pm \tilde{\rho}(B)I]/2 \tag{24}$$

$$C^\pm = [C \pm \tilde{\rho}(C)I]/2$$

where $\tilde{\rho}(A) = max[|\lambda(A)|]$ and represent a spectral radius of the Jacobian matrix A with the eigenvalues $\lambda(A)$. These eigenvalues are, e.g., $\lambda(A) = [U, U, U, U \pm a\sqrt{s_x^2 + s_y^2 + s_z^2}]_j$, where the metric terms are again defined at cell centers by Eq. (19). Other methods of increasing the diagonal dominance of the LU operator include the addition some approximation of the viscous terms and the artificial dissipation in the implicit operator [11].

The inversion of equation (21) is done in three steps. The block inversion along the diagonal is eliminated if the approximate Jacobian of Eqn (24) are used instead of Eqn (23). A Newton-type of iteration is obtained simply by setting $\hat{h} = \infty$. Eliminating the time step from the algorithm provides the practical advantage of side-stepping the need to find an optimal CFL number or the time step scaling to reduce the overall computational effort to achieve steady state solutions. As mentioned previously, this is an important consideration for a multizonal code. If first-order one-sided differences are used, Eq. (22) reduces to

$$L = \tilde{\rho}I - A^+_{j-1,k,l} - B^+_{j,k-1,l} - C^+_{j,k,l-1}$$

$$D = \tilde{\rho}I \tag{25}$$

$$U = \tilde{\rho}I + A^-_{j+1,k,l} + B^-_{j,k+1,l} + C^-_{j,k,l+1}$$

where

$$\tilde{\rho} = \tilde{\rho}(A) + \tilde{\rho}(B) + \tilde{\rho}(C)$$

This algorithm requires only scalar diagonal inversions since the diagonal of L or $U = D$. The true Jacobians matrices of Eq. (23) may permit better convergence rates, but require block diagonal inversions with approximately twice the computational effort per iteration. This study considers the scalar version without the viscous or artificial dissipation only.

The scheme is completely vectorizable on $j + k + l =$ constant oblique planes of sweep, Fig. 2. This can be achieved by reordering the three dimensional arrays into two-dimensional arrays as follows:

$$P(i_p, p) = P(j, k, l)$$

where i_p is the serial number of the oblique plane of sweep and p is the address of point (j, k, l) in that plane. The specific value of p in terms of (j, k, l) is

$$p = j + k + l - 2$$

and the number of points in the plane p is

$$n_p = \frac{1}{2}p(p-1)$$

$$-\frac{1}{4}[(p - j_{max})(p - j_{max} + 1)(1 + isign(1, p - j_{max} - 1))$$

$$+(p - k_{max})(p - k_{max} + 1)(1 + isign(1, p - k_{max} - 1))$$

$$+(p - l_{max})(p - l_{max} + 1)(1 + isign(1, p - l_{max} - 1))$$

$$-(p - k_{max} - l_{max})(p - k_{max} - l_{max} + 1)$$

$$\cdot(1 + isign(1, p - k_{max} - l_{max} - 1))$$

$$-(p - j_{max} - l_{max})(p - j_{max} - l_{max} + 1)$$

$$\cdot(1 + isign(1, p - j_{max} - l_{max} - 1))$$

$$-(p - j_{max} - k_{max})(p - j_{max} - k_{max} + 1)$$

$$\cdot(1 + isign(1, p - j_{max} - k_{max} - 1))]$$

The maximum number of points in the planes is $n_{p,max} = n_p(p_{\frac{1}{2}})$, where $p_{\frac{1}{2}} = (p_{max} + 1)/2$. The vector lengths change from 1 at the beginning of the inversion sweep to a maximum of $n_{p,max}$ at the center of the sweep and back to 1 at the end. While $n_{p,max}$ can be large, the vector lengths at the beginning and end of the sweep are very small and inhibit efficient vectorization. Nevertheless for reasonably-sized zones, $n_{p,max}$ is sufficiently large so that good vectorization can be achieved. Other reordering sequences are possible, but additional sweeps will then be necessary and the overall performance of the LU-SGS inversion is less. In addition, the oblique planes of sweep allow for very efficient autotasking procedures on multi-processor computers; the efficiency is less with other sweep directions.

Boundary Conditions and Zonal Interfaces

To complete the equation set boundary conditions must be specified. With the use of curvilinear coordinates the physical boundaries have been mapped into computational boundaries, which simplifies the application of boundary conditions. The boundary conditions to be implemented for external viscous or inviscid flows include (1) inflow or far field, (2) outflow, (3) inviscid and (4) viscous impermeable wall, and (5) symmetry conditions. For external three-dimensional flow fields about closed bodies, the topology of the grid usually introduces (6) grid singularities which require special boundary conditions. The use of zonal methods can avoid the generation of grid singularities, but requires (7) special zonal interface boundary conditions. For compressible flows these zonal boundary conditions must be conservative to maintain global conservation. In this study we consider nonconservative interfaces. However, conservative interzonal communication has been developed and reported in a previous study [8].

A patched zonal procedure is used for the grid system. The zonal interfaces are general three-dimensional surfaces. The zoning procedure has a general capability so that a face of one zone can be in contact with several other zones. Information is transfered between zones with a bilinear interpolation procedure. The search method for determining the interpolation coefficients is automatic in that no user input is required other than identifying the zonal faces in contact with each other. The grid topology of the various zones is arbitrary and the zonal interfaces are not, in general, mesh continuous. In this study we use the ghost cell concept to obtain boundary and interface fluxes. Although there is no grid overlap at the zonal interfaces, the cell-centered conserved variables are reflected across surface and zonal boundaries and are determined by interpolating from the the adjacent zonal values. To determine the dissipative fluxes for the fourth-order dissipation requires two ghost cells. However, here we simply reduce the fourth-order dissipation to second-order so that only one ghost cell is needed.

Results

The test case chosen to validate the LU-SGS scheme and the code is the attached turbulent transonic flow over an ONERA M6 wing. This case has good experimental data [12] available and has been used extensively for Navier-Stokes code validation. The reason for choosing an attached flow case instead of a separated flow is that the emphasis of this study is to validate the flow solver and not the turbulence model. For attached flows the Baldwin-Lomax model [5] is sufficiently accurate. The flow conditons are $M_\infty = 0.84$, $\alpha = 3.06^0$, $Re_{mac} = 11.72 \times 10^6$. The wing surfaces are adiabatic and the fluid is a perfect gas with $\gamma = 1.4$. The computational grid consists of a C-O mesh split into four zones. There are two zones around the wing, one each on the upper and lower surface of the wing. Similarly, the wake region is split into an upper and lower grids. This grid, in the single zone version, seems to be the standard grid used for various validations of Navier-Stokes codes, see, e.g., ref. 13. Figure 3 shows a partial view of the grids and the zones. The total grid size is $193 \times 34 \times 49$ points, and the individual zones consist of $73 \times 34 \times 49$ and $25 \times 34 \times 49$ points for the wing and wake zones, respectively. The wall normal grid spacing at the surface is on the order of $y^+ = 4$. This is sufficient to properly resolve the boundary layer.

The converged results in terms of pressure contours on the upper surface of the wing are compared with experimental data [12] are shown in Fig. 4. The agreement between computation and experiment is good for the four span stations shown. These results are comparable to the results reported in ref. 13. The numerical predicted shocks are slightly smeared as compared to experiment. This is to

be expected since the numerical scheme uses central differencing with nonlinear scalar dissipation. Increased mesh resolution or an upwind scheme will sharpen the shock wave structure.

Figure 5 presents the performance comparison between the diagonal Beam-Warming scheme and the LU-SGS scheme. These views show the convergence rate for the upper wing zone with the two schemes. The first view shows that the LU-SGS scheme is not that much faster than the diagonal Beam-Warming scheme in terms of iterations required to reach convergence. However, in terms of cpu times, the LU-SGS scheme is at least twice as efficient, as shown in the second view.

Most of the gains in efficiency with the LU-SGS scheme is in the reduced operation count. Table 1 shows the unit cpu time (μsec/ gridpoint/ iteration) for both schemes, as measured on the NASA-Ames Cray C90 computer. The unit cpu time for the right hand side (RHS) includes the calculation required for the RHS fluxes, boundary conditions, interface interpolation, swapping data in/out of machine core for each zone at each time step, writeout of convergence data, and other miscellaneous operations. This unit time is the same for both schemes at $4.5\mu sec$. The left hand side (implicit) unit cpu time is substantially different. The times are $5.9\mu sec$ for the diagonal Beam-Warming and $2.2\mu sec$ for the LU-SGS scheme. The total times are $6.7\mu sec$ and $10.4\mu sec$ for the LU-SGS and diagonal Beam-Warming schemes, respectively. Furthermore, the diagonal Beam-Warming scheme required user intervention to obtain the optimum time step size as well as the time step scaling. The LU-SGS scheme uses no time step scaling and an infinitely large time step as explained in LU-SGS section above.

Conclusions

A multi-zonal compressible Navier-Stokes code has been improved by replacing the implicit diagonal Beam-Warming algorithm with the lower-upper symmetric Gauss Seidel scheme. With the new scheme the code is now much more robust, requires no user intervention to determine the optimum time step or time step scaling, converges faster, and requires only half the cpu time to obtain the same level of convergence. Most of the gain in efficiency is due to the lower operation count required by the LU-SGS algorithm. The algorithm and code have been validated with a turbulent simulation of an attached transonic flow around an Onera M6 wing.

Acknowledgements

The first author's work was supported by NASA Grant NCC 2-616 and the second author's by NCC 2-505.

References

1. Beam, R. and Warming, R. F.; "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations." AIAA Journal, Vol. 16, Apr. 1978, pp. 393-402.

2. Pulliam, T. H. and Chaussee, D. S.; "A Diagonal Form of an Implicit Approximate Factorization Algorithm." Journal of Computational Physics, Vol. 39, 1981, pp. 347-363.

3. Yoon, S. and Jameson, A.; "Lower-Upper Symmetric Gauss Seidel Method for the Euler and Navier-Stokes Equations," AIAA Journal, Vol. 26, Sept. 1987, pp. 1025-1026.

4. Yoon, S. and Kwak, D.; "An Implicit Three-Dimensional Navier-Stokes Solver for Compressible Flow," AIAA Paper 91-1555-CP, June 1991.

5. Baldwin, B. S. and Lomax, H.; "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flow," AIAA Paper 78-0257, Jan. 1978.

6. Flores, J. and Chaderjian, N. M.; "Zonal Navier-Stokes Methodology for Flow Simulations about Complete Aircraft," Journal of Aircraft, Vol. 27, No. 7, July 1990, pp. 583-590.

7. Vinokur, M.; "An Analysis of Finite Difference and Finite Volume Formulation of Conservation Laws," NASA CR-177416, June 1986.

8. Klopfer, G. H. and Molvik, G. A.; "Conservative Multizonal Interface Algorithm for the 3-D Navier-Stokes Equations," AIAA Paper 91-1601-CP, June 1991.

9. Chaussee, D. S. and Klopfer, G. H.; "The Numerical Study fo 3-D Flow Past Control Surfaces," AIAA Paper 92-4650, August 1992.

10. Turkel, E. and Vatsa, V. N.; "Effect of Artificial Viscosity on Three-Dimensional Flow Solutions," AIAA Paper 90-1444, June 1990.

11. Shih, T. I-P., Steinthorsen, E., and Chyu, W. J.; "Implicit Treatment of Diffusion Terms in Lower-Upper Algorithms," AIAA J., Vol. 31. No. 4, 1993, pp. 788-791.

12. Schmitt, V. and Charpin, F.; "Pressure Distributions on the ONERA M6 Wing at Transonic Mach Numbers," AGARD-AR-138, Report of the Fluid Dynamics Panel, May 1979.

13. Rumsey, C. L. and Vatsa, V. N.; "A Comparison of the Predictive Capabilities of Several Turbulent Models Using Upwind and Central-Difference Computer Codes," AIAA Paper 93-0192, Jan. 1993.

Scheme	CNSFV - diag BW	CNSFV - LUSGS
Total	10.4μsec (340 mflop)	6.7μsec (330 mflop)*
LHS	5.9μsec xiinv etainv zetainv vpenta, etc	2.2μsec lusgs (oblique planes of sweeps)
RHS, etc.	4.5μsec	4.5μsec

* average vector length ~70, optimum length ~128

Table 1. Perftrace performance comparison of diagonal Beam-Warming and LU-SGS schemes on Cray C90 computer. (unit cpu time μsec/grid point/iteration)

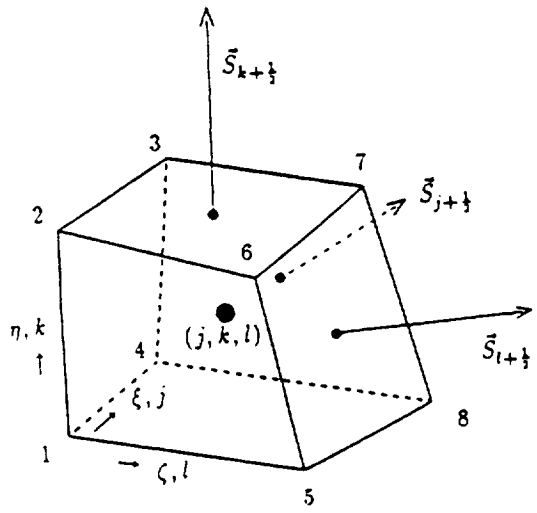


Fig. 1. Finite volume cell nomenclature.

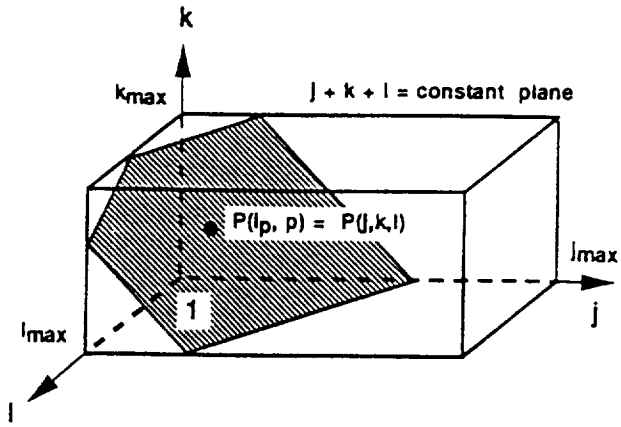


Fig. 2. Oblique plane of sweep for vectorization.

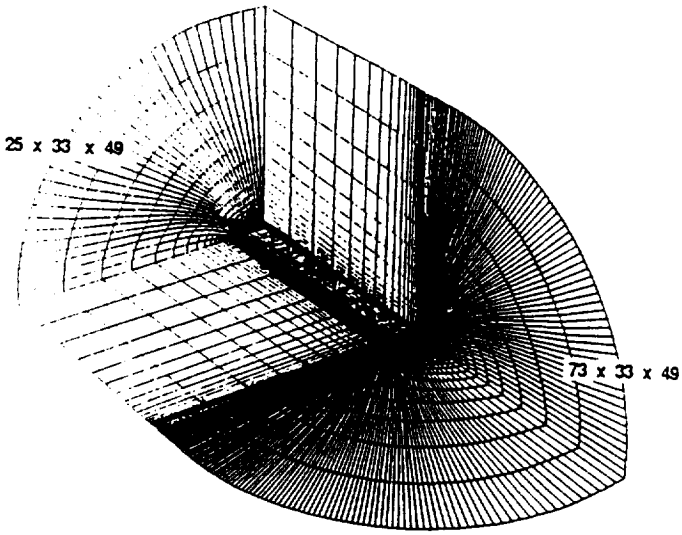
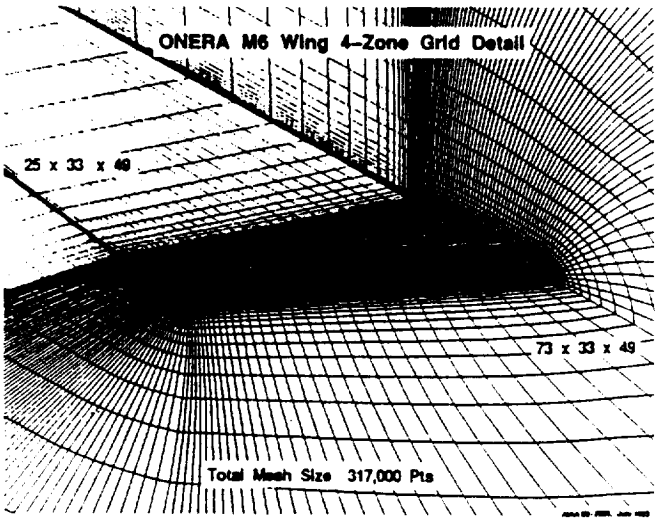


Fig. 3. Four zone C-O Grid for the ONERA M6 wing; only the two zones on the upper surface of wing and wake are shown.



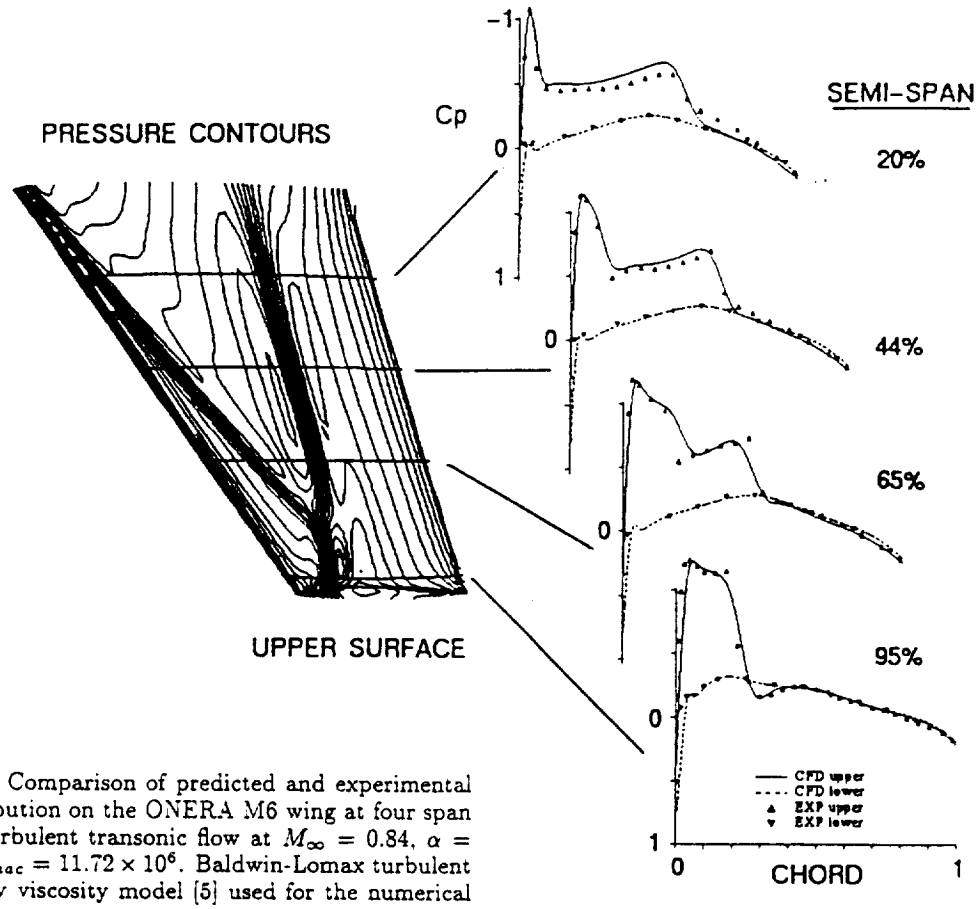


Fig. 4. Comparison of predicted and experimental [12] C_p distribution on the ONERA M6 wing at four span locations. Turbulent transonic flow at $M_\infty = 0.84$, $\alpha = 3.06$, and $Re_{mac} = 11.72 \times 10^6$. Baldwin-Lomax turbulent algebraic eddy viscosity model [5] used for the numerical prediction.

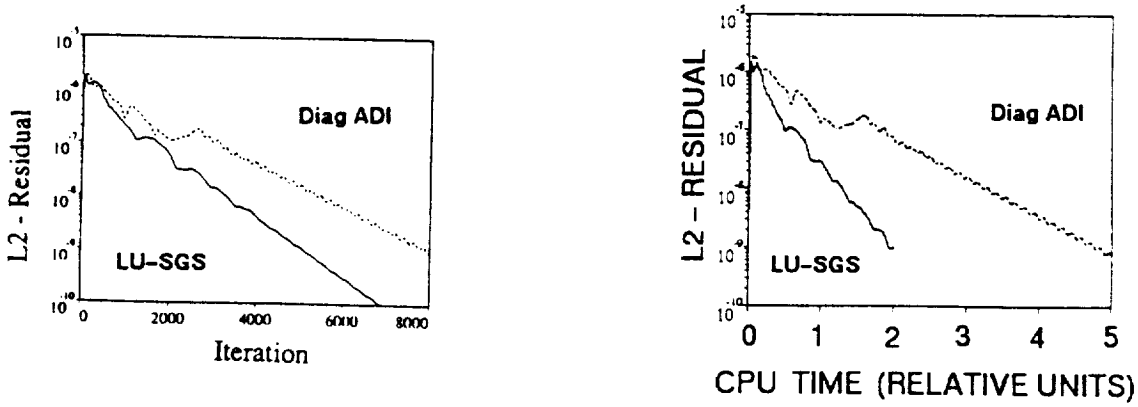


Fig. 5. Comparison of the convergence rates versus iteration and cpu times between the diagonal Beam-Warming and LU-SGS schemes.