Newton Solution of Inviscid and Viscous Problems

V. Venkatakrishnan*
Analytical Services and Materials, Inc., Hampton, Virginia

The application of Newton iteration to inviscid and viscous airfoil calculations is examined. Spatial discretization is performed using upwind differences with split fluxes. The system of linear equations that arises as a result of linearization in time is solved directly by using either a banded or a sparse matrix solver. In the latter case, the solver is used in conjunction with the nested dissection strategy, whose implementation for airfoil calculations is discussed. The boundary conditions are also implemented in a fully implicit manner, thus making quadratic convergence possible. Complexities such as the ordering of cell nodes and the use of a far-field vortex to correct freestream for a lifting airfoil are addressed. Various methods to accelerate convergence and improve computational efficiency while using Newton iteration are discussed. Results are presented for inviscid, transonic nonlifting and lifting airfoils and also for laminar viscous cases.

Introduction

IN the past, iterative methods have been used with great success by many researchers to solve the equations governing inviscid and viscous flows. One of the popular explicit schemes is the finite-volume scheme of Jameson et al.[1] On the other hand, a number of researchers have used implicit schemes so as to be able to use large time steps, e.g., Beam and Warming[2] and Thomas and Walters.[3] In these schemes, the large linear system of equations that results from linearization in time is solved by iterative means, such as the alternating direction implicit (ADI) or line-by-line Gauss-Seidel. However, direct solution to this banded system of linear equations represents Newton's method in the limit of infinitely large time step. This has not been attempted so far due to large memory requirements and large operation counts. Since the advent of large, extremely powerful computers such as the Cray-2, it appears possible to overcome these limitations. The purpose of this paper is to assess the practicality of the application of Newton's method for airfoil calculations. Such a solver should be extremely robust, thus enabling one to solve problems where iterative methods fail or become prohibitively slow.

Newton's method has been applied recently for potential flows by Hafez et al.,[4] for incompressible two-dimensional viscous flows by Bender and Khosla,[5] and for multielement airfoil potential calculations by Wigton.[6] It is the intent of this paper to examine the application of Newton's method to solve the Euler and Navier-Stokes equations for steady-state solutions. The convective and pressure terms are differenced using a second-order upwind method employing the flux vector splitting technique of van Leer.[7] This technique has been used for two- and three-dimensional calculations by Anderson et al.[8] and has very good shock-capturing properties. The viscous terms for the Navier-Stokes equations are discretized using central differences. The equations then are linearized in time leading to a delta formulation. The linear system of equations is solved directly by Gaussian elimination at each time step. Consistent difference approximations are used on the implicit and explicit sides of the equations so that quadratic convergence can be obtained.

Received Nov. 19, 1987; presented as Paper 88-0413 at the AIAA 26th Aerospace Science Meeting, Reno, NV, Jan. 11-14, 1988; revision received July 29, 1988. Copyright © 1988 American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Research Scientist. Member AIAA.

Discretization

The equations governing fluid flow are

$$\frac{\partial w}{\partial t}+\frac{\partial f}{\partial x}+\frac{\partial g}{\partial y}=\frac{\partial R}{\partial x}+\frac{\partial S}{\partial y} \tag{1}$$

where w, f, g, R, and S are, respectively, the vector unknown, convective flux vectors, and viscous flux vectors in the x and y directions and are given by

$$w=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{bmatrix},\quad f=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uH\end{bmatrix},\quad g=\begin{bmatrix}\rho v\\ \rho vu\\ \rho v^2+p\\ \rho vH\end{bmatrix}$$

$$R=\begin{bmatrix}0\\ \sigma_{xx}\\ \sigma_{xy}\\ u\sigma_{xx}+v\sigma_{xy}-q_x\end{bmatrix},\quad S=\begin{bmatrix}0\\ \sigma_{xy}\\ \sigma_{yy}\\ u\sigma_{xy}+v\sigma_{yy}-q_y\end{bmatrix}$$

where

$$\sigma_{xx}=2\mu u_x-\tfrac{2}{3}\mu\,(u_x+v_y)$$

$$\sigma_{yy}=2\mu v_y-\tfrac{2}{3}\mu\,(u_x+v_y)$$

$$\sigma_{xy}=\sigma_{yx}=\mu(u_y+v_x)$$

$$q_x=-\kappa\frac{\partial T}{\partial x}$$

$$q_y=-\kappa\frac{\partial T}{\partial y}$$

The pressure p is obtained from the equation of state

$$P=(\gamma-1)\rho[E-\tfrac{1}{2}(u^2+v^2)]$$

The nondimensionalization here is with respect to the freestream density, pressure, and viscosity (the factors $1/Re$ and $1/Pr$ are absorbed in μ and κ). The viscosity is taken to be proportional to the temperature.

Upwind differencing is used for the convective and pressure terms based on the flux vector splitting method of van Leer[7] to second-order accuracy. This splitting technique has been found to be very effective in transonic airfoil calculations[8] and represents steady shocks with at most two interior zones. The inviscid flux f, for example, is split into f^+ and f^- according to the eigenvalues of the Jacobian matrix $\partial f/\partial w$ and is differenced as

$$\frac{\partial f}{\partial x} = \frac{1}{\delta x}(\delta_x^- f^+ + \delta_x^+ f^-) \tag{2}$$

where δ_x^- and δ_x^+ denote backward and forward difference operators, respectively. The split fluxes are represented as a flux balance across a cell,

$$\delta_x^- f^+ + \delta_x^+ f^- = [f^+(w^-) + f^-(w^+)]_{i+1/2,j} - [f^+(w^-) + f^-(w^+)]_{i-1/2,j} \tag{3}$$

where w^- and w^+ are obtained by second-order, one-sided differences as

$$w^-_{i+1/2,j} = 1.5w_{i,j} - 0.5w_{i-1,j} \tag{4a}$$

$$w^+_{i+1/2,j} = 1.5w_{i+1,j} - 0.5w_{i+2,j} \tag{4b}$$

Limiters have not been employed in this work since it has been found that transonic normal shocks can be captured without oscillations using the fully one-sided interpolation without limiting. The viscous terms are discretized using second-order central differences and the second derivatives are treated conservatively as differences across cell interfaces of first-order derivative terms. The scheme is implemented in generalized coordinates and the thin-layer form of the Navier-Stokes equations is used for the viscous calculations.

The system of Eqs. (1) thus can be discretized in space as shown above. Application of Euler implicit integration and linearization in time leads to a system of liner equations

$$\left(\frac{I}{\delta t} + \frac{\partial R}{\partial w}\right)\delta w_{i,j} = -R_{i,j}$$

$$R = D_\xi F + D_\eta G - \bar{D}_\eta S$$

$$\delta w_{i,j} = (w^{n+1} - w^n)_{i,j} \tag{5}$$

where the operator D refers to upwind differencing, the operator $\bar{D}$ to central differencing, ξ and η are the generalized coordinates, and R is the residual and vanishes at steady state Equation (5) represents a large linear (banded for a topologically rectangular computational domain) system of equations for the vector of unknowns and needs to be solved at each time step. It represents a modified Newton's method with the term $I/\delta t$ acting as a damping term. As δt tends to infinity, one obtains the standard Newton's method having the property that the error decays quadratically (for an isolated root of the equation). The term $(\partial R/\partial w)$ δw symbolically represents the implicit side after linearization and involves the Jacobian matrices of the flux vectors, which can be derived analytically. For example, consider the linearization of Eq. (3). The term $f^+(w^-_{i+1/2,j})$ is linearized as

$$f^+(w^-_{i+1/2,j})^{n+1} = f^+(w^-_{i+1/2,j})^n + a^+(w^-_{i+1/2,j})^n\,\delta w^-_{i+1/2,j}$$

Using Eq. (4a),

$$\delta w^-_{i+1/2,j} = 1.5\delta w_{i,j} - 0.5\delta w_{i-1,j}$$

where $a^+(q)$ stands for the 4×4 Jacobian matrix $\partial f^+(q)/\partial q$. Similarly, the rest of the terms in Eq. (3) can be linearized. The Jacobian matrices a^+ and a^- for van Leer's flux vector splitting can be systematically derived in generalized coordinates.[8,9] For more complicated schemes and for the implementation of turbulence models, it may difficult or impossible to carry out the analytical evaluation. In those cases, one can derive the Jacobians numerically. The latter alternative is very expensive and should be avoided unless absolutely necessary. The time-stepping strategy has been described by Mulder and van Leer.[10] At any level, the time step is obtained from

$$\delta t = \frac{\delta t_0}{[\,\|R^n\|_2/\|R^0\|_2]}$$

where $\|R^n\|_2$ is the L_2 norm of the residual at the nth iteration and δt_0 is the initial time step. The typical start-up Courant number is around 50 and the subsequent time steps increase rapidly as the steady state is approached. For cases with strong shocks, it has been found necessary to reduce the start-up Courant number to lower values.

Boundary Conditions

Fully implicit boundary conditions are incorporated in the present scheme. On a solid wall, the boundary conditions are imposed by considering a dummy cell inside the airfoil. The normal velocity on the airfoil is set equal to zero and zeroth order extrapolation of pressure onto the airfoil is used as well. For the viscous case, in addition, the velocity component tangential to the airfoil surface is set to zero. These conditions are all implemented in an implicit manner by linearization in time.

In the far field, assuming a locally one-dimensional flow normal to the boundary, Riemann invariants are employed. Assuming that the flow is subsonic in the far field, the two Riemann invariants are

$$R_0 = q_{n_0} - 2\frac{c_0}{\gamma - 1}$$

$$R_e = q_{n_e} + 2\frac{c_e}{\gamma - 1}$$

where q_n is the velocity normal to the boundary and c the speed of sound. The subscripts 0 and e refer to the freestream values and values extrapolated from the interior, respectively. Adding and subtracting these equations, we get relations for q_n and c that can be easily linearized. In addition, entropy and tangential velocity are specified at inflow and are extrapolated from the interior at outflow and these are linearized as well. Thus, for both the wall and far-field boundary conditions, we obtain a matrix relation

$$A\,\delta w_e + B\,\delta w_i = 0$$

where subscript e refers to the dummy cell either in the far field or in the airfoil and subscript i to the adjacent interior point. One point to note is that since the far-field boundary condition is dependent on the direction of the flow, implementing it in a delta form is not correct when a switch in sign of the normal velocity occurs. Therefore, to account for this possible change in sign, it is necessary to implement the far-field boundary condition explicitly after the fully implicit implementation for the first few cycles.

For a lifting airfoil, it has been shown by Thomas and Salas[11] that modifying the freestream in the far field by a point vortex representation of the airfoil leads to accurate results. This enables one to use a far-field boundary only 20 chords from the airfoil in order to obtain the correct value of the lift. Implementation of this condition in an implicit manner is difficult, since the vortex strength is proportional to the lift coefficient. This leads to a coupling between every point in the far field and points on the airfoil, thereby destroying the banded structure. Two ways to circumvent this difficulty are presented in the next section.

Acceleration Techniques

In this section, various means to accelerate convergence as well as to improve efficiency in the context of Newton iteration are discussed.

Mesh Sequencing

The initial condition for airfoil calculations is usually assumed to be freestream condition. Starting with this initial condition on a fine grid leads to a large plateau in the convergence plot before one can observe quadratic convergence. Since each matrix inversion is rather expensive, it is necessary to minimize the number of such inversions, especially on the fine grid. Therefore, the solution is first obtained on a coarse grid and then transferred to a finer grid sequentially to establish a better initial guess on the fine grid. Since the computational work goes up by a factor of 16 when dimensions are doubled, the work done on the coarser grids is insignificant in comparison with that done on the finest grid.

Superconvergence

Superconvergence refers to a technique that enables one to achieve better than quadratic convergence while using Newton's method. It has been shown by Traub (see Ortega and Rheinboldt,[12] p. 315) that one can easily achieve cubic or even quartic convergence with little extra effort while solving a system of nonlinear equations by Newton's method. Consider a system of nonlinear equations,

$$F(x) = 0$$

Then, in this modified Newton's method $F'(x)$ is evaluated only every m steps. The iteration becomes

$$x^{k,i} = x^{k,i-1} - F'(x^k)^{-1} F(x^{k,i-1}), \quad i = 1,2,\ldots,m$$

$$x^{k,0} = x^k$$

$$x^{k+1} = x^{k,m}$$

This iteration may be considered as a composition of one Newton step with $(m-1)$ simplified Newton steps. (A simplified Newton step uses a frozen value for F' and can be shown to have linear convergence.) Note that $m = 1$ corresponds to the standard Newton's method, and it can be shown the $m = 2$ leads to cubic convergence, $m = 3$ to quartic convergence, etc. A Newton step involves an inversion of the matrix requiring $\mathcal{O}(n\,bw^2)$ operations, whereas a simplified Newton step requires on $\mathcal{O}(2nbw)$ operations, where n is the total number of unknowns and bw is the half-bandwidth. Since the bandwidth is $\mathcal{O}(100)$, the work involved in the simplified Newton steps is negligible. This technique again cuts down the number of inversions of the big matrix.

Ordering of Cells

Figure 1 shows an airfoil with a C-mesh with a particular ordering of cell nodes. This ordering is denoted as ordering A. This ordering leads to a fully banded matrix with a half-bandwidth for a second-order-accurate scheme of $(16{*}J2 + 3)$, where $J2$ is the number of points in normal direction. This is solved using a banded matrix solver.

Figure 2 shows another ordering denoted as ordering B. This ordering is the usual ordering when dealing with a C-mesh in most iterative codes. Ordering B does not lead to a banded structure because of coupling between cells on either side of the wake cut. The matrix on the left side of Eq. (2) is made up essentially of a banded matrix with a half-bandwidth of $(8{*}J2 + 3)$ with extraneous entries outside the band. In this case, the system of equations is solved by splitting the matrix and using simple Richardson iteration. Let the system of equations be denoted by

$$Lu = f \tag{6}$$

Consider the splitting

$$L = M + N$$

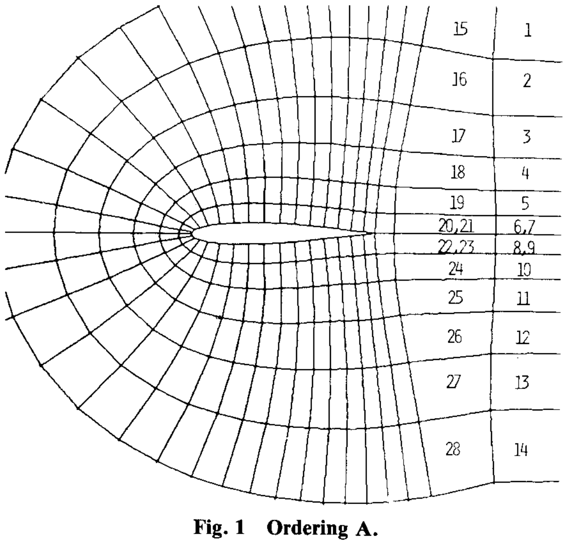


Fig. 1 Ordering A.

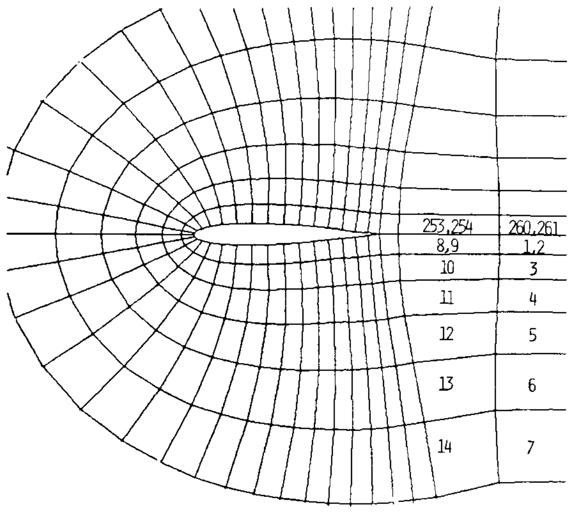


Fig. 2 Ordering B.

where M represents the banded part of L and N the entries outside the band. Then, the system of Eq. (6) is solved in two steps as

$$Mu^{(0)} = f \tag{7a}$$

$$Mu^{(k)} = -Nu^{(k-1)} + f, \quad k = 1,2,\ldots p \tag{7b}$$

Equation (7a) is solved using a banded solver, and Eq. (7b) is solved easily since M^{-1} is known. Provided enough number of subiterations [Eq. (4b)] are performed, one gets an exact solution to Eq. (6). The operation count for each of the subiterations is $\mathcal{O}(2nbw)$, and therefore it is necessary to use as few a number of subiterations as possible. The typical number of subiterations is around 10.

The advantage of using ordering A is that it is possible to use the technique of superconvergence since we have an exact solution to the system of equations. In conjunction with mesh sequencing, it is possible to drive the residual to machine zero in three to five iterations and leads to a robust code. One drawback of ordering A is that programming is cumbersome, with a lot of exceptional situations to handle. In the case of ordering B, however, since the number of subiterations needs to be kept small, we do not have an exact solution, and hence the idea of superconvergence does not help. The computation time at each time step [Eq. (7a) + p iterations of Eq. (7b)] goes down by a factor of three. (The main iteration is four times faster since the bandwidth is halved.) It is further possible to freeze the Jacobian matrices after two time steps or so, if we do not want to converge to machine zero. Another advantage of ordering B is that the far-field vortex condition for a lifting airfoil can be incorporated in the subiterations, which amounts to a fully implicit implementation of this condition. While using ordering A, this condition is applied explicitly at the beginning of computation in every grid in grid sequence (any explicit application of boundary conditions during Newton iteration will slow down convergence).

Nested Dissection Strategy and Sparse Matrix Solvers

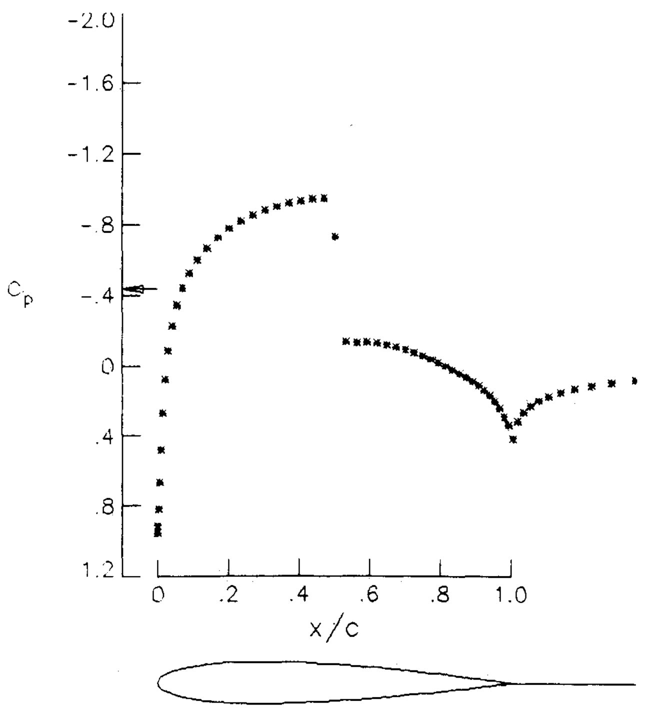
The banded solvers employed pose a few problems. One troublesome feature is the fact that doubling the dimensions increases the effort by a factor of 16 (the number of unknowns increases by factor of 4 and the bandwidth by a factor of 2). Another problem is that certain topological restrictions apply in order to be able to arrive at a banded matrix. Ordering A, for example, is mandated for a C-mesh to meet this requirement. For other topologies, such as an O-mesh, it is impossible to devise a similar strategy. There is, of course, the unattractive alternative of ordering cells in the streamwise direction, in which case the bandwidth is proportional to the number of cells around the airfoil. To alleviate these problems, the nested dissection strategy is implemented. This has been used by Wigton[6] in multielement airfoil calculations. The algorithm leads to an efficient ordering of cells, such that the fill-in that occurs during the factorization is minimized. This leads to substantial savings in computational speed and memory requirements.[13] For instance, the computational effort increases by a factor of 8 when the dimensions are doubled. The recursive ordering algorithm is first derived for a rectangular topology and then is extended to handle other topologies. The ordering is done at the preprocessor stage and takes up little time. Once the ordering is completed, a sparse matrix solver such as the YSMP[14] is invoked to carry out the factorization. The nested dissection strategy is discussed in the Appendix. In addition to the advantages mentioned above, the implementation of the far-field vortex becomes simple, since the change in the lift coefficient can be included in the ordering as an additional unknown.

Results and Discussion

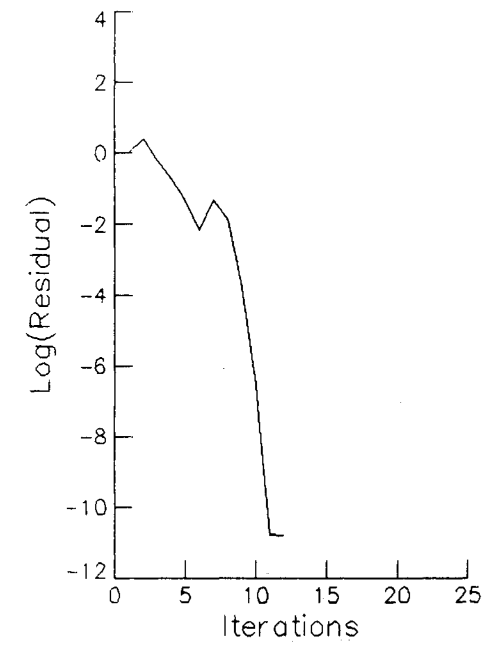
This section presents some representative results of inviscid and viscous airfoil calculations. A special banded solver was written for implementation on the Cray-2 that makes use of local memory. This solver brings in a fixed number of columns into local memory, performs an LU decomposition, then modifies the rest of the matrix by bringing in a column at a time into local memory. This improved the performance by about 30%. All the inviscid computations were done on a fine mesh of 160 × 32, generated using a hyperbolic grid generation technique.

Figures 3a and 3b present, respectively, the pressure profile and the convergence history for flow over a NACA 0012 airfoil at an angle of attack of 0 deg and freestream Mach number of 0.8 obtained using ordering A. The convergence plot shows that quadratic convergence is achieved after only 10 iterations. The kink in the residual history is due to the movement of the shock. This is a very expensive computation as it took about 200 s for each inversion on the Cray-2. With the sparse matrix solver the time to perform the LU decomposition is cut by a factor of two. Figures 4a and 4b present the results obtained using ordering B and a mesh that is finer near the airfoil. Here, the Jacobian and, consequently, the LU decomposition, is frozen after two cycles. About 10 subiterations [Eq. (7b)] are performed at each time step. The solution is essentially converged. Since the flux vector splitting technique does not preserve constant total enthalpy, the error in average total enthalpy in the field at steady state is of the order of the truncation error and this is observed to reach a constant value. The shock is captured very well and the result agrees with previously published results. However, the ordering employed is not general enough and also, in more general problems, it is not clear as to when the decomposition may be completely frozen.

The next set of results pertains to a lifting case. The flow is over an NACA 0012 at angle of attack of 1.25 deg and a freestream Mach number of 0.8. Figures 5a and 5b show the results for this case. Here, mesh sequencing is employed and four levels of grid are used. The startup CFL number is 100. The convergence history is shown in Fig. 5b for the fine grid. The Jacobian and LU decomposition are frozen twice at each time step, yielding quartic convergence (theoretically). The nested dissection strategy is used to solve the linear system of equations. The solution is accurate and the shock on the bottom surface is captured well too. The solution converges to machine zero in five iterations (solutions for nonlifting cases have been obtained in as few as two to three iterations on the

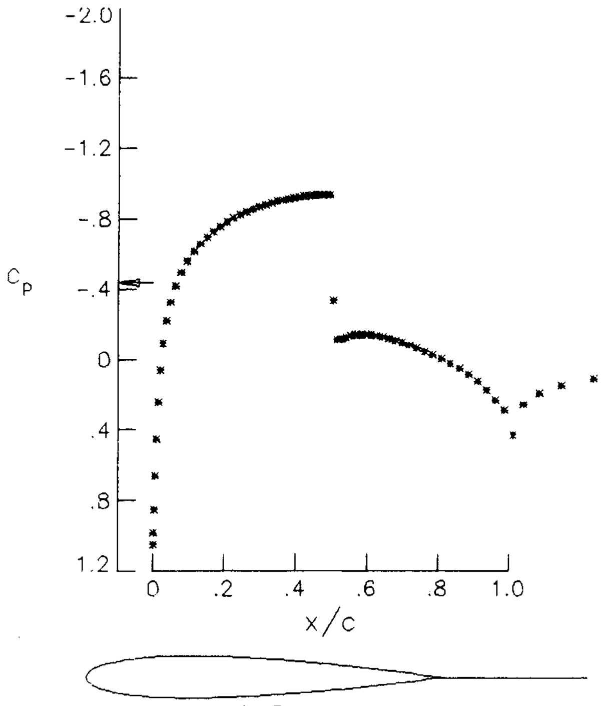


a) Pressure profile

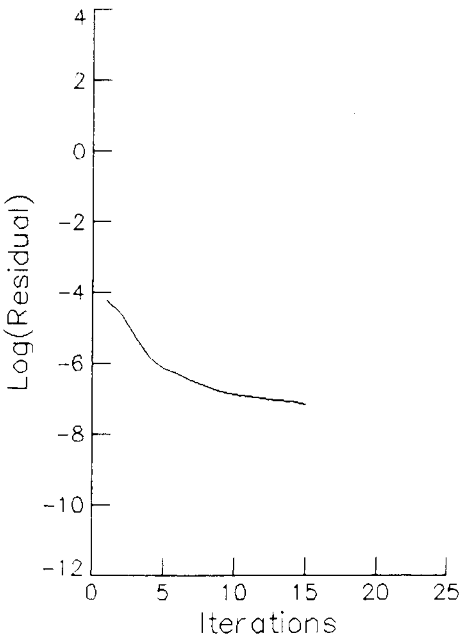


b) Convergence history

Fig. 3 Ordering A of flow over an NACA0012 airfoil ($M = 0.8$, = 0 deg).

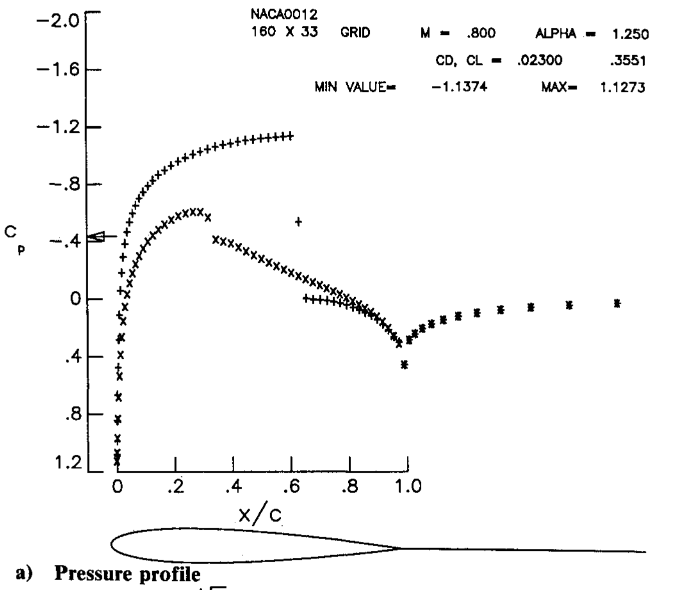


a) Pressure profile

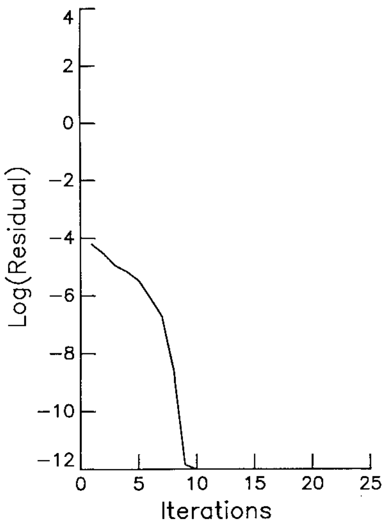


b) Convergence history

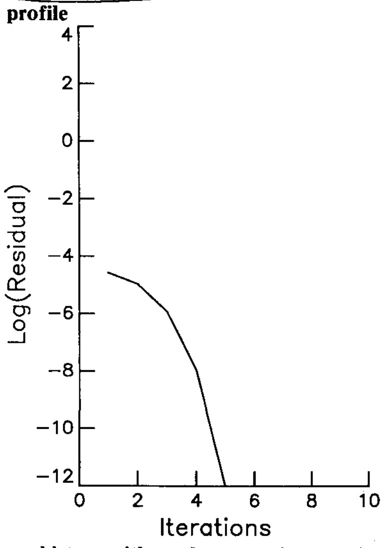
Fig. 4 Ordering B of flow over an NACA0012 airfoil ($M = 0.8$, $= 0$ deg).



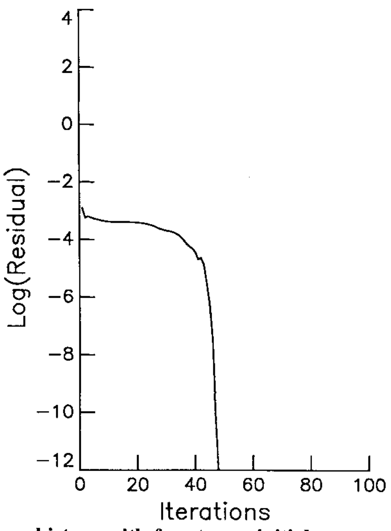
a) Pressure profile



c) Convergence history with mesh sequencing only

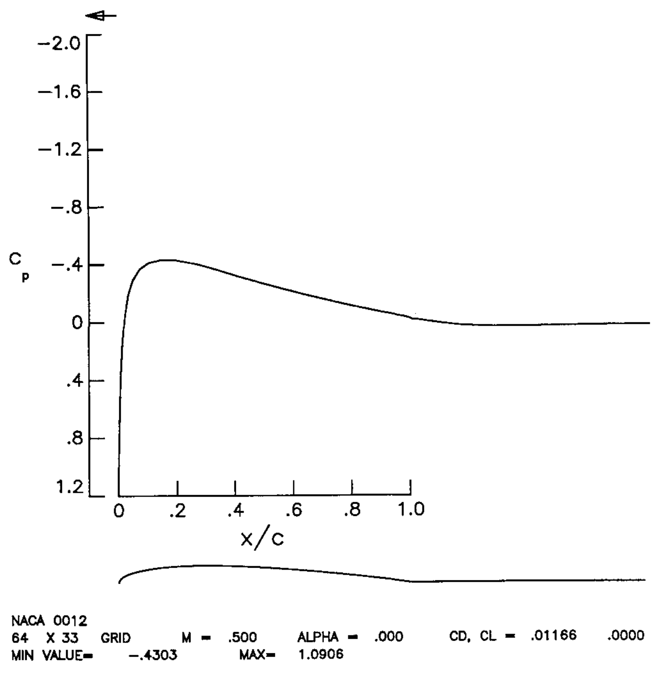


b) Convergence history with mesh sequencing and Jacobian freezing

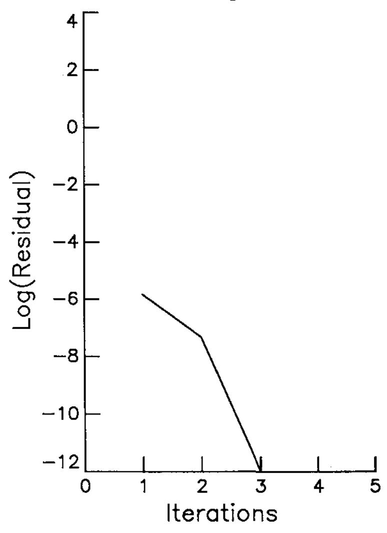


d) Convergence history with freestream initial guess

Fig. 5 Flow over an NACA0012 airfoil ($M = 0.8$, $= 1.25$ deg).



a) Pressure profile



b) Convergence history

Fig. 6 Laminar flow over an NACA0012 airfoil ($M = 0.5$, $= 0$ deg, $Re = 5000$).

finest grid to machine zero). Figure 5c shows the convergence history for the same case with mesh sequencing when the decomposition is done at each time step, i.e., no freezing of the decomposition. This exhibits quadratic convergence and takes nearly 10 iterations to converge to machine zero. Figure 5d shows the convergence history for the standard Newton's method without mesh sequencing, starting with the freestream as the initial guess on a 80×16 grid. It is seen to take about 50 iterations to converge and is thus prohibitively expensive. Thus, the benefits of freezing the Jacobian and mesh sequencing (or any other good preconditioner) are overwhelming.

Finally, results are presented for a laminar viscous case. The symmetric problem is studied using a 80×32 grid. The case presented is laminar flow over an NACA 0012 airfoil at a Reynolds number of 5000 based on the chord. The freestream Mach number is 0.5. Figures 6a and 6b show, respectively, the pressure profile and the convergence history. The solution converges to machine zero in three iterations. In Fig. 6a, we notice little pressure recovery, indicating that the viscous effects are dominant. Separation occurs at a distance of 82% from the trailing edge and reattachment occurs at 13% from the trailing edge. This agrees with previous computation.[15] Most important to note, however, is that the viscous case takes about the same time as the inviscid case, and there is no apparent effect of Reynolds number on the convergence.

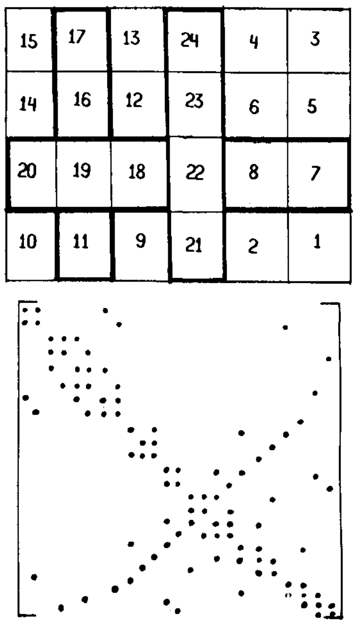


Fig. 7 Natural ordering for Poisson equation on a rectangle and the resultant matrix.

A robust direct solver that uses Newton iteration has been developed for obtaining steady state solutions of fluid dynamic problems. Different techniques to improve efficiency and convergence in the context of Newton iteration are implemented. In particular, the nested dissection strategy has been implemented for aerodynamic problems. The time of computation is thus substantially reduced, but the method is not competitive with iterative methods for standard inviscid problems. Given the flurry of activity in the area of algorithms for the solution of sparse linear systems, there could be substantial gains in efficiency in the near future. On the other hand, viscous problems may be more suited as a lot of iterative codes exhibit a deterioration in convergence as the Reynolds number increases. It is proposed to use Roe's flux-difference splitting scheme[16] for the viscous problems. A semi-iterative technique wherein the main iteration is solved by LU decomposition followed by subiterations also has been investigated. Various ways to accelerate the convergence during subiteration have been examined. These include the minimum residual method, the epsilon algorithm, and some others. A turbulence model needs to be incorporated to do practical viscous problems. The Jacobians then could be derived numerically. It is clear that if one used a coarse enough mesh, the direct solvers will fare better than iterative methods. Therefore, direct solvers can be used in the multigrid context on coarser meshes. Another possible application is in the cross planes in three-dimensional calculations.

Appendix

Some of the basic ideas behind the nested dissection strategy and its application to aerodynamic problems are presented below. Consider the solution of Poisson's equation in a rectangle by using central differences. If one were to solve the resulting linear system of equations by Gaussian elimination an important issue that arises is the ordering of cells. Figure 7 shows a natural ordering by columns that minimizes the bandwidth and the banded matrix that arises as a result. Upon LU decomposition, the entire region between the bands gets filled in, even though most of the entries were zero initially. In order to minimize this fill-in, George and Liu[13] have devised the nested dissection strategy. Central to this strategy is the con-

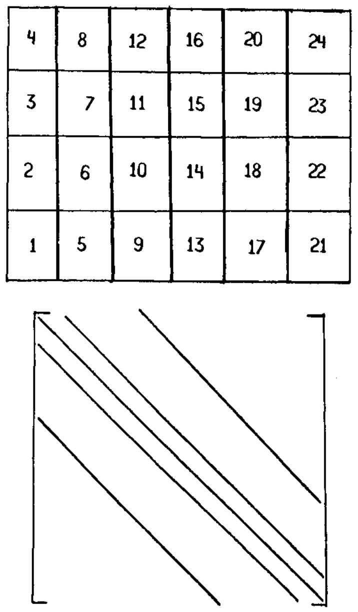


Fig. 8 Nested dissection ordering for Poisson equation on a rectangle and the resultant matrix.

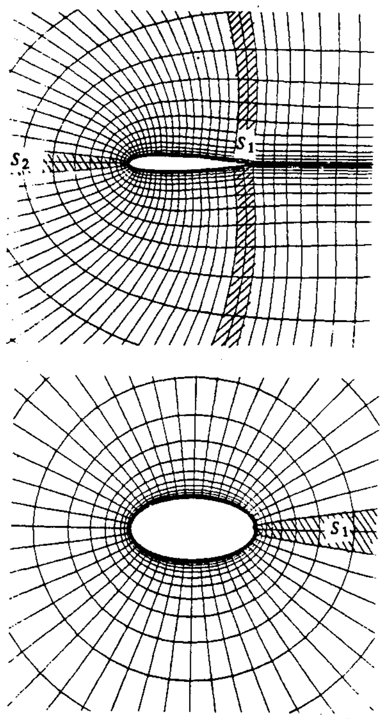


Fig. 9 Initial ordering steps for C- and O-meshes.

cept of a separator, which is a set of nodes that divides the computational domain into nearly equal halves that do not communicate directly with each other. This leads to segments of the matrix that remain zero during the factorization. An example of a separator is the set of nodes numbered 24,23,22, and 21 in Fig. 8. It is easy to see that the separator need be only one cell wide in this case, since we are dealing with a three-point stencil in each direction. To get the nested dissection ordering, we continue dissecting the two remaining components, always choosing the separator of the smallest length and numbering the cells in a descending manner. The final ordering has the desired property of incurring least fill-in during the decomposition. Figure 8 shows the ordering for Poisson's equation and the associated matrix. The separators are highlighted. The number of operations required to factor a matrix associated with an n by n grid is $\mathcal{O}(n^3)$ and the storage required is $\mathcal{O}(n^2 \log_2 n)$. The banded solver on the other hand, requires $\mathcal{O}(n^4)$ operations and storage of $\mathcal{O}(n^3)$.

Extension of this physically based strategy to aerodynamic problems is easily achieved. Since we deal with a five-point stencil in each direction, the separator becomes two cells wide. Figure 9 shows the initial ordering steps for C and O-meshes. Separator S_1 is required as the first separator in each case so that the domain is divided into simpler regions. It is possible to use this strategy for zonal meshes well.

Acknowledgments

The author thanks NASA Langley Research Center for supporting this project. The author also expresses his appreciation to Manuel Salas, Jim Thomas, and Jay Lambiotte of Langley Research Center for many fruitful suggestions during the course of this project.

References

[1]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, June 1981.

[2]Beam, R. and Warming, R. F., "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law-Form," *Journal of Computational Physics,* 1976, pp. 87–110.

[3]Thomas, J. L. and Walters, R. W., "Upwind Relaxation Algorithms for the Navier-Stokes Equations," *AIAA Journal,* Vol. 25, April 1987, pp. 527–534.

[4]Hafez, M., Palaniswamy, S., and Mariani, P., "Calculations of Transonic Flows with Shocks using Newton's Method and Direct Solver: Part II," AIAA Paper 88-0226, Jan. 1988.

[5]Bender, E. E. and Khosla, P. K., "Solution of the Two-Dimensional Navier-Stokes Equations using Sparse Matrix Solvers," AIAA Paper 87-0603, Jan. 1987.

[6]Wigton, L. B., "Application of MACSYMA and Sparse Matrix Technology to Multielement Airfoil Calculations," AIAA Paper 87-1124-CP, June 1987.

[7]van Leer, B., "Flux-Vector Splitting for the Euler Equations," Inst. for Computer Applications in Science and Engineering, Hampton, VA, Rept. 82-30, Sept. 1982.

[8]Anderson, W. K., Thomas, J. L., and Whitfield, D. L., "Multigrid Acceleration of the Flux Split Euler Equations," AIAA Paper 86-0274, Jan. 1986.

[9]Steger, J. L., "Implicit Finite-Difference Simulation of Flow about Arbitrary Two-Dimensional Geometries," *AIAA Journal,* Vol. 16, July 1978, pp. 679–686.

[10]Mulder, W. A. and van Leer, B., "Implicit Upwind Methods for the Euler Equations," AIAA Paper 83-1930, July 1983.

[11]Thomas, J. L. and Salas, M. D., "Far-Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations," *AIAA Journal*, Vol. 24, July 1986, pp. 1074–1080.

[12]Ortega, J. M. and Rheinboldt, W. C., *Iterative Solution of Nonlinear Equations in Several Variables*, Academic, New York, 1970.

[13]George, A. and Liu, J. W., *Computer Solution of Large Sparse Positive Definite Systems,* Prentice-Hall, Englewood Cliffs, NJ, 1981.

[14]Eisenstadt, S. C., Bursky, M. C., Schultz, M. H., and Sherman, A. H., "Yale Sparse Matrix Package: II The Nonsymmetric Codes," Dept. of Computer Sciences, Yale University, New Haven, CT, Res. Rept. 114, 1977.

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

[16]van Leer, B., Thomas, J. L., Roe, P. L., and Newsome, R. W., "A Comparison of Numerical Flux Formulas for the Euler and Navier-Stokes Equations," AIAA Paper 87-1104, June 1987.