GRANT/AMES

Aerodynamics of Engine-Airframe Interaction

29P

Annual Progress Report

August 30, 1986

NASA Grant: NAG 2-373   *IN-23727*
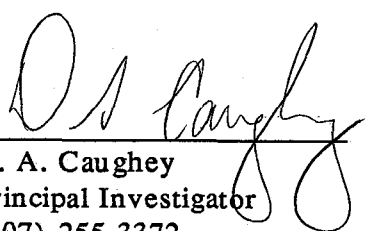
Submitted to: University Affairs Office
                      Mail Stop 241-25
                      NASA Ames Research Center
                      Moffett Field, California 94035

Submitted by: Cornell University
                      Ithaca, New York 14853

Principal Investigator: D. A. Caughey (Soc. Sec. # ████████)
                          Professor
                          Sibley School of Mechanical and Aerospace Engineering

Period of Support: October 1, 1985 - September 30, 1986.

D. A. Caughey
Principal Investigator
(607)-255-3372

Preface

This document describes work performed during the past year under Grant NAG 2-373. The work is directed towards the development of efficient computational methods for the solution of both inviscid and viscous transonic flow problems, with emphasis on problems of complex, three-dimensional geometry. Solution-adaptive techniques are being developed, using the multigrid method to achieve computational efficiency, and the work is focussed upon the aerodynamic problems associated with complex wing-fuselage shapes, including engine inlets. The Principal Investigator for the project is David A. Caughey, Professor in the Sibley School of Mechanical and Aerospace Engineering at Cornell University. The Grant Technical Monitor is

Dr. W. J. Chyu
Applied Aerodynamics Branch
Mail Stop 227-6
NASA Ames Research Center

Further information can be obtained from:

Professor David A. Caughey
218 Upson Hall
Cornell University
Ithaca, New York 14853
Tel.: (607) - 255-3372

Table of Contents

I. Introduction

This report describes progress in research directed towards the efficient solution of the inviscid Euler and Reynolds-averaged Navier-Stokes equations for transonic flows through engine inlets, and past complete aircraft configurations, with emphasis on the flowfields in the vicinity of engine inlets. The research focusses upon the development of solution-adaptive grid procedures for these problems, and the development of multi-grid algorithms in conjunction with both, implicit and explicit time-stepping schemes for the solution of three-dimensional problems. The work includes further development of mesh systems suitable for inlet and wing-fuselage-inlet geometries using a variational approach. One portion of the research forms the basis of the Ph.D. thesis of Mr. Dun C. Liu, and another portion forms the basis of the M.S. thesis work of Mr. Ravi Iyer. Mr. Liu is supported independently by a Fellowship, and Mr. Iyer will be supported next year as a Graduate Research Assistant if the grant is renewed.

II. Review of Work in Past Year

Work in the past year has concentrated upon two-dimensional problems, and has been in two general areas:

(1) the development of solution-adaptive procedures to cluster the grid cells in regions of high (truncation) error, and

(2) the development of a multigrid scheme for solution of the two-dimensional Euler equations using a diagonalized ADI smoothing algorithm.

The work in each of these areas will now be described briefly.

A. Adaptive Grid Generation

This section contains a review of the formulation of the Euler-Lagrangian equations for grid generation, and some test results demonstrating their applicability.

The solution of a partial differential equation can be greatly simplified by a properly constructed grid. It is also true that a grid which is not well suited to the problem may lead to an unsatisfactory result, or even to instability and a lack of convergence. Therefore, one of the central problems in computing numerical solutions to partial differential equations is that of grid generation. Several advantages are obtained when the partial differential equation, approximated on an unequally spaced physical domain, is transformed to a uniformly spaced computational domain. For example, a curved body surface can be treated as a boundary in the computational domain, which fact permits easy application of the boundary conditions. The transformation methods fall into three basic classes: complex variable methods, algebraic methods, and partial differential equation methods. A concise review of these methods is given by Thompson.[1]

The partial differential equation method has been developed over the past decade. Thompson et al.[2] have developed a particular way of using elliptic PDE's to generate two dimensional computational grids about arbitrary bodies. Brackbill and Saltzman[3] have derived the elliptic PDE's by minimizing an integral which characterizes undesirable properties of the grid.

In order to determine the best grid point distribution, an *a priori* knowledge of the solution of the physical problem is desired. Unfortunately, this knowledge is unavailable and only the general features of the solution may initially be understood. The concept of a solution adaptive grid is needed to incorporate information from the solution in locating the grid points. With the use of an adaptive grid, the physical behavior of events on both large and small scales can be adequately accommodated by a variable mesh size. The effect of this is to reduce the discretization error of the numerical scheme by clustering the grid, in adaptation to the solution, in high gradient regions.

For conceptual simplicity, an ability to handle problems with a number of length scales, and extendibility to three-dimensional problems, the method of Brackbill and Saltzman has been chosen for the present study. In this section, the formulation of Euler-Lagrangian equations for grid generation are reviewed, and some preliminary test results are presented to demonstrate their applicability.

Adaptive Grid Schemes

Brackbill and Saltzman,[3] and Brackbill[4] have extended Winslow's[5] method for generating a computational mesh to include grid adaptation. In this approach a variational technique is used to minimize a linear combination of measures of grid smoothness, orthogonality, and adaptation. The smoothness is measured by integrating the change of the computational coordinates over the physical domain, and can be written as the following form for a two dimensional problem

$$I_s = \int_D ((\nabla \xi)^2 + (\nabla \eta)^2) dA \qquad (1.1)$$

where dA represents a differential area in the physical space, and D denotes the physical domain. In addition to smoothness requirement, control of mesh skewness is obtained if a measure of grid non-orthogonality, expressed as

$$I_o = \int_D (\nabla \xi . \nabla \eta)^2 J^3 dA \qquad (1.2)$$

is minimized over the domain, where J is the Jacobian. The grid adaptation is provided by minimizing a weighted average of the area variation over the field; an appropriate integral is

$$I_w = \int_D wJ dA \qquad (1.3)$$

where w is the weight function which produces the grid adaptation. If $w = w(x,y)$ the solution for minimized I_w corresponds to $wJ^2 = const.$ while for $w = w(\xi,\eta)$, it corresponds to $wJ = const.$.

In order to incorporate smoothness, orthogonality, and adaptation in the grid generator, a linear combination of the functionals is considered,

$$I = I_s + f_w I_w + f_o I_o \qquad (1.4)$$

and the corresponding Euler equations are

$$b_1 x_{\xi\xi} + b_2 x_{\xi\eta} + b_3 x_{\eta\eta} + a_1 y_{\xi\xi} + a_2 y_{\xi\eta} + a_3 y_{\eta\eta} = f_w G_1 \qquad (1.5)$$

and

$$a_1 x_{\xi\xi} + a_2 x_{\xi\eta} + a_3 x_{\eta\eta} + c_1 y_{\xi\xi} + c_2 y_{\xi\eta} + c_3 y_{\eta\eta} = f_w G_2 \qquad (1.6)$$

where

$$a_i = a_{si} + f_w a_{wi} + f_o a_{oi}$$

$$b_i = b_{si} + f_w b_{wi} + f_o b_{oi}$$

$$c_i = c_{si} + f_w c_{wi} + f_o c_{oi}$$

for $i = 1, 2, 3$. Here a_{si}, b_{si}, c_{si} are the coefficients arising from minimization of the smoothness integral, a_{oi}, b_{oi}, c_{oi} are the coefficients arising from minimization of the orthogonality integral, and a_{wi}, b_{wi}, c_{wi} are the coefficients arising from minimization of the weighting-function integral. If $w = w(x,y)$,

$$G_1 = -\frac{J^2}{2w}\frac{\partial w}{\partial x}$$

$$G_2 = -\frac{J^2}{2w}\frac{\partial w}{\partial y}$$

or if $w = w(\xi,\eta)$,

$$G_1 = -\frac{J}{w}(w_\xi y_\eta - w_\eta y_\xi)$$

$$G_2 = -\frac{J}{w}(w_\eta x_\xi - w_\xi x_\eta)$$

Normalization of the Functional

For a uniformly spaced grid on a square, the initial value of the smoothness functional is $I_s \sim 2n^2$, where n represents the number of cells in both the x and y directions. For the solution adaptation functional, $I_w \sim \frac{\hat{w}}{n^2}$, where $\hat{w}$ is the mean value of w in the computational domain. For the orthogonality functional, $I_o \sim \frac{4}{n^2}$. To have the same order of magnitude, f_w and f_o are scaled to

$$\overline{f}_w = f_w \frac{2n^4}{\hat{w}} \tag{1.7}$$

and

$$\overline{f}_o = f_o \frac{n^4}{2} \tag{1.8}$$

respectively.

For more general geometries, it is reasonable to assume

$$h \sim x_\xi \sim x_\eta \sim y_\xi \sim y_\eta$$

where h is the step size. With m × n cells,

$$I_s \sim mn$$

$$I_w \sim mn\hat{w}h^4$$

$$I_o \sim mnh^4$$

therefore

$$\bar{f}_w = \frac{f_w}{\hat{w}h^4} \tag{1.9}$$

and

$$\bar{f}_o = \frac{f_o}{h^4} \tag{1.10}$$

Scaling and Smoothing of the Weight Function

To have some external control of the range of the weight function, w_{ij} is scaled, as proposed by Brackbill and Saltzman,[3] such that after scaling

$$\bar{w}_{ij}J_{ij} = 1 \ .$$

The range of $\bar{w}_{ij}$ is determined by the lesser of the externally set value σ_0 and the ratio of the weight function

$$\sigma = \min\left[\sigma_0, \frac{\max(w_{ij})}{\min(w_{ij})}\right]$$

In order that the largest zone fits into the computational domain, q, the minimum value of w must satisfy the equation

$$q = \min(\bar{w}_{ij}) = \min\left[\frac{1}{\sum\limits_{ij}(J_{ij})}, \frac{1}{\sigma}\right]$$

The scaled $\bar{w}_{ij}$ is calculated from

$$\bar{w}_{ij} = q\left[\frac{(\sigma^2 - 1)w_{ij}}{\max(w_{ij})} + 1\right] \tag{1.11}$$

Computational experience has indicated that the convergence rate is poor when the weight function has very large gradients. The weight function can be smoothed by the following equation

$$\bar{w}_{ij}^{(m+1)} = (1-\mu)\bar{w}_{ij}^{(m)} + 0.25\mu(\bar{w}_{i+1,j}^{(m)} + \bar{w}_{i-1,j}^{(m)} + \bar{w}_{i,j+1}^{(m)} + \bar{w}_{i,j-1}^{(m)}) \tag{1.12}$$

where $\mu \leq 0.5$ to ensure stability.

Numerical Schemes

All derivatives in Eqs. (1.5) and (1.6) are approximated by central differences

$$x_\xi \sim 0.5(x_{i+1,j} - x_{i-1,j})$$

$$x_\eta \sim 0.5(x_{i,j+1} - x_{i,j-1})$$

and

$$x_{\xi\xi} \sim x_{i+1,j} - 2x_{i,j} + x_{i-1,j}$$

$$x_{\xi\eta} \sim 0.25(x_{i+1,j+1} + x_{i-1,j-1} - x_{i+1,j-1} - x_{i-1,j+1})$$

$$x_{\eta\eta} \sim x_{i,j+1} - 2x_{i,j} + x_{i,j-1}$$

If the residuals at the $m-th$ iteration of this approximation to Eqs. (1.5) and (1.6) are denoted by $R_1^{(m)}$ and $R_2^{(m)}$, respectively, the Jacobi method gives

$$R_1^{(m)} = 2(b_1^{(m)} + b_3^{(m)})\Delta x_{i,j}^{(m+1)} + 2(a_1^{(m)} + a_3^{(m)})\Delta y_{i,j}^{(m+1)} \qquad (1.13)$$

$$R_2^{(m)} = 2(a_1^{(m)} + a_3^{(m)})\Delta x_{i,j}^{(m+1)} + 2(c_1^{(m)} + c_3^{(m)})\Delta y_{i,j}^{(m+1)} \qquad (1.14)$$

One can solve for $\Delta x_{i,j}^{(m+1)}$ and $\Delta y_{i,j}^{(m+1)}$ using Eqs. (1.13) and (1.14). The iteration is terminated when $R_1^{(m)}$ and $R_2^{(m)}$ are everywhere smaller than a prescribed tolerance.

Numerical Examples

Two cases are studied. The first deals with a model singular perturbation problem and the second deals with the nonlifting transonic flow over an NACA 0012 airfoil.

Case 1 - A Model Singular Perturbation Problem

Consider the following convection-diffusion equation

$$\phi_t + \nabla \cdot (\phi \overline{u}) - \nabla \cdot \nabla \phi = 0$$

where

$$\overline{u} = -P_e \ h^+(\overline{r})h^-(\overline{r}) \ \hat{i}_r$$

$$h^+(\overline{r}) = (1 + \exp(P_e(\overline{r}-1)))^{-1}$$

$$h^-(\overline{r}) = (1 + \exp(P_e(1-\overline{r})))^{-1}$$

and $P_e \equiv u_0 r_0 / k_0$. If the boundary conditions are such that a steady state is reached,

$$\overline{u} \ \phi = \nabla \phi \qquad (1.15)$$

and,

$$\phi(\overline{r}) = \exp(-P_e \ \overline{r} \ h^-(\overline{r}) + \log(h^+(\overline{r})) - \log(h^+(0)))$$

It is noted that ϕ has value 1 within $\overline{r} = 1$ and sharply drops to zero outside $\overline{r} = 1$.

If the weight function is chosen as

$$w = \left(\frac{1}{\phi}\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{1}{\phi}\frac{\partial\phi}{\partial y}\right)^2$$

with (1.15)

$$w = (\overline{u}\cdot\hat{i})^2 + (\overline{u}\cdot\hat{j})^2$$

In order to maintain symmetry, a simple point-Jacobi method is used to solve Eqs. (1.13) and (1.14). For the weight function displayed in Figure A.1, the grids for $P_e = 12$, $r_0 = 0.25$ and $f_w = 5$ are shown in Figures A.2 and A.3. The mesh in Figure A.2 was computed with a simple Dirichlet boundary condition to enforce uniform spacing on the boundaries. That in Figure A.3 is obtained by including an orthogonality constraint (i.e, $f_o = 1$), and orthogonality is also enforced at the boundaries.

Case 2 - Transonic Flow over an NACA 0012 Airfoil

For the calculation of compressible flows with shock waves, the pressure,[3] Mach number,[6] or internal energy[6] have been proposed to serve as weight function for solution adaptation. However, when using multigrid a more natural weight function has been proposed by Berger and Jameson,[7] which is naturally related to the local truncation error of the scheme. This procedure has also been adopted for use in the present work.
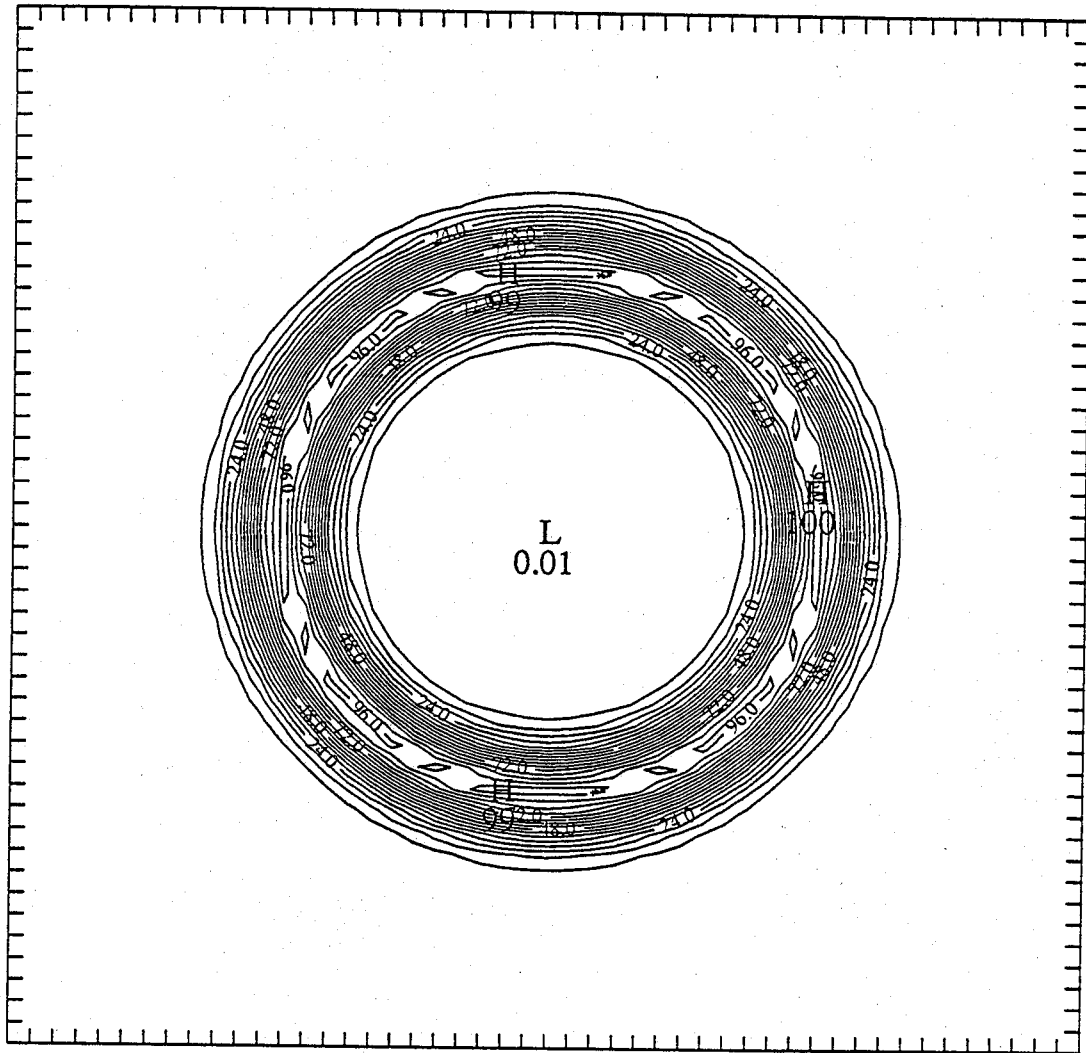
contour from   0.   to 96.000   contour interval of 6.0000   pt(3,3)= 0.10000e-01

Figure A.1   Weight function for circularly-symmetric test case.
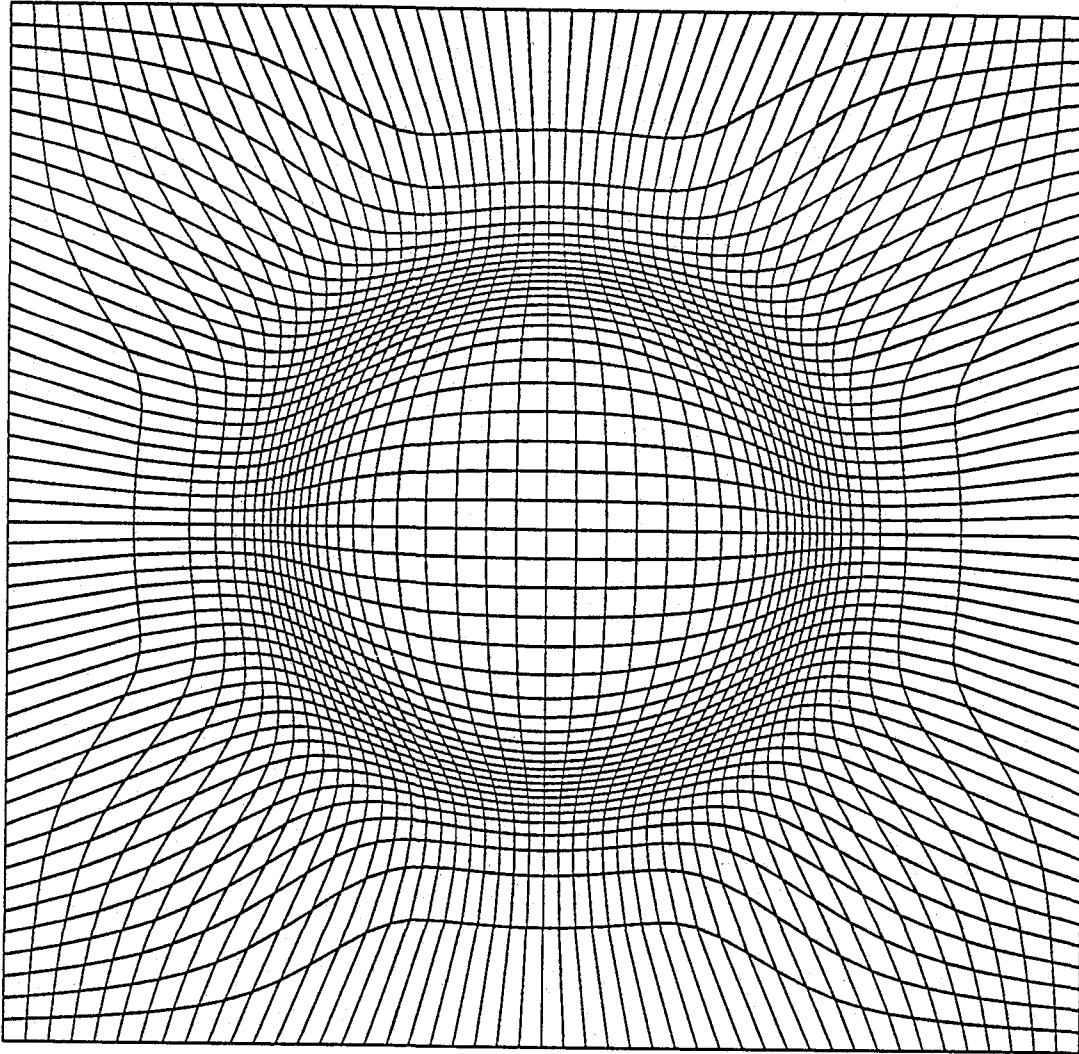
Fig. 2

Grid 48 x 48   Iter= 500

Figure A.2   Mesh computed with Dirichlet boundary conditions
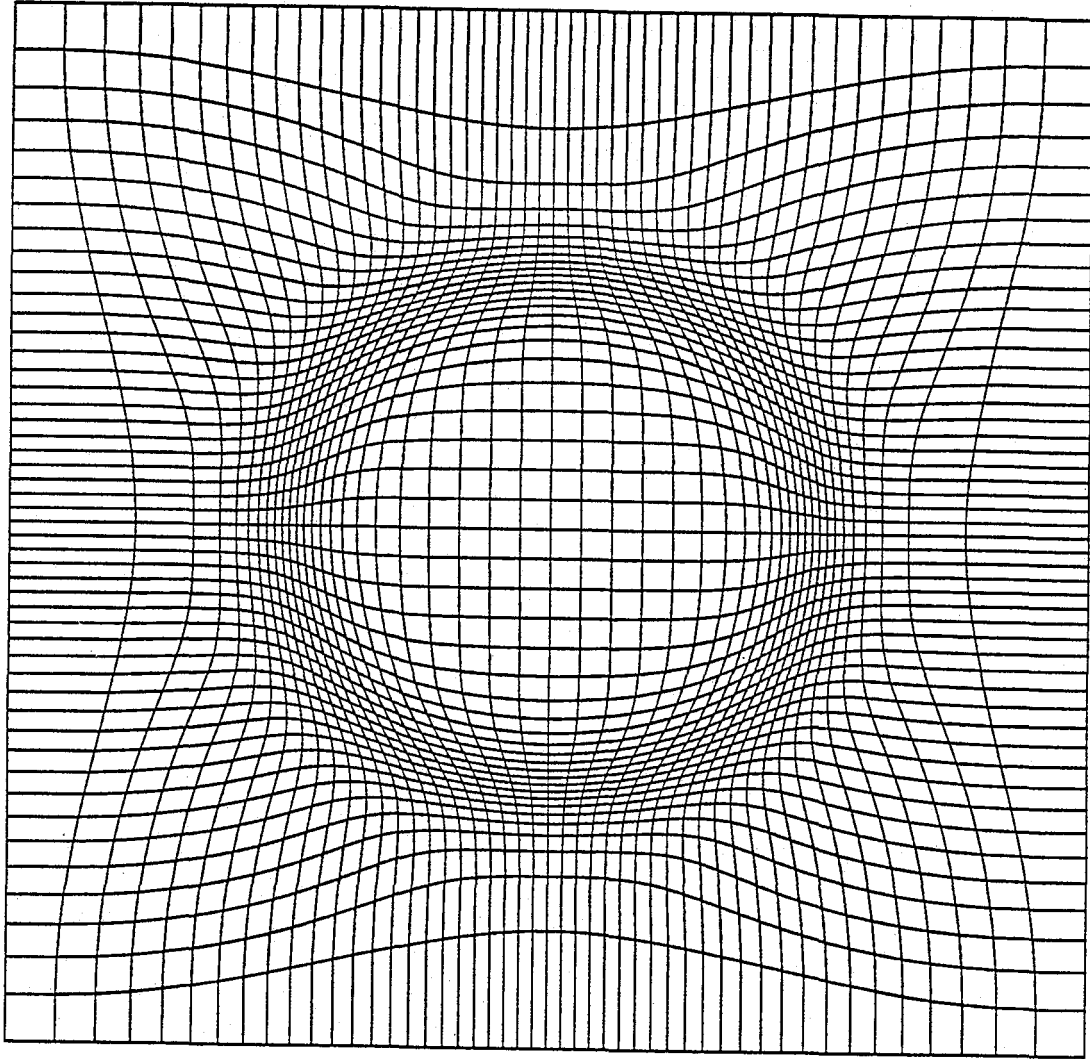   and no orthogonality control.

Fig. 3

Grid 48 x 48   Iter= 500

Figure A.3 Mesh generated with orthogonality constraint.

Let the equation we wish to solve be represented as

$$L(u) = 0,$$

where L is a differential operator and u is the exact solution. Let the numerical approximation to this equation be represented by

$$N^h(U) = 0$$

where N^h is the numerical operator on the finest grid with step size h, and U is the numerical approximation to u (i.e., is the exact solution to the difference equations). Since

$$u = U + \Delta U,$$

the local truncation error is

$$N^h(u) \sim h^n D,$$

where D contains derivatives of u, and $n = 2$ for a method which is second order accurate. The goal of the adaptive procedure is to find where D is large and to reduce the mesh spacing h there. To estimate this error we use the relation on a coarser grid having twice the mesh spacing of the fine grid

$$N^{2h}(u) \sim (2h)^n D.$$

If ΔU is small,

$$N^{2h}(U) \sim (2h)^n D,$$

and interpolating back to the fine grid gives

$$I^h_{2h}(N^{2h}(U)) \sim h^n D.$$

For steady state calculations, this relation is modified as

$$h^n D \sim I^h_{2h}(N^{2h}(U)) - N^h(U). \tag{1.16}$$

If the calculation is started with a uniform mesh distribution, regions where $|I^h_{2h}(N^{2h}(U)) - N^h(U)|$ is large indicates D is large and h needs to be reduced.

Since the solution minimizing I_w for $w = w(\xi, \eta)$ corresponds to $wJ = const.$, the weight function is taken as

$$w = \frac{|I^h_{2h}(N^{2h}(U)) - N^h(U)|}{J} \tag{1.17}$$

with the adaptation process terminating when $h^n D$ reaches a constant.

The 64×16 O-grid used in Flo52 in calculating flow field over an NACA 0012 airfoil at Mach number 0.80 and zero angle of attack is taken as a test case for applying the solution adaptive algorithm. The residual calculation, based on the root mean square of the term $\partial \rho / \partial t$ after 50 multigrid cycles, indicates that the regions of largest truncation error are located near the leading and trailing edges of the airfoil. With the weight function defined as in Eq. (1.17), the Jacobi method is used to solve Eqs. (1.13) and (1.14), with the far field boundary points held fixed and the inner boundary points allowed to move along the profile surface to satisfy grid orthogonality. After 100 iterations the new grid is shown in Figures A.4 - A.7. The pressure
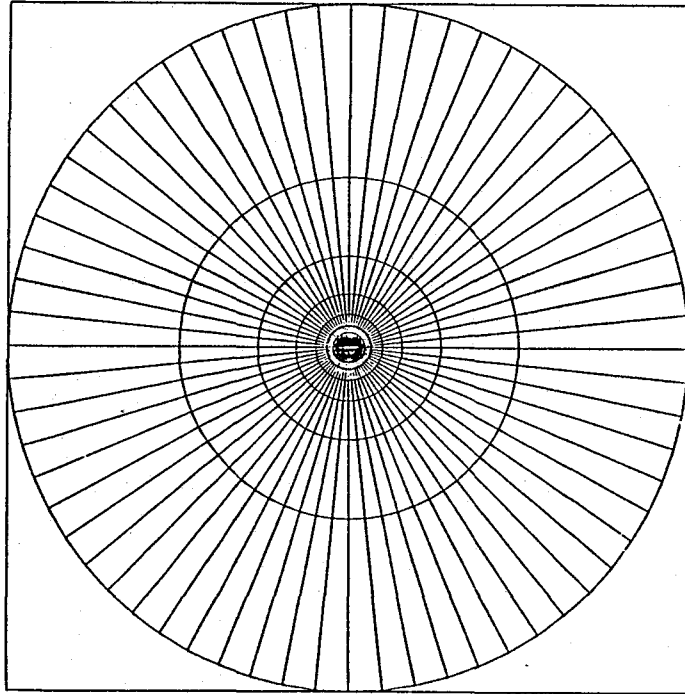
Old Mesh from flo52
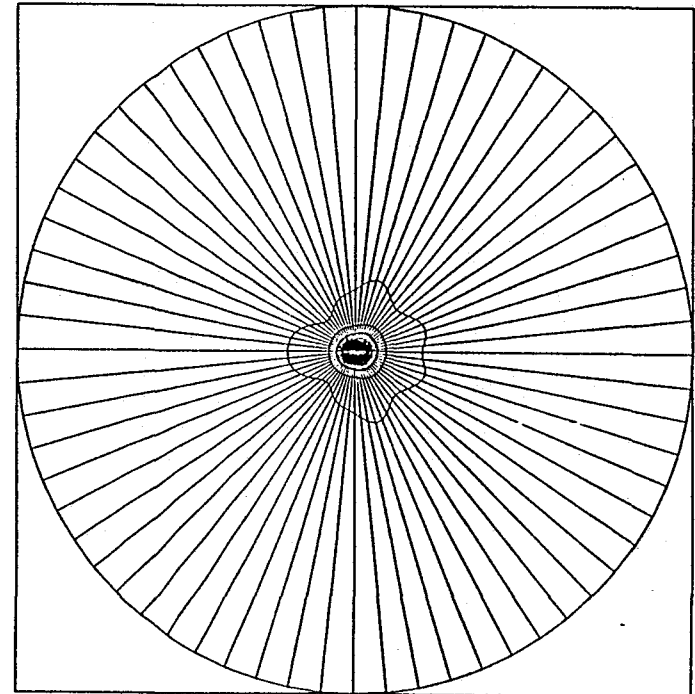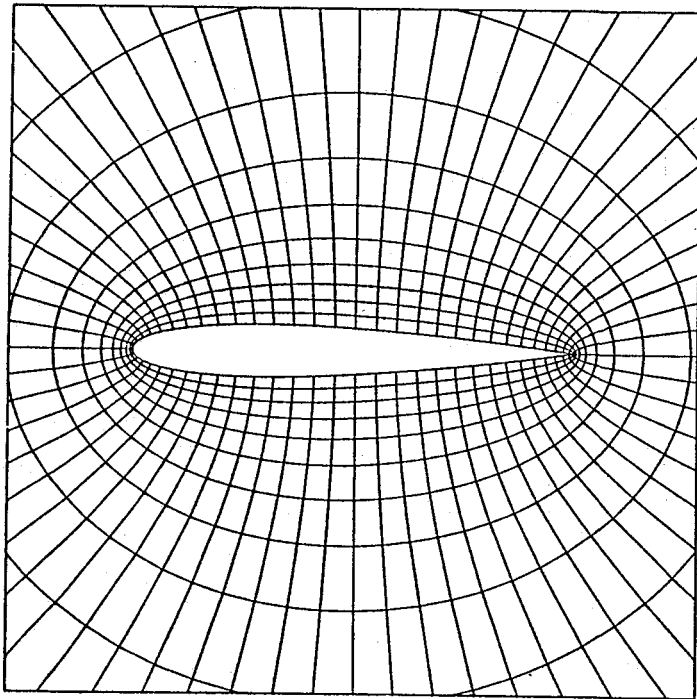Grid   64 x 16
Window Size X1 = -0.0 X2 = 200.0

Fig. 4 New Mesh after Adaption
Grid   64 x 16   Iter= 100
Window Size X1 = -0.0 X2 = 200.0

Figure A.4   Original and adaptive meshes for transonic airfoil problem.

Old Mesh from flo52
Grid 64 x 16
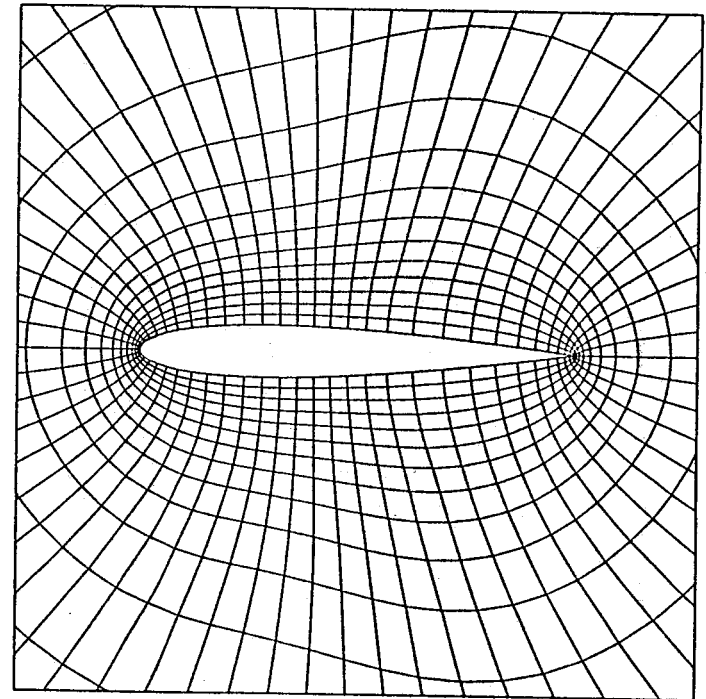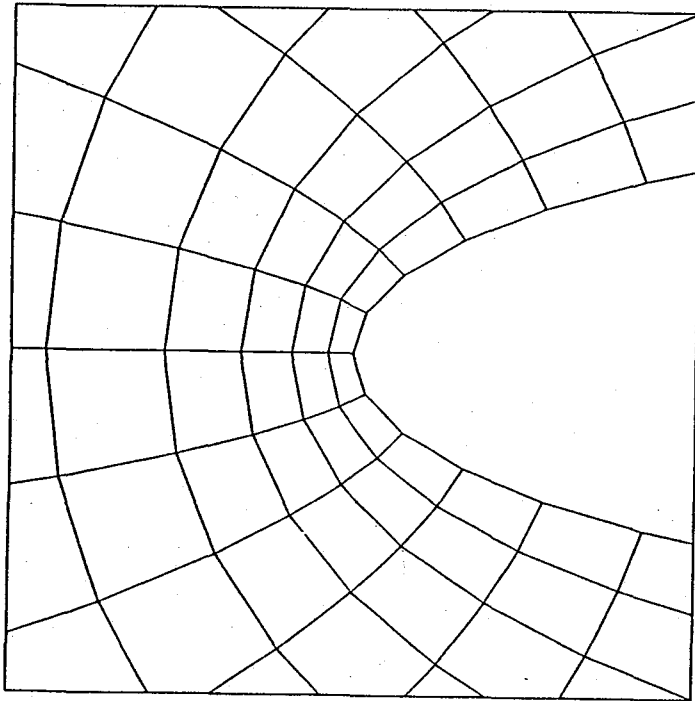Window Size X1 = 97.0 X2 = 103.0

Fig. 5 New Mesh after Adaption
Grid 64 x 16   Iter= 100
Window Size X1 = 97.0 X2 = 103.0

Figure A.5   Original and adaptive meshes for transonic airfoil problem.

Old Mesh from flo52
Grid   64 x 16
Window Size X1 = 97.8 X2 = 98.4
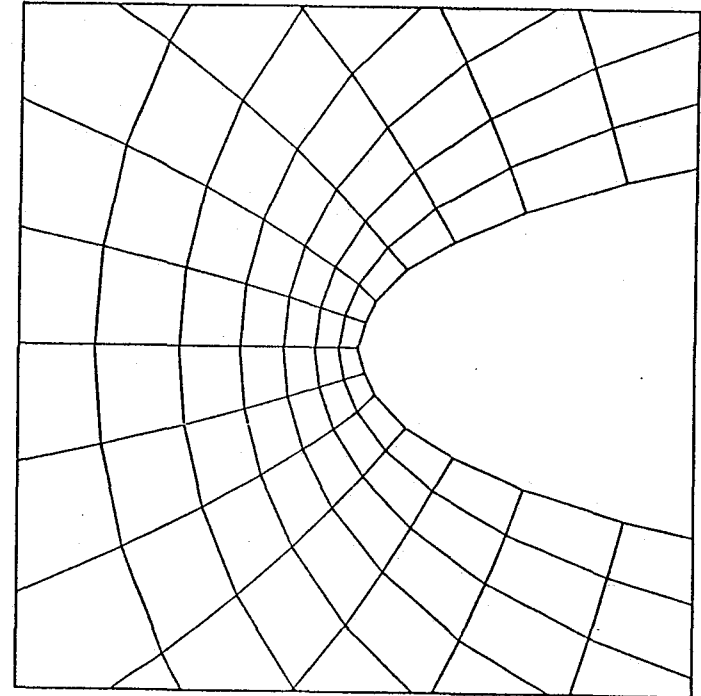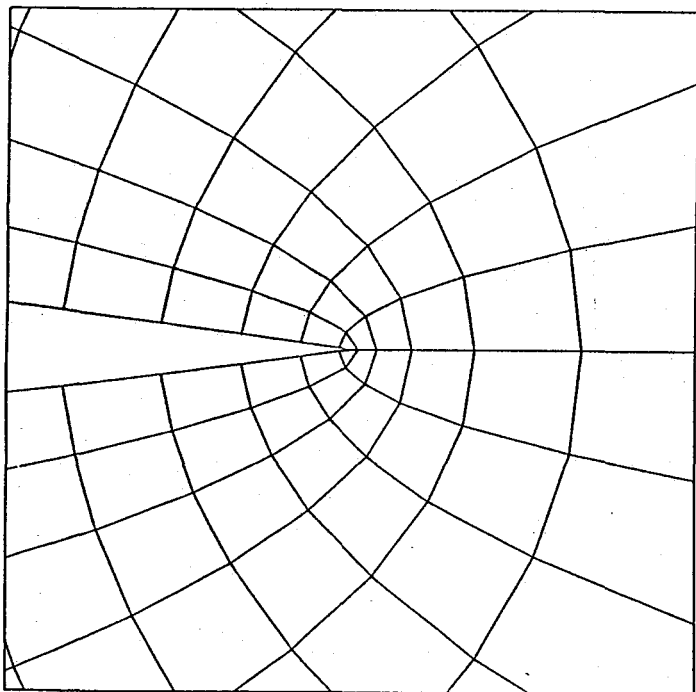
Fig. 6 New Mesh after Adaption
Grid   64 x 16   Iter= 100
Window Size X1 = 97.8 X2 = 98.4

Figure A.6   Original and adaptive meshes for transonic airfoil problem.

Old Mesh from flo52
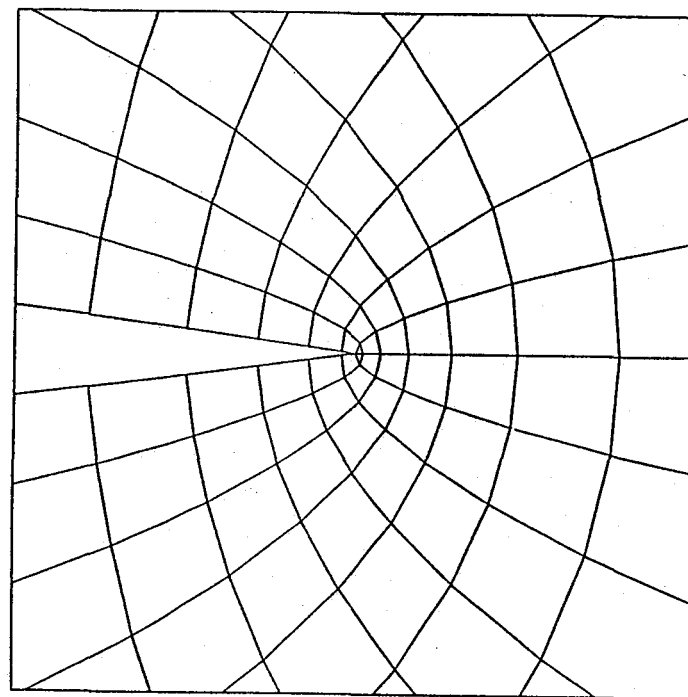Grid   64 x 16
Window Size X1 = 101.6 X2 = 102.2

Fig. 7 New Mesh after Adaption
Grid   64 x 16   Iter= 100
Window Size X1 = 101.6 X2 = 102.2

Figure A.7 Original and adaptive meshes for transonic airfoil problem.

distribution and convergence history after the calculation is continued for another 50 multigrid cycles on this new grid are shown in Figures A.8 and A.9, respectively. It is worth noting that after the same number of multigrid cycles the solution adaptive grid has reduced the largest truncation errors at the leading and trailing edges, and the standard deviation of the truncation errors on the original grid, by almost fifty per cent.

B. Diagonal Implicit Alternating Direction Multigrid Scheme

In order to circumvent the problem of slow convergence of explicit methods on grids containing cells of high aspect ratio, it seems attractive to develop implicit alternating direction multigrid methods. These methods have also been studied recently by Jameson and Yoon.[8] In order for the implicit method to be an effective smoothing algorithm when used in conjunction with the multigrid algorithm, it is important to include an accurate representation of the dissipative terms. Since these usually include fourth-differences to maintain high accuracy, their inclusion in the implicit operator requires the inversion of block pentadiagonal systems for each one-dimensional factor. To avoid the high cost of solving block pentadiagonal systems, the equations in the present scheme are first diagonalized at each point using a local similarity transformation, following Chaussee and Pulliam.[9] This has the effect of decoupling the equations, and requiring the solution of four (in two-dimensional problems) *scalar* pentadiagonal equations for each factor. The resulting method has good high-wavenumber damping, so is a good smoothing algorithm for use in conjunction with the multigrid method. It is also computationally efficient because of the need to solve only scalar systems. Additional computation is required to calculate the local similarity transformations, and to perform matrix multiplies of the residual, and of the intermediate and final corrections, but this is a small fraction of the work required to solve the block systems, and also can be vectorized to further reduce the required CPU time.

The Euler equations of inviscid compressible flow can be written (in two space dimensions) as

$$\partial \vec{w}/\partial t + \partial \vec{f}/\partial x + \partial \vec{g}/\partial y = 0 \,, \tag{2.1}$$

where

$$\vec{w} = \{ \rho, \rho u, \rho v, e \}^T \,, \tag{2.2a}$$

is the vector of dependent variables, and

$$\vec{f} = \{ \rho u, \rho u^2 + p, \rho uv, (e+p)u \}^T \,, \tag{2.2b}$$

$$\vec{g} = \{ \rho v, \rho uv, \rho v^2 + p, (e+p)v \}^T \,, \tag{2.2c}$$

are the flux vectors in the x and y coordinate directions, respectively. Here, ρ and p are the fluid density and pressure, u,v are the velocity components in the x and y directions, and e is the total energy per unit volume. The pressure is related to the total energy per unit volume e by the equation of state

$$p = (\gamma - 1)\{e - \rho(u^2 + v^2)/2\} \,, \tag{2.3}$$

where γ is the ratio of specific heats.
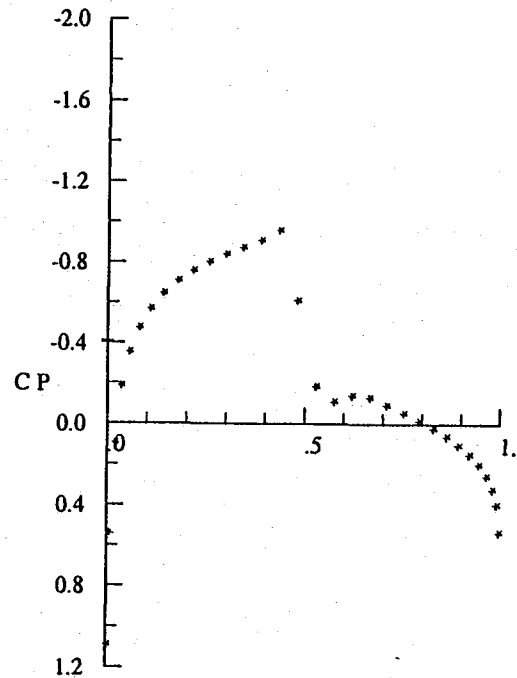
NACA 0012
Mach 0.800   Alpha 0.
Cl -0.0000   Cd 0.0101   Cm 0.0000
Grid 64x16   Ncyc   100   Res0.639e-04
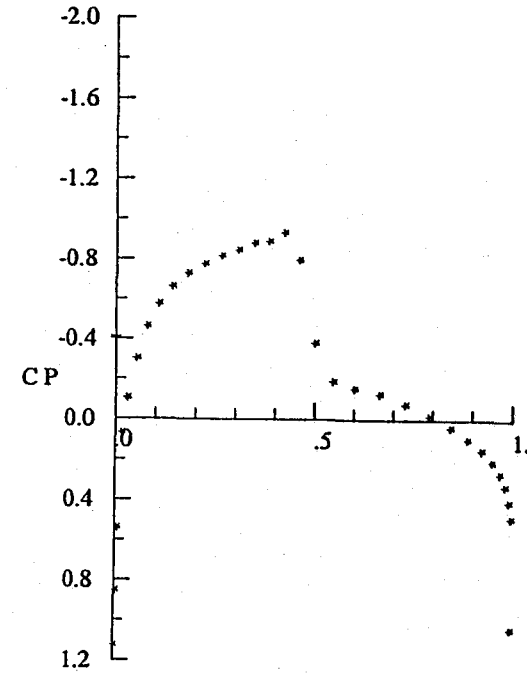
Fig. 8
Mach 0.800   Alpha 0.
Cl -0.0000   Cd 0.0099   Cm 0.0000
Grid 64x16   Ncyc   101   Res0.849e-05

Figure A.8   Pressure distributions on airfoil surface for original and adaptive solutions.
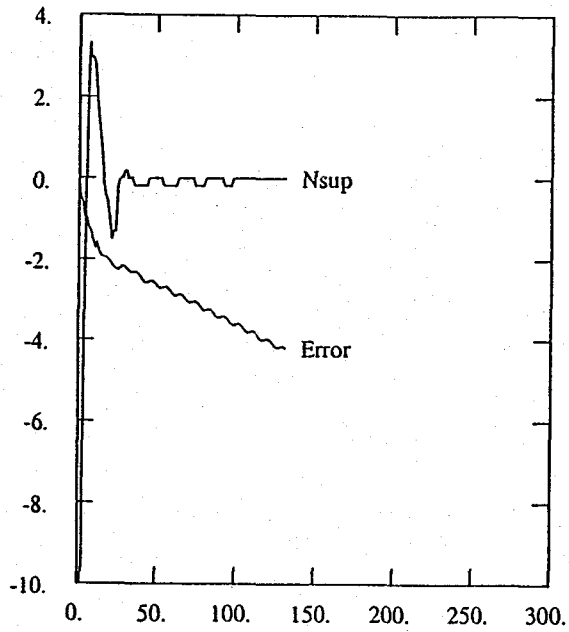
NACA 0012
Mach   0.800   Alpha   0.
Resid1 0.102e+01   Resid2 0.639e-04
Work   131.48   Rate   0.9290
Grid   64x16

Fig. 9
Mach   0.800   Alpha   0.
Resid1 0.102e+01   Resid2 0.849e-05
Work   132.81   Rate   0.9157
Grid   64x16

Figure A.9   Convergence histories for original and adaptive grid solutions.

The algorithm has been implemented within a finite-volume framework to allow the treatment of essentially arbitrary geometries. Under an arbitrary nonsingular transformation of dependent variables the equations can be written
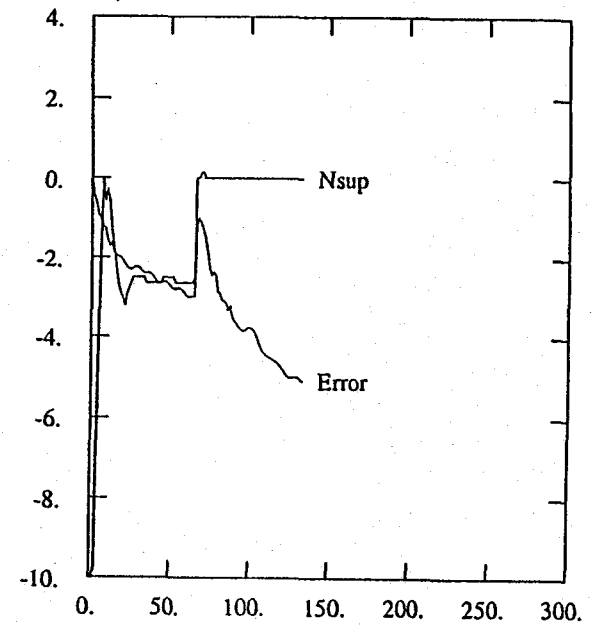
$$\partial \vec{W}/\partial t + \partial \vec{F}/\partial \xi + \partial \vec{G}/\partial \eta = 0 , \tag{2.4}$$

where $\vec{W} = h\vec{w}$ is the transformed dependent variable and

$$\vec{F} = \{ \rho h U, \rho h U u + y_\eta p, \rho h U v - x_\eta p, (e+p)hU \}^T , \tag{2.5a}$$

$$\vec{G} = \{ \rho h V, \rho h V u - y_\xi p, \rho h V v + x_\xi p, (e+p)hV \}^T , \tag{2.5b}$$

are the transformed flux vectors. Here $h = x_\xi y_\eta - x_\eta y_\xi$ is the determinant of the Jacobian of the transformation (which corresponds to the cells area), and U and V are the contravariant components of the velocity given by

$$h \{U,V\}^T = \begin{Bmatrix} y_\eta u - x_\eta v \\ -y_\xi u + x_\xi v \end{Bmatrix} \tag{2.6}$$

In a finite-volume method, the spatial derivatives are represented by approximating the net flux across the faces of each mesh cell, using constant values of the velocities to evaluate the flux across each face. In the present method, the dependent variables are defined at the cell centers; alternatively, these values can be considered as averages for the cell. The value on each face is taken to be the average of the cells sharing the face, so that for example

$$(hU)_{i+\frac{1}{2},j} = \frac{1}{2}\{(y_{i,j+\frac{1}{2}} - y_{i,j-\frac{1}{2}})(u_{i+1,j} + u_{i,j}) -$$
$$(x_{i,j+\frac{1}{2}} - x_{i,j-\frac{1}{2}})(v_{i+1,j} + v_{i,j})\} \tag{2.7}$$

This approximation is equivalent to a centered difference scheme which is second-order accurate in the mesh spacing in the physical domain when the mesh is smooth. In order to prevent the decoupling of the solution at odd and even numbered cells in the grid, dissipative terms must be added. Following Jameson,[10] the dissipative terms are constructed as an adaptive blend of second- and fourth-differences. As pointed out by Jameson, the fourth-difference terms are necessary if the scheme is to converge to a steady state, while the second-difference terms are necessary to prevent excessive oscillation of the solution in the vicinity of shock waves.

The difference approximation, including the dissipative terms, can be written

$$d(\vec{W}_{i,j})/dt + Q\vec{w}_{i,j} - D\vec{w}_{i,j} = 0, \tag{2.8}$$

where Q is the operator corresponding to the differences introduced by approximations of the form of Eq.(2.7) for the fluxes, and D is an operator corresponding to the dissipative terms, which will now be defined. The dissipative operator can be written

$$D\vec{w} = D_\xi \vec{w} + D_\eta \vec{w}, \tag{2.9}$$

where

$$D_\xi \vec{w} = \delta_\xi(d_\xi \vec{w}), \tag{2.10a}$$

$$D_\eta \vec{w} = \delta_\eta(d_\eta \vec{w}), \tag{2.10b}$$

where δ_ξ and δ_η are central difference operators spanning a single mesh cell, and

$$(d_\xi \vec{w})_{i+\frac{1}{2}} = \{\varepsilon^{(2)}\delta_\xi - \varepsilon^{(4)}\delta_\xi^3\}\vec{w} \ . \tag{2.11}$$

The coefficients $\varepsilon^{(2)}$ and $\varepsilon^{(4)}$ are adapted to the solution and are defined as follows. Let

$$\nu_{i,j} = |(p_{i+1,j} - 2p_{i,j} + p_{i-1,j})/(p_{i+1,j} + 2p_{i,j} + p_{i-1,j})| \ , \tag{2.12}$$

then

$$\varepsilon^{(2)}_{i+\frac{1}{2},j} = \frac{h_{i+\frac{1}{2},j}}{\Delta t^*}\kappa^{(2)} \max(\nu_{i+1,j}, \nu_{i,j}) \tag{2.13a}$$

and

$$\varepsilon^{(4)}_{i+\frac{1}{2},j} = \max(0, \frac{h_{i+\frac{1}{2},j}}{\Delta t^*}\kappa^{(4)} - \varepsilon^{(2)}_{i+\frac{1}{2},j}) \ , \tag{2.13b}$$

where Δt^* is the time-step corresponding to unit Courant number for the cell, and $\kappa^{(2)}$ and $\kappa^{(4)}$ are constants (chosen to be 1 and 1/64, respectively, in the present study. This scaling of the dissipative terms with $1/\Delta t^*$ makes them proportional to the propagation speed of that characteristic which limits the time step. The dissipative terms corresponding to the operator D_η are similarly defined. This tailoring of the dissipative coefficients simultaneously activates the second-difference terms and turns off the fourth-difference terms near shock waves, allowing them to be captured with little, or no, overshoot.

In smooth regions of the flow, the equations corresponding to Eqs.(2.4) plus the dissipative terms just described can be written in the quasilinear form

$$\partial\vec{W}/\partial t + A\,\partial\vec{W}/\partial\xi + B\,\partial\vec{W}/\partial\eta =$$

$$\partial\{(\varepsilon^{(2)}\partial w/\partial\xi - \varepsilon^{(4)}\partial^3 w/\partial\xi^3)\}/\partial\xi +$$

$$\partial\{(\varepsilon^{(2)}\partial w/\partial\eta - \varepsilon^{(4)}\partial^3 w/\partial\eta^3)\}/\partial\eta \ , \tag{2.14}$$

where $A = \{\partial\vec{F}/\partial\vec{W}\}$ and $B = \{\partial\vec{G}/\partial\vec{W}\}$ are the Jacobians of the flux vectors with respect to the solution. These matrices are given by Chaussee and Pulliam.[9]

The basis of Alternating Direction Implicit (ADI) methods is to approximate the spatial derivatives as weighted averages of differences taken at the old and new time levels, linearizing the changes in the flux vectors using the Taylor series expansions in time

$$\vec{F}^{n+1}_{i,j} = \vec{F}^n_{i,j} + A^n_{i,j}\Delta\vec{W}^n_{i,j} + O(\Delta t^2), \tag{2.15a}$$

$$\vec{G}^{n+1}_{i,j} = \vec{G}^n_{i,j} + B^n_{i,j}\Delta\vec{W}^n_{i,j} + O(\Delta t^2), \tag{2.15b}$$

where

$$\Delta\vec{W}^n_{i,j} = \vec{W}^{n+1}_{i,j} - \vec{W}^n_{i,j}$$

is the correction to be added to the solution in the i,j cell in going from the n to the $n+1$ time level. This gives a scheme of the form

$$\{I + \theta\Delta t[A_{i,j}\delta_\xi + B_{i,j}\delta_\eta - \varepsilon^{(2)}_{i,j}(\delta_\xi^2 + \delta_\eta^2) + \varepsilon^{(4)}_{i,j}(\delta_\xi^4 + \delta_\eta^4)]\}^n\Delta\vec{W}^n_{i,j} =$$

$$- \Delta t\{\delta_\xi\vec{F}_{i,j} + \delta_\eta\vec{G}_{i,j} + \varepsilon^{(2)}_{i,j}(\delta_\xi^2 + \delta_\eta^2)\vec{w} - \varepsilon^{(4)}_{i,j}(\delta_\xi^4 + \delta_\eta^4)\vec{w}]\}^n, \tag{2.16}$$

where θ is a parameter determining the degree of implicitness of the scheme (with $\theta = 1$ corresponding to a fully implicit scheme.

Note that for simplicity, Eqs.(2.16) are written as if the coefficients of the ξ- and η-difference dissipation terms are the same, but they are, of course, different, depending as they do on the second derivative of the pressure in the appropriate coordinate direction.

Finally, to make the scheme computationally efficient, the operator on the left-hand side of Eq.(2.16) is approximated as the product of two one-dimensional factors to give

$$\{I + \theta \Delta t [A_{i,j}^n \delta_\xi - \varepsilon_{i,j}^{(2)} \delta_\xi^2 + \varepsilon_{i,j}^{(4)} \delta_\xi^4]\}$$
$$\{I + \theta \Delta t [B_{i,j}^n \delta_\eta - \varepsilon_{i,j}^{(2)} \delta_\eta^2 + \varepsilon_{i,j}^{(4)} \delta_\eta^4]\} \Delta \vec{W}_{i,j}^n =$$
$$- \Delta t \{\delta_\xi \vec{F}_{i,j} + \delta_\eta \vec{G}_{i,j} + \varepsilon_{i,j}^{(2)} (\delta_\xi^2 + \delta_\eta^2) \vec{w} - \varepsilon_{i,j}^{(4)} (\delta_\xi^4 + \delta_\eta^4) \vec{w}]\}^n . \qquad (2.17)$$

The solution of Eqs.(2.17) is achieved in two steps. First, the intermediate correction

$$\Delta \vec{W}_{i,j}^* = \{I + \theta \Delta t [B_{i,j}^n \delta_\eta - \varepsilon_{i,j}^{(2)} \delta_\eta^2 + \varepsilon_{i,j}^{(4)} \delta_\eta^4]\} \Delta \vec{W}_{i,j}^n \qquad (2.18a)$$

is determined by solving

$$\{I + \theta \Delta t [A_{i,j}^n \delta_\xi - \varepsilon_{i,j}^{(2)} \delta_\xi^2 + \varepsilon_{i,j}^{(4)} \delta_\xi^4]\} \Delta \vec{V}_{i,j}^* = \vec{R}_{i,j} , \qquad (2.18b)$$

where $\vec{R}_{i,j}$ is the residual vector corresponding to the right hand side of Eqs.(2.17). Thus, to advance the solution one time-step, Eqs.(2.17) require the inversion of one block pentadiagonal system along each line of constant η, followed by the inversion of one block pentadiagonal system along each line of constant ξ. The size of the blocks is $(4x4)$ for this two dimensional problem; for a three-dimensional problem, there are three factors to be inverted, and the blocks are $(5x5)$.

The scheme described by Eqs.(2.17) would be reasonably efficient if it were not necessary to add numerical dissipation to stabilize the central-difference approximation. Even so, if only second-difference terms were added (i.e., if $\varepsilon^{(4)}=0$), it would still be necessary only to solve block tridiagonal systems. As pointed out above, however, the inclusion of fourth-differences is essential if the solution is ultimately to converge to a steady state, and it is important to treat them implicitly if the solution is to converge rapidly. This can be done in a straightforward manner, following the development above, but leads to a requirement to solve block *pentadiagonal* systems for each factor. This requires approximately twice the computational labor of the block tridiagonal inversion, and begins to become computationally prohibitive.

An alternative is to diagonalize the equations at each mesh point, yielding a decoupled set of equations which can be solved using a *scalar* pentadiagonal solver. This requires approximately one-quarter the computational labor of the block pentadiagonal solution (and requires, in fact, only about half the computation required to solve the *block tridiagonal* systems). Let M_A and M_B be the modal matrices of the Jacobians A and B, so that $Q_A^{-1} A Q_A = \Lambda_A$ and $Q_B^{-1} B Q_B = \Lambda_B$ are diagonal matrices whose elements are the eigenvalues of A and B, respectively. At each point the equations can then be written

$$\{I + \theta \Delta t [\Lambda_A^n \delta_\xi - \varepsilon^{(2)} \delta_\xi^2 + \varepsilon^{(4)} \delta_\xi^4]\}$$

$$Q_A^{-1} Q_B \{I + \theta \Delta t [\Lambda_B^n \delta_\eta - \varepsilon^{(2)} \delta_\eta^2 + \varepsilon^{(4)} \delta_\eta^4]\} \Delta \vec{V}_{i,j}^n =$$

$$- \Delta t \, Q_A^{-1} \{\delta_\xi \vec{F}_{i,j} + \delta_\eta \vec{G}_{i,j} + \varepsilon_{i,j}^{(2)} (\delta_\xi^2 + \delta_\eta^2) \vec{w} - \varepsilon_{i,j}^{(4)} (\delta_\xi^4 + \delta_\eta^4) \vec{w}]\}^n \, (2.19)$$

where

$$\Delta \vec{W}_{i,j}^n = Q_B \Delta \vec{V}_{i,j}^n ,$$

i.e., $\Delta \vec{V}_{i,j}^n$ is the change in the characteristic variable corresponding to the Jacobian of the η flux vector. The solution procedure is similar in structure to that of the block scheme. First, the intermediate correction

$$\Delta \vec{V}_{i,j}^* = Q_A^{-1} Q_B \{I + \theta \Delta t [\Lambda_B^n \delta_\eta - \varepsilon^{(2)} \delta_\eta^2 + \varepsilon^{(4)} \delta_\eta^4]\} \Delta \vec{V}_{i,j}^n \qquad (2.20a)$$

is determined by solving the equations

$$\{I + \theta \Delta t [\Lambda_A^n \delta_\xi - \varepsilon^{(2)} \delta_\xi^2 + \varepsilon^{(4)} \delta_\xi^4]\} \Delta \vec{V}_{i,j}^* = Q_A^{-1} \vec{R}_{i,j} , \qquad (2.20b)$$

then the correction itself is determined by solving Eqs.(2.20a). As described above, the solution of Eqs.(2.19) requires the determination of the modal matrices of the A and B Jacobians (and their inverses) at each point, a matrix multiply of the residual vector, the solution of four scalar pentadiagonal systems along each ξ−line, another matrix multiply of the intermediate correction vector, a second set of four scalar pentadiagonal solutions along each η−line, and a final matrix multiply to extract the corrections to the primitive variables. The cost of computing the elements of the modal matrices is about the same as that of computing the elements of the Jacobian matrices in the block method, and even with the extra matrix-vector multiplies, the diagonalized procedure is still considerably more efficient than solution of the block pentadiagonal systems. The advantage of the diagonal scheme would be even greater on a vector machine, since the additional matrix-vector multiplies are vectorizable.

The incorporation of the scheme within the multigrid algorithm is straightforward, following the procedure developed by Jameson.[11] The procedure will not be further described here. Since Jameson found it desirable to use only a fixed-coefficient, second-difference form of the dissipation on coarser grids, additional efficiency with the present formulation can be achieved by using scalar *tridiagonal* solvers on the coarser grid levels.

The algorithm described above has been coded for the problem The diagonal implicit scheme has been coded for the problem of transonic flow past an airfoil, including the blended second- and fourth-difference dissipation terms and the multigrid implementation. Results have been obtained for a variety of airfoils and freestream conditions, to verify the accuracy of the basic algorithm. Here, the results of a single computation will be presented to verify the efficiency of the multigrid implementation.

The calculation was performed on a C-grid containing 192 x 32 grid cells in the wraparound and normal directions, respectively. The farfield boundary of the grid is located approximately 50 chords upstream and downstream, and 100 chords laterally, from the airfoil, and the most elongated cells have aspect ratios of approximately 168 and 114 in the η and ξ directions, respectively. The grid structure in the vicinity of the airfoil surface is shown for a NACA 0012 profile in Figure B.1. The calculations were performed on an FPS AP-264 Scientific Processor attached to an IBM 3084 host.
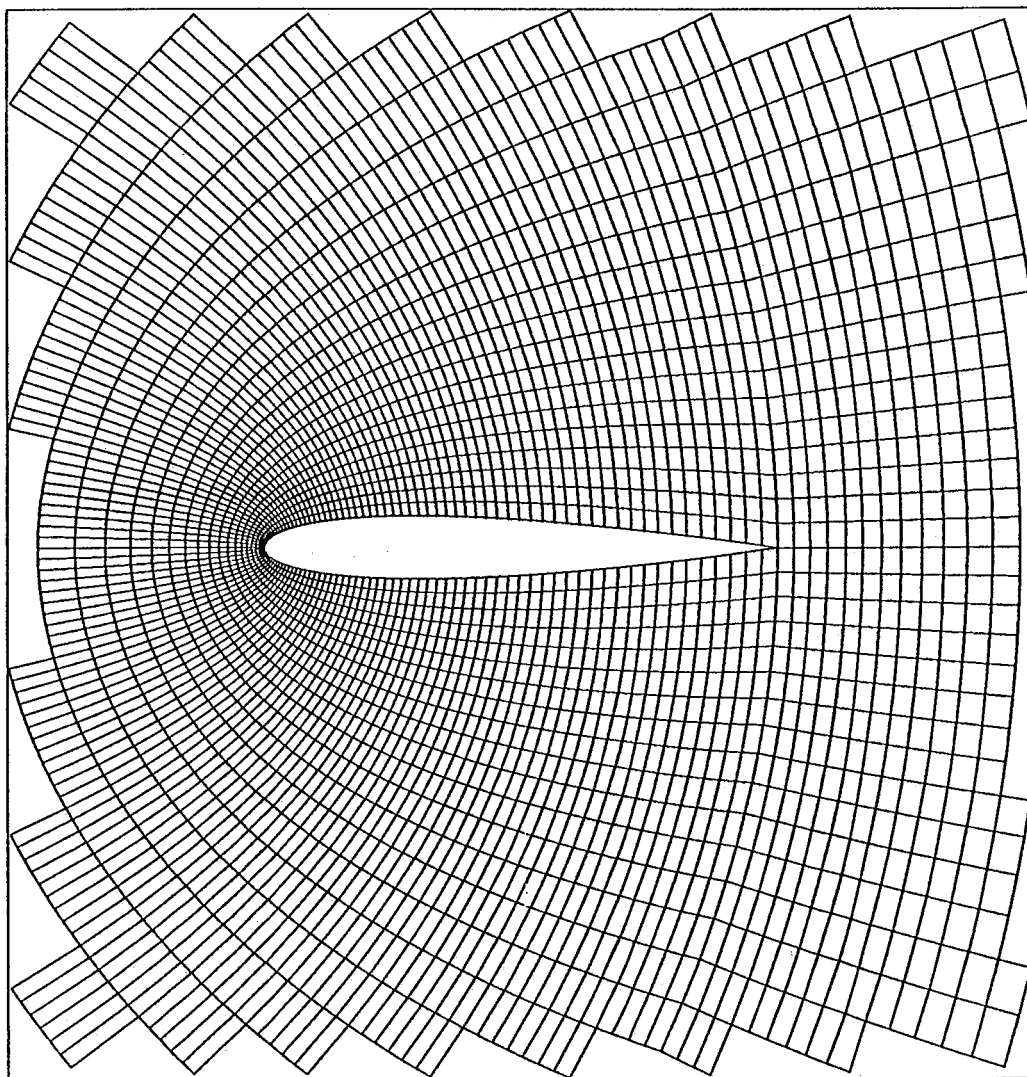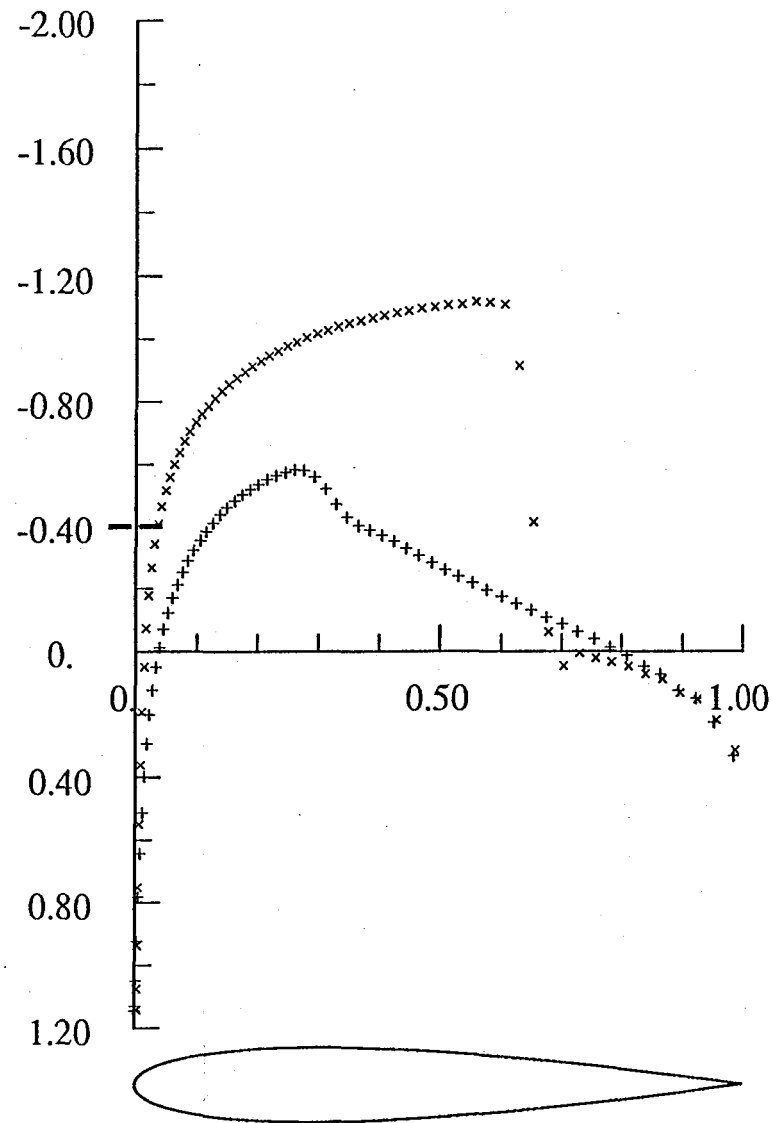
NACA 0012 Airfoil

Grid 192 x 32

Figure B.1 Computational grid in vicinity of airfoil surface.

A typical calculation (consisting of 100 work units) requires approximately 3.0 minutes for the decoupled pentadiagonal implicit scheme, or 8.4 minutes for the tridiagonal block implicit scheme, demonstrating the relative computational efficiency of the diagonalized scheme.

The airfoil surface pressure distribution for the case of the flow past the NACA 0012 airfoil at a freestream Mach number of 0.80 and 1.5 degrees angle of attack is presented in Figure B.2, showing a strong shock on the upper surface, and a weak shock on the lower surface of the profile.

Convergence results are presented in Figure B.3 for this case. The solid lines in Figure B.3 represent the convergence history for the present diagonal implicit multigrid scheme for this transonic, lifting case, while the broken line represents the convergence history for a single-grid implementation of the diagonal implicit algorithm. The logarithm of the average residual (average over all the cells of the absolute value of $\Delta\rho/\Delta t$) and the total number of cells in which the local Mach number is supersonic are plotted vs. work units. The latter quantity is a measure of the global convergence of the solution. The Courant number for the implicit multigrid calculation is 8., while that for the single-grid calculation is 12. For both calculations, local time-stepping is used. (That is, the time step in each cell is adjusted to correspond to a fixed Courant number). The implicit scheme remains stable at higher Courant numbers, but the overall convergence rate is degraded because of the poorer high-wavenumber damping at the larger values of time step. The average residual in the multigrid calculation has been reduced by almost seven orders of magnitude in only 100 multigrid cycles (corresponding to approximately 200 Work units) and the number of supersonic cells has converged to its final value after only 50 multigrid cycles (corresponding to about 100 work units). The single-grid residual has been reduced by less than three orders of magnitude in the same 200 work units, and the number of supersonic cells is still in error by more than 7 per cent.

Work is continuing on the development of the scheme, with particular attention being given to the construction of the dissipative terms, and the performance of the scheme on more highly stretched grids. In addition, the algorithm is being applied to compute the flow in a supersonic inlet. A computational grid suitable for the multigrid scheme has been developed using a modification of the procedure of Caughey and Jameson;[12] a sample of the grid is shown in Figure B.4. Attempts are also being made to modify the GRAPE code[13] for this class of geometries in order to provide greater control over local mesh spacing. Work on adapting the diagonalized ADI multigrid algorithm for this problem is currently underway.

NACA 0012 Airfoil
Mach 0.800   Alpha 1.250
Cl   0.3742   Cd   0.0237   Cm   -0.1376
Grid 192x32   Work 200.61   Res 0.366e-05

Figure B.2 Airfoil surface pressure distribution.
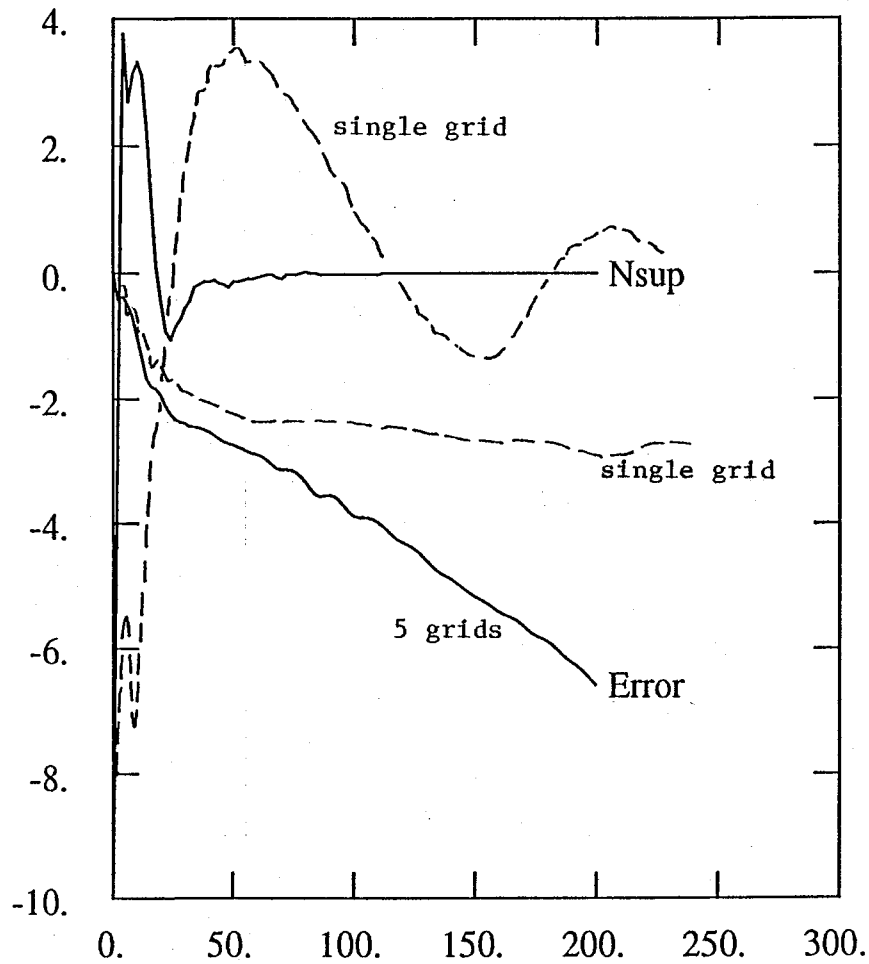
NACA 0012 Airfoil
Mach 0.800   Alpha 1.250
Res1 0.138e+02   CFL   8.00
Res2 0.366e-05   Grid 192x32
Work 199.61   Rate 0.9270   Nmesh   5
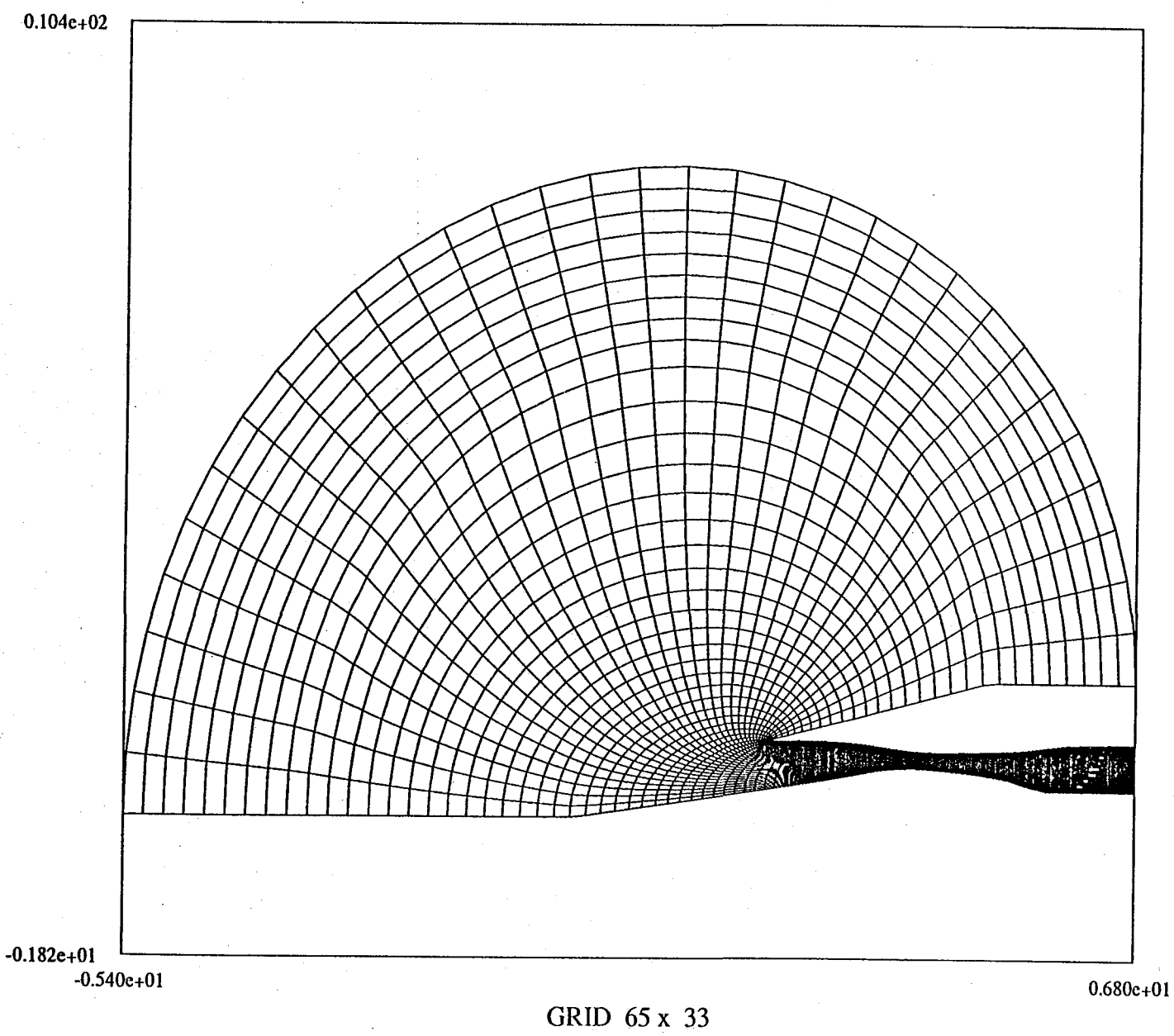
Figure B.3 Convergence histories with and without multigrid.

GRID 65 x 33

Figure B.4 Computational grid for inlet flow field.

References

1. J. F. Thompson, "Grid Generation Techniques in Computational Fluid Dynamics," *AIAA Journal*, vol. 22, no. 11, pp. 1505-1523, November 1984.

2. J. F. Thompson, F. C. Thames, and C. Wayne Mastin, "Automatic Numerical Generation of Body-Fitted Curvilinear Coordinates System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies," *Journal of Computational Physics*, vol. 15, pp. 299-319, 1974.

3. J. U. Brackbill and J. Saltzman, "Adaptive Zoning for Singular Problems in Two-Dimensions," *Journal of Computational Physics*, vol. 46, pp. 342-368, January 1982.

4. J. U. Brackbill, "Coordinate Systems Control: Adaptive Meshes," in *Numerical Grid Generation*, ed. J. F. Thompson, pp. 277-294, Elsevier Science Publishing Co. Inc., 1982.

5. A. Winslow, "Numerical Solution of the Quasilinear Poisson Equation," *Journal of Computational Physics*, vol. 1, pp. 149-172, 1966.

6. P. A. Gnoffo, "A Vectorized, Finite Volume, Adaptive Grid Algorithm Applied to Planetary Entry Problems," *AIAA Journal*, vol. 21, p. 1249, 1983.

7. M. J. Berger and A. Jameson, "Automatic Adaptive Grid Refinement for the Euler Equations," *AIAA Journal*, vol. 23, no. 4, pp. 561-568, 1985.

8. Antony Jameson and S. Yoon, "Multigrid Solution of the Euler Equations Using Implicit Schemes," AIAA Paper 85-0293, AIAA 23rd Aerospace Sciences Meeting, Reno, Nevada, 1985.

9. D. S. Chaussee and T. H. Pulliam, "Two-Dimensional Inlet Simulation Using a Diagonal Implicit Algorithm," *AIAA Journal*, vol. 19, pp. 153-159, 1981.

10. Antony Jameson, "Steady State Solution of the Euler Equations for Transonic Flow," *Proc. Symposium on Transonic, Shock, and Multi-Dimensional Flows*, pp. 37-70, Academic Press, Madison, Wisconsin, 1982.

11. Antony Jameson, *Solution of the Euler Equations by a Multigrid Method*, Princeton University MAE Report 1613, 1983.

12. D. A. Caughey and A. Jameson, "Accelerated Iterative Calculation of Transonic Nacelle Flowfields," *AIAA Journal*, vol. 15, pp. 1474-1480, 1977.

13. R. L. Sorensen, "A Computer Program to Generate Two-dimensional Grids about Airfoils and Other Shapes by the use of Poisson's Equation," NASA TM-81198, 1980.