A SECOND ORDER FINITE ELEMENT METHOD FOR THE SOLUTION OF THE TRANSONIC EULER AND NAVIER-STOKES EQUATIONS

Guido S. Baruzzi

A Thesis
in
The Department
of
Mechanical Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montréal, Québec, Canada

April 1995

ISBN 0-612-01279-4

Canada

ABSTRACT

A Second Order Finite Element Method for the Solution of the Transonic Euler and Navier-Stokes Equations

Guido S. Baruzzi, Ph.D.
Concordia University, 1995

The numerical solution of the compressible Euler and Navier-Stokes equations in primitive variables form requires the use of artificial viscosity or upwinding. Methods that are first order accurate are too dissipative and reduce the effective Reynolds number substantially, unless a very fine grid is used. A first order finite element method for the solution of the Euler and Navier-Stokes equations can be constructed by adding Laplacians of the primitive variables to the governing equations. Second order schemes may require a fourth order dissipation and higher order elements.

A finite element approach is proposed in which the fourth order dissipation is recast as the difference of two Laplacian operators, allowing the use of bilinear elements. The Laplacians of the primitive variables of the first order scheme are thus balanced by additional terms obtained from the governing equations themselves, tensor identities or other forms of nodal averaging. Finally, considerable speedup in solution execution is obtained by parallelizing significant portions of the finite element code, namely matrix assembly and the linear equation solver.

To formally demonstrate the accuracy of this scheme, an exact solution is introduced which satisfies the continuity equation identically and the momentum equations through forcing functions. The solutions of several

transonic and supersonic inviscid and laminar viscous test cases are also presented and compared to other available numerical data.

Furthermore, the numerical error stemming from an arbitrary distributed mesh can be reduced by dynamically adapting the grid to the solution. The adapting method presented, applied but not restricted to structured meshes, is based on a spring analogy, where each grid line is assumed to consist of springs connecting each pair of nodes. Each spring coefficient is proportional to the gradient of a selected solution variable. The adaptation criterion is the minimization of the energy stored in the springs of each mesh line, for which there exists a variational principle. The variational integral is discretized using the finite element method and the resulting system of equations of each line of the structured grid is solved using a tridiagonal solver. Additional constraints can be introduced to enforce a measure of orthogonality and smoothness in the adapted grid. This procedure is applied to supersonic flow over a wedge, to underline the substantial improvement in the quality of the results that this method can provide.

ACKNOWLEDGMENTS

First and foremost, I wish to thank my thesis supervisor, Professor Wagdi G. Habashi, for his continued guidance, support and remarkable patience shown during the course of this research. Many thanks are also due to Professor Mohamed M. Hafez, of the University of California at Davis, for his lively and fruitful discussions and his valuable advice regarding this research.

I also owe my colleagues at the Concordia CFD Lab many thanks for their help and suggestions. In particular I would like to thank Professor Grant Guèvremont for his valuable suggestions, for helping me hone some of the skills that I have acquired and for the great fun that was shared in our numerical investigations. Many thanks are due to Mr. Richard Lefebvre, who, as a colleague at the CFD Lab and as a Professional at CERCA, devoted many hours to developing a fancy PostScript library to help with pictures and visualization, and without whom I could not have spilled so much toner on these pages. Thanks are also due to my colleague Mr. Pierre Gauthier for his help with those recalcitrant turbulence models.

The CFD group at Pratt & Whitney Canada also helped in making this research interesting, through our many technical discussions and information exchange. In that respect, thanks are due to Mr. M. Peeters, Drs. M. Robichaud and W. Ghaly.

A particular mention should be reserved for CERCA (Center for Research on Computation and its Applications) for complementing the resources of the CFD

Lab by giving me access to its computational facilities and its staff who helped with many logistical problems.

Finally, this Thesis is dedicated to my parents, Giorgio and Luciana, and my family for their support and encouragement, and for constantly reminding me of my ultimate goal when I strayed from the right path.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

A	Ratio of the max. and min. spring lengths
$\boldsymbol{A}$	Jacobian of the flux vector E
$\boldsymbol{B}$	Jacobian of the flux vector F
B	Exponent of the spring coefficients
C	Coefficient of the orthogonality/smoothness springs
$\boldsymbol{E}$	x-component of the flux vector
$\boldsymbol{F}$	y-component of the flux vector
$\vec{F}$	Balancing term (Laplacian of pressure)
$\vec{G}$	Balancing term (Laplacian of velocity)
$\boldsymbol{G}$	x-component of the stress tensor
$\boldsymbol{H}$	y-component of the stress tensor
H	Total enthalpy per unit mass
I	Variational integral
M	Mach number
N	Finite element shape functions
N	Number of unknowns of the linear system
Re	Reynolds number
S	Length of mesh line
T	Temperature
$\boldsymbol{U}$	State vector of the conservative variables
$\lvert\vec{V}\rvert$	Magnitude of the velocity vector
$\tilde{X}$	Left eigenvector
$\boldsymbol{X}$	Matrix of the left eigenvectors $\tilde{X}$
W	Galerkin weight functions (W=N)

W	State vector of the primitive variables
c	Speed of sound
e	Total energy
f	Gradient of the spring-driving function
f	Components of the continuity balancing terms
g	Components of the momentum balancing terms
h	Enthalpy per unit mass
k	Spring coefficient
n	Number of parallel processors
p	Pressure
$\vec{q}$	Heat transfer vector
s	Tangential coordinate of the adapted line
u	x-component of velocity
v	y-component of velocity

Greek symbols

α	Relaxation factor
	Angle of attack
β	Orthogonality/smoothness blending coefficient
γ	Ratio of the specific heats
λ	Eigenvalue
λ	Spring coefficients (orthogonality/smoothness)
μ	Molecular viscosity coefficient
ρ	Density
$\vec{\tau}$	Viscous stress tensor

Subscripts

I	Internal energy
c	Centroid
e	Element
i	Node index
j	Node index
∞	Freestream condition

1. INTRODUCTION

From an aerodynamicist's standpoint a major goal in aircraft design is to obtain the required amount of lift with a minimum of drag, so as to minimize the costs incurred in producing the thrust needed to fly. The behavior of viscous fluids is modeled by a set of nonlinear conservation laws for mass, momentum and energy: the Navier-Stokes equations. Whereas both lift and drag strongly depend on the aerodynamic body geometry, the drag is particularly influenced by viscous effects. Except for simple geometries and flow conditions, few analytical solutions can be derived for the Navier-Stokes equations. While such solutions are of great value to physical understanding, they find limited practical application because of the simplicity of the situations they cover. The numerical prediction of the drag is the weak link in the aerodynamics design process and is the current focus of computational aerodynamics research [Rubbert (1991)].

Currently, engineers still depend greatly on wind tunnel experiments of scaled models to gain better insight into the properties of proposed designs. Despite the continuing successes and a considerable amount of experience derived from such tests, it is virtually impossible to simultaneously match the Mach and Reynolds numbers of the model and the full-scale aircraft. Correlations must then be used to extrapolate the behavior in flight from wind tunnel measurements. A few cryogenic tunnels can be used to alleviate this problem, but their availability is severely limited and their cost prohibitive, providing an added economic justification for the development of cost-effective computational techniques suitable for modern civil transports flying in the transonic regime.

The numerical solution of the Navier-Stokes equations still has to wait for the emergence of sufficiently powerful computers able to handle their exceedingly demanding algorithms. Currently available computers do not have even a fraction of the power required to study the full spectrum of turbulent fluctuations characteristic of fluid flows. The direct numerical solution of the Navier-Stokes equations is confined to the study of the mechanisms that generate and sustain turbulence and to provide explanations for the causes and effects of this phenomenon [Bestek et al. (1992), Lesieur, Comte and Metais (1992)].

The numerical approaches most in use today for viscous flows are based on the solution of the Reynolds-averaged Navier-Stokes (RANS) equations, obtained from the governing equations through a time-filtering process to eliminate the small-scale fluctuations. The effect of turbulence in the solution of the RANS equations is provided through suitable models, used to simulate the Reynolds stress terms introduced by the averaging process [see Jaeger and Dhatt (1992) for an informative review]. Direct numerical simulation also plays a role in providing a database for the calibration of the wide variety of turbulence models used in conjunction with the RANS equations [Miner *et al.* (1991)].

The Euler equations are a simpler subset of the Navier-Stokes equations for the simulation of high-speed flows outside the boundary layers, obtained by ignoring the viscous terms. From the computational point of view, even though no information regarding the viscous friction forces can be obtained, the Euler equations are an important stepping stone towards the solution of the compressible Navier-Stokes equations because they can reproduce the physics of wave propagation, an important feature of high-speed flows.

An artificial viscosity must be added to the Euler equations, which have no entropy control mechanism, for the dual purpose of preventing the numerical occurrence of unphysical phenomena, such as expansion shocks, and suppressing the instabilities that can be caused by a symmetric discretization of the first order derivatives. Even for the Navier-Stokes equations, in which the molecular viscosity is the dissipative mechanism that permits the formation of compression shocks, an additional artificial viscosity must be included, partly to ensure the overall stability of the numerical schemes and partly because the grids used for the solutions are not sufficiently fine to capture the physics of shocks in detail.

1.1. Monotonicity and Total Variation

Research on the numerical discretization of scalar conservation laws or hyperbolic systems with constant coefficients has led to a greater understanding of the numerical difficulties associated with modeling wave phenomena. In particular, two important concepts have emerged: monotonicity and control of total variation. A scalar conservation law is written in the general form

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{1.1}$$

with the initial condition

$$u(0,x) = g(x) \quad \text{on} \quad -\infty < x < \infty \tag{1.2}$$

where u is the dependent variable and $f(u)$ is a flux function. The solutions of this scalar conservation law may admit discontinuities. However, since there is no entropy control mechanism present, both expansion and compression shocks can appear. The solution is said to be monotonic if the entropy can only increase in the direction of the flow. This property is violated if spurious oscillations are present in the numerical solution, particularly in the vicinity of a shock.

The other concept that has emerged is that of total variation diminishing, i.e.

$$TV\left[u\left(t^{n+1},x\right)\right] \le TV\left[u\left(t^{n},x\right)\right] \tag{1.3}$$

where the total variation is defined as

$$TV\left[u\left(t^{n},x\right)\right] = \sum_{i=-\infty}^{\infty} \left|u_{i+1} - u_i\right|^n \tag{1.4}$$

Even though it has not been proven that the TVD property exists for nonlinear multidimensional systems of conservation laws, such as the Euler equations, the schemes developed for scalar conservation laws provide a general guideline for the construction of schemes for the Euler equations with high shock resolution, which are also free from spurious oscillations.

1.2. Discretization Methods

The numerical discretization methods available for the study of the Euler and Navier-Stokes equations in primitive variables form can generally be divided into four main classes:

i) Finite difference methods,

ii) Finite volume methods,

iii) Finite element methods,

iv) Boundary element methods.

Some schemes do not fit precisely in these categories, such as control volume-based finite elements and others, but are rather combinations of two different methods in an attempt to capitalize on the favorable characteristics of both. The boundary element methods, while meeting some success for incompressible

flows, have not found any application in viscous transonic flows and hence will not be included in the discussion.

1.3. Finite Difference Schemes

Boris and Book (1973) outline a Flux Corrected Transport (FCT) scheme for the solution of scalar conservation laws based on a two-step, predictor-corrector procedure: the transport stage and the correction (anti-diffusion) stage. Hoffmann and Chiang (1993) give a simple example of this algorithm using the Lax-Wendroff scheme as the transport stage. This step conserves the flux but is quite diffusive. A flux-based corrector step is then introduced to diminish the error of diffusion and, at the same time, preserve monotonicity. A limiter function acts on the flux corrector to suppress oscillations by not allowing local extrema to grow. In a sense, the FCT algorithm is a precursor of the TVD schemes that will be mentioned in this survey.

Beam and Warming (1976) have developed one of the first successful implicit procedures, based on central differencing, for the solution of the 2-D Euler equations on structured grids. The time-stepping scheme is based on the integration of the time derivative with a trapezoidal formula, while the flux vectors of the governing equations are linearized with a truncated Taylor expansion. The implicit operator is the product of two one-dimensional operators that are solved sequentially in an Alternating Direction Implicit (ADI) manner. The ADI factorization requires the decomposition of a pentadiagonal matrix for each grid line, instead of a large global matrix, which results in an appreciable reduction in storage. A fourth order artificial viscosity provides global second order accuracy, while an artificial viscosity in the form of Laplacians of the conservative variables is used for shock capturing. Two- and three-dimensional

versions of this scheme have been implemented by Steger (1978) and Pulliam and Steger (1980) and the method has been considerably improved by Pulliam (1985) and Pulliam and Barton (1985).

Variant of this scheme, using flux-vector splitting, have also been introduced. Later, Pulliam (1985) as well as others have shown that such flux vector splitting with forward and backward differencing are equivalent to a symmetric discretization with an added artificial viscosity. Webster and Shang (1991) have used Pulliam's code, ARC3D, to study the supersonic turbulent viscous flow over a delta wing at high Reynolds numbers and high angles of attack. In another application of the Beam and Warming scheme, Miyakawa *et al.* (1987) have conducted a 3-D simulation of high-Reynolds number transonic viscous flows over a wing-body configuration.

Harten (1983), Yee and Harten (1987) and Yee (1989) have developed a new class of explicit and implicit monotone schemes, based on the Total Variation Diminishing (TVD) property of scalar conservation laws or hyperbolic systems. Some of these schemes, based on central differencing, have an artificial viscosity that can switch the stencil bias from upwind to downwind according to the signs of the characteristics of the coefficient matrices of the system of governing equations. The TVD property essentially provides a feedback mechanism that suppresses oscillations of the solution near local maxima and minima. An anti-diffusive flux is incorporated to ensure second order accuracy, but it must be turned off by a limiter near shocks to prevent violations of the TVD property. Several different limiters have been studied by Sweby (1982, 1985) and others. However, Roe and van Leer (1988) have shown that limiters may produce spurious solutions not related to the real solution of the governing differential equations.

When 2-D or 3-D curvilinear geometries are encountered, or if non-uniform grids are required, a considerable degree of complexity is introduced and the simplicity that originally made the finite difference discretization method popular quickly vanishes. Usually, a mapping that transforms the geometry into a rectangle or a square computational domain is required, along with an inverse mapping to revert from the computational to the physical domain. For complex geometries, the finite volume method has proven to be a more efficient discretization method that retains part of the original simplicity of finite differences without suffering from the same limitations.

1.4. Finite Volume Schemes

During the last fifteen years the finite volume method has become one of the most, if not the most, widely used discretization technique in the CFD community.

Perhaps one of the most visible applications of this discretization procedure is the explicit time-marching scheme for the solution of the Euler equations developed by Jameson (1985) for aerodynamic components and then for a full airplane. He proposed a multi-stage Runge-Kutta time-integration approach and a symmetric discretization with an added fourth order artificial viscosity in the form of a biharmonic operator for global second order accuracy and a Laplacian operator for shock-capturing. The addition of multigrid convergence acceleration improved the convergence properties considerably. Mavriplis (1987) and Mavriplis and Jameson (1990) have extended this scheme to unstructured non-nested triangular grids for inviscid flows over multiple airfoils. By using linear triangular elements and lumped mass matrices, they have also shown that this scheme is equivalent to a Galerkin finite element formulation.

Cambier and Escande (1990) have proposed an explicit finite volume scheme for the simulation of three-dimensional shock wave-turbulent boundary-layer interaction in internal flows. The time-marching scheme is based on a two-step form of the Lax-Wendroff method with multigrid convergence acceleration. Staggered grids are used: the first step consists in updating the fluxes at the centers of the control volumes, while in the second step the fluxes are updated at the nodes. An artificial viscosity in the form of a non-linear symmetric second order term and a fourth order symmetric linear operator is added for shock capturing and stability.

Recently Jameson (1993) has incorporated several high resolution symmetric and upwind flux formulae into his finite volume scheme to improve shock resolution. Using these schemes and an algebraic turbulence model, Tatsumi, Martinelli and Jameson (1995) have captured transonic high-Reynolds-number viscous solutions for flows over airfoils and wings. A comparative study of different flux formulae for upwind schemes is also provided by van Leer *et al.* (1987).

The high resolution schemes, such as the TVD methods mentioned in the previous section, can be viewed as modifications of the classical flux splitting technique. The flux splitting method consists in linearizing and separating the fluxes of the governing equations into a positive part convected downstream and a negative part convected upstream. Van Ransbeeck and Hirsch (1993) have developed finite volume schemes for the solution of the 2-D Euler and Navier-Stokes equations, based on a multidimensional approach to the flux splitting technique, to model more accurately the multidimensional nature of wave propagation. These schemes are more robust and may not suffer from grid-induced problems when strong discontinuities present in the solution are not aligned with the grid.

The effort to produce monotone schemes which can convect waves with the correct propagation velocity and capture both compression waves and contact discontinuities within few grid points has led to several interesting schemes for the solution of the 1-D Euler equations based on different forms of wave tracing. These methods are based on Godunov's scheme, in essence an explicit method that advances the solution to the next time level by solving a set of one-dimensional Riemann problems, thereby projecting each separate wave along its characteristic direction. Godunov's scheme is first order accurate because the data is assumed to be piecewise constant between nodes. However, van Leer (1979) has extended it to second order accuracy by considering a linear distribution of the data between the nodes. This scheme is called Monotonic Upwind Scheme for Conservation Laws, or MUSCL. In an effort to capture contact and entropy discontinuities with higher accuracy, Yamamoto and Daigujii (1991) have developed a fifth-order-accurate upwind scheme and a fourth order-accurate TVD scheme. The higher order schemes are obtained by introducing additional terms and widening the stencil of the second-order flux formula.

The characteristics of nonlinear systems of equations are curved, hence at each time step an additional iterative procedure must be introduced to accurately compute their trajectory. To avoid this complication, Roe (1981) has shown that for approximate discrete solutions it is not necessary to solve the Riemann problem exactly as long as the general non-linear behavior is respected. Unlike Roe, Osher and Chakravarthy (1983) use an integration scheme for the solution of the approximate Riemann problem. A good review of these methods can be found in the articles of Roe (1986) and Harten *et al.* (1987).

These complex schemes are not easily extended to higher dimensions. In two space dimensions the characteristics are conical and the picture is even more complex in three dimensions, where the number of waves increases and the characteristics propagate in an infinite number of directions. However, Rumsey, van Leer and Roe (1993) have obtained solutions for two-dimensional inviscid flows over airfoil geometries using a finite volume multistage time-marching scheme.

1.5. Finite Element Schemes

Some finite element schemes for the solution of the Navier-Stokes equations are based on Galerkin weighted residual integrals, where the weights are identical to the shape functions used to interpolate the unknowns across each element. Schemes that use weights that differ from the shape functions are called Petrov-Galerkin weighted residual methods.

Hood and Taylor (1974) introduced one of the first Galerkin finite element schemes for the solution of the incompressible 2-D Navier-Stokes equations in primitive variables form. In the absence of an artificial viscosity, stable finite difference approximations require staggered grids for pressure and velocity. The Galerkin finite element scheme of Hood and Taylor dispenses with the inconvenience of dual meshes by using unequal order elements for pressure and velocity. The combination of quadratic interpolation for velocity and linear for pressure or cubic interpolation for velocity and quadratic for pressure produces stable and accurate solutions.

Fortin and Fortin (1985) have extended the concept of unequal order interpolation to other types of elements. They construct new elements for the velocity by adding internal nodes to the standard triangular and quadrilateral

elements. The extra node supports a bubble function, so-called because it resembles a bubble over the element, which has value of one at the internal node and is zero on the perimeter of the element. It can be shown that the bubble function is equivalent to adding an artificial viscosity. The effect of the new extra node can be transferred to the corner nodes of the element in a procedure called static condensation. In this case, no new degrees of freedom are added to the solution of the problem, while stability is maintained. Boivin and Fortin (1993) have used unequal order triangular elements of this type and added an artificial viscosity to solve the compressible 2-D Navier-Stokes equations for flows with shocks.

Fletcher (1979) devised a least-squares method for the steady Euler equations based on group variables, where the group variables are the entries of the flux vectors. If the energy equation is not solved, the scheme has six variables instead of the customary three. However, the discretization leads to a symmetric matrix and from the computational standpoint the penalty in storage and solution times is not excessive. The least-squares method is naturally dissipative and no artificial viscosity is required.

Jiang and Carey (1990) introduced a least-squares method for the Euler equations in primitive variables form. The method is based on a Newton linearization and a backward-Euler first order time-accurate formula for the time derivative. An artificial viscosity was, however, added to control the instability arising with higher order elements.

Lefebvre, Peraire and Morgan (1993) have also applied the least-squares method to the solution of the Euler equations in 2-D, and they also used an implicit time-marching scheme obtained by discretizing the time derivative with a backward

difference They obtained solutions with strong shocks for supersonic flows over a circular cylinder and for transonic airfoils, using both linear and quadratic triangular elements, with an adaptive mesh refinement strategy but without the addition of artificial viscosity terms.

Hassan, Morgan and Peraire (1990) adopted a flux-corrected transport strategy to reduce the effects of the artificial viscosity and improve the quality of the solution. Lyra, Morgan, Peraire and Peiró (1994) have formulated an explicit multistage TVD edge-based Galerkin finite element scheme for unstructured meshes. A variety of upwind and symmetric limiters have been developed for the solution of inviscid flows with strong shocks.

Löhner, Morgan and Zienkiewicz (1984) used a Galerkin finite element discretization of the explicit Lax-Wendroff scheme to solve the Euler equations. An added artificial viscosity is required to capture shocks. Lerat (1985) has shown that a simple implicit predictor-corrector scheme can be constructed, with the Lax-Wendroff scheme as the predictor and a corrector step based on an approximation of its first truncation error term. The convergence rate improves dramatically and shocks can be captured within two mesh points. A one-dimensional finite element discretization of this scheme has been reported by Baruzzi (1989).

Hughes and Brooks (1982) and Tezduyar and Hughes (1983) have adopted the Petrov-Galerkin weighted residual strategy for the solution of the Euler Equations. In this method the weight functions are equal to the finite element shape functions with an added perturbation which depends on the solution and must be adjusted at each iteration. There are strong similarities between this method and the least-squares method.

Bey and Oden (1991) have developed a Runge-Kutta time-marching scheme based on discontinuous finite elements (RKDG) for the solution of the Euler equations. While the finite element shape functions are continuous inside each element, they are normally discontinuous at the edges. A constraint can be introduced to ensure that the fluxes are continuous at the element interfaces. While this constraint cannot satisfy monotonicity and the TVD condition together, by relaxing the TVD property into a Total Variation Bounded (TVB) condition the total variation is allowed to increase within some bound, as long as the solution remains smooth. Shocks not aligned with the mesh can be captured within two grid points.

Nicolaides (1993) has outlined the principles underlying the co-volume approach, which uses complementary pairs of control volumes, such as the Delaunay–Voronoi mesh pairs, to discretize the fluxes, in an effort to find lower order methods of discretization which do not produce spurious pressure solutions. The Delaunay–Voronoi mesh pairs have the property that the edges of each set of control volumes are perpendicular to the faces of the other set of control volumes, hence the velocity vectors normal to one control volume are tangential to the other. By evaluating the divergence of velocity on the faces of the triangles and the curl of velocity on the boundary of the Voronoi tessellation, the scheme exploits the orthogonality of the boundaries of the two control volumes to provide a stable discretization of system of equations containing curl and divergence terms, such as the Navier-Stokes equations. Natural boundary conditions are provided by the continuity equation, while the curl component is used to enforce no-slip at the walls. Unlike traditional discretizations, which employ two velocity components and one control volume, this method uses one

velocity component normal to the faces of the triangulation and two control volumes.

Masson, Saabas and Baliga (1994) have proposed a Control Volume Finite Element Method (CVFEM) that attempts to exploit the positive aspects of the finite volume and finite element formulations for the solution of the incompressible Navier-Stokes equations. The triangular elements posses linear shape functions, similar to finite elements, however the integration of the governing equations is performed on control volumes surrounding each node, instead of over the surface of each element. Special care must be exercised to prevent the appearance of negative coefficients in the discrete governing equations, which would generate instabilities.

1.6. Justification for the Proposed Finite Element Method

All of the discretization methods mentioned in the previous sections have attractive features that can be applied effectively to the construction of robust schemes for the solution of the Navier-Stokes equations. The finite element method, however, offers some advantages over other techniques: natural boundary conditions, geometric flexibility, and the possibility of building a library of specialized elements that may be easily interchanged or combined for special purposes.

In particular, for turbulent flows it is possible to construct elements that mimic the logarithmic behavior of the velocity near walls, removing the need to replace the solution of the governing equations in near-wall elements with empirical wall functions [Haroutunian and Engelman (1991), Manouzi and Fortin (1991), Cochran (1992)]. Even though turbulence models are not covered in this dissertation, the possibility of including logarithmic elements is in itself a

sufficient reason to warrant the selection of the finite element discretization technique for the present work, in anticipation of its eventual extension to the solution of high Reynolds number turbulent flows.

1.7. Justification for the Proposed Artificial Viscosity Scheme

The schemes for the numerical solution of the Euler equations in primitive variables form require the use of artificial viscosity or of upwinding to eliminate odd-even decoupling and for numerical stability. The viscous terms of the compressible Navier-Stokes equations at high Reynolds numbers, on the other hand, are dominant only in a thin layer outside which the flow is nearly inviscid. For numerical stability, an artificial viscosity is needed in the outer inviscid region, however it must be minimized throughout the viscous layer, or be eliminated altogether, in order not to contaminate the numerical solution with artificial dissipation.

The various types of formulations for the solution of the compressible Euler and Navier-Stokes equations that were reviewed in the previous sections can be classified as either artificial viscosity-based or as upwind-biased. The first type uses symmetric discretization operators with the addition of an artificial viscosity (with a symmetric operator) for stability and to eliminate unphysical phenomena. To bias the computational stencil in a preferred direction, the second type of schemes uses either directional operators or symmetric operator, but with an artificial viscosity that is not symmetric. Symmetric schemes with artificial viscosity are simple and are well-suited for a canonical finite element formulation, whereas TVD and Godunov-type upwind schemes are more complex than symmetric schemes and trade better resolution for an increase in computational cost and complexity.

Baruzzi, Habashi and Hafez (1989, 1991) had proposed a simple first order artificial viscosity in the form of Laplacians of the pressure and velocity components, added to the continuity and momentum equations, respectively. The amount of artificial viscosity necessary for stable solutions of such first order methods is proportional to the mesh size and its detrimental effects can only be reduced by mesh refinement. However it is impractical to use a uniformly fine mesh throughout the solution domain, and a cost-effective grid should reflect the disparate characteristic lengths of the viscous and inviscid regions. A practical alternative to fine meshes would be either to adopt a higher order artificial viscosity, dynamically adapt the grid to the solution or, preferably, a combination of both.

The existing first-order-accurate finite element solver could be extended to higher order by replacing the Laplacians of the artificial viscosity with fourth order operators, requiring high-order elements, or by introducing an upwind formulation along the lines of the high-resolution TVD or Godunov's schemes reviewed in the previous sections. Both options would have resulted in the formulation of a very complex method. To avoid this, an alternative is to recast the fourth order artificial viscosity as the difference of two second order Laplacian operators. Thus, the Laplacians of the first order scheme can be balanced with correction terms, effectively yielding a fourth order dissipation. This type of scheme has been tested in the context of viscous incompressible flows by Hafez and Soliman (1991), for transonic inviscid flows by Fernandez and Hafez (1991) and for transonic viscous flows by Baruzzi, Habashi and Hafez (1992a, 1992b).

The present work is a natural continuation of previous research work for a Masters degree [Baruzzi (1989)]. It formalizes a common second order approach

framework and addresses the issue of solution acceleration and precision by parallelizing the major time-consuming parts of the finite element code, namely assembly and matrix solution, and developing a mesh adaptation algorithm for accurate and cost-effective capture of shocks and other flow features exhibiting high gradients.

2. THE EULER AND NAVIER-STOKES EQUATIONS

The full Navier-Stokes equations constitute a set of non-linear conservation laws for mass, momentum and energy. These conservation laws provide an accurate mathematical model of the physics of fluid flow. However, due to the considerable variations in the spatial and temporal scales of the gradients of the flow velocity, pressure and density, which may differ by several orders of magnitude from one region to another, their numerical solution requires resources that greatly exceed the capability of even the most advanced supercomputers available now or expected to come into service in the near future.

A more manageable set of equations can be obtained from the full Navier-Stokes equations by filtering out the small-scale temporal variations with a time-averaging process. The modified equations, known as the Reynolds-averaged Navier-Stokes equations, are valid for laminar viscous flows only, but can be adapted to the solution of turbulent flows with the addition of suitable models for the Reynolds stress tensor resulting from the averaging procedure.

For flows at moderate or high Reynolds number the effects of viscosity are confined to thin layers near the solid surfaces in contact with the fluid, while the bulk of the flow can be considered to be virtually inviscid. For a sufficiently smooth flow, the thickness of these layers vanishes in the limit as the Reynolds number approaches infinity, hence the flow in this conditions is convection-dominated and reasonable solutions can be obtained even if the effect of viscosity is completely neglected.

Because of the mathematical and computational complexity associated with the viscous stresses, the initial effort in the development of large-scale numerical simulation techniques for compressible flows was initially directed towards the apparently simpler Euler equations, obtained from the Navier-Stokes equations by neglecting the viscous terms.

For transonic flows the Euler equations form a set of non-linear, strongly-coupled conservation laws, therefore their numerical solution, which may contain discontinuities or singularities whose positions are not known a priori, poses considerable difficulties. Furthermore, by itself the discretization of the governing equations is not sufficient to produce a solution. The formulation of a well-posed problem requires the specification of boundary conditions that are both physically relevant and mathematically valid.

In this chapter the nature of the Euler equations is analyzed in some detail and the outline of a rational approach for the specification of the boundary conditions for this system of equations is provided. The rules obtained from this analysis are subsequently extended heuristically to the Navier-Stokes equations.

2.1. *Mathematical Classification of the Euler Equations*

The Euler equations in the compressible conservative variables form can be written in vector-matrix notation as follows

$$\frac{\partial U}{\partial t}+\frac{\partial E}{\partial x}+\frac{\partial F}{\partial y}=0 \tag{2.1}$$

The vectors U, E and F are defined as

$$U=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho e\end{bmatrix} \qquad E=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(\rho e+p)\end{bmatrix} \qquad F=\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(\rho e+p)\end{bmatrix}$$

An additional relation is required to express the pressure as a function of the other variables. The equation of state, which defines the link among pressure, density, velocity and energy, represents a local scalar property and not a global conservation law. For this reason it appears separately from equation (2.1) and is written in the form

$$p=(\gamma-1)\rho\left(e-\frac{u^2+v^2}{2}\right) \tag{2.2}$$

More information on the nature of Eq. (2.1) can be gathered by recasting the flux vectors E and F in terms of the vector of the conservative variables U. Thus

$$\frac{\partial U}{\partial t}+A\frac{\partial U}{\partial x}+B\frac{\partial U}{\partial y}=0 \tag{2.3}$$

The coefficient matrices $A=\partial E/\partial U$ and $B=\partial F/\partial U$ are the Jacobians of the flux vectors E and F with respect to the conservative variables U. The components of these matrices are shown below.

$$A=\begin{bmatrix} 0 & 1 & 0 & 0 \\ -u^2+\frac{\gamma-1}{2}\left(u^2+v^2\right) & -(\gamma-3)u & -(\gamma-1)v & \gamma-1 \\ -uv & v & u & 0 \\ -u\left[\frac{c^2}{\gamma-1}-\left(\frac{\gamma}{2}-1\right)\left(u^2+v^2\right)\right] & \frac{c^2}{\gamma-1}+\frac{1}{2}\left(u^2+v^2\right)-(\gamma-1)u^2 & -(\gamma-1)uv & \gamma u \end{bmatrix}$$

$$B=\begin{bmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ -v^2+\frac{\gamma-1}{2}\left(u^2+v^2\right) & -(\gamma-1)u & -(\gamma-3)v & \gamma-1 \\ -v\left[\frac{c^2}{\gamma-1}-\left(\frac{\gamma}{2}-1\right)\left(u^2+v^2\right)\right] & -(\gamma-1)uv & \frac{c^2}{\gamma-1}+\frac{1}{2}\left(u^2+v^2\right)-(\gamma-1)v^2 & \gamma v \end{bmatrix}$$

where c is the local speed of sound, obtained from Eq. (2.2) and defined as

$$c^2 = \gamma \frac{p}{\rho} = \gamma(\gamma-1)\left(e - \frac{u^2+v^2}{2}\right) \tag{2.4}$$

To compute the eigenvalues of A and B, which are the characteristic speeds at which information is transmitted from one region of the fluid to another, let

$$(A-\lambda_A I)\tilde{X}_A = 0 \qquad (B-\lambda_B I)\tilde{X}_B = 0 \tag{2.5}$$

where (λ_A, λ_B) are the eigenvalues of A and B, respectively, $(\tilde{X}_A, \tilde{X}_B)$ are the associated left eigenvectors and I is the identity matrix. The eigenvalues of the matrices A and B are

$$\lambda_A = \{u, u, u+c, u-c\} \tag{2.6a}$$

$$\lambda_B = \{v, v, v+c, v-c\} \tag{2.6b}$$

Furthermore, each Jacobian matrix can be rewritten as a function of the diagonal matrix of its eigenvalues, such that

$$A = X_A D_A X_A^{-1} \qquad B = X_B D_B X_B^{-1}$$
$$D_A = \mathrm{diag}(u, u, u+c, u-c) \qquad D_B = \mathrm{diag}(v, v, v+c, v-c)$$

X_A and X_B are the matrices formed by the eigenvectors of A and B, and are shown below.

$$X_A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & u & u+c & u-c \\ 1 & v & v & v \\ v & \frac{u^2+v^2}{2} & \frac{c^2}{\gamma-1}+\frac{u^2+v^2}{2}+uc & \frac{c^2}{\gamma-1}+\frac{u^2+v^2}{2}-uc \end{bmatrix}$$

$$X_B = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & u & u & u \\ 0 & v & v+c & v-c \\ u & \frac{u^2+v^2}{2} & \frac{c^2}{\gamma-1}+\frac{u^2+v^2}{2}+vc & \frac{c^2}{\gamma-1}+\frac{u^2+v^2}{2}-vc \end{bmatrix}$$

Their inverse matrices are

$$X_A^{-1} = \left[\begin{array}{c|c|c|c} -v & 0 & 1 & 0 \\ \hline 1-\frac{\gamma-1}{2}\frac{u^2+v^2}{c^2} & (\gamma-1)\frac{u}{c^2} & (\gamma-1)\frac{v}{c^2} & -\frac{\gamma-1}{c^2} \\ \hline \frac{\gamma-1}{4}\frac{u^2+v^2}{c^2}-\frac{u}{2c} & \frac{1}{2c}-\frac{\gamma-1}{2}\frac{u}{c^2} & -\frac{\gamma-1}{2}\frac{v}{c^2} & \frac{\gamma-1}{2c^2} \\ \hline \frac{\gamma-1}{4}\frac{u^2+v^2}{c^2}+\frac{u}{2c} & -\left(\frac{1}{2c}+\frac{\gamma-1}{2}\frac{u}{c^2}\right) & -\frac{\gamma-1}{2}\frac{v}{c^2} & \frac{\gamma-1}{2c^2} \end{array}\right]$$

$$X_B^{-1} = \left[\begin{array}{c|c|c|c} -u & 1 & 0 & 0 \\ \hline 1-\frac{\gamma-1}{2}\frac{u^2+v^2}{c^2} & (\gamma-1)\frac{u}{c^2} & (\gamma-1)\frac{v}{c^2} & -\frac{\gamma-1}{c^2} \\ \hline \frac{\gamma-1}{4}\frac{u^2+v^2}{c^2}-\frac{v}{2c} & -\frac{\gamma-1}{2}\frac{u}{c^2} & \frac{1}{2c}-\frac{\gamma-1}{2}\frac{v}{c^2} & \frac{\gamma-1}{2c^2} \\ \hline \frac{\gamma-1}{4}\frac{u^2+v^2}{c^2}+\frac{v}{2c} & -\frac{\gamma-1}{2}\frac{u}{c^2} & -\left(\frac{1}{2c}+\frac{\gamma-1}{2}\frac{v}{c^2}\right) & \frac{\gamma-1}{2c^2} \end{array}\right]$$

Since the eigenvalues of A and B are real, the system of PDEs (2.1) or (2.3) is hyperbolic. In addition, since the coefficient matrices A and B are functions of the solution variables, the Euler equations are classified as non-linear hyperbolic wave equations, where the waves are represented by the eigenvectors $\tilde{X}$.

If the fluid is moving, some of the waves produced by the fluid particles are carried by the particles themselves, while others emanate radially outward from the moving particles. Since all vectors in two-dimensions can be represented by two independent components, the eigenvalues are the x- and y-components of the actual speeds at which the waves propagate, namely the velocity of the particle $|\vec{V}| = \sqrt{u^2+v^2}$ and the velocity of the particle plus the additional signal speed $|\vec{V}+\vec{c}|$. Note that, even though the fluid particles are moving in a definite direction, some of the waves that they emit radiate outwards along circular fronts (spherical fronts in 3-D), hence for subsonic flow $|\vec{V}+\vec{c}|$ may have components directed upstream as well as downstream. In general, four

characteristic speeds, or simply characteristics, are identified, namely $(V, V, V+c, V-c)$.

From Eqs. (2.6), if $0<|\vec{V}|<c$ it follows that three characteristics are positive and one is negative. For supersonic flows $0<c<|\vec{V}|$, all the characteristics $(V, V, V+c, V-c)$ are positive, nevertheless one or both of the velocity components (u, v) may actually be subsonic.

Four different cases for the boundary conditions on the state vector $\boldsymbol{U} = \{\rho, \rho u, \rho v, \rho e\}^T$ can be identified:

i) At a subsonic inflow boundary, $0<|\vec{V}|<c$ and according to the signs of the characteristics, three waves are entering the domain and one is exiting. Since part of the information reaching the boundary comes from the interior of the domain, only three boundary conditions for the state vector are required, and these are customarily, but not necessarily, the values of $(\rho, \rho u, \rho v)$. The outgoing wave carries information from inside the domain to the boundary, hence the boundary condition on the fourth component of the state vector is replaced with a value extrapolated from grid points inside the solution domain.

ii) At a supersonic inflow boundary, $0<c<|\vec{V}|$, therefore all four characteristics are positive, four waves are entering the domain and all four values of the state vector are specified.

iii) At a subsonic outflow boundary, $0<|\vec{V}|<c$, hence three characteristics are positive and one is negative. Three waves are exiting the domain and one is entering. Three boundary conditions for the state vector are replaced with formulae to extrapolate the values of $(\rho, \rho u, \rho v)$ from inside the

domain to the boundary, while the last condition, usually the value of (ρs) or (p), is specified.

iv) At a supersonic outflow boundary $0 < c < |\vec{V}|$, all four characteristics are positive, hence all the values of the state vector are extrapolated from the interior of the solution domain to the boundary.

Finally, it will be shown in the following chapters that one of the advantages of using a finite element discretization is that it is unnecessary to explicitly replace some of the boundary conditions with extrapolation formulae.

2.2. *Steady-State Euler Equations — Mixed Formulation*

If the numerical scheme for the solution of the Euler equations is specifically designed to capture only the steady-state solutions, it may be convenient to neglect the time derivatives in Eq. (2.1) and write the Euler equations in the form

$$\frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} = 0 \tag{2.7}$$

Following Anderson *et al.* (1984), Eq. (2.7) can be rewritten as

$$\frac{\partial W}{\partial x} + A\frac{\partial W}{\partial y} = 0 \tag{2.8}$$

where

$$W = \begin{bmatrix} u \\ v \\ p \\ \rho \end{bmatrix} \qquad A = \frac{1}{u^2 - c^2}\begin{bmatrix} uv & -c^2 & -\frac{v}{\rho} & 0 \\ 0 & \frac{v}{u}\left(u^2 - c^2\right) & \frac{u^2 - c^2}{\rho u} & 0 \\ -\rho v c^2 & \rho u c^2 & uv & 0 \\ -\rho v & \rho u & \frac{v}{u} & \frac{v}{u}\left(u^2 - c^2\right) \end{bmatrix}$$

W is the vector of primitive variables and A is obtained from the Jacobians of E and F with respect to the vector W.

The eigenvalues of the coefficient matrix A are

$$\lambda_1 = \frac{v}{u} \tag{2.9a}$$

$$\lambda_2 = \frac{v}{u} \tag{2.9b}$$

$$\lambda_3 = \frac{uv + c\sqrt{u^2 + v^2 - c^2}}{u^2 - c^2} \tag{2.9c}$$

$$\lambda_4 = \frac{uv - c\sqrt{u^2 + v^2 - c^2}}{u^2 - c^2} \tag{2.9d}$$

The first two eigenvalues are real, while the last two eigenvalues are imaginary for subsonic flow, when $0 < V^2 = u^2 + v^2 < c^2$, and real for supersonic flows, when $0 < c^2 \leq V^2 = u^2 + v^2$. Thus this formulation has a mixed nature, elliptic for subsonic flows and hyperbolic for supersonic flows. This mixed nature can create difficulties in the implementation of the numerical solution schemes, limiting them to the solution of supersonic flows. Furthermore, the numerical schemes that could be developed for the solution of this system of equations may not be extended to the time-dependent form of the Euler or Navier-Stokes equations, limiting their usefulness.

2.3. *Steady-State Euler Equations — Hyperbolic Formulation*

In this form, the time derivatives are also discarded and the Euler equations are written in the form of Eq. (2.7). The solution vector can either be based on the conservative variables $U = \{\rho, \rho u, \rho v, \rho e\}^T$ or any other set of primitive variables, for example $Q = \{\rho, u, v, e\}^T$, as for example in Baruzzi, Habashi and Hafez (1989). To reduce the system from four to three equations [Lytton (1987)], it is possible to avoid solving the energy equation by assuming that the total enthalpy is constant everywhere in the solution domain, provided that the following four assumptions are valid:

i) the flow is steady,

ii) the effect of heat-transfer is negligible,

iii) the total enthalpy is constant along the entire boundary,

iv) the fluid is chemically inert.

In this case the energy equation has the form

$$\frac{\partial}{\partial x}[u(\rho e+p)]+\frac{\partial}{\partial y}[v(\rho e+p)]=0 \tag{2.10}$$

The symbol e represents the total energy per unit mass of the fluid, which is the sum of the internal and the kinetic energy, hence the term $(\rho e+p)$ can be recast in the form

$$\rho e+p=\rho\left(e+\frac{p}{\rho}\right)=\rho\left[e_I+\frac{1}{2}\left(u^2+v^2\right)+\frac{p}{\rho}\right] \tag{2.11}$$

where e_I is the internal energy per unit mass and $\left(u^2+v^2\right)/2$ is the kinetic energy. Eq. (2.11) can be rearranged to yield

$$e_I+\frac{p}{\rho}+\frac{1}{2}\left(u^2+v^2\right)=h+\frac{1}{2}\left(u^2+v^2\right)=H \tag{2.12}$$

where H is the total enthalpy per unit mass. The energy equation can be recast in terms of the total enthalpy after substituting Eq. (2.12) into Eq. (2.10). Hence

$$\frac{\partial}{\partial x}(\rho uH)+\frac{\partial}{\partial y}(\rho vH)=0 \tag{2.13}$$

Eq. (2.13) can be simplified even further by using the continuity equation from the system of equations (2.1), to produce the final energy equation

$$\frac{\partial H}{\partial x}+\frac{\partial H}{\partial y}=0 \tag{2.14}$$

and therefore

$$H=\text{const.}$$

provided that the total enthalpy is constant along the boundaries of the solution domain. Since the total enthalpy is constant everywhere if the four assumptions listed above are satisfied, the system of equations (2.1) can be modified as follows

$$\frac{\partial \tilde{E}}{\partial x}+\frac{\partial \tilde{F}}{\partial y}=0 \tag{2.15}$$

where

$$\tilde{E}=\begin{bmatrix} \rho u \\ \rho u^2+p \\ \rho uv \end{bmatrix} \qquad \tilde{F}=\begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2+p \end{bmatrix}$$

The system is closed by introducing the definition of total enthalpy, namely

$$H=\frac{\gamma}{\gamma-1}\frac{p}{\rho}+\frac{1}{2}\left(u^2+v^2\right) \tag{2.16}$$

Finally, the system of equations (2.15) must be classified to ascertain whether it is still hyperbolic. Initially one may be tempted to recast Eq. (2.15) in the form

$$\tilde{A}\frac{\partial \tilde{U}}{\partial x}+\tilde{B}\frac{\partial \tilde{U}}{\partial y}=0 \tag{2.17}$$

where

$$\tilde{U}=\begin{bmatrix} \rho \\ \rho u \\ \rho v \end{bmatrix}$$

$$\tilde{A}=\begin{bmatrix} 0 & 1 & 0 \\ \frac{c^2}{\gamma}-\frac{u^2}{\gamma}+\frac{\gamma-1}{\gamma}v^2 & \frac{\gamma+1}{\gamma}u & -\frac{\gamma-1}{\gamma}v \\ -uv & v & u \end{bmatrix}$$

$$\tilde{B}=\begin{bmatrix} 0 & 0 & 1 \\ -uv & v & u \\ \frac{c^2}{\gamma}+\frac{\gamma-1}{\gamma}u^2-\frac{v^2}{\gamma} & -\frac{\gamma-1}{\gamma}u & \frac{\gamma+1}{\gamma}v \end{bmatrix}$$

The eigenvalues of the coefficient matrices $\tilde{A}$ and $\tilde{B}$ are

$$\tilde{\lambda}_A=\left\{u,\frac{\gamma+1}{2\gamma}u+\frac{1}{2\gamma}\sqrt{(\gamma-1)^2u^2+4\gamma c^2},\frac{\gamma+1}{2\gamma}u-\frac{1}{2\gamma}\sqrt{(\gamma-1)^2u^2+4\gamma c^2}\right\} \tag{2.18a}$$

$$\tilde{\lambda}_B = \left\{ v, \frac{\gamma+1}{2\gamma}v + \frac{1}{2\gamma}\sqrt{(\gamma-1)^2 v^2 + 4\gamma c^2}, \frac{\gamma+1}{2\gamma}v - \frac{1}{2\gamma}\sqrt{(\gamma-1)^2 v^2 + 4\gamma c^2} \right\} \qquad (2.18b)$$

All characteristics are real and the system of equations (2.15) could be classified as a n . inear hyperbolic standing wave system. Note that for $0 < u, v < c$ two characteristics are positive and one is negative for each of the coefficient matrices.

The characteristics (2.18), however, are substantially different from Eqs. (2.6). This is very surprising be.:se if system (2.1) is solved over a domain with a constant value of the total enthalpy specified along the entire boundary, a condition that must satisfy the energy equation identically as shown above, the solution would appear to have a set of characteristics different from Eqs. (2.18). Note, however, that even though in this particular case the assumption of constant tctal enthalpy satisfies the energy equation identically, removing the need to solve the equation explicitly, the energy equation cannot be assumed to have vanished simply because it does not have to be solved. Whereas H is constant everywhere in the solution domain, e is not. To avoid the inconsistency caused by the discrepancy of the value of the characteristics of the two systems, the boundary conditions will be applied according to Eqs. (2.6) rather than according to Eqs. (2.18). Hence for both subsonic and supersonic inflow boundaries the values of (ρ, u, v) are specified, the pressure (p) is specified at subsonic outflow boundaries and no conditions are imposed at supersonic outflow boundaries.

2.4. Reynolds-Averaged Navier-Stokes Equations

The Reynolds-averaged Navier-Stokes Equations in primitive variables and conservative form are

$$\frac{\partial U}{\partial t}+\frac{\partial E}{\partial x}+\frac{\partial F}{\partial y}-\frac{1}{\mathrm{Re}}\left(\frac{\partial G}{\partial x}+\frac{\partial H}{\partial y}\right)=0 \tag{2.19}$$

where

$$U=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho e\end{bmatrix} \qquad E=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(\rho e+p)\end{bmatrix} \qquad F=\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(\rho e+p)\end{bmatrix}$$

$$G=\begin{bmatrix}0\\ \tau_{xx}\\ \tau_{xy}\\ u\tau_{xx}+v\tau_{xy}-q_x\end{bmatrix} \qquad H=\begin{bmatrix}0\\ \tau_{yx}\\ \tau_{yy}\\ u\tau_{yx}+v\tau_{yy}-q_y\end{bmatrix}$$

The components of the stress tensor $\vec{\tau}$ and the heat-transfer vector $\vec{q}$ are

$$\tau_{xx}=\frac{2\mu}{3}\left(2u_x-v_y\right)$$

$$\tau_{yy}=\frac{2\mu}{3}\left(2v_y-u_x\right)$$

$$\tau_{xy}=\tau_{yx}=\mu\left(u_y+v_x\right)$$

$$q_x=-kT_x$$

$$q_y=-kT_y$$

The system of equations is completed by the scalar equation of state

$$p=(\gamma-1)\rho\left(e-\frac{u^2+v^2}{2}\right)$$

and by an additional scalar equation to express the coefficient of molecular viscosity as a function of the thermodynamic variables

$$\mu=\mu(\rho,p,T)$$

For steady, chemically inert flows without heat transfer, the system of equations can be simplified by assuming that the total enthalpy is constant everywhere. However, in this case the assumption does not satisfy the energy equation

identically since the viscous dissipation term is not negligible. Some loss of accuracy with respect to the full system of equations (2.19) should be expected.

There are no set rules for the classification of the Navier-Stokes equations. Even though the characteristics of the Jacobians of the flux vectors are identical to those of Eqs. (2.6), the system (2.19) is not strictly hyperbolic due to the higher order derivatives introduced by the viscous stresses. The thickness of the boundary layers, however, is small for high Reynolds number flows, therefore the convection processes are dominant almost everywhere. Assuming that the effect of viscosity is negligible at the inflow and outflow boundaries, at least for external flows, the boundary conditions outlined in Section 2.1 are applicable.

3. A SECOND ORDER ARTIFICIAL VISCOSITY SCHEME

A simple, first order accurate finite element scheme for the solution of the steady compressible Euler and Navier-Stokes equations will be briefly outlined. Even though this simple scheme can produce good results for inviscid flows, a higher order scheme is required for viscous flows. The objective of this chapter is to show that by introducing a simple modification into the first order scheme, it is possible to extend the accuracy to second order.

3.1. A First Order Artificial Viscosity Scheme

The governing equations (2.19) can be rewritten as

$$\nabla\cdot\left(\rho\vec{V}\right)=0 \tag{3.1a}$$

$$\nabla\cdot\left(\rho\vec{V}\vec{V}\right)+\nabla p-\nabla\cdot\vec{\tau}=0 \tag{3.1b}$$

$$H_\infty=\frac{\gamma}{\gamma-1}\frac{p}{\rho}+\frac{\vec{V}\cdot\vec{V}}{2} \tag{3.1c}$$

$$\frac{\mu}{\mu_\infty}=\left(\frac{T_\infty+110°k}{T+110°k}\right)\left(\frac{T}{T_\infty}\right)^{3/2} \tag{3.1d}$$

$$\nabla\cdot\vec{\tau}=\frac{1}{Re}\left[-\frac{2}{3}\nabla\left(\mu\nabla\cdot\vec{V}\right)+\nabla\times\mu\left(\nabla\times\vec{V}\right)+2(\nabla\cdot\mu\nabla)\vec{V}\right]$$

Eq. (3.1c) is the definition of constant total enthalpy, an approximation valid only for steady flows without heat transfer and viscous energy dissipation. Eq. (3.1d) is Sutherland's law for air. After replacing the density in Eqs. (3.1a,b) by Eq. (3.1c), and lagging the viscosity coefficient during the iterative solution, the system, in two dimensions, reduces to three coupled equations with variables $\{u,v,p\}$.

If a symmetric discretization scheme such as the finite element method were used to discretize Eqs. (3.1) directly, with equal order of interpolation for all variables, the convection terms would cause a numerical instability in the form of even-odd point decoupling. To circumvent this problem, a pressure dissipation term is introduced in the continuity equation to permit the use of equal order interpolation polynomials for pressure and velocity. Thus Eq. (3.1a) becomes

$$\nabla \left(\rho \vec{V}\right) - \varepsilon_1 \nabla^2 p = 0 \tag{3.2a}$$

An artificial dissipation (viscosity), proportional to the Laplacian of the velocity, can be introduced in the momentum equation as follows

$$\nabla \left(\rho \vec{V} \vec{V}\right) + \nabla p - \nabla \cdot \vec{\tau} - \varepsilon_1 \nabla^2 \vec{V} = 0 \tag{3.2b}$$

This first order artificial viscosity scheme successfully met the Pulliam challenge [Pulliam (1990)] for inviscid non-lifting flows over circular cylinders and ellipses, a set of simple test cases whose accurate numerical solutions, to our knowledge, have yet to be obtained by other methods. This approach has also been extended to internal flows by Peeters, Habashi *et al.* (1991, 1992).

Even though good results have been obtained based on Eqs. (3.2), the penalty is that conservation of mass and momentum can only be achieved within an error proportional to the magnitude of the artificial dissipation terms. Furthermore, for viscous flows, first order schemes do not produce sufficiently accurate solutions. The amount of extra diffusion produced by the artificial viscosity adds significantly to boundary layer growth, producing solutions resembling flows with a much lower effective Reynolds number than the one specified. These flaws cannot be avoided completely even with massive grid refinement, therefore schemes with lower dissipation and greater accuracy must be utilized for the simulation of viscous flows.

3.2. *A Second Order Artificial Viscosity Scheme*

The present work addresses the reduction of the artificial dissipation in order to improve the quality of the overall solution to second order accuracy. Global second order accuracy can be achieved by introducing fourth order operators in place of the Laplacians in Eqs. (3.2). Normally these operators would require higher order elements for a correct discretization, however, at the discrete level, they can also be recast as the difference of two Laplacian operators, which require only linear elements. It is thus proposed to balance the Laplacian of pressure in Eq. (3.2a) by the scalar quantity $\nabla\cdot\vec{F}$ and the Laplacians of velocity in Eq. (3.2b) by the vector $\nabla\cdot\vec{G}$, where $\vec{G}$ is a tensor, as follows

$$\nabla\cdot\left(\rho\vec{V}\right)-\varepsilon_1\nabla\cdot\left(\nabla p-\vec{F}\right)=0 \tag{3.3a}$$

$$\nabla\cdot\left(\rho\vec{V}\vec{V}\right)+\nabla p-\nabla\cdot\vec{\tau}-\varepsilon_1\nabla\cdot\left(\nabla\vec{V}-\vec{G}\right)=0 \tag{3.3b}$$

At the discrete level, the terms $\vec{F}$ and $\vec{G}$ must be constructed in a way that guarantees that the differences between the Laplacians and the balancing terms yield a fourth order artificial dissipation. It can be shown that, for the artificial dissipation to be a fourth order operator, the balancing terms $\vec{F}$ and $\vec{G}$ must be the averaged gradients of pressure and velocity, respectively. To illustrate this point, it is more convenient to switch temporarily to the finite difference notation. Starting with a five-point stencil, the Laplacian of p, for example, can be written in terms of the gradient as follows

$$\nabla^2 p=\frac{\nabla p|_{i+1/2}-\nabla p|_{i-1/2}}{\Delta x}=\frac{(p_{i+1}-p_i)-(p_i-p_{i-1})}{(\Delta x)^2}$$

which reduces to the classical formula

$$\nabla^2 p=\frac{p_{i+1}-2p_i+p_{i-1}}{(\Delta x)^2} \tag{3.4}$$

Similarly, for the balancing term

$$\nabla\cdot\vec{F}=\frac{\overline{\nabla p}\big|_{i+1}-\overline{\nabla p}\big|_{i-1}}{2\Delta x}=\frac{\frac{1}{2}\left(\frac{p_{i+2}-p_{i+1}}{\Delta x}+\frac{p_{i+1}-p_i}{\Delta x}\right)-\frac{1}{2}\left(\frac{p_i-p_{i-1}}{\Delta x}+\frac{p_{i-1}-p_{i-2}}{\Delta x}\right)}{2\Delta x}$$

Collecting terms

$$\nabla\cdot\vec{F}=\frac{p_{i+2}-2p_i+p_{i-2}}{(2\Delta x)^2} \tag{3.5}$$

This is also a Laplacian operator, however it encompasses a stencil with five points instead of the three points of Eq. (3.4). The difference of Eq. (3.4) and (3.5) yields

$$\nabla\cdot\left(\nabla p-\vec{F}\right)=\frac{p_{i+1}-2p_i+p_{i-1}}{(\Delta x)^2}-\frac{p_{i+2}-2p_i+p_{i-2}}{(2\Delta x)^2}=-\frac{p_{i+2}-4p_{i+1}+6p_i-4p_{i-1}+p_{i-2}}{(2\Delta x)^2}$$

which is the discrete equivalent of

$$\nabla\cdot\left(\nabla p-\vec{F}\right)=-\frac{1}{4}\nabla^4 p \tag{3.6}$$

The same argument also applies to the tensor $\vec{G}$.

The artificial viscosity terms must also incorporate a function that removes the balancing terms in the neighborhood of a shock, allowing the solution to become locally first-order accurate to avoid spurious oscillations, at the same time preserving the conservative nature of the equations. For these reasons, Eqs. (3.3) are modified as follows

$$\nabla\cdot\left(\rho\vec{V}\right)-\nabla\cdot\left[(\varepsilon_1+\varepsilon_2)\nabla p-\varepsilon_2\vec{F}\right]=0 \tag{3.7a}$$

$$\nabla\cdot\left(\rho\vec{V}\vec{V}\right)+\nabla p-\nabla\cdot\vec{\tau}-\nabla\left[(\varepsilon_1+\varepsilon_2)\nabla\vec{V}-\varepsilon_2\vec{G}\right]=0 \tag{3.7b}$$

The coefficients ε_1 and ε_2 denote the first and second order artificial viscosity coefficients, which are functions of the solution, such that ε_2 tends to zero in the neighborhood of a shock and ε_1 tends to zero away from a shock.

The weak-Galerkin weighted residual forms of Eqs. (3.7) are

Continuity

$$\iint_A W_i \left\{ \left[\rho u - (\varepsilon_1 + \varepsilon_2) p_x + \varepsilon_2 f_1 \right]_x + \left[\rho v - (\varepsilon_1 + \varepsilon_2) p_y + \varepsilon_2 f_2 \right]_y \right\} dA = 0$$

weak form

$$\iint_A \left\{ \left[\rho u - (\varepsilon_1 + \varepsilon_2) p_x + \varepsilon_2 f_1 \right] W_{i_x} + \left[\rho v - (\varepsilon_1 + \varepsilon_2) p_y + \varepsilon_2 f_2 \right] W_{i_y} \right\} dA = \oint_S W_i \left[\rho \vec{V} \cdot \vec{n} - (\varepsilon_1 + \varepsilon_2) p_n + \varepsilon_2 \vec{F} \cdot \vec{n} \right] dS \tag{3.8a}$$

x-momentum

$$\iint_A W_i \left\{ \left[\rho u^2 + p - \frac{2\mu}{3Re}(2u_x - v_y) - (\varepsilon_1 + \varepsilon_2) u_x + \varepsilon_2 g_{11} \right]_x + \left[\rho uv - \frac{\mu}{Re}(u_y + v_x) - (\varepsilon_1 + \varepsilon_2) u_y + \varepsilon_2 g_{12} \right]_y \right\} dA = 0$$

weak form

$$\iint_A \left\{ \left[\rho u^2 + p - \frac{2\mu}{3Re}(2u_x - v_y) - (\varepsilon_1 + \varepsilon_2) u_x + \varepsilon_2 g_{11} \right] W_{i_x} + \left[\rho uv - \frac{\mu}{Re}(u_y + v_x) - (\varepsilon_1 + \varepsilon_2) u_y + \varepsilon_2 g_{12} \right] W_{i_y} \right\} dA = \oint_S W_i p \, dy + \oint_S W_i \left[\rho u \vec{V} \cdot \vec{n} - (\varepsilon_1 + \varepsilon_2) u_n + \varepsilon_2 \vec{G} \cdot \vec{n} \right] dS - \oint_S W_i \left[\frac{2\mu}{3Re}(2u_x - v_y) \frac{dy}{dS} - \frac{\mu}{Re}(u_y + v_x) \frac{dx}{dS} \right] dS \tag{3.8b}$$

y-momentum

$$\iint_A W_i \left\{ \left[\rho uv - \frac{\mu}{Re}(u_y + v_x) - (\varepsilon_1 + \varepsilon_2) v_x + \varepsilon_2 g_{21} \right]_x + \left[\rho v^2 + p - \frac{2\mu}{3Re}(2v_y - u_x) - (\varepsilon_1 + \varepsilon_2) v_y + \varepsilon_2 g_{22} \right]_y \right\} dA = 0$$

weak form

$$\begin{aligned}
&\iint_A \left\{ \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right) - \left(\varepsilon_1 + \varepsilon_2\right)v_x + \varepsilon_2 g_{21} \right] W_{i_x} + \right. \\
&\left. \left[\rho v^2 + p - \frac{2\mu}{3Re}\left(2v_y - u_x\right) - \left(\varepsilon_1 + \varepsilon_2\right)v_y + \varepsilon_2 g_{22} \right] W_{i_y} \right\} dA = \\
&-\oint_S W_i p\, dx + \oint_S W_i \left[\rho v \vec{V} \cdot \vec{n} - \left(\varepsilon_1 + \varepsilon_2\right)v_n + \varepsilon_2 \vec{G} \cdot \vec{n} \right] dS - \\
&\oint_S W_i \left[\frac{\mu}{Re}\left(u_y + v_x\right)\frac{dy}{dS} - \frac{2\mu}{3R^e}\left(2v_y - u_x\right)\frac{dx}{dS} \right] dS
\end{aligned} \tag{3.8c}$$

with W_i denoting the weight functions. At the discrete level $\{f_1, f_2, g_{11}, g_{12}, g_{21}, g_{22}\}$ are nodal variables and would be represented in the canonical finite element framework as

$$\left\{\vec{F}, \vec{G}\right\} = \sum_{k=1}^{4} \left\{\vec{F}_k, \vec{G}_k\right\} N_k \tag{3.9}$$

where $\left\{\vec{F}_k, \vec{G}_k\right\}$ are the nodal values of $\left\{\vec{F}, \vec{G}\right\}$ and N_k denotes the finite element shape functions. The functions ε_1 and ε_2 are

$$\varepsilon_1 = \left(\left|\vec{V}\right| + c\right) \cdot \psi \qquad\qquad \varepsilon_2 = \left(\left|\vec{V}\right| + c\right) \cdot \phi \tag{3.10}$$

where

$$\psi = \kappa^{(2)} \varepsilon \left(\max_k T_j^k \right) \tag{3.11}$$

$$\phi = \max\left(0, \kappa^{(4)} \varepsilon - \psi\right) \tag{3.12}$$

and

$$T_j^k = \frac{\left| \max_k p_j^k + \min_k p_j^k - 2p_j \right|}{\max_k p_j^k + \min_k p_j^k + 2p_j} \tag{3.13}$$

is an operator for the detection of the shock. The notation $(\cdot)_j^k$ refers to all the neighbors k of node j. The artificial viscosity is controlled by the parameter ε and the values of the coefficients $\kappa^{(2)}$ and $\kappa^{(4)}$ are similar to those of Schmidt and Jameson (1985) and Pulliam (1985), namely $\kappa^{(2)} = 1/4$ and $\kappa^{(4)} = 0.009$.

After replacing the density by Eq. (3.1c), the system of equations (3.8) is linearized by Newton's method, discretized with bilinear quadrilateral isoparametric elements and integrated by Gauss-Legendre quadrature with (3X3) Gauss points. At each iteration, a fully-coupled, sparse linear algebraic system of the form:

$$[J]\{\Delta U\} = -\alpha\{Res\} \tag{3.14}$$

is solved with a direct solver. Here $\Delta U = \{\Delta u, \Delta v, \Delta p\}$, $\{Res\}$ is the residual and α is a relaxation factor. The details of Eq.(3.14) can be found in the Appendix.

In this deferred correction approach for the introduction of the second order artificial viscosity, the added balancing terms $\{f_1, f_2, g_{11}, g_{12}, g_{21}, g_{22}\}$ are functions of the solution at the previous iteration, hence the Jacobian matrix $[J]$ is unaltered with respect to the corresponding first order approach. The residual, however, is of higher order accuracy.

3.3. *Boundary Conditions*

The following boundary conditions are imposed with all contour integrals of Eqs. (3.8) evaluated, unless otherwise noted:

Inflow: u, v, p are specified.

Walls: For viscous flows $u, v = 0$. For inviscid flows, the term $\rho\vec{V}\cdot\vec{n}$ in the continuity equation contour integral vanishes.

Outflow: For a subsonic outflow, p is specified in the contour integrals of the momentum equations. For a supersonic outflow, no back pressure is specified.

For all other boundary conditions, the practice has been to replace the governing equations by extrapolation formulae, locally one-dimensional equations or Riemann invariants. However, since the finite element shape functions are smooth continuous polynomials over the element, this situation can be avoided altogether. Hence, once the influence matrices of the elements on the boundary are assembled into a global one, no further treatment is required for quantities such as pressure, velocities, etc. The advantage is therefore that the discrete governing equations are not replaced by simpler approximations at the boundary.

4. EXPLICIT SECOND ORDER BALANCING TERMS

In the previous chapter it was shown that a second order finite element scheme which does not require higher order elements can be produced by adding two balancing terms, $\vec{F}$ and $\vec{G}$, to the artificial viscosity of a first order scheme. Three different methods to compute the balancing terms will be outlined in this chapter. In the first method, the balancing terms are obtained from tensor identities and nodal averaging, while in the second scheme the tensor identities are discarded and the balancing terms are obtained directly through the smoothing provided by nodal averaging. In these two schemes the balancing terms are functions of the solution at the previous iteration and cannot be linearized due to the intermediate procedure required for their evaluation. To overcome the linearization problem a third scheme has been developed which dispenses with the intermediate Galerkin averaging step and considerably improves the efficiency of the overall second order finite element solver.

4.1. Method 1: Tensor Identities plus Galerkin Averaging

The term $\vec{F}$ of Eq. (3.3a) is obtained by rearranging the original momentum equations (3.1b) and solving for the pressure gradient

$$\vec{F} = f_1\vec{i} + f_2\vec{j} = \nabla p = -\nabla\left(\rho\vec{V}\vec{V}\right) + \nabla\cdot\vec{\tau} \tag{4.1}$$

where

$$f_1 = -\left\{\left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right)\right]_x + \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]_y\right\} \tag{4.2a}$$

$$f_2 = -\left\{\left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]_x + \left[\rho v^2 - \frac{2\mu}{3Re}\left(2v_y - u_x\right)\right]_y\right\} \tag{4.2b}$$

The artificial term of Eq. (3.3a), $\nabla^2 p - \nabla \cdot \vec{F}$, does not vanish identically, even in the steady-state when the discrete momentum equations are satisfied, since the divergence of the momentum equations is not necessarily zero in a discrete sense. Nevertheless, conservation of mass is improved at least an order of magnitude over the original method represented by Eq. (3.2a).

Similarly, the momentum equations are modified by adding artificial viscosity terms proportional to the Laplacian of the velocity components, with these terms balanced by a tensor obtained from the following identity:

$$\nabla^2 \vec{V} - \nabla\left(\nabla \ \vec{V}\right) + \nabla \times \left(\nabla \times \vec{V}\right) = 0 \tag{4.3}$$

$$\nabla^2 \vec{V} - \nabla s + \nabla \times \omega = 0 \tag{4.4}$$

The tensor $\bar{G}$ therefore becomes:

$$g_{11} = s \qquad g_{12} = -\omega \tag{4.5a}$$

$$g_{21} = \omega \qquad g_{22} = s \tag{4.5b}$$

The correction terms $f_1, f_2, g_{11}, g_{12}, g_{21}, g_{22}$ are evaluated at the nodes in accordance with the standard weak-Galerkin framework. For f_1, for example, the Galerkin integral is:

$$\iint_A W_i \left\{ f_1 + \left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right) \right]_x + \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right) \right]_y \right\} dA = 0 \tag{4.6}$$

Let

$$f_1 = \sum_{j=1}^{4} \overline{f_1}_j N_j \tag{4.7}$$

where N_j represents the finite element shape functions and $\overline{f_1}_j$ denotes the value of f_1 at the four nodes of an element. After integrating Eq. (4.6) by parts and introducing Eq. (4.7)

$$\sum_{j=1}^{4}\overline{f_{1j}}\iint_A W_i N_j dA = \iint_A \left\{\left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right)\right]W_{i_x} + \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]W_{i_y}\right\}dA - \oint_S W_i\left\{\left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right)\right]\frac{dy}{dS} - \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]\frac{dx}{dS}\right\}dS \tag{4.8}$$

$$[M]\{\overline{f_{1j}}\} = \iint_A \left\{\left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right)\right]W_{i_x} + \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]W_{i_y}\right\}dA - \oint_S W_i\left\{\left[\rho u^2 - \frac{2\mu}{3Re}\left(2u_x - v_y\right)\right]\frac{dy}{dS} - \left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]\frac{dx}{dS}\right\}dS \tag{4.9}$$

where $[M]$ is the mass matrix and the integral on the right-hand-side is a function of the solution at the previous iteration.

The other balancing terms can be written as

$$f_2 = -\left\{\left[\rho uv - \frac{\mu}{Re}\left(u_y + v_x\right)\right]_x + \left[\rho v^2 - \frac{2\mu}{3Re}\left(2v_y - v_x\right)\right]_x\right\} \tag{4.10}$$

$$g_{11} = -\nabla\cdot\vec{V} \qquad g_{12} = \nabla\times\vec{V} \tag{4.11a}$$

$$g_{21} = -\nabla\times\vec{V} \qquad g_{22} = -\nabla\cdot\vec{V} \tag{4.11b}$$

where $g_{22} = g_{11}$, $g_{21} = -g_{12}$ and

$$\iint_A W_i\left(g_{11} + \nabla\cdot\vec{V}\right)dA = 0 \qquad \iint_A W_i\left(g_{12} - \nabla\times\vec{V}\right)dA = 0 \tag{4.12}$$

In summary, for this scheme, four new variables, $\{f_1, f_2, g_{11}, g_{12}\}$, need to be defined at the nodes as functions of the variables $\{u,v,p\}$ at the previous iteration.

4.2. Method 2: Galerkin Averaging

In this scheme, the Laplacians of the artificial viscosity are balanced in a slightly different manner

$$\nabla\cdot\left(\rho\vec{V}\right) - \varepsilon_1\nabla\cdot\left(\nabla p - \nabla\hat{p}\right) = 0 \tag{4.13a}$$

$$\nabla\cdot\left(\rho\vec{V}\vec{V}\right) + \nabla p - \nabla\cdot\vec{\tau} - \varepsilon_1\nabla\cdot\left(\nabla\vec{V} - \nabla\vec{G}\right) = 0 \tag{4.13b}$$

The vector $\nabla\hat{p}$ and the Cartesian tensor $\nabla\vec{G}$ are constructed as follows:

$$\nabla\hat{p} = f_1\vec{i} + f_2\vec{j} = p_x\vec{i} + p_y\vec{j} \tag{4.14}$$

$$\nabla\vec{G} = \vec{G}_x + \vec{G}_y = (g_{11} + g_{21})\vec{i} + (g_{12} + g_{22})\vec{j} = (\vec{u}_x + \vec{v}_x)\vec{i} + (\vec{u}_y + \vec{v}_y)\vec{j} \tag{4.15}$$

where $\{f_1, f_2, g_{11}, g_{12}, g_{21}, g_{22}\}$ are again evaluated within the weak-Galerkin framework, outlined in Eqs. (4.6), (4.8) and (4.9), such that

$$\iint_A W_i(f_1 - p_x)dA = 0 \tag{4.16a}$$

$$\iint_A W_i(f_2 - p_y)dA = 0 \tag{4.16b}$$

$$\iint_A W_i(g_{11} - u_x)dA = 0 \tag{4.16c}$$

$$\iint_A W_i(g_{12} - u_y)dA = 0 \tag{4.16d}$$

$$\iint_A W_i(g_{21} - v_x)dA = 0 \tag{4.16e}$$

$$\iint_A W_i(g_{22} - v_y)dA = 0 \tag{4.16f}$$

This is a more general scheme, allowing a balanced artificial viscosity even for the full energy equation, should it be included. In this approach, the number of extra variables to be computed increases from four to six. However, one should note that the mass matrix in both schemes is identical for all the additional variables and needs to be assembled and decomposed only once, in a post-processing step, during each global Newton iteration. Therefore the increase in the number of variables of Method 2 represents a minor penalty in terms of computational cost.

Four different methods can be used to compute the nodal values of the additional variables: the mass matrix of Eq. (4.9) (CMM), the lumped mass

matrix (LMM), and, as an alternative to the Galerkin averaging, extrapolation from the Gauss points to the nodes (EXT) through a 9th order polynomial [Hinton *et al.* (1975)] and, finally, least-squares fitting (LSQ) involving sampling the derivatives of the variables at all the Gauss points of the elements surrounding each node [Langtangen (1989)].

Note that the solutions obtained with the second order schemes are much less sensitive to the value of the artificial viscosity coefficient than those obtained with the first order formulation. The convergence rate of the second order schemes, however, is directly controlled by the value of the artificial viscosity coefficient since the balancing terms are lagged from the previous iteration and do not contribute to the Jacobian of the Newton linearization. A partial remedy to this situation is presented in the following section.

4.3. *Method 3: Semi-Implicit Second Order Balancing Terms*

The various schemes presented above for the construction of the second order balancing terms have one common feature: the balancing terms are obtained from an averaging procedure. Indeed some of the schemes are based on a sophisticated but expensive weighted-averaging procedure, the Galerkin integral. The complexity of this procedure requires both the solution of a matrix and that the balancing terms be lagged at the previous iteration, with a detrimental effect on both the convergence rate and the solution time. Since the solution of Eqs. (3.8) — the linear system of coupled equations of the Navier-Stokes FEM scheme — can be an expensive undertaking when a direct solver is used, a fully-implicit scheme with quadratic convergence would be desirable to minimize solution cost. A fully-implicit second order scheme, however, requires a wider finite element stencil and would produce a system of equations with

twice the current bandwidth, for which direct solvers would prove too expensive and unwieldy. Hence, a compromise must be sought.

The Galerkin weighted averaging can be replaced by a simpler averaging procedure which requires neither integration nor the solution of an intermediate system of equations. This scheme will be outlined through the construction of the balancing term f_1, with the procedure being identical for the other five balancing terms introduced in Method 2.

The definition of f_1 is:

$$f_1 = p_x \tag{4.17}$$

Introducing the finite element discretization into Eq. (4.17), the following equation is obtained

$$f_1 = \sum_{j=1}^{4} \bar{p}_j \frac{\partial N_j}{\partial x} \tag{4.18}$$

where f_1 is intended to be a nodal value but p_x is defined only inside the element in the discrete approximation. The nodal value of f_1 is obtained by an area averaging of the values of the derivatives computed at the centroids of the elements surrounding each node, such that

$$f_1 = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \bar{p}_j \frac{\partial N_j}{\partial x}\bigg|_c \right)_l \tag{4.19}$$

where $\bar{p}_j$ is the value of pressure at the nodes of the element and

$$A_{total} = \sum_{l=1}^{M} A_l \tag{4.20}$$

is the sum of the areas of the element surrounding each node, M is the number of elements surrounding a node and the subscript c indicates the centroid of the element. Similarly, the other balancing terms are

$$f_2 = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \overline{v}_j \frac{\partial N_j}{\partial y}\bigg|_c \right)_l \tag{4.21a}$$

$$g_{11} = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \overline{u}_j \frac{\partial N_j}{\partial x}\bigg|_c \right)_l \tag{4.21b}$$

$$g_{12} = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \overline{u}_j \frac{\partial N_j}{\partial y}\bigg|_c \right)_l \tag{4.21c}$$

$$g_{21} = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \overline{v}_j \frac{\partial N_j}{\partial x}\bigg|_c \right)_l \tag{4.21d}$$

$$g_{22} = \frac{1}{A_{total}} \sum_{l=1}^{M} \left(A_l \sum_{j=1}^{4} \overline{v}_j \frac{\partial N_j}{\partial y}\bigg|_c \right)_l \tag{4.21e}$$

The expressions for $\{f_1, f_2, g_{11}, g_{12}, g_{21}, g_{22}\}$ in Eqs. (4.19) and (4.21) are substituted directly into Eqs. (3.8) and are eventually discretized like nodal-based quantities via the finite element approximation. The areas of the elements and the derivatives of the shape functions are constant for stationary grids, therefore, since the balancing terms can be recast in terms of the dependent variables, the intermediate Galerkin averaging procedure is eliminated and the balancing terms can be linearized with Newton's method.

Since a full linearization would double the bandwidth, compared to the schemes where the balancing terms are lagged, a partial linearization is suggested. The balancing terms contributions to the right-hand side of Eqs. (3.8) must be included in their entirety. Some elements have balancing terms stiffness matrices which can only be partially assembled into the Jacobian matrix without expanding the bandwidth. It was found that if these stiffness matrices where partially assembled, the Jacobian matrix would loose its topological symmetry and the iterative procedure would diverge. Therefore only the balancing terms

stiffness matrices which can be fully assembled should be included into the Jacobian matrix to ensure stability. This is possible because the unknowns in Eqs. (3.8) are the changes in the solution and the Jacobian matrix can be viewed as a preconditioner for the system of equations. This procedure is stable, conserves the topological symmetry and, by including more information into the Jacobian matrix, produces better convergence rates.

5. PARALLELIZATION STRATEGIES

In the last decade, ever since the introduction of cheap DRAM memory units capable of storing several Megabytes of data, most computer manufacturers have switched from CISC (Complex Instruction Set Computer) to RISC (Reduced Instruction Set Computer) architectures. Before these memory units became available, the scarcity of on-board memory dictated that computers be able to recognize complex instructions so that program size could be kept as small as possible to maximize the amount of the limited data space available to the user [Dowd (1993)].

In order to permit a wide variety of operations, the set of complex instructions tends to be very large and requires a dedicated processing unit on the CPU for each of these instructions. As the amount of on-board memory grows, however, a smaller set of simpler instructions makes more efficient use of space on the CPU and also lends itself to optimization, since the instructions that once required special processing units are now composed of a string of commands that can be executed in assembly-line-style in a limited number of dedicated processing units arranged in pipelines. Most modern RISC architectures offer several pipelines to handle the flow of commands and data with considerable efficiency and speed and can also handle conditional branches more efficiently than CISC architectures.

While CPU clock speeds are constantly increasing as computers evolve, the access speed of the memory systems has not kept pace. The reason for this discrepancy is economical rather than technical. The huge cost of modern supercomputers is due in large measure to the very fast and very expensive

memory systems installed. The installation of such expensive memory systems in desktop workstation is out of the question, therefore computer manufacturers provide slower but more economical memory units. A small amount of very fast memory is built into each CPU to handle the local computational tasks more efficiently. A moderate amount of fast memory, called the cache, also resides in a separate buffer between the CPU and the main memory. The cache acts as an intermediary, processing memory requests received in advance from the CPU, storing and retrieving chunks of data and instructions from main memory and passing them to the CPU as they are needed. Provided that the data and instructions are organized sequentially in main memory, this arrangement can be very effective.

Some of the techniques used in the computational sciences are easily amenable to parallelization, hence some computer manufacturers offer lines of parallel-processors computers that are almost as fast as multi-million dollar supercomputers were just a few years ago, but cost a fraction of the price. These parallel computers are progressively differentiating into two categories: coarse- and fine-grained. Coarse-grained parallel computers are generally endowed with a small number of very fast processors, generally from two to twenty-four, while fine-grained parallel computers have thousands of cheaper and slower processors linked by a sophisticated communication system. The memory units are either pooled into a single shared-memory system or reside in small amounts on each processor to form a distributed-memory system.

The computational aspects of the techniques developed in the previous sections will now be examined in more detail to show how they can be efficiently implemented on modern coarse-grained parallel workstations to take full advantage of pipelining and cached memory systems. In actual practice the

entire finite element solver boils down to a sequence of matrix assemblies and factorizations. All other operations are secondary — as far as solution times are concerned — and would actually make the computation inefficient if they were executed in parallel.

5.1. Parallel Matrix Factorization

The finite element discretization shown above leads to large sparse linear systems of equations which are not well conditioned. Two solution strategies are available: direct and iterative solvers. An efficient direct solver has been developed for the current two-dimensional applications for which speed, rather than memory reduction, is the main requirement. For three-dimensional problems, where the memory requirements of direct solvers would be prohibitive, conjugate gradient-like iterative solvers are a more practical alternative [Dutto, Habashi et al. (1993)].

Among the various direct solver techniques, the Gauss elimination algorithm was chosen for its robustness and ability to handle matrices that are not diagonally dominant, even without pivoting. The Gauss elimination algorithm is divided into two processes: the matrix factorization and the back substitution, with the former being the most expensive and the one that benefits most from parallelization.

The matrix is stored with a constant bandwidth and its indices are reversed with respect to the customary matrix notation, so that it resides in the computer memory with all the rows arranged in sequential order, in the same way in which it will be accessed during row-elimination. The factorization step, being the most time-consuming, receives particular attention: IF statements are eliminated by storing selected critical parameters into three integer vectors of

length N, the number of unknowns. The first vector maps the number of rows scheduled for decomposition below each diagonal entry, the second maps the address of the end of each row and the third is used during the back substitution to map the number of the unknowns already solved that affect the calculation of the current unknown. The three vectors are assembled before the global iterative process and need not be evaluated repeatedly if the matrix size remains constant.

The factorization step is represented schematically by three nested loops: the first loop sweeps all the entries along the diagonal of the matrix, the middle loop sweeps all the entries in each column under the diagonal and finally the inner loop handles the factorization of all the entries in the rows. Either the inner or the middle loop can be parallelized, however the parallelization of the inner loop requires frequent synchronization and will yield modest speedups of the order of $\sqrt{n}$, where n is the number of processors, while the parallelization of the middle loop generates much less synchronization overhead and easily reaches 100% efficiency. The two parallelization strategies are shown below in schematic form. The C$DOACROSS machine-dependent command is actually added by the parallel pre-compiler, but is shown to illustrate the differences in parallelization.

Example 1: Inefficient parallelization

```
      DO 10 I=1,N-1
         DO 15 II=I+1,NO. OF ROWS TO ELIMINATE
            RATIO=ELIMINATION COEFFICIENT

C$DOACROSS SHARE( I, II, RATIO, A), LOCAL(J)
            DO 20 J=DIAGONAL,LAST ENTRY IN PIVOT ROW
               A(J,II)=A(J,II)-(ENTRY IN PIVOT ROW)*RATIO
 25            CONTINUE

            RHS(II)=RHS(II)-RATIO*RHS(I)
 15         CONTINUE
 10      CONTINUE
```

Example 2: Efficient parallelization

```
      DO 10 I=1,N-1

C$DOACROSS SHARE(I, RHS, A), LOCAL(II, RATIO, J)
         DO 15 II=I+1,NO. OF ROWS TO ELIMINATE
            RATIO=ELIMINATION COEFFICIENT
            DO 20 J=DIAGONAL,LAST ENTRY IN PIVOT ROW
               A(J,II)=A(J,II)-(ENTRY IN PIVOT ROW)*RATIO
25             CONTINUE
            RHS(II)=RHS(II)-RATIO*RHS(I)

15          CONTINUE
10       CONTINUE
```

In the second example, an efficient factorization is constructed so that each processor handles the multiplication and addition of a complete row. If the bandwidth is kept constant, a strict synchronization of the processors is no longer necessary as all of them handle an equal amount of work, with little idle time at the completion of the row operation spent waiting for the other processors to complete their task.

An additional gain can be obtained by unrolling the inner row factorization loop. In this case the processors memory caches handle the I/O of data in a more efficient manner, reducing the overhead required for fetching and storing data by increasing the size of the data blocks processed. The Parallel FORTRAN Accelerator software on Silicon Graphics computers can automatically unroll loops at the user's request, however since this operation is beneficial on virtually any modern RISC workstation, it has been incorporated explicitly in the matrix factorization routine. Furthermore, on vector architectures the inner loop could also be vectorized, with dramatic gains in execution speed. The details of loop unrolling are shown in the next example.

Example 3: Efficient parallelization with loop unrolling

```
      DO 10 I=1,N-1

C$DOACROSS SHARE(I, RHS, A), LOCAL(II, RATIO, J)
        DO 15 II=I+1,NO. OF ROWS TO ELIMINATE
           RATIO=ELIMINATION COEFFICIENT
           DO 20 J=DIAGONAL,LAST ENTRY IN PIVOT ROW,STRIDE
              A(J,II)=A(J,II)-(ENTRY IN PIVOT ROW)*RATIO
              A(J+1,II)=A(J+1,II)-(ENTRY IN PIVOT ROW)*RATIO
              A(J+2,II)=A(J+2,II)-(ENTRY IN PIVOT ROW)*RATIO
                                  . . .
              A(J+S,II)=A(J+S,II)-(ENTRY IN PIVOT ROW)*RATIO
25            CONTINUE
           RHS(II)=RHS(II)-RATIO*RHS(I)

15         CONTINUE
10      CONTINUE
```

The solver is written in FORTRAN, contains no machine-dependent instructions and parallelizes automatically on Silicon Graphics parallel workstations with the Power FORTRAN Accelerator software. Fig. 1 shows the performance gains obtainable with this solver on an SGI Power Series 280 GTX shared-memory computer with eight 25 MHz MIPS R3000 processors. The problem considered is transonic flow over a NACA0012 airfoil at M_∞=0.8 and 1.25° angle of attack, discretized using a 254X30 C-grid. The matrix size is (23,637X371) and the solution requires 98 seconds with eight processors. Fig. 2 shows the performance of the same solver on a Silicon Graphics Challenge workstation equipped with twenty 50 MHz R4400 processors. The problem considered is M_∞=2 supersonic flow in a channel with a 4% circular arc airfoil using a (80X200) grid. The matrix size is (47,920X1,209) and the solution requires 295 seconds with 20 processors.

5.2. Parallel Assembly of the Global Matrix

Two processes dominate the execution time during each Newton iteration: the assembly and the factorization of the global matrix. With the factorization

parallelized as shown above, the global matrix assembly becomes the dominant process and it should also be parallelized in the interest of overall execution performance.

The global matrix is the sum of all the element stiffness matrices and is normally constructed in a sequential element-by-element order. Two strategies for the parallelization of the assembly can be adopted. The first strategy is based on the parallelization of the assembly of each stiffness matrix, with all the stiffness matrices added sequentially into the global matrix, while the second strategy requires the subdivision of the solution domain into n blocks of elements of roughly equal size, each block of elements being assigned to one of the n processors.

This second strategy is more efficient because it involves a minimum of synchronization overhead, but requires some changes in the FORTRAN code The operation is accomplished by nesting a call to a subroutine which assembles a block of elements in a DO-loop that traverses all the element blocks. Because the subroutine CALL statement is inside the loop, this DO-loop will not parallelize automatically, since the parallel pre-processor does not check inside the routine to identify possible conflicts between processors.

The matrix assembly DO-loop must be parallelized manually by inserting a simple C$DOACROSS directive which instructs the processors to classify all variables as either SHAREd or LOCAL. The loop index and loop limits must be explicitly declared as local variables since they differ from processor to processor. All the element stiffness matrices are declared as local variables by virtue of being defined inside the routine, but the global matrix and all the ancillary variables needed for the assembly are shared by all the processors.

A schematic arrangement of the parallel assembly process, complete with the machine-dependent instructions, is shown in the next example.

Example 4: Parallel matrix assembly

```
C$DOACROSS
C$&   SHARE( XLONG    , YLONG    , ZMU      , VISCL    , UVAR     ,
C$&          DNXGLBL , DNYGLBL , DETERJ   , SHOKPTO , DNXVISC ,
                                    ...
C$&          KOUNT    , NVCYCLE , KMASSTR , VMASSTR , NPOINTR ,
C$&   LOCAL( NTHREAD , NELBGN   , NELEND )

      DO 10 NTHREAD=1,NAPROC
         NELBGN=NPOINTR(NTHREAD,1)
         NELEND=NPOINTR(NTHREAD,2)
         CALL GLOBMXV( XLONG    , YLONG    , ZMU      , VISCL    ,
             UVAR     , DNXGLBL , DNYGLBL , DETERJ   , SHOKPTO ,
                                    ...
             NGAUSB   , NBOUND   , NELEMB   , KOUNT    , NVCYCLE ,
             KMASSTR , VMASSTR )
  10        CONTINUE
```

The variable NAPROC identifies the number of parallel processors, NELBGN and NELEND are the element numbers at the beginning and end of each block and GLOBMXV is the routine that performs the stiffness and global matrix assembly in each block. The rest of the variables are needed for the assembly and are included only to show that such complex constructions do not hamper the parallelization process. The machine-dependent instructions that have been included do not destroy the portability of the code since they are be identified as comments by the FORTRAN compilers on other computers.

The assembly can be parallelized without summation conflicts provided that the stiffness matrices at the interfaces of neighboring blocks are not assembled and stored in the global matrix simultaneously. If the elements are not numbered sequentially, or if the geometry of the domain is complicated, some checking of the element assembly sequence on the interfaces of the blocks is required to

ensure that potential storage conflicts are eliminated. More complicated methods of block subdivision based on checkerboarding or coloring the elements can also be applied, if necessary.

6. GRID ADAPTING BASED ON A SPRING ANALOGY

A numerical scheme for the solution of fluid problems in an industrial production environment must be both efficient and accurate. This is a contradiction in terms, since for a discrete solution to approach the exact solution the mesh has to be progressively refined, to the great detriment of efficiency.

Ideally, a grid should be constructed in such a way that the solution obtained from it has a uniformly distributed numerical error that meets or exceeds the level of accuracy required. A uniformly fine grid would seem to be a crude but effective way of ensuring high accuracy, however this approach has two serious flaws. First, even the most powerful computers available today cannot handle a uniformly fine mesh for the solution of high Reynolds number flows over a simple geometry. Secondly, a uniformly fine grid still does not guarantee that the error is distributed equally everywhere, hence the features of some regions, free of important details, will be captured with an accuracy that is several orders of magnitude better than required, at the expense of solution speed and economy.

In numerical computations, given that the speed and storage capacity of the available computer are the limiting factors, computational efficiency constrains the finest mesh size that is practical and therefore the level of accuracy that can be attained. Once an estimate of the size of the mesh necessary to obtain a specified level of accuracy is established, the crucial problem is to determine how the elements should be distributed. Unfortunately the solution is never known a priori, hence the construction of the grid customarily proceeds according to empirical rules providing stretching and deformations to fit the geometry and the expected solution.

These difficulties can be circumvented by using the solution obtained on an initial arbitrary mesh to change the grid distribution and minimize some norm of the numerical error. Ideally the next solution, obtained on the adapted grid, will have a constant numerical error throughout, but in practice several cycles of this solution-adaptation strategy must be completed before the error becomes uniform everywhere. Grid adapting is therefore based on two main ingredients: error estimates and mechanisms to distribute the elements according to the error estimates. There exists a considerable variety of methods for both tasks, however in this chapter the emphasis is on illustrating how the nodes of a structured mesh can be redistributed with a simple mechanism based on a spring analogy.

6.1. Node Movement Generated by Spring Forces

The present grid adapting method traces its origins to the idea of Gnoffo (1983) of using a spring analogy to adapt the grid to the gradients of selected functions of the solution. A sophisticated application of the spring analogy, with control over mesh size and orthogonality, was introduced by Nakahashi and Deiwert (1986, 1987) for 2-D and 3-D meshes. Subsequent improvements were reported by Harvey *et al.* (1991,1992) who applied the method to supersonic 2-D and 3-D viscous flows. A variant of this method developed by Baruzzi (1993) will be outlined in this chapter.

Structured grids based on quadrilateral elements can be generated by two families of lines, roughly perpendicular to one another. Each grid line is assumed to consist of a chain of springs connected in series, the length of each spring representing the length of the corresponding element edge, as shown in Fig. 3. Each spring has a different stiffness coefficient, which is proportional to some

local estimate of the error. The springs are free to move along the grid line, subject to the following constraints:

i) the stiffness coefficient is constant within each spring,

ii) the two ends of the chain of springs are stationary,

iii) the shape of a grid line is not deformed as the springs move along it,

iv) the geometry of the boundary must be rigorously conserved.

In addition, it is desirable to retain full control over the minimum and maximum spring lengths, $\Delta x_{\min}$ and $\Delta x_{\max}$.

The stiffness coefficient is a function of the local solution error. Since the solution error is proportional to the truncation error and hence to the gradients of the solution, a crude but effective approach is to obtain the stiffness coefficients from the gradients of the solution. For inviscid compressible flows the gradient of the Mach number or the pressure is a good choice for the stiffness coefficients. For viscous flows the gradients of pressure or density are preferable.

Since the force stored in the stretched springs is constant along the chain, we can cut the chain at any node and examine the force exerted on the node. From Hooke's law:

$$F = k_e \Delta x_e = \text{Const.} \tag{6.1}$$

where k_e is the stiffness coefficient and Δx_e is the length of the spring. The stiffness coefficient can assume any arbitrary value. However, some constraints must be imposed on it to ensure that the smallest springs will cluster in the regions of highest gradients. Some control over the size of the smallest and

largest springs must be imposed to prevent excessive stretching of the grid. The coefficient k_e can be written as

$$k_e = 1 + A\bar{f}_e^B \tag{6.2}$$

where A and B are constants and $\bar{f}$ is the non-dimensional gradient of the solution, computed as follows

$$\bar{f}_e = \frac{f_e - f_{\min}}{f_{\max} - f_{\min}} \tag{6.3}$$

such that

$$f_e = f_{\min} \rightarrow \bar{f}_e = 0 \qquad\qquad f_e = f_{\max} \rightarrow \bar{f}_e = 1$$

where f is the gradient of the function that drives the system of springs. The relationship between the shortest and longest springs can be determined from Eq. (6.1) as

$$k_{\max}\Delta x_{\min} = k_{\min}\Delta x_{\max}$$

which can be rearranged in the form

$$\frac{k_{\max}}{k_{\min}} = \frac{\Delta x_{\max}}{\Delta x_{\min}}$$

Consider Eq. (6.2): if the driving function has a constant value everywhere, Δx_e must be constant throughout, therefore the lower limit of the constant A must be zero. Define A as

$$A = \frac{\Delta x_{\max}}{\Delta x_{\min}} - 1 = \frac{k_{\max}}{k_{\min}} - 1 \tag{6.4}$$

The lower and upper values of k_e can now be determined. From Eqs. (6.2) and (6.4)

$$\bar{f} = 0 \rightarrow k_{\min} = 1 \qquad\qquad \bar{f} = 1 \rightarrow k_{\max} = 1 + A$$

The constant of Eq. (6.1) can be determined from the first expression as

$$\text{Const.} = k_e\Delta x_e = k_{\min}\Delta x_{\max} = \Delta x_{\max} \tag{6.5}$$

and subsequently all the spring lengths Δx_e can be expressed as functions of the maximum value of the mesh spacing in the equation

$$\Delta x_e = \frac{\Delta x_{\max}}{k_e} \tag{6.6}$$

The value of the exponent B in Eq. (6.2) is computed as follows. Let

$$\sum_{e=1}^{M} \Delta x_e = S \tag{6.7}$$

where S is the length of the mesh line, which must not vary while the grid is adapted. Substituting Eq. (6.6) into Eq. (6.7) yields

$$\Delta x_{\max} \sum_{e=1}^{M} \frac{1}{k_e} = S \tag{6.8}$$

Eq. (6.8) can be rewritten as

$$\sum_{e=1}^{M} \frac{1}{k_e} - \frac{S}{\Delta x_{\max}} = 0 \tag{6.9}$$

An expression in terms of B can be obtained by substituting Eq. (6.2) into Eq. (6.9)

$$F(B) = \sum_{e=1}^{M} \frac{1}{1 + A\bar{f}_e^{B}} - \frac{S}{\Delta x_{\max}} = 0 \tag{6.10}$$

The value of B can be computed iteratively using Newton's method. Let

$$F'(3) = \frac{dF(B)}{dB} = -\sum_{e=1}^{M} \frac{A\bar{f}_e^{B} \ln(\bar{f}_e)}{\left(1 + A\bar{f}_e^{B}\right)^2} \tag{6.11}$$

The value of the exponent B is updated using Eqs. (6.10) and (6.11) as follows

$$B^{n+1} = B^n - \frac{F(B)}{F'(B)} \tag{6.12}$$

The initial value of B for the examples shown in Chapter 8 is 2.

An expression for the spring coefficients has been derived which will force grid clustering at the location of the steepest gradients, at the same time maintaining

control over the minimum and maximum mesh sizes. An iterative scheme based on Eq. (6.6) can be derived to evaluate the new positions of the nodes, however the system of springs is undamped and therefore such a scheme would be unconditionally unstable. A damping term could be added, however the damping coefficient would negatively affect the convergence rate of the adapting scheme.

6.2. *Variational Approach*

A different approach that does not require the addition of an additional damping term can be constructed by considering Hooke's law in differential equation form

$$k_e \frac{\partial x}{\partial s} = \text{Const.} \tag{6.13}$$

where s is the coordinate of the fixed frame of reference along the original grid line and x is the new location of the nodal points along this line. The constant can be eliminated by differentiating with respect to s

$$\frac{\partial}{\partial s}\left(k_e \frac{\partial x}{\partial s}\right) = 0 \tag{6.14}$$

From a variational principle standpoint [Brackbill and Saltzman (1982)], the solution of this second order equation corresponds to the minimization of the integral

$$I = \frac{1}{2}\int_0^S k_e \left(\frac{\partial x}{\partial s}\right)^2 ds \tag{6.15}$$

Note that the expression inside the integral represents the energy of a spring, so that minimizing the integral is equivalent to finding the minimum energy state of a system of springs along a given line. The positions of the nodes of the springs can be described by a finite element interpolation. Let

$$x = \sum_{j=1}^{N} \hat{x}_j N_j$$

where the N_j are linear finite element shape functions, $\hat{x}_j$ denotes the nodal positions and the elements are the segments of the original grid line. Substituting this expression into Eq. (6.15) and omitting the summation symbol

$$I = \frac{1}{2}\int_0^S k_e \left[\frac{\partial}{\partial s}\left(\hat{x}_j N_j\right)\right]^2 ds \tag{6.16}$$

To minimize the integral I, set the variation

$$\delta I = \frac{\partial I}{\partial \hat{x}_j}\delta \hat{x}_j = 0$$

which is satisfied either by $\partial I/\partial \hat{x}_j = 0$ or by $\delta \hat{x}_j = 0$. In the first case

$$\frac{\partial I}{\partial \hat{x}_j} = \hat{x}_j \int_0^S k_e \frac{\partial N_i}{\partial s}\frac{\partial N_j}{\partial s} ds = 0 \tag{6.17}$$

In the second case Eq. (6.17) is replaced by the boundary conditions $x = 0$ at $s = 0$ and $x = S$ at $s = S$. For simple linear one-dimensional finite element shape functions, the integral in Eq. (6.17) is evaluated exactly.

Eq. (6.17) is nonlinear in k_e. Even though the springs can move along a grid line, the driving function remains fixed to the old grid. As the grid adapts, the values of the spring coefficients must be continually updated. To improve convergence to the final stationary position, it is useful to apply Newton's method to Eq. (6.17). The linearized equation can be written as

$$\Delta \hat{x}_j \int_0^S k_e^n \frac{\partial N_i}{\partial s}\frac{\partial N_j}{\partial s} ds = -\int_0^S k_e^n \frac{\partial N_i}{\partial s}\frac{\partial x^n}{\partial s} ds \tag{6.18}$$

The value of k_e^n is lagged at the previous iteration, therefore convergence will only be linear. Eq. (6.18) yields a tridiagonal system of algebraic linear equations of the form

$$[A]\{\Delta x\} = -\alpha\{\mathrm{Res}\}$$

that is solved with the Thomas algorithm. Note that here Δx represents the change in nodal position, rather than the spring length. The solution can be

underrelaxed to overcome any instability created by the small radius of convergence of Newton's method. The nodal positions are updated by setting

$$\hat{x}_j^{n+1} = \hat{x}_j^n + \Delta\hat{x}_j \tag{6.19}$$

The iterative procedure consisting of the solution of Eq. (6.18), followed by Eq. (6.19), is repeated until the RMS norm of the residual drops below a pre-set tolerance, usually 1×10^{-7}.

6.3. 2-D Adaptive Scheme

A general structured grid in two dimensions can be subdivided into two families of lines. The adaptation procedure consists in relocating the nodes of the grid, one line at a time, starting from a selected surface and moving to the final one along the same family of lines, in a parabolic sweep.

It is also desirable to include orthogonality and smoothness control to ensure that the final adapted grid is not excessively skewed. When the body surface is adapted, all the discontinuities of the slope of the surface must be identified before moving the nodes to ensure that they will be modeled properly by the adapted grid.

After the surface nodes have been rearranged, the next grid lines of the same family are progressively adapted, one by one. Starting with the second grid line, an additional set of torsion springs, centered at the nodes of the previous grid line, is added to control orthogonality. Similarly, starting with the third grid line, another set of springs is added for smoothness control. The effect of these two additional spring sets, however, makes control over the minimum and maximum mesh sizes less accurate. Fig. 4 shows the location of the orthogonal projection point H and the smoothness projection point S on the line being adapted.

Fig. 5 shows how a torsion spring to enforce orthogonality can be replaced by a conventional spring. The orthogonality and smoothness constraints are introduced into the adaptation procedure by adding a new term to Eq. (6.16)

$$I = \frac{1}{2}\int_0^S k_e \left[\frac{\partial}{\partial s}\left(\hat{x}_j N_j\right)\right]^2 ds + \frac{C}{2}\left(\hat{x}_j - \hat{x}_{HSj}\right)^2 \tag{6.20}$$

where $\hat{x}_{HSj} = \beta x_S + (1-\beta)x_H$ is the location of $\hat{x}_j$ that would maximize a blend of orthogonality and smoothness controlled by the parameter β and C is the orthogonality/smoothness spring coefficient obtained by averaging all the spring coefficients along the line. The parameter C is given by

$$C = \frac{\lambda}{\overline{AB}}\left(\frac{1}{M}\sum_{e=1}^{M} K_e\right) \tag{6.21}$$

where M is the number of springs along the line, $\overline{AB}$ is the distance between consecutive lines in the same family, as shown in Fig. 5, and λ is the parameter that controls the magnitude of the spring coefficient C. For the tests reported in Chapter 7, the values of the coefficients are $\beta = 0.1$, $\lambda = 3 \times 10^{-4}$.

7. RESULTS

The results of several test-cases are shown in this section. The tests, which are well documented in the literature, have been selected for their difficulty, with the intent to show the robustness of the present second order schemes.

7.1. Second Order Accuracy Test

A test case with an exact solution is proposed to verify the accuracy of the three second order schemes. The Navier-Stokes equations are solved on a square domain, with a uniform mesh spacing, satisfying the following exact solution:

$$\rho = 2\sin(x)\sin(y) \qquad u = \sin(x)\cos(y) \qquad v = -\cos(x)\sin(y)$$

$$0.5 \le x \le 0.7 \qquad 0.5 \le y \le 0.7$$

The original continuity equation, Eq. (1), is satisfied identically and a forcing function for each of the momentum equations can be derived from the exact solution. Five progressively refined grids have been used: (16X16), (24X24), (32X32), (48X48) and (64X64) elements. The Reynolds number is set at 1,000. An error norm defined as:

$$Error = \iint_A |U_{Exact} - U_{Numerical}| dA$$

is calculated for density, velocities and pressure. From Figs. 6–9 it appears that the (ρ,u,v,p) errors of Scheme 1, for both the lumped and consistent matrices, do not tend to zero as the mesh is refined, suggesting that only Schemes 2 and 3 are truly second order accurate, irrespective of the method used to compute the balancing terms.

7.2. Inviscid Supersonic Flow

The second test case is for supersonic inviscid flow over a 15° wedge, at M_∞=2. The solution procedure is started with the first-order scheme and uniform artificial viscosity coefficients. The artificial viscosity parameters are initially set to $\varepsilon_1 = 0.05, \varepsilon_2 = 0$ uniformly throughout and ε_1 is lowered to 0.01 and 0.005 at the intermediate RMS residual value of 10^{-6}. The first grid adapting is performed and the code is switched to second order accuracy with the balancing terms of Scheme 3 and artificial viscosity parameter $\varepsilon = 0.2$. After the intermediate residual is reached a second adaptation is carried out and the solution is finally allowed to converge to the final residual of 10^{-8} with $\varepsilon = 0.1$. The relaxation factor in Eq. (3.14) is set to $\alpha = 0.5$ to stabilize the second order solution in the presence of strong normal and oblique shocks.

The (120X48) grid was initially nearly uniform in the streamwise direction and during the solution procedure it was adapted using the mesh adapting strategy based on a spring analogy outlined in Chapter 6. The initial, intermediate and final grids are shown in Fig. 10. The Mach number contours obtained from the three grids are shown in Fig. 11. The final grid was obtained by specifying minimum and maximum mesh spacings of 3.5% and 150% of the average mesh size along each grid line, respectively. Fig. 12 is the convergence curve which shows the effects of the five cycles of artificial viscosity and mesh refinement. Solution times are of the order of 30 seconds per iteration for Scheme 3 on a Silicon Graphics Challenge computer, with four 150 MHz R-4400 processors.

7.3. Inviscid Transonic Flow Around an Airfoil

The third test is the AGARD01 case [Pulliam and Barton (1985)], namely that of transonic inviscid flow over a NACA0012 airfoil, at M_∞=0.8 and 1.25° angle of

attack. This test is very demanding because a strong shock, a very weak shock and a slip line should appear simultaneously in the solution. However, due to their excessive diffusion, most first order schemes are unable to capture the very weak shock and the slip line.

A (200X32) O-grid is used, with 200 elements on the airfoil surface (Fig. 13). For this test case, the solution is started with the second order dissipation but with the viscosity coefficient set to a large value in order to stabilize the Newton method. This high value is then reduced in 4 successive steps, ε=0.25, 0.15, 0.10 and 0.05, converging at each value to an intermediate residual of $5x10^{-6}$, with the last step carried to 10^{-12}. The relaxation factor in Eq. (3.14) is set to $\alpha = 0.5$ to stabilize the second order solution in the presence of the strong shock. The three discontinuities are clearly captured, as shown in the Mach number and pressure contours of Figs. 14 and 15. The Mach number distribution on the surface is compared in Fig. 16 to the results of Pulliam and Barton (1985) obtained on a (561X65) C-grid and to those of Jameson [Viviand (1985)] on a (320X64) O-grid.

Note that in spite of the finer grids, the results of Pulliam and Barton and those of Jameson show shocks captured across several grid points. The convergence of the solution is shown in Fig. 17. The saw-tooth appearance of the convergence history is due to the four artificial viscosity cycles. Solution times are of the order of 33 seconds per iteration on a Silicon Graphics Challenge computer with eight 150 MHz R-4400 processors, achieving speeds of 20 Mflops per processor.

7.4. Viscous Transonic Flow Around an Airfoil

The fourth case is for transonic viscous flow over a NACA0012 airfoil, at M_∞=0.9, Re=5,000 and 0° angle of attack. A detail of the (200X48) C-grid, with 120 elements on the surface, is shown in Fig. 18. The solution was started by

marching in Reynolds number from 1,000 to 5,000 with uniform first order dissipation and $\varepsilon_1 = 0.005, \varepsilon_2 = 0$. Fig. 19 shows the convergence curve, with machine accuracy reached in 10 Newton iterations. The fully converged first order scheme was then used as input for the second order schemes 1, 2 and 3.

To illustrate the difference between the first and second order accuracy of the finite element schemes, consider the following: the scale factor of the natural viscosity terms of Eq. (2b) is 1/Re, while the scale factor of the artificial viscosity terms is ε_1 The overall scale factor of the combined viscosities is $1/Re'=(1/Re)+\varepsilon_1=0.0052$ for this particular test case, yielding an effective Reynolds number, Re', of 192.3. The second order accurate scheme at this Reynolds number and M_∞=0.9 produces results that are almost identical to those of the first order scheme at Re=5,000, as can be seen by comparing the Mach number contours on the upper and lower halves of Figs. 20. The uniform artificial viscosity of the first order scheme has therefore effectively lowered the Reynolds number by a factor of 26! Note also that above Re=200 the first order artificial viscosity dominates over the natural viscosity, hence the value of $\varepsilon_1$must be kept lower than the inverse of the Reynolds number. The only way to achieve this objective is to refine the grid progressively as the Reynolds number increases, with the solution cost quickly becoming prohibitive.

Fig. 21 shows the Mach number contours for M_∞=0.9 and Re=5,000 obtained with Scheme 3. The artificial viscosity parameters were set at $\varepsilon_1 = 0, \varepsilon_2 = 0.001$ throughout the solution domain. The relaxation factor in Eq. (3.14) is set to $\alpha = 0.9$ to stabilize the second order solution. The improvement in the results achieved with the second order scheme is noticeable. The surface Cp distribution obtained with the first and all the second order schemes is compared in Fig. 22 to the results of the finite volume scheme of Hollanders and Ravalason (1986). Note

again the good agreement between the Cp distribution of the first order scheme at Re=5,000 and the second order scheme at Re=192.3. Fig. 23 is a comparison of the convergence rates of Scheme 1, 2 and 3 using the various methods to compute the balancing terms. Scheme 1 and 2 with balancing terms evaluated via a consistent mass matrix (CMM) have the slowest convergence rates. Convergence is improved by switching to other methods of evaluating the second order balancing terms. Scheme 3 has the best convergence rate for this transonic test case. Solution times where of the order of 136 seconds per iteration on a Silicon Graphics Challenge computer with four 150 MHz R-4400 processors.

6. CONCLUSIONS

This Thesis has presented some developments in terms of finite element methods for the solution of the compressible Euler and Navier-Stokes equations. By itself a straightforward discretization of these governing equations is not sufficient to produce a solution: the formulation of a well-posed problem requires the specification of boundary conditions that are both physically relevant and mathematically valid and a discretization method that is accurate and stable.

An analysis of the nature of the Euler equations and a consistent methodology for the specification of their boundary conditions, formulated according to the theory of characteristics, was outlined in Chapter 2. The rules obtained from this analysis were also adopted for the Navier-Stokes formulation.

In Chapter 3, a first order accurate finite element scheme for the solution of the steady compressible Euler and Navier-Stokes equations is briefly outlined. Even though this simple scheme can produce good results for inviscid flows, a higher order scheme is required for viscous flow simulations. A second order scheme is constructed by balancing the Laplacians of pressure and velocity of the first order scheme with additional terms obtained from an averaging procedure.

In Chapter 4, three methods are presented to compute the balancing terms with vector and tensor terms obtained from the averaged gradients of the pressure and velocity. In the first scheme, the balancing terms are obtained from tensor identities and nodal averaging. In the second scheme, the balancing terms are obtained directly through the smoothing provided by nodal averaging, with four different methodologies for the evaluations of the averaged gradients. In the first

two schemes the balancing terms are functions of the solution at the previous iteration and cannot be linearized due to the intermediate procedure required for their evaluation. To overcome this limitation, a third scheme was developed to dispense with the intermediate Galerkin averaging step. A partial linearization of the balancing terms considerably improves the convergence rate without affecting the bandwidth of the linear system of equations.

Chapter 5 addresses the issues of computational speed and efficiency. A simple and effective strategy for the parallelization of the finite element global assembly procedure is presented. The finite element scheme is linearized by Newton's method and a highly efficient parallel banded solver is developed for the fully-coupled linear system of equations. The solver demonstrates a parallel efficiency of nearly 100% on coarse-grained superpipelined parallel architectures such as the Silicon Graphics R8000 family of advanced workstations.

To overcome the fact that the grids used in CFD applications are generated heuristically, a grid adaptation procedure is developed and proves to be a powerful tool for mesh re-distribution to smooth solution error estimates. The adaptation procedure is based on a spring analogy, allowing control of the maximum and minimum mesh spacing as well as grid smoothness and orthogonality. The advantages of grid adaptation were clearly shown in the case of supersonic flow in a channel. The small Mach stem at the top channel wall did not appear in the first solution with a uniform grid, but was clearly captured after the mesh was twice adapted.

In Chapter 7, the second order accuracy of the scheme was formally demonstrated through a test for which an exact solution exists. Other test cases were used to assess the performance of the second order solver for more

complicated geometries. The viscous test case showed clearly that physically meaningful solutions can only be obtained with schemes that are at least second order accurate.

Current work is concentrating on extending and improving the techniques developed in FENSAP (Finite Element Navier-Stokes Analysis Package), a code based on this work. The extension to 3-D of the second order scheme, with grid adaptation and iterative solvers, is perhaps the most important current project and has been initiated at the Concordia CFD Lab. It will lead to a M.A.Sc. Thesis in 1995 by Mr. Martin Aubé. While a 3-D approach based on direct solvers cannot be considered because of the prohibitive size of the matrices that would be generated, preconditioned iterative solvers are continuously improved, sometimes combined with acceleration techniques such as multigrid, at the CFD Lab [Dutto (1993)].

The development of a flexible and reliable library of turbulence models is underway at the CFD Lab and will lead to a M.A.Sc. Thesis by Mr. Guillaume Houzeaux in 1995. The first order scheme has already been extended to the simulation of hypersonic reacting flows in the current Ph.D. work of Mr. Ait Ali Yahia et al. (1994)].

REFERENCES

Ait Ali Yahia, D., Habashi, W.G. and Baruzzi, G.S. (1994) A Finite Element Method for Hypersonic Reacting Flow, *Advances in Finite Element Analysis in Fluid Dynamics*, ASME.

Anderson, D.A., Tannehill, J.C. and Pletcher, R.H. (1984) *Computational Fluid Mechanics and Heat Transfer*, Hemisphere, New York.

Baruzzi, G.S. (1989) Finite Element Solutions of the Euler Equations in Primitive Variables Form, Master Thesis, Department of Mechanical Engineering, Concordia University, Montreal, Canada.

Baruzzi, G.S., Habashi, W.G. and Hafez, M.M. (1989) Non-Unique Solutions of the Euler Equations, *Advances in Fluid Dynamics*, eds. W.F. Ballhaus, Jr. and Y.M. Hussaini, Springer-Verlag, New York, 1–10.

Baruzzi, G.S., Habashi, W.G. and Hafez, M.M. (1991) Finite Element Solutions of the Euler Equations for Transonic External Flows, *AIAA Journal*, **29**, 11, 1886–1893.

Baruzzi, G.S., Habashi, W.G. and Hafez, M.M. (1992a) A Second Order Method for the Finite Element Solution of the Euler and Navier-Stokes Equations, *Proceedings of the 13th International Conference on Numerical Methods in Fluid Dynamics*, eds. M. Napolitano and F. Sabetta, Springer-Verlag, Rome, Italy, 509–513.

Baruzzi, G.S., Habashi, W.G. and Hafez, M.M. (1992b) An Improved Finite Element Method for the Solution of the Compressible Euler and Navier-Stokes

Equations, *Proceedings of the First European Computational Fluid Dynamics Conference* –Volume 2 (ECCOMAS), eds. Ch. Hirsch, J. Périaux, E. Oñate, Elsevier, Brussels, Belgium, 643–650.

Baruzzi, G.S. (1993) Structured Mesh Grid Adapting Based on a Spring Analogy, *Proceedings of the CFD'93 Conference*, CERCA (Centre for Research on Computation and its Applications), Montreal, 425–436.

Beam, R.M. and Warming, R.F. (1976) An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law Form, *Journal of Computational Physics*, **22**, 87–110.

Bestek, H., Thumm, A. and Fasel, H. (1992) Direct Numerical Simulation of Three-Dimensional Breakdown to Turbulence in Compressible Boundary Layers, *Proceedings of the 13th International Conference on Numerical Methods in Fluid Dynamics*, eds. M. Napolitano and F. Sabetta, Springer-Verlag, Rome, Italy, 145–149.

Bey, K.S. and Oden, J.T. (1991) A Runge-Kutta Discontinuous Finite Element Method for High Speed Flows, *AIAA Paper* 91-1575.

Boivin, S. and Fortin, M. (1993) A New Artificial Viscosity Method for Compressible Viscous Flow Simulations by FEM, *International Journal of Computational Fluid Dynamics*, **1**, 25–41.

Boris, J.P. and Book, D.L. (1973) Flux-Corrected Transport. I. SHASTA, A Fluid Transport Algorithm That Works, *Journal of Computational Physics*, **11**, 38–69.

Brackbill, J.U. and Saltzman, J.S. (1982) Adaptive Zoning for Singular Problems in Two Dimensions, *Journal of Computational Physics*, **46**, 342–368.

Cambier, L., and Escande, B. (1990) Calculation of a Three-Dimensional Shock Wave-Turbulent Boundary-Layer Interaction, *AIAA Journal*, **28**, 11, 1901–1908.

Cochran, R.J. (1992) Laminar and Turbulent Incompressible Fluid Flow Analysis with Heat Transfer by the Finite Element Method, Ph.D. Thesis, Department of Mechanical Engineering, University of Washington.

Dowd, K. (1993) *High Performance Computing*, O'Reilly and Associates, New York.

Dutto, L.C., Habashi, W.G., Fortin, M. and Robichaud, M.P. (1993) Parallelizable Block-Diagonal Preconditioners for 3D Viscous Compressible Flow Calculations, *AIAA Paper* 93-3309, *Proceedings of the 11th AIAA Computational Fluid Dynamics Conference*– Volume 2, Orlando, Florida, 135–143.

Fernandez, G. and Hafez, M.M. (1991) Finite Element Simulation of Compressible Flow with Shocks, *AIAA Paper* 91-1551.

Fletcher, C.A.J. (1979) A Primitive Variable Finite Element Formulation for Inviscid, Compressible Flow, *Journal of Computational Physics*, **33**, 301–312.

Fortin, M. and Fortin, A. (1985) Newer and Newer Elements for Incompressible Flows, *Finite Elements in Fluids*–Volume 6, eds. R.H. Gallager, G.F. Carey, J.T. Oden and O.C. Zienkiewicz, John Wiley & Sons, New York, 171–187.

Gnoffo, P.A. (1983) A Finite-Volume Adaptive Grid Algorithm Applied to Planetary Entry Flow Fields, *AIAA Journal*, **21**, 9, 1249–1254.

Hafez, M.M. and Soliman, M. (1991) Numerical Solution of the Incompressible Navier-Stokes Equations in Primitive Variables on Unstructured Grids, *AIAA Paper* 91-1561.

Haroutounian, V. and Engelman, M.S. (1991) On Modeling Wall-Bound Turbulent Flows Using Specialized Near-Wall Finite Elements and the Standard k-ε Turbulence Model, *Advances in Numerical Simulation of Turbulent Flows*, ASME, Vol. 117, 97–105.

Harvey, A.D., Acharya, S., Lawrence, S.L. and Cheung, S. (1991) Solution-Adaptive Grid Procedure for High-Speed Parabolic Flow Solvers, *AIAA Journal*, **29**, 8, 1232–1240.

Harvey, A.D., Acharya, S., Lawrence, S.L. and Cheung, S. (1992) Solution-Adaptive Grid Procedure for the Parabolized Navier-Stokes Equations, *AIAA Journal*, **30**, 4, 953–962.

Harten, A. (1983) High Resolution Schemes for Hyperbolic Conservation Laws, *Journal of Computational Physics*, **49**, 357–393.

Harten, A., Engquist, B., Osher, S. and Chakravarthy, S. (1987) Uniformly High Order Accurate Essentially Non-Oscillatory Schemes III, *Journal of Computational Physics*, **71**, 231–303.

Hassan, O., Morgan, K. and Peraire, J. (1990) An Implicit Finite Element Method for High Speed Flows, *AIAA Paper* 90-0402.

Hinton, E., Scott, F.C. and Ricketts, R.E. (1975) Local Least Squares Stress Smoothing for Parabolic Isoparametric Elements, *International Journal for Numerical Methods in Engineering*, **9**, 235–239.

Hoffmann, K.A. and Chiang, S.T. (1993) *Computational Fluid Dynamics for Engineers–Volume I*, Engineering Education Systems, Wichita, Kansas.

Hollanders, H. and Ravalason, W. (1986) Résolution des Équations de Navier-Stokes en Fluide Compressible par Méthode Implicite, *La Recherche Aérospatiale*, **1**, 23–46.

Hood, P. and Taylor, C. (1974) Navier-Stokes Equations Using Mixed Interpolation, *Finite Element Methods in Flow Problems*, eds. J.T. Oden, O.C. Zienkiewicz, R.H. Gallager and C. Taylor, UAH Press, Huntsville Alabama.

Hughes, T.J.R. and Brooks, A. (1982) A Theoretical Framework for Petrov-Galerkin Methods with Discontinuous Weight Functions: Application to the Streamline Upwind Procedure, *Finite Element in Fluids*–Volume 4, eds. R.H. Gallagher *et al.*, John Wiley & Sons, New York, 47–65.

Jaeger, M. and Dhatt, G. (1992) An extended k-ε Finite Element Model, *International Journal for Numerical Methods in Fluids*, **14**, 1325–1345.

Jameson, A. (1985) Euler Solvers as an Analysis Tool for Aircraft Aerodynamics, *Advances in Computational Transonics*, ed. W.G. Habashi, Pineridge Press, U.K., 371–404.

Jameson, A. (1993) Artificial Diffusion, Upwind Biasing, Limiters and Their Effect on Accuracy and Multigrid Convergence in Transonic and Hypersonic Flows, *AIAA Paper* 93-3359.

Jiang, B.-N. and Carey, G.F. (1990) Least-Squares Finite Element Method for Compressible Euler Equations, *International Journal for Numerical Methods in Fluids*, **10**, 557–568.

Langtangen, H.P. (1989) A Method for Smoothing Derivatives of Multilinear Finite Element Fields, *Communications in Applied Numerical Methods*, **5**, 275–281.

Lefebvre, D., Peraire, J. and Morgan, K. (1993) Finite Element Least Squares Solution of the Euler Equations Using Linear and Quadratic Approximations, *International Journal of Computational Fluid Dynamics*, **1**, 1–23.

Lerat, A. (1985) Implicit Methods of Second-Order Accuracy for the Euler Equations, *AIAA Journal*, **23**, 1, 33–40.

Lesieur, M., Comte, P. and Metais, O. (1992) Direct Numerical Simulation of Turbulence, *Proceedings of the First European Computational Fluid Dynamics Conference* (ECCOMAS), eds. Ch. Hirsch, J. Périaux, E. Oñate, Elsevier, New York, 37–42.

Löhner, R, Morgan, K. and Zienkiewicz, O.C. (1984) The Solution of Non-Linear Hyperbolic Equation Systems by the Finite element Method, *International Journal for Numerical Methods in Fluids*, **4**, 1043–1063.

Lyra, P.R.M., Morgan, K., Peraire, J., and Peiró, J. (1994) TVD Algorithms for the Solution of the Compressible Euler Equations on Unstructured Meshes, *International Journal for Numerical Methods in Fluids*, **19**, 827–827.

Lytton, C.C. (1987) Solution of the Euler Equations for Transonic Flow over a Lifting Aerofoil—The Bernoulli Formulation, *Journal of Computational Physics*, **73**, 395–431.

van Leer, B. (1979) Towards the Ultimate Conservative Difference Scheme. V. A Second-Order Sequel to Godunov's Method, *Journal of Computational Physics*, 32, 101–136.

van Leer, B., Thomas, J., Roe, P.L. and Newsome, R. (1987) A Comparison of Numerical Flux Formulas for the Euler and Navier-Stokes Equations, *AIAA Paper* 87-1104.

Manouzi, H. and Fortin, M. (1991) A Treatment of Wall Boundaries for Turbulent Flows by the Use of a Transmission Finite Element Method, *International Journal for Numerical Methods in Fluids*, **31**, 113–126.

Masson, C., Saabas, H.J. and Baliga, B.R., (1994) Co-Located Equal-Order Control-Volume Finite Element Method for Two-Dimensional Axisymmetric Incompressible Fluid Flow, *International Journal for Numerical Methods in Fluids*, **18**, 1–26.

Mavriplis, D.J. (1987) Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes, Ph.D. Thesis, Department of Mechanical Engineering, Princeton University, Princeton, New Jersey.

Mavriplis, D.J. and Jameson, A. (1990) Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes, *AIAA Journal*, **28**, 8, 1415-1425.

Miner, E.W., Swean, T.F. Jr., Handler, R.A. and Leighton, R.I. (1991) Examination of Wall Damping for the k–ε Turbulence Model Using Direct Simulations of Turbulent Channel Flow, *International Journal for Numerical Methods in Fluids*, **12**, 609–624.

Miyakawa, J., Takanashi, S., Fujii, K. and Amano, K. (1987), Searching the Horizon of Navier-Stokes Simulation of Transonic Aircraft, *AIAA Paper* 87-0524.

Nakahashi, K. and Deiwert, G.S. (1986) Three-Dimensional Adaptive Grid Method, *AIAA Journal*, **24**, 6, 948–954.

Nakahashi, K. and Deiwert, G.S. (1987) Self-Adaptive Grid Method with Application to Airfoil Flow, *AIAA Journal*, **25**, 4, 513–520.

Nicolaides, R.A. (1993) The Covolume Approach to Computing Incompressible Flows, *Incompressible Computational Fluid Dynamics. Trends and Advances*, eds. M. Gunzburger and R.A. Nicolaides, Cambridge University Press, 295–333.

Osher, S. and Chakravarthy, S. (1983) Upwind Schemes and Boundary Conditions with Applications to Euler Equations in General Geometries, *Journal of Computational Physics*, **50**, 447–481.

Peeters, M.F., Habashi, W.G. and Nguyen, B.Q. (1991) Finite Element Solution of the Incompressible Navier-Stokes Equations by a Helmholtz Velocity Decomposition, *International Journal for Numerical Methods in Fluids*, **13**, 2, 135–144.

Peeters, M.F., Habashi, W.G., Nguyen, B.Q. and Kotiuga, P.L. (1992) Finite Element Solutions of the Navier-Stokes Equations for Compressible Internal Flows, *AIAA Journal of Propulsion and Power*, **8**, 1, 192–198.

Pulliam, T.H. and Steger, J.L. (1980) Implicit Finite-Difference Simulation of Three-Dimensional Compressible Flow, *AIAA Journal*, **18**, 2, 159–167.

Pulliam, T.H. (1985) Implicit Finite-Difference Methods for the Euler Equations, *Advances in Computational Transonics*, ed. W.G. Habashi, Pineridge Press, U.K., 503–542.

Pulliam, T.H. and Barton, J.T. (1985) Euler Computations of AGARD Working Group 07 Airfoil Test Cases, *AIAA Paper* 85-0018.

Pulliam, T.H. (1990) A Computational Challenge: Euler Solution for Ellipses, *AIAA Journal*, **28**, 10, 1703–1704.

van Ransbeeck, P. and Hirsch, Ch. (1993) New Upwind Dissipation Models with a Multidimensional Approach, *AIAA Paper* 93-3304.

Roe, P.L. (1981) Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes, *Journal of Computational Physics*, **43**, 357–372.

Roe, P.L. (1986) Characteristics-Based Schemes for the Euler Equations, *Annual Review of Fluid Mechanics*, **18**, 337–365.

Roe, P.L. and van Leer, B. (1988) Non-Existence, Non-Uniqueness and Slow Convergence in Discrete Conservation Laws, *Numerical Methods for Fluid Dynamics*, eds. K. Morton and M. Baines, Clarendon, Oxford, England, U.K.

Rubbert, P.E. (1991) On the Leverage of Computational Fluid Dynamics (CFD) for Airplane Design, *Proceedings of the Fourth International Symposium on Computational Fluid Dynamics*, University of California, Davis, California, 983–986.

Rumsey, C.L., van Leer, B. and Roe, P.L. (1993) A Multidimensional Flux Function with Applications to the Euler and Navier-Stokes Equations, *Journal of Computational Physics*, **105**, 306–323.

Schmidt, W. and Jameson, A. (1985) Euler Solvers as an Analysis Tool for Aircraft Aerodynamics, *Advances in Computational Transonics*, ed. W.G. Habashi, Pineridge Press, U.K., 371–404.

Steger, J.L. (1978) Implicit Finite-Difference Simulation of Flow About Arbitrary Two-Dimensional Geometries, *AIAA Journal*, **16**, 7, 679–686.

Sweby, P.K. (1982) High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws, *SIAM Journal of Numerical Analysis*, **21**, 5, 995–1011.

Sweby, P.K. (1985) Flux Limiters, *Numerical Methods for the Euler Equations of Fluid Dynamics*, ed. F. Angrand et al., SIAM, Philadelphia, 48–65.

Tatsumi, S., Martinelli, L. and Jameson, A. (1995) A new High Resolution Scheme for Compressible Viscous Flow with Shocks, *AIAA Paper* 95-0466.

Tezduyar, T.E. and Hughes, T.J.R. (1983) Finite Element Formulation for Convection Dominated Flows with Particular Emphasis on the Compressible Euler Equations, *AIAA Paper* 83-0125.

Viviand, H. (1985) *Test Cases for Inviscid Flow Field Methods*, AGARD Advisory Report No. 211, 6–21,22.

Webster, W.P. and Shang, J.S. (1991) Thin-Layer Full Navier Stokes Simulations over a Supersonic Delta Wing, *AIAA Journal*, **29**, 9, 1363–1369.

Yee, H.C. and Harten, A. (1987) Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates, *AIAA Journal*, **25**, 2, 266–274.

Yee, H.C., (1989) A Class of High Resolution Explicit and Implicit Shock Capturing-Methods, *NASA Technical Memorandum* 101088.

Yamamoto, S. and Daiguji, H. (1991) Higher-Order Accurate Upwind Schemes for Solving the Compressible Euler and Navier-Stokes Equations, *Proceedings of the International Symposium on Computational Fluid Dynamics*, Davis, California, 1269–1274.

APPENDIX

This section illustrates the detailed form of Eq. (3.14). Eq. (3.1c) can be solved for the density so that

$$\rho = \frac{ap}{b} \tag{A.1}$$

where

$$a = \frac{2\gamma}{\gamma - 1}$$

$$b = 2H_{\infty} - \left(u^2 + v^2\right)$$

The canonical finite element discretization is introduced as follows

$$\{\Delta p, \Delta u, \Delta v\} = \sum_{j=1}^{4} \{\Delta \hat{p}_j, \Delta \hat{u}_j, \Delta \hat{v}_j\} N_j \tag{A.2}$$

Newton's method is applied by setting

$$\{p, u, v\}^{n+1} = \{p, u, v\}^n + \{\Delta p, \Delta u, \Delta v\} \tag{A.3}$$

After substituting Eqs. (A.1), (A.2) and (A.3) in Eqs. (3.8), Eq. (3.14) can be written in detail as

$$\begin{bmatrix} [J_{11}] & [J_{12}] & [J_{13}] \\ [J_{21}] & [J_{22}] & [J_{23}] \\ [J_{31}] & [J_{32}] & [J_{33}] \end{bmatrix} \begin{Bmatrix} \Delta \hat{p}_j \\ \Delta \hat{u}_j \\ \Delta \hat{v}_j \end{Bmatrix} = -\alpha \begin{Bmatrix} \mathrm{Res}_1 \\ \mathrm{Res}_2 \\ \mathrm{Res}_3 \end{Bmatrix} \tag{A.4}$$

where the $[J_{ij}]$ are component matrices of the Jacobian $[J]$. The component matrices are shown below

$$[J_{11}] = \sum_{j=1}^{N_e} \iint_A \left[\frac{a}{b} \left(u \frac{\partial W_i}{\partial x} + v \frac{\partial W_i}{\partial y} \right) N_j - (\varepsilon_1 + \varepsilon_2) \left(\frac{\partial W_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial W_i}{\partial y} \frac{\partial n_j}{\partial y} \right) \right]^n dA$$

$$[J_{12}] = \sum_{j=1}^{N_e} \iint_A \left\{ \frac{ap}{b^2} \left[\left(b + 2u^2 \right) \frac{\partial W_i}{\partial x} + 2uv \frac{\partial W_i}{\partial y} \right] N_j \right\}^n dA$$

$$[J_{13}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{ap}{b^2}\left[2uv\frac{\partial W_i}{\partial x}+\left(b+2v^2\right)\frac{\partial W_i}{\partial y}\right]N_j\right\}^n dA$$

$$[J_{21}]=\sum_{j=1}^{N_e}\iint_A\left\{\left(1+\frac{au^2}{b}\right)\frac{\partial W_i}{\partial x}+\frac{auv}{b}\frac{\partial W_i}{\partial y}\right\}^n N_j dA$$

$$[J_{22}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{ap}{b^2}\left[2u\left(b+u^2\right)\frac{\partial W_i}{\partial x}+v\left(b+2u^2\right)\frac{\partial W_i}{\partial y}\right]N_j-\left(\frac{4\mu}{3\mathrm{Re}}+\varepsilon_1+\varepsilon_2\right)\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial x}\right.$$
$$\left.-\left(\frac{\mu}{\mathrm{Re}}+\varepsilon_1+\varepsilon_2\right)\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial y}\right\}^n dA$$

$$[J_{23}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{apu}{b^2}\left[2uv\frac{\partial W_i}{\partial x}+\left(b+2v^2\right)\frac{\partial W_i}{\partial y}\right]N_j+\frac{\mu}{\mathrm{Re}}\left(\frac{2}{3}\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial y}-\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial x}\right)\right\}^n dA$$

$$[J_{31}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{auv}{b}\frac{\partial W_i}{\partial x}+\left(1+\frac{av^2}{b}\right)\frac{\partial W_i}{\partial y}\right\}^n N_j dA$$

$$[J_{32}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{apv}{b^2}\left[\left(b+2u^2\right)\frac{\partial W_i}{\partial x}+2uv\frac{\partial W_i}{\partial y}\right]N_j-\frac{\mu}{\mathrm{Re}}\left(\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial y}-\frac{2}{3}\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial x}\right)\right\}^n dA$$

$$[J_{33}]=\sum_{j=1}^{N_e}\iint_A\left\{\frac{ap}{b^2}\left[u\left(b+2v^2\right)\frac{\partial W_i}{\partial x}+2v\left(b+v^2\right)\frac{\partial W_i}{\partial y}\right]N_j-\left(\frac{\mu}{\mathrm{Re}}+\varepsilon_1+\varepsilon_2\right)\frac{\partial W_i}{\partial x}\frac{\partial N_j}{\partial x}\right.^n$$
$$\left.-\left(\frac{4\mu}{3\mathrm{Re}}+\varepsilon_1+\varepsilon_2\right)\frac{\partial W_i}{\partial y}\frac{\partial N_j}{\partial y}\right\}dA$$

$$\{Res_1\}=\sum_{j=1}^{N_e}\iint_A\left\{\left[\frac{apu}{b}-(\varepsilon_1+\varepsilon_2)\frac{\partial p}{\partial x}+\varepsilon_2 f_1\right]\frac{\partial W_i}{\partial x}+\left[\frac{apv}{b}-(\varepsilon_1+\varepsilon_2)\frac{\partial p}{\partial y}+\varepsilon_2 f_2\right]\frac{\partial W_i}{\partial y}\right\}dA$$
$$-\oint_S W_i\left(\frac{ap}{b}\vec{V}\cdot\vec{n}-(\varepsilon_1+\varepsilon_2)\frac{\partial p}{\partial n}+\varepsilon_2\vec{F}\cdot\vec{n}\right)dS$$

$$\{Res_2\} = \iint_A \left\{ \left[\frac{apu^2}{b} + p - \frac{2\mu}{3Re}\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) - (\varepsilon_1 + \varepsilon_2)\frac{\partial u}{\partial x} + \varepsilon_2 g_{11} \right] \frac{\partial W_i}{\partial x} \right.$$
$$\left. + \left[\frac{apuv}{b} - \frac{\mu}{Re}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) - (\varepsilon_1 + \varepsilon_2)\frac{\partial u}{\partial y} + \varepsilon_2 g_{12} \right] \frac{\partial W_i}{\partial y} \right\} dA$$
$$- \oint_S W_i \left(\frac{apu}{b} \vec{V} \cdot \vec{n} - (\varepsilon_1 + \varepsilon_2)\frac{\partial u}{\partial n} + \varepsilon_2 \vec{G} \cdot \vec{n} \right) dS - \oint_S p \frac{dy}{dS} dS$$
$$+ \oint_S W_i \frac{\mu}{Re}\left[\frac{2}{3}\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right)\frac{dy}{dS} - \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\frac{dx}{dS} \right] dS$$

$$\{Res_3\} = \iint_A \left\{ \left[\frac{apuv}{b} - \frac{\mu}{Re}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) - (\varepsilon_1 + \varepsilon_2)\frac{\partial v}{\partial x} + \varepsilon_2 g_{21} \right] \frac{\partial W_i}{\partial x} \right.$$
$$\left. + \left[\frac{apv^2}{b} + p - \frac{2\mu}{3Re}\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right) - (\varepsilon_1 + \varepsilon_2)\frac{\partial v}{\partial y} + \varepsilon_2 g_{22} \right] \frac{\partial W_i}{\partial y} \right\} dA$$
$$- \oint_S W_i \left(\frac{apv}{b} \vec{V} \cdot \vec{n} - (\varepsilon_1 + \varepsilon_2)\frac{\partial v}{\partial n} + \varepsilon_2 \vec{G} \cdot \vec{n} \right) dS + \oint_S p \frac{dx}{dS} dS$$
$$+ \oint_S W_i \frac{\mu}{Re}\left[\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\frac{dy}{dS} - \frac{2}{3}\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right)\frac{dx}{dS} \right] dS$$

FIGURES

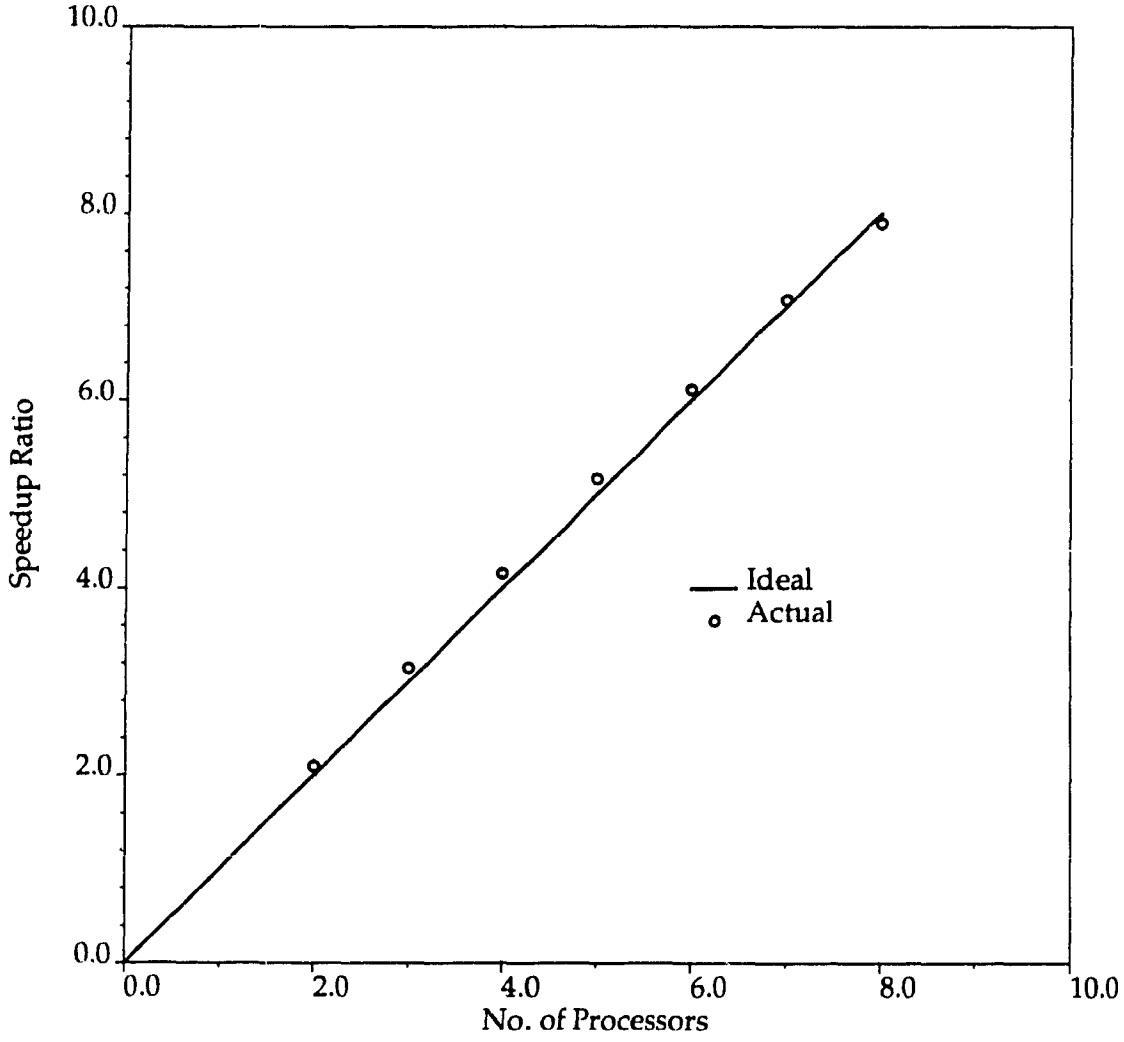

Figure 1. Performance of the direct solver on an 8–processor SGI Power Series 280 GTX with 25 MHz MIPS R3000 Processors.

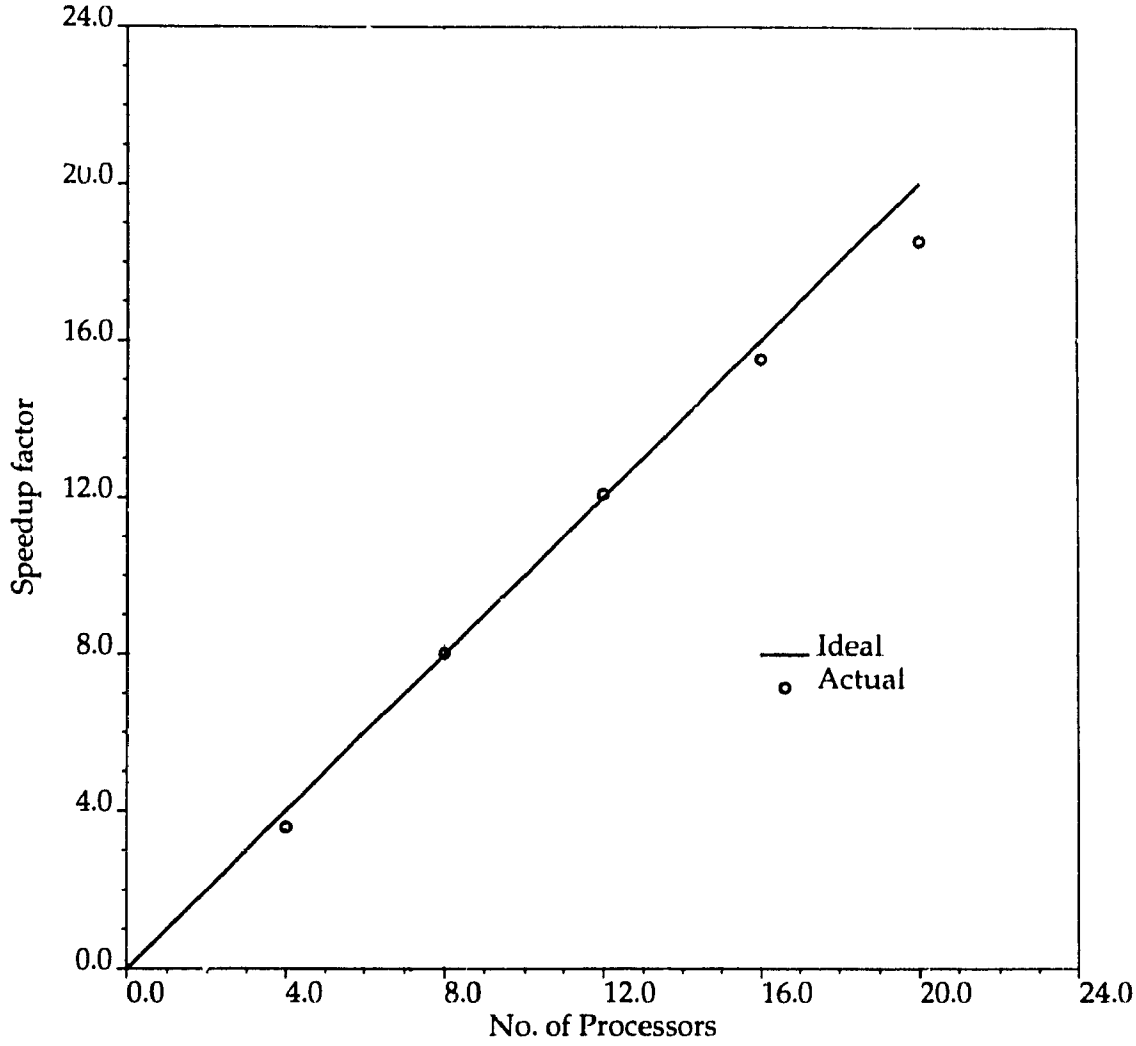

Figure 2. Performance of the direct solver on a 20–processor SGI Challenge with 50 MHz MIPS R4400 processors.

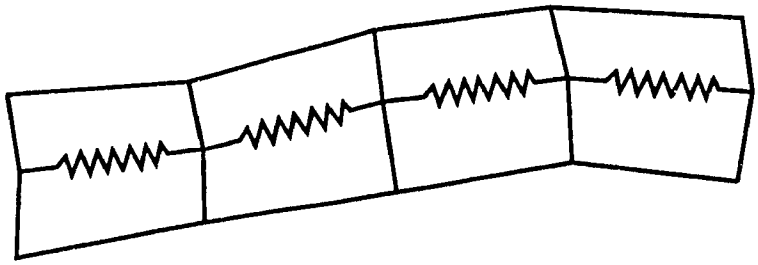

Figure 3. Mesh line replaced by a chain of springs.

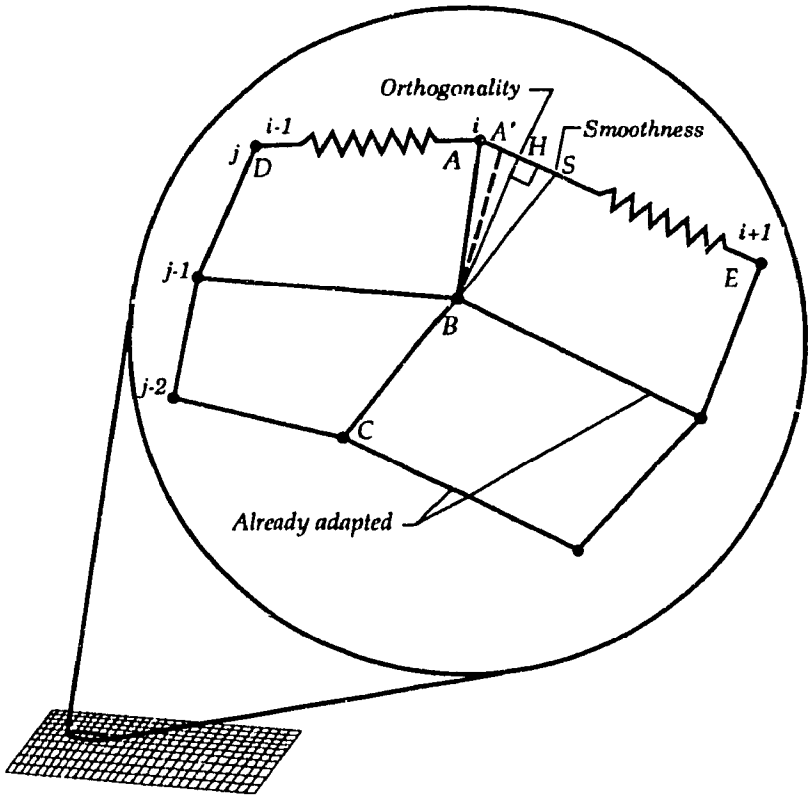

Figure 4. Locations of the projection points H and S.

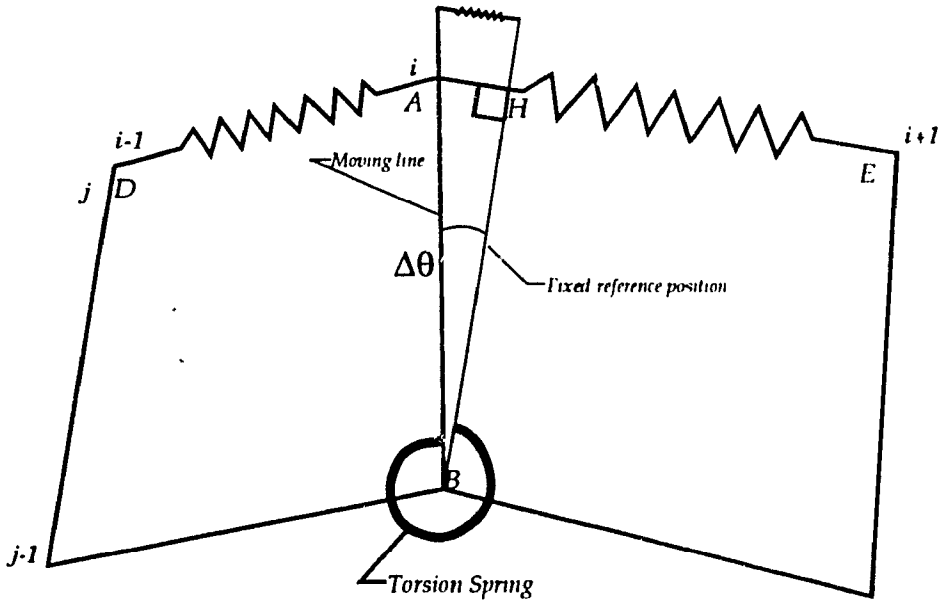

Figure 5. Torsion spring replaced by a conventional spring.

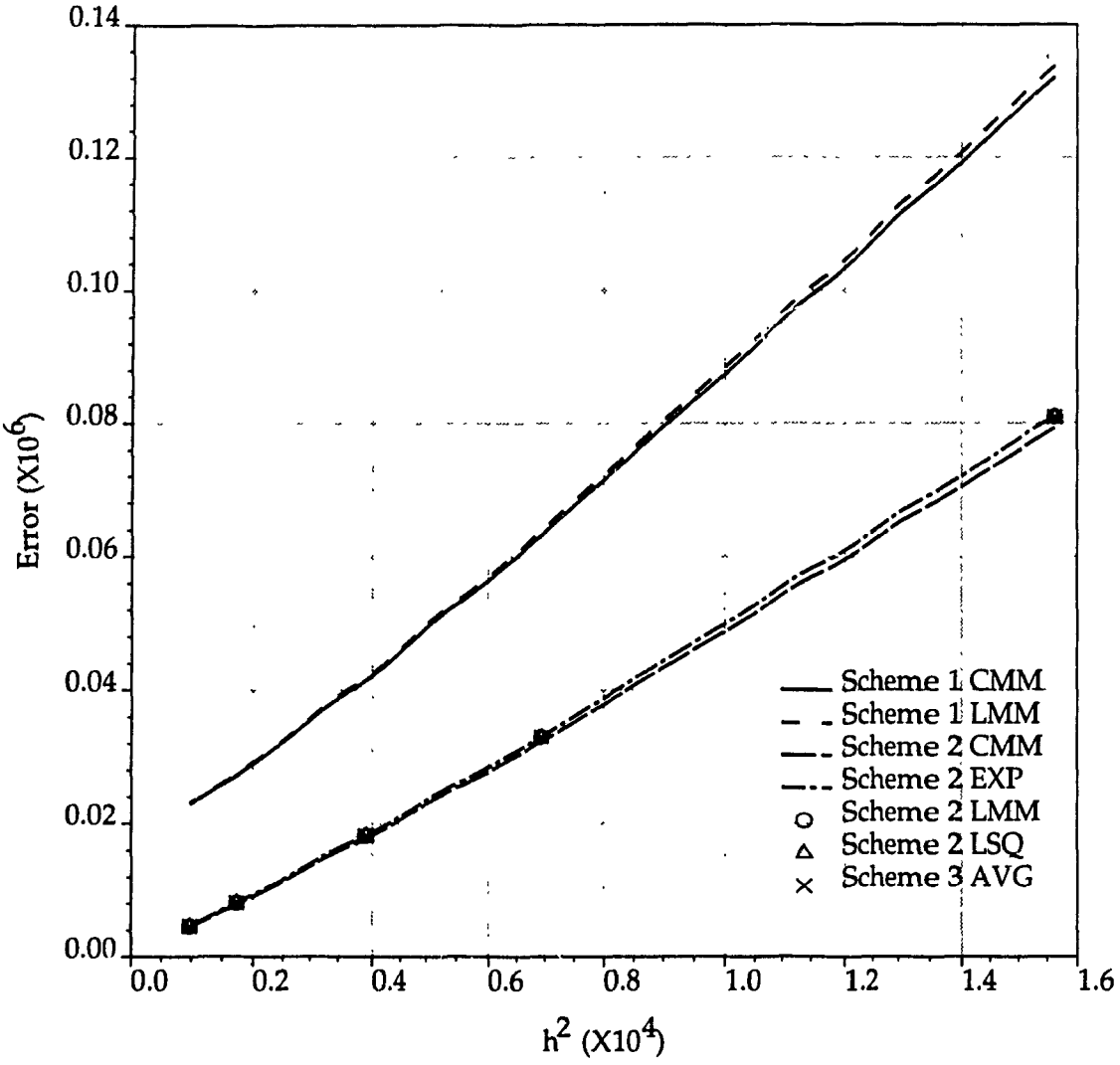

Figure 6. Density error as a function of h^2 for the order of accuracy test.

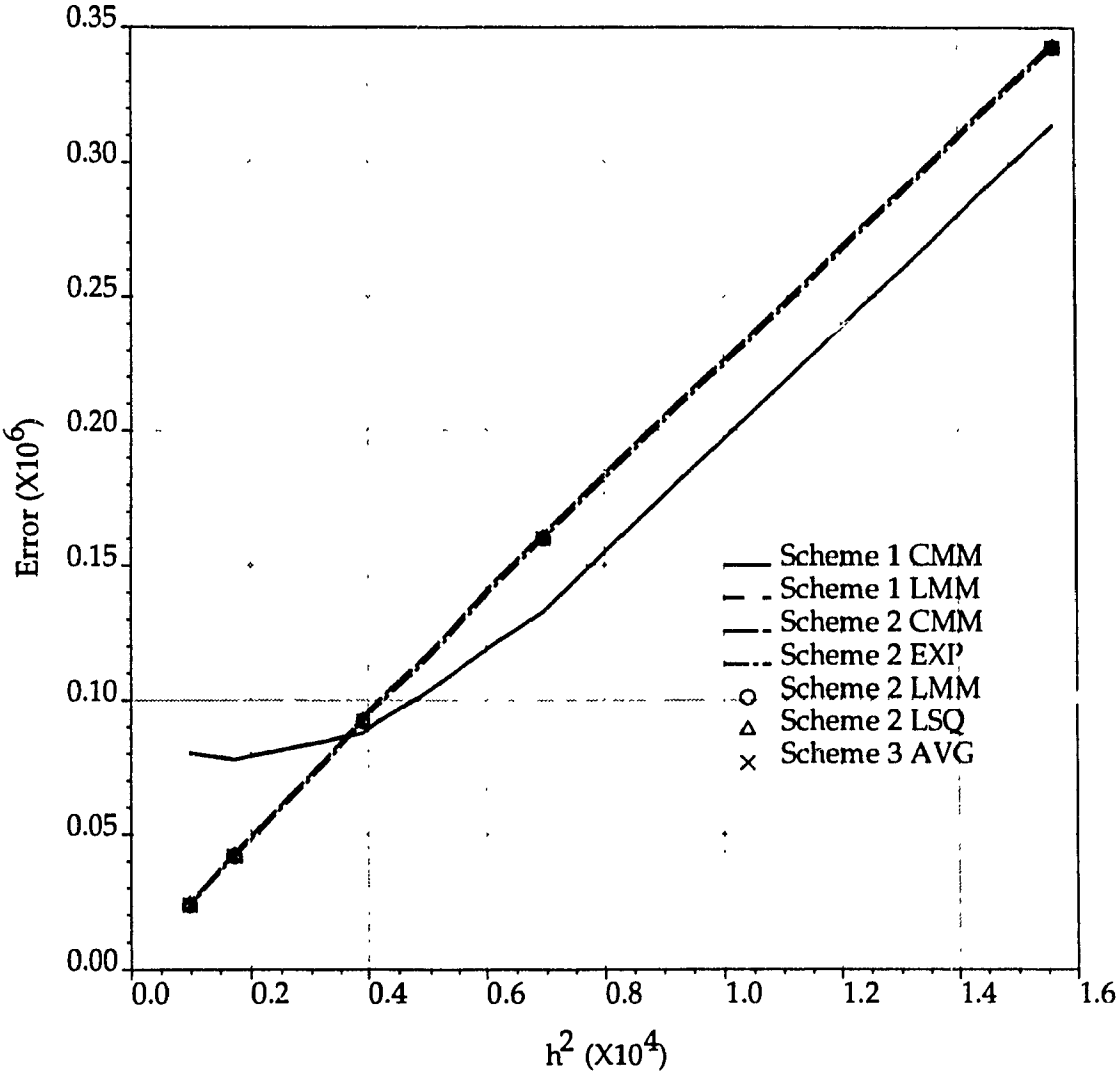

Figure 7. U–velocity error as a function of h^2 for the order of accuracy test.

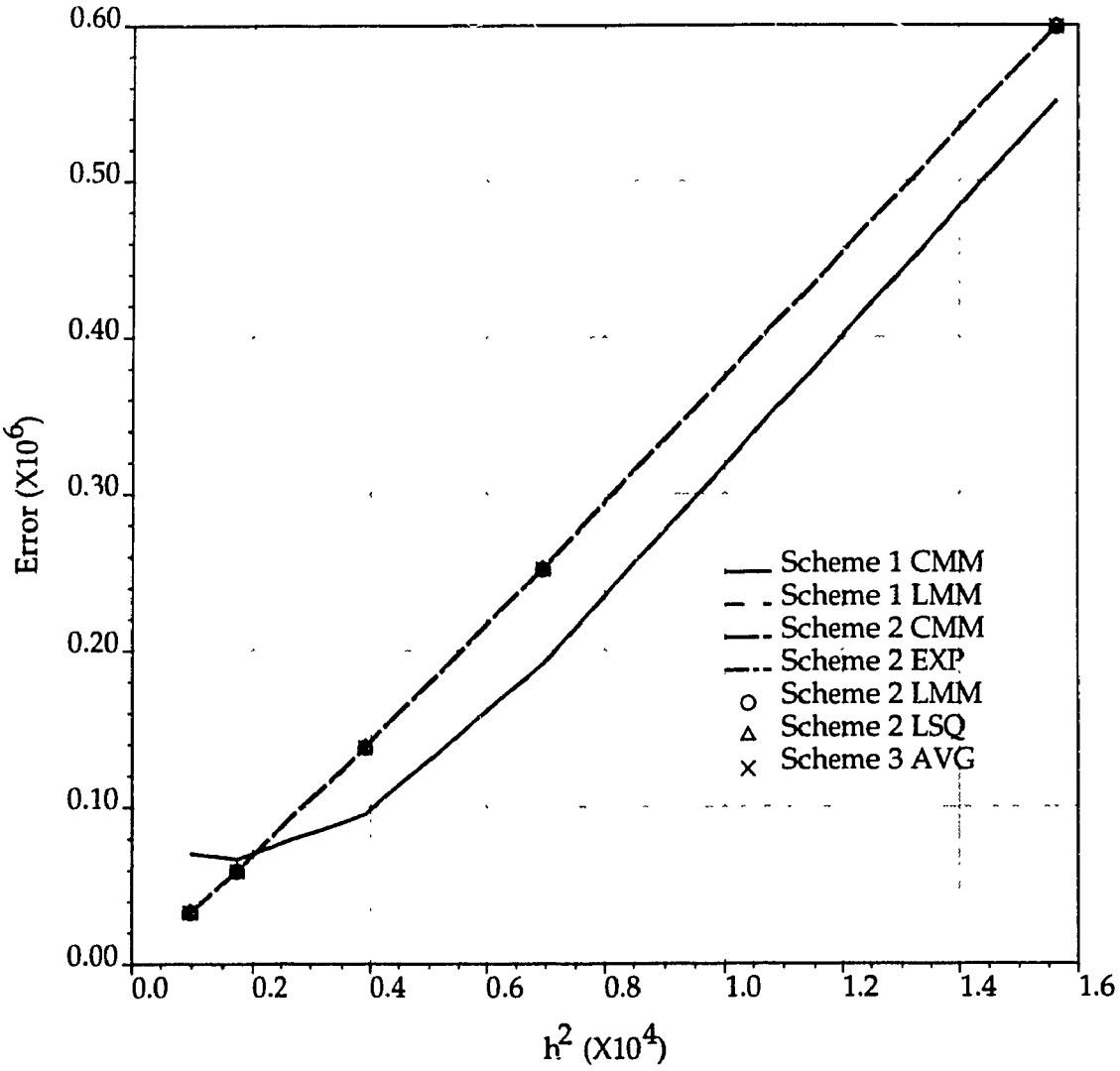

Figure 8. V–velocity error as a function of h^2 for the order of accuracy test.

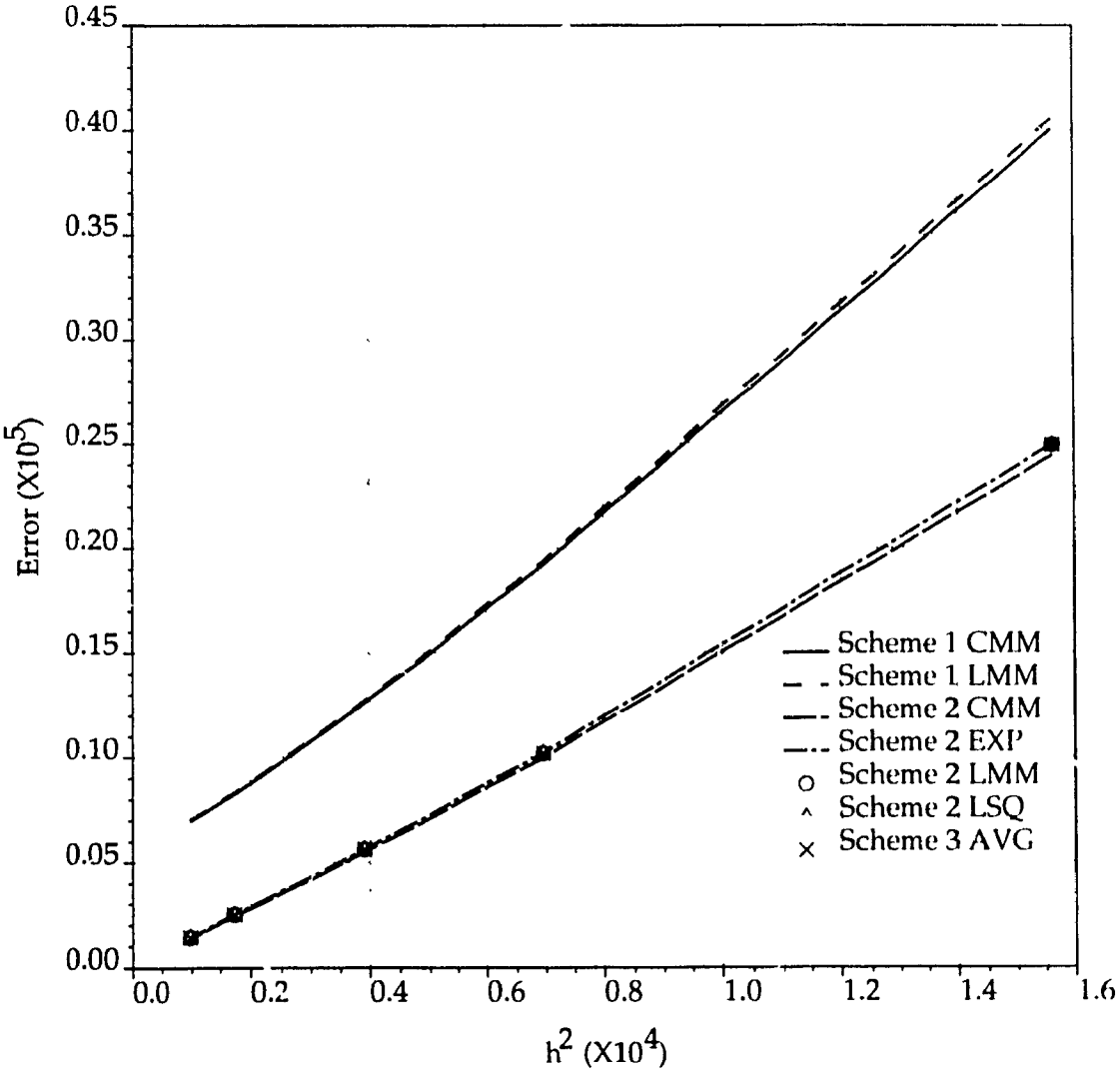

Figure 9. Pressure error as a function of h^2 for the order of accuracy test.

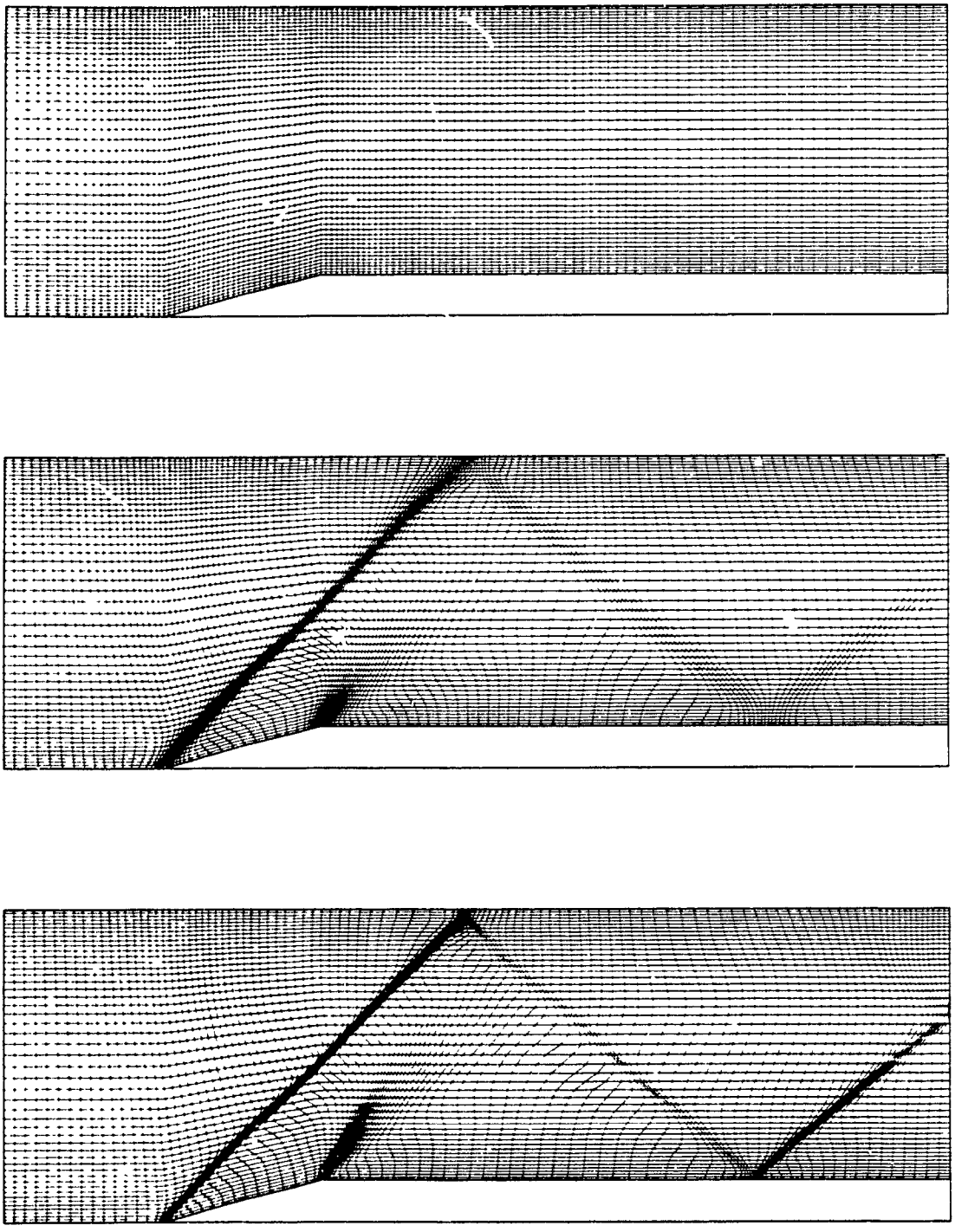

Figure 10. Initial, intermediate and final adapted grid for supersonic flow over a 15^0 wedge.

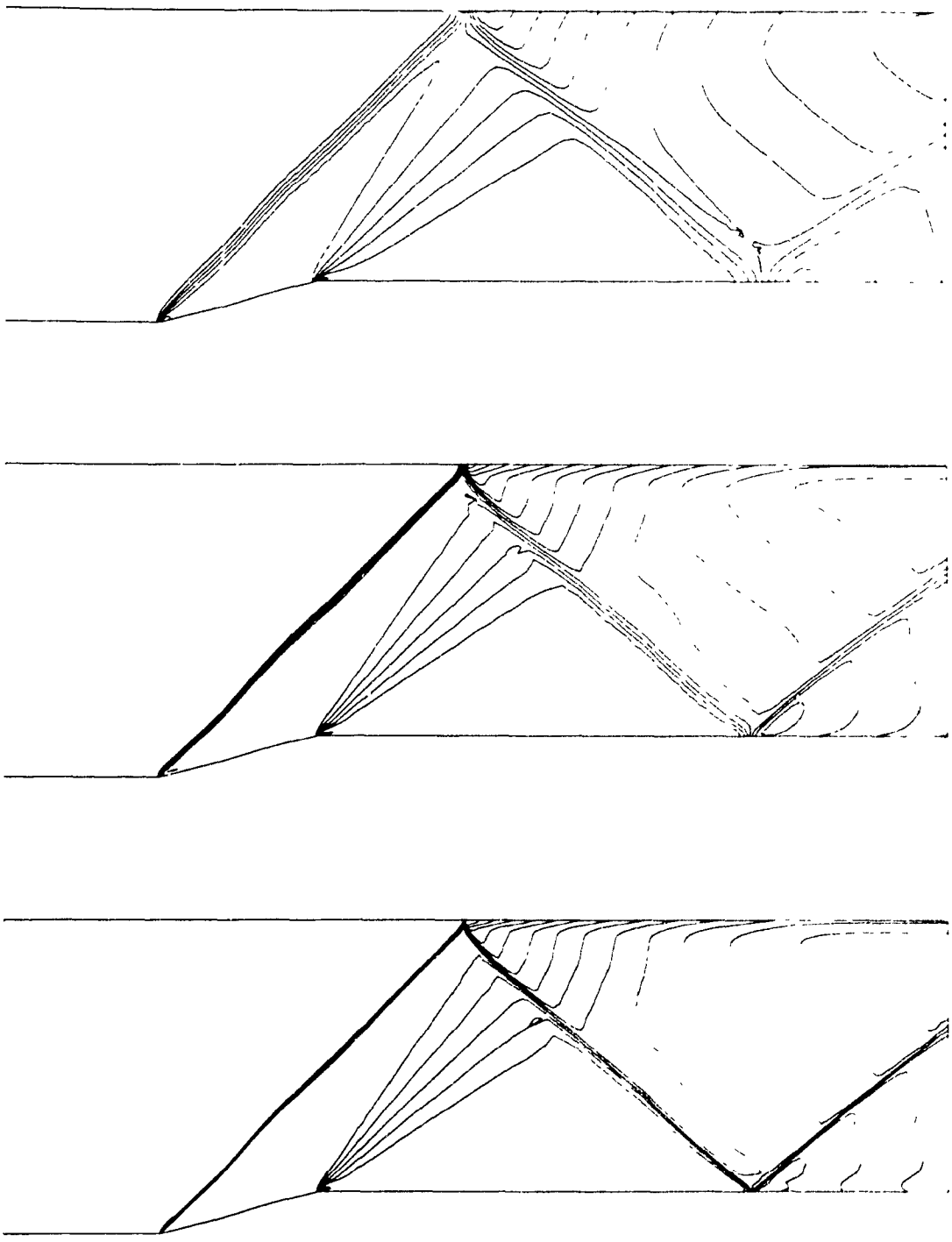

Figure 11. Initial, intermediate and final Mach number contours for supersonic flow over a 15° wedge (M_{min}=0.65, ΔM=0.05).

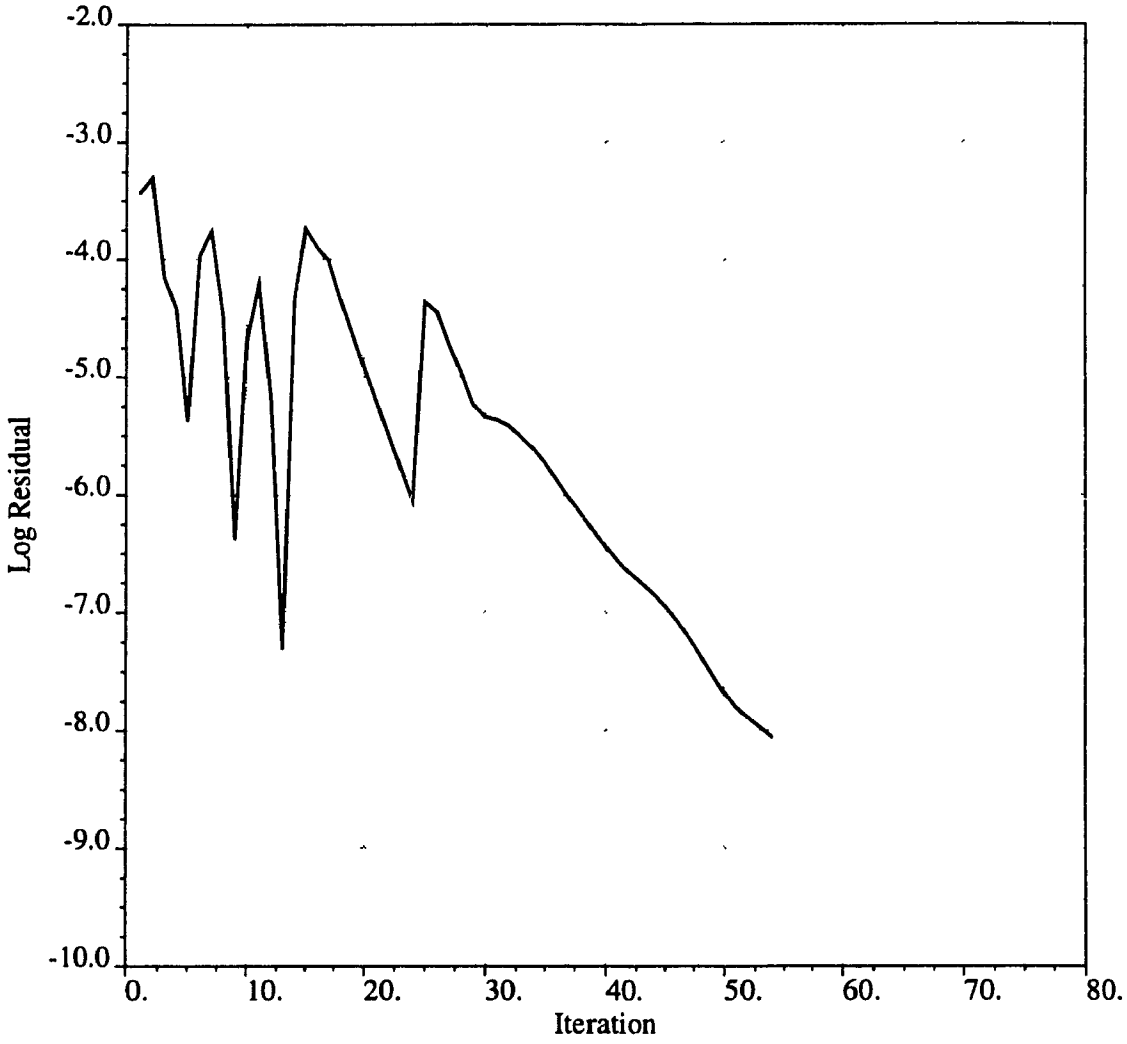

Figure 12. Convergence history for supersonic flow over a 15^{o} wedge.

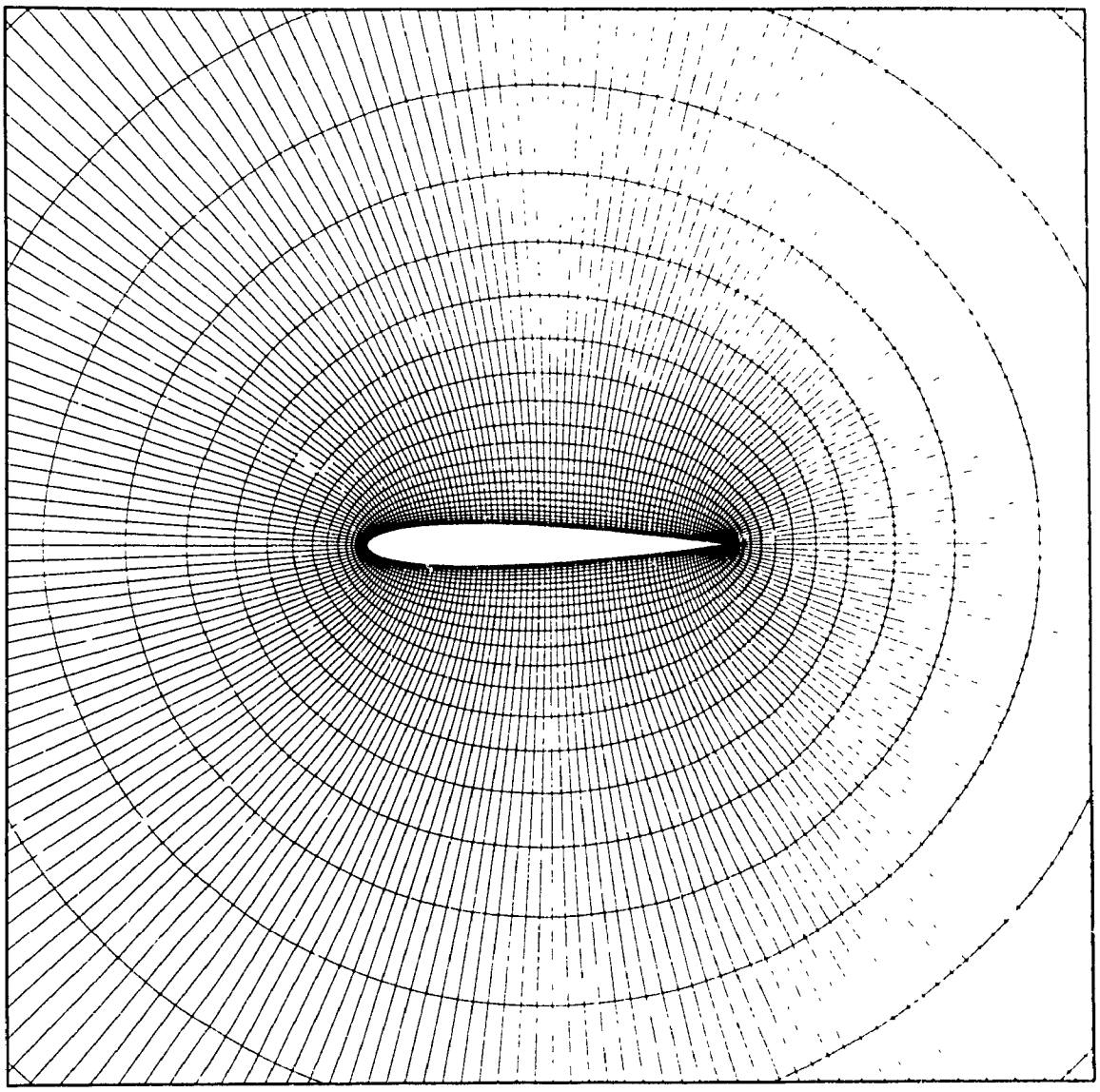

Figure 13. Detail of the (200X32) O–grid around a NACA0012 airfoil.

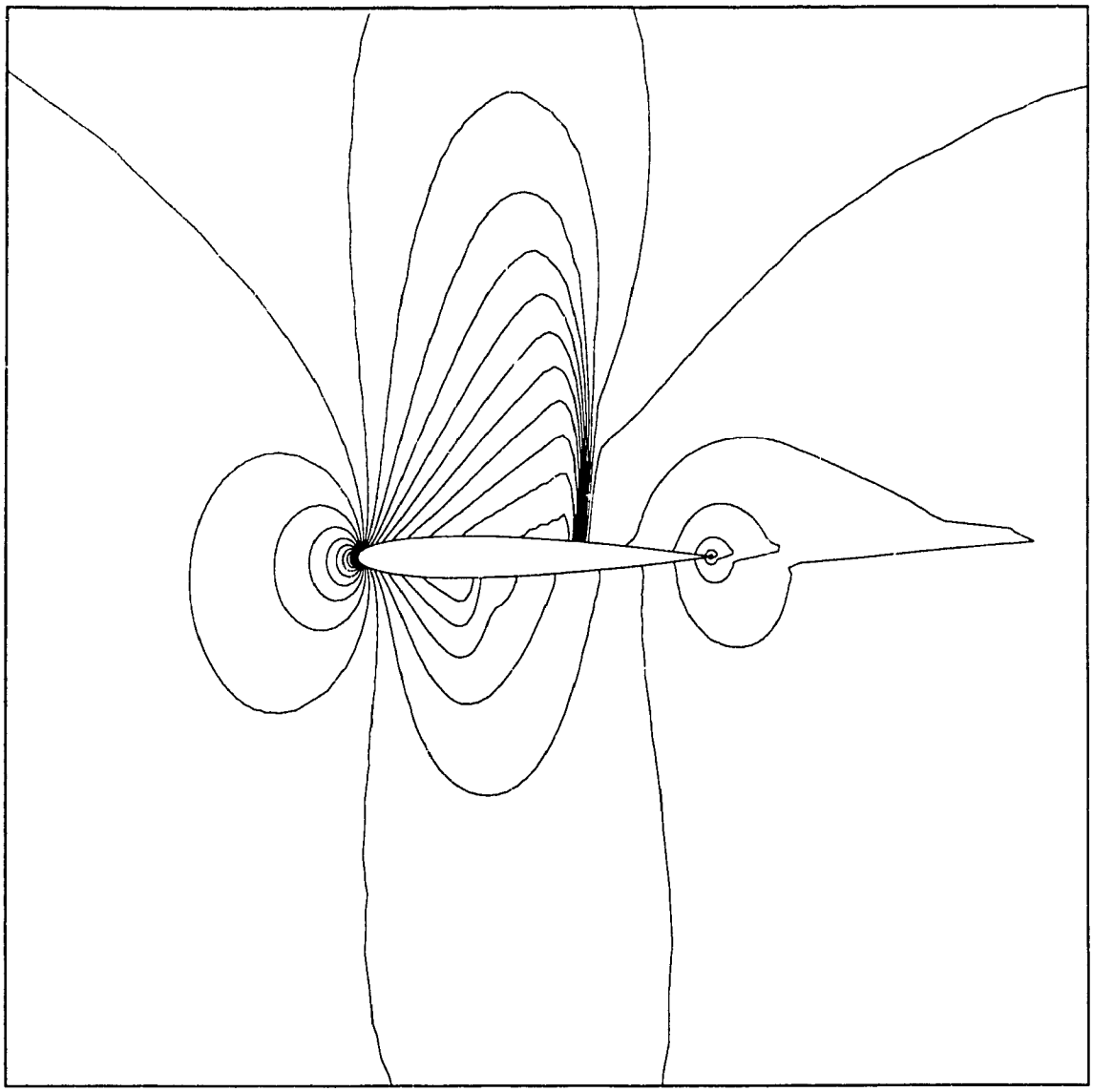

Figure 14. Mach number contours for inviscid flow over a NACA0012 airfoil at M=0.8 and $\alpha=1.25^{o}$ ($M_{min}=0.1$, $\Delta M=0.05$).

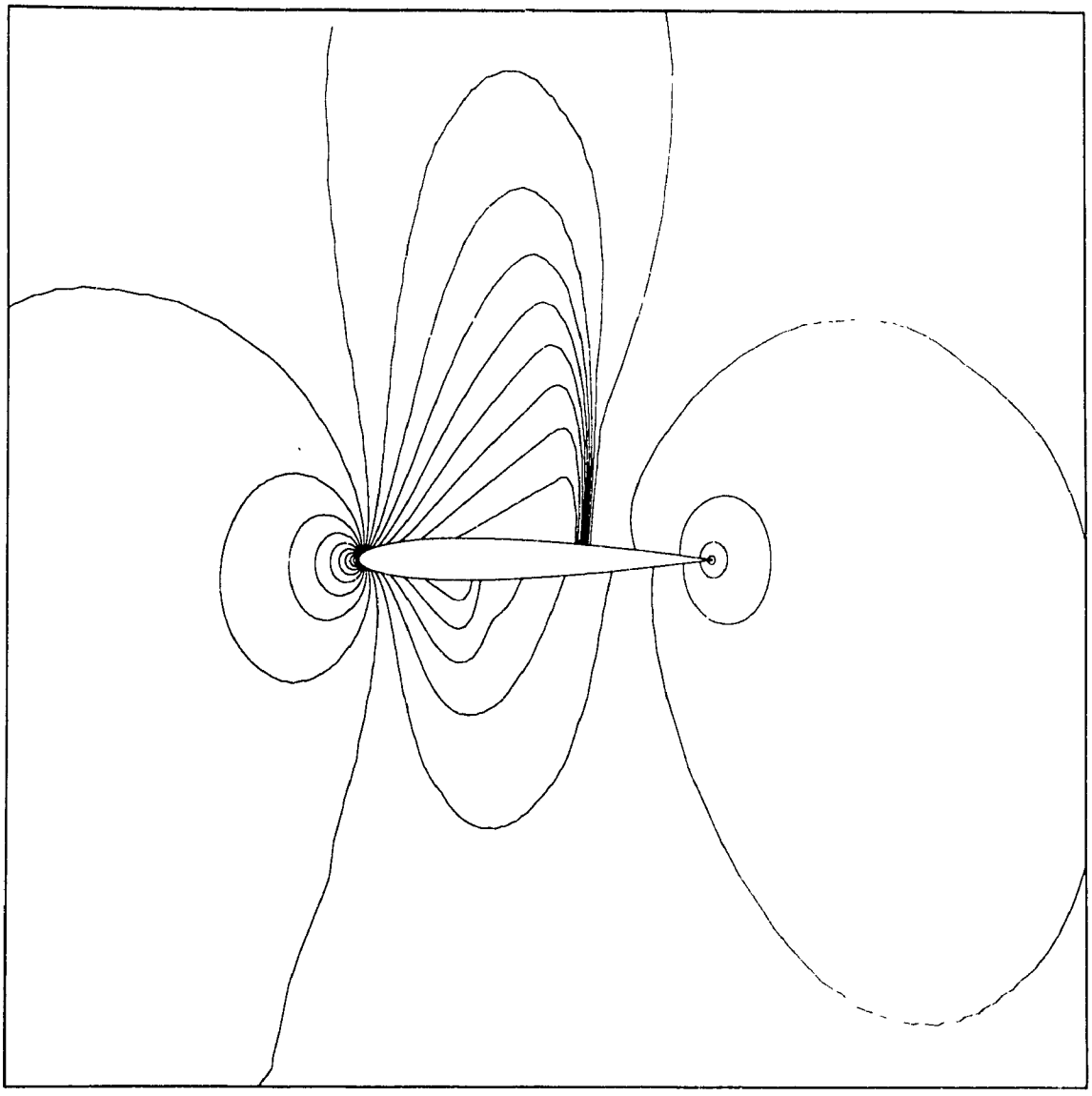

Figure 15. Nondimensional pressure contours for inviscid flow over a NACA0012 airfoil at M=0.8 and α=1.25^{o}: (P_{min}=0, ΔP=0.05).

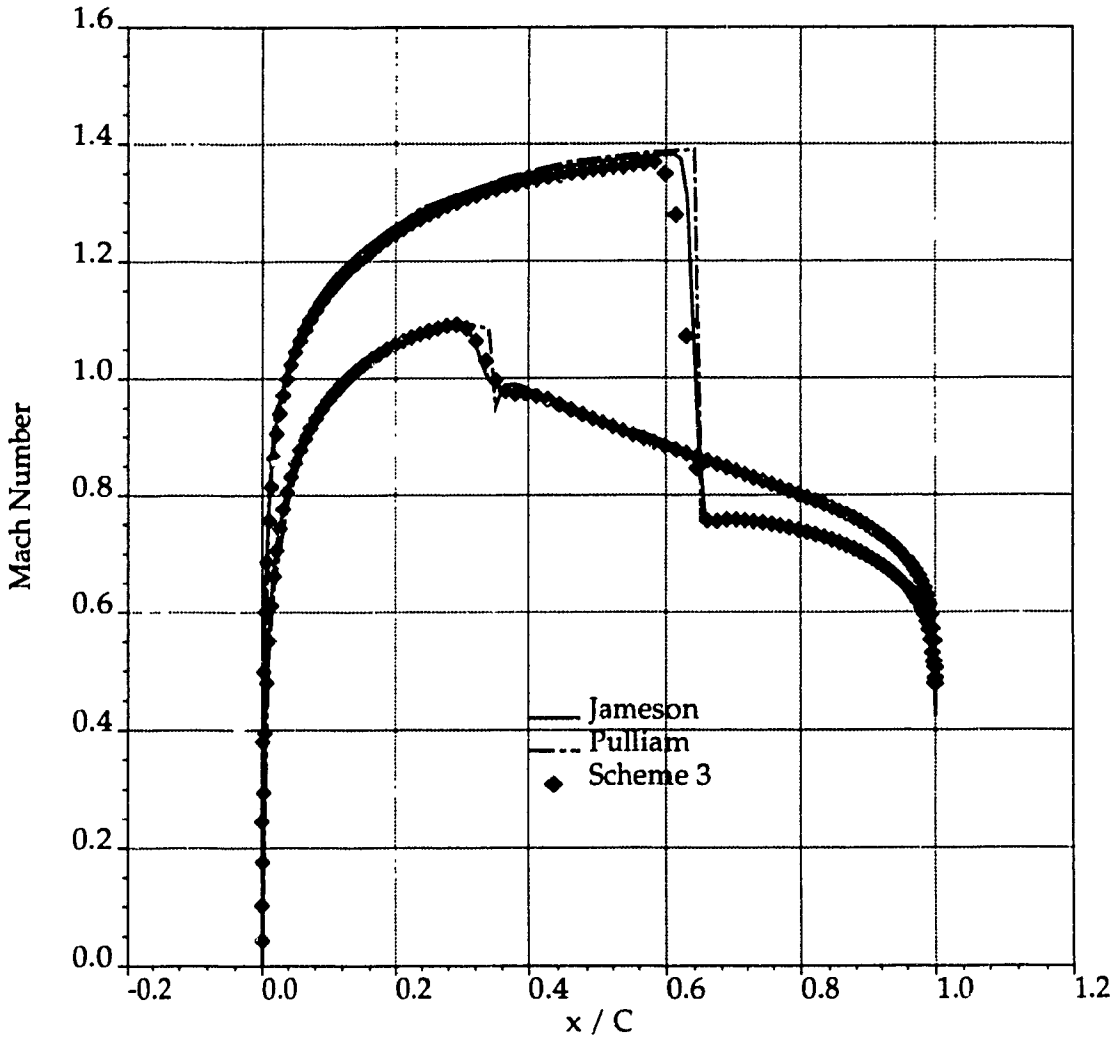

Figure 16. Surface Mach number distribution for inviscid flow over a NACA0012 airfoil at M=0.8 and $\alpha=1.25^{o}$.

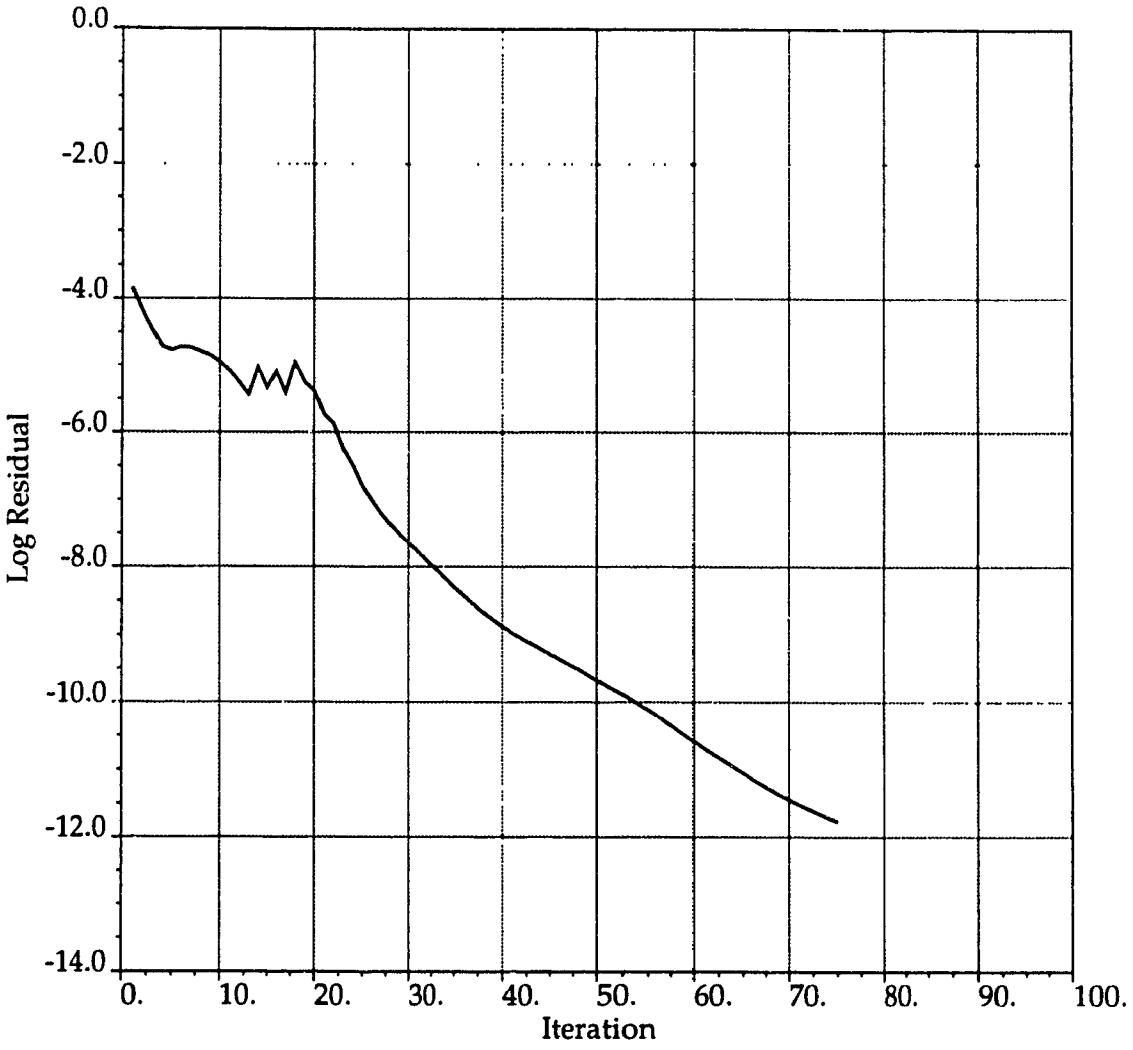

Figure 17. Convergence history, showing the effect of four artificial viscosity cycles, for inviscid flow over a NACA0012 airfoil at M=0.8 and $\alpha=1.25^{o}$.

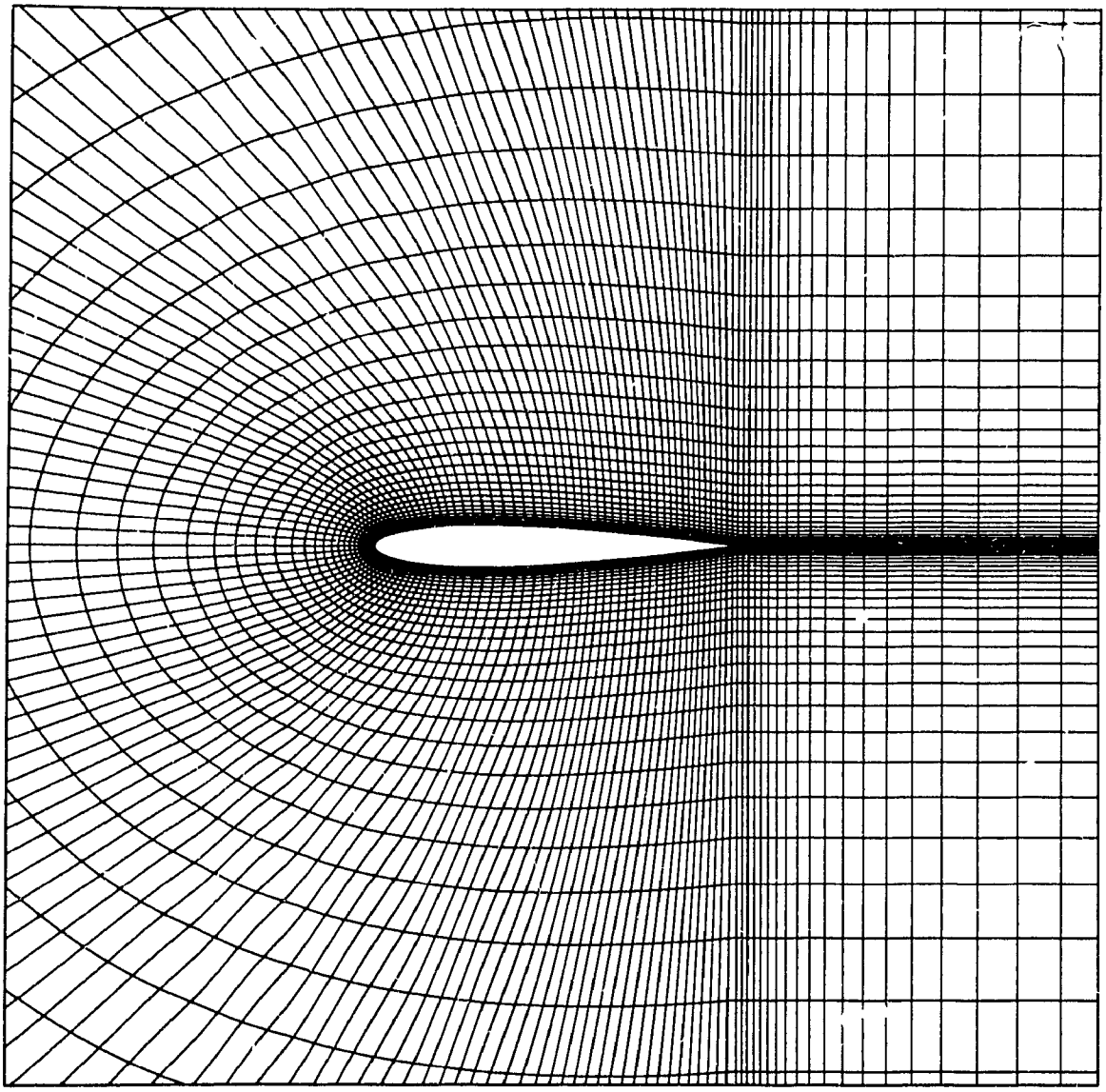

Figure 18. Detail of the (200X48) C–grid for laminar transonic viscous flow over a NACA0012 airfoil.

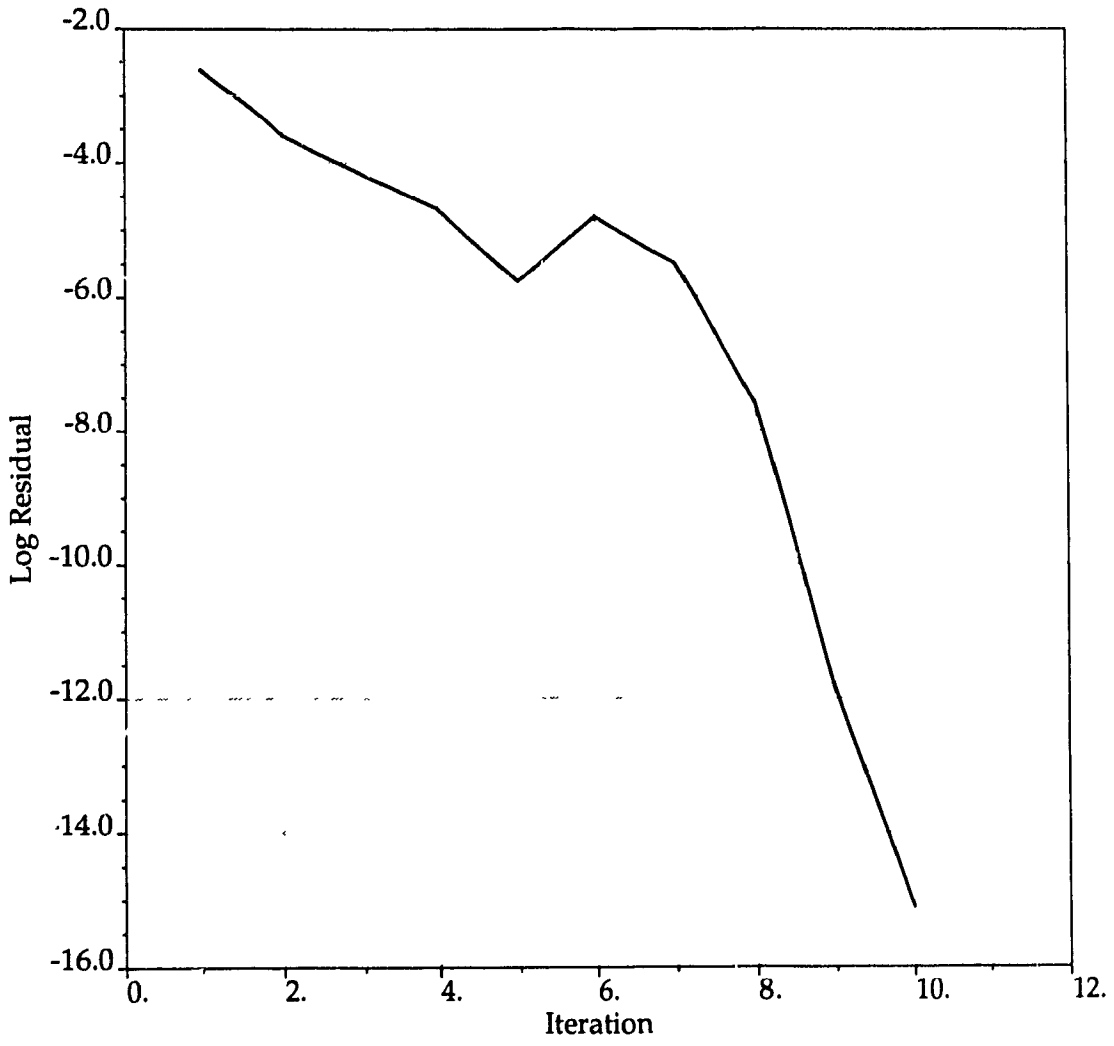

Figure 19. Convergence history of the first order solution for laminar transonic viscous flow over a NACA0012 airfoil at M=0.9, Re=5,000 and $\alpha=0^{o}$.

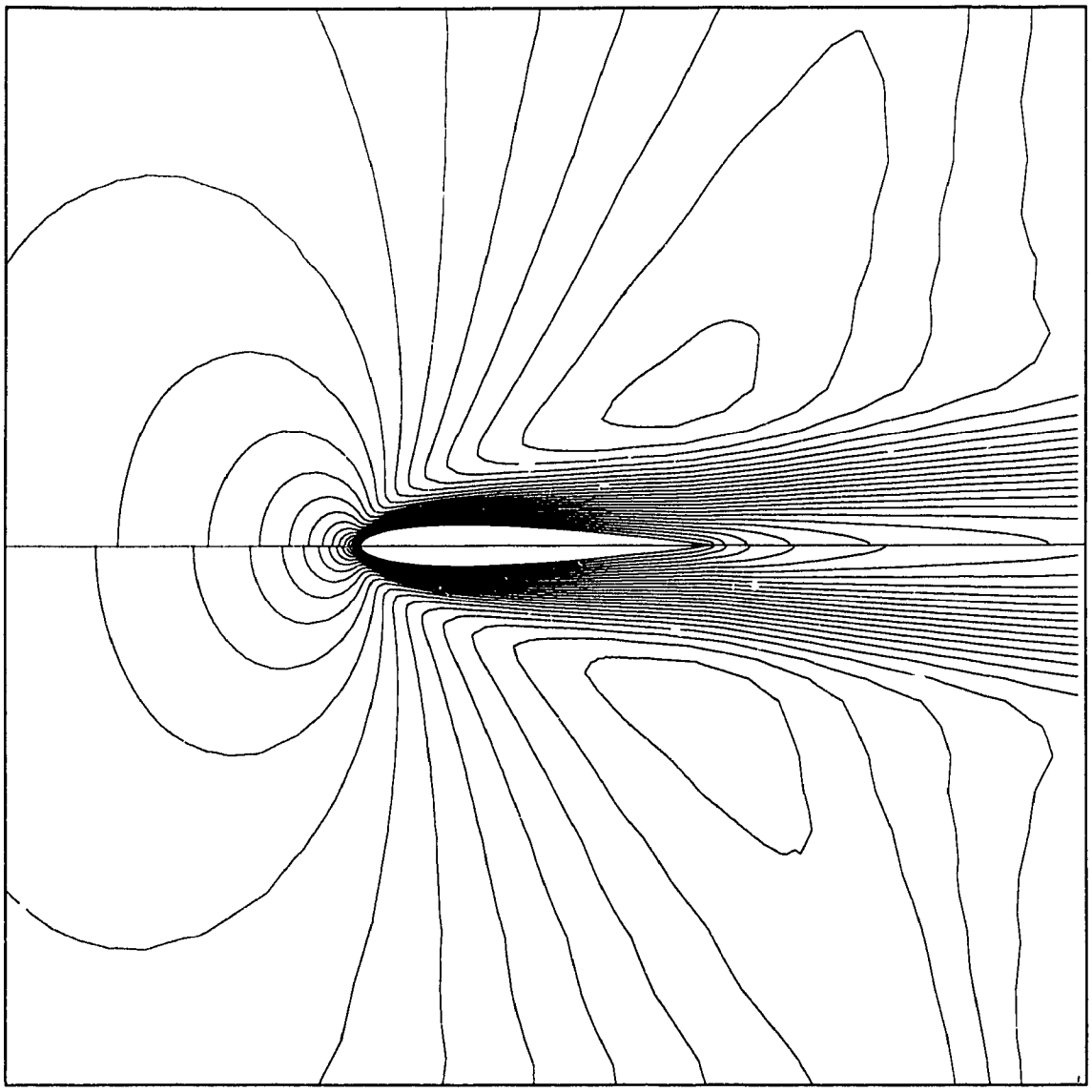

Figure 20. Comparison of the Mach number contours of the first order solution at M=0.9, Re=5,000 and $\alpha=0^{o}$, at the top of the figure, with those of the second order solution at M=0.9, Re=192.3 and $\alpha=0^{o}$ at the bottom ($M_{min}=0$, $\Delta M=0.05$).

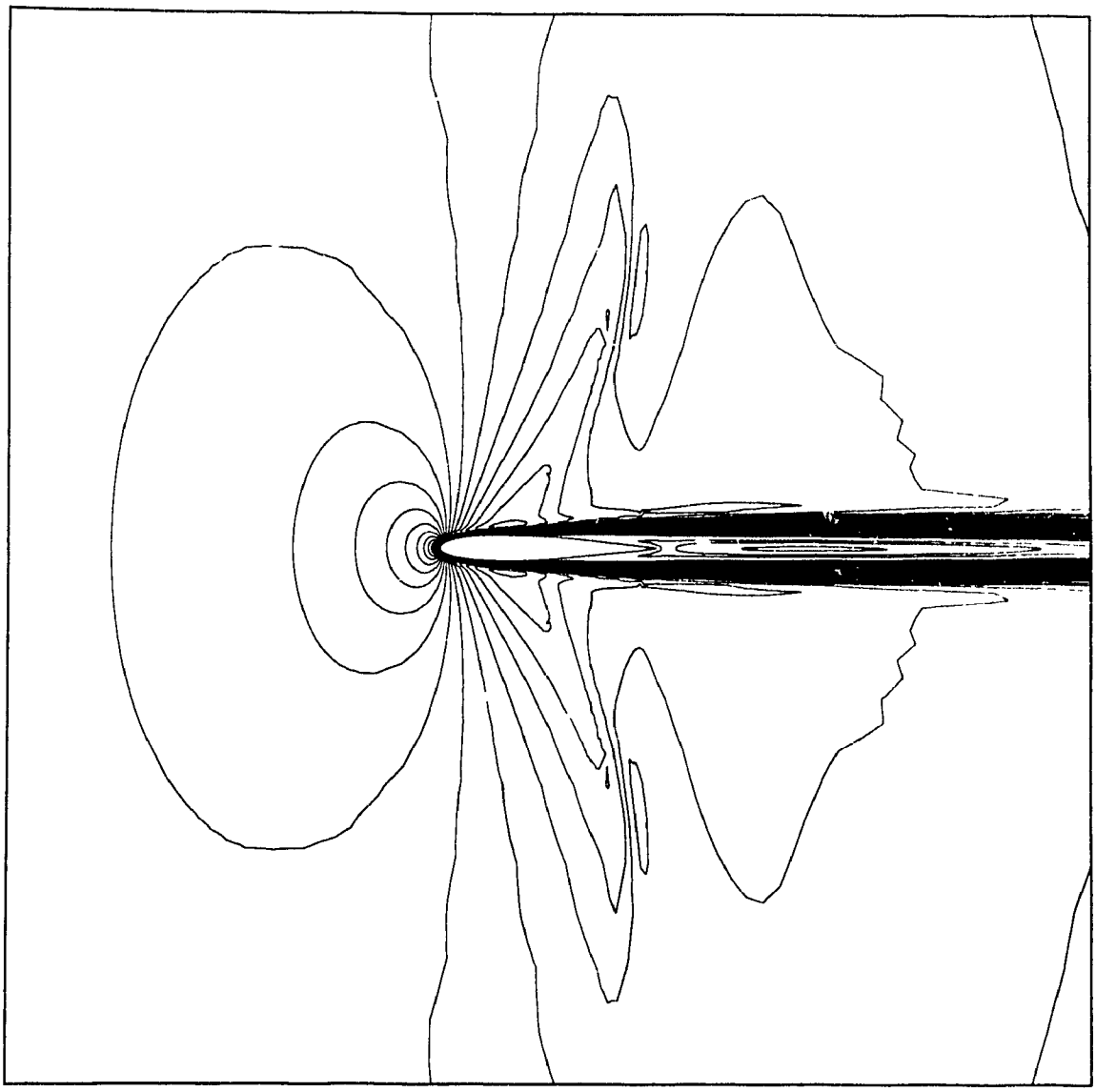

Figure 21. Mach number contours of the second order solution for laminar transonic viscous flow over a NACA0012 airfoil at M=0.9, Re=5,000 and $\alpha=0^{o}$ (Scheme 3 , $M_{min}=0$, $\Delta M=0.05$).

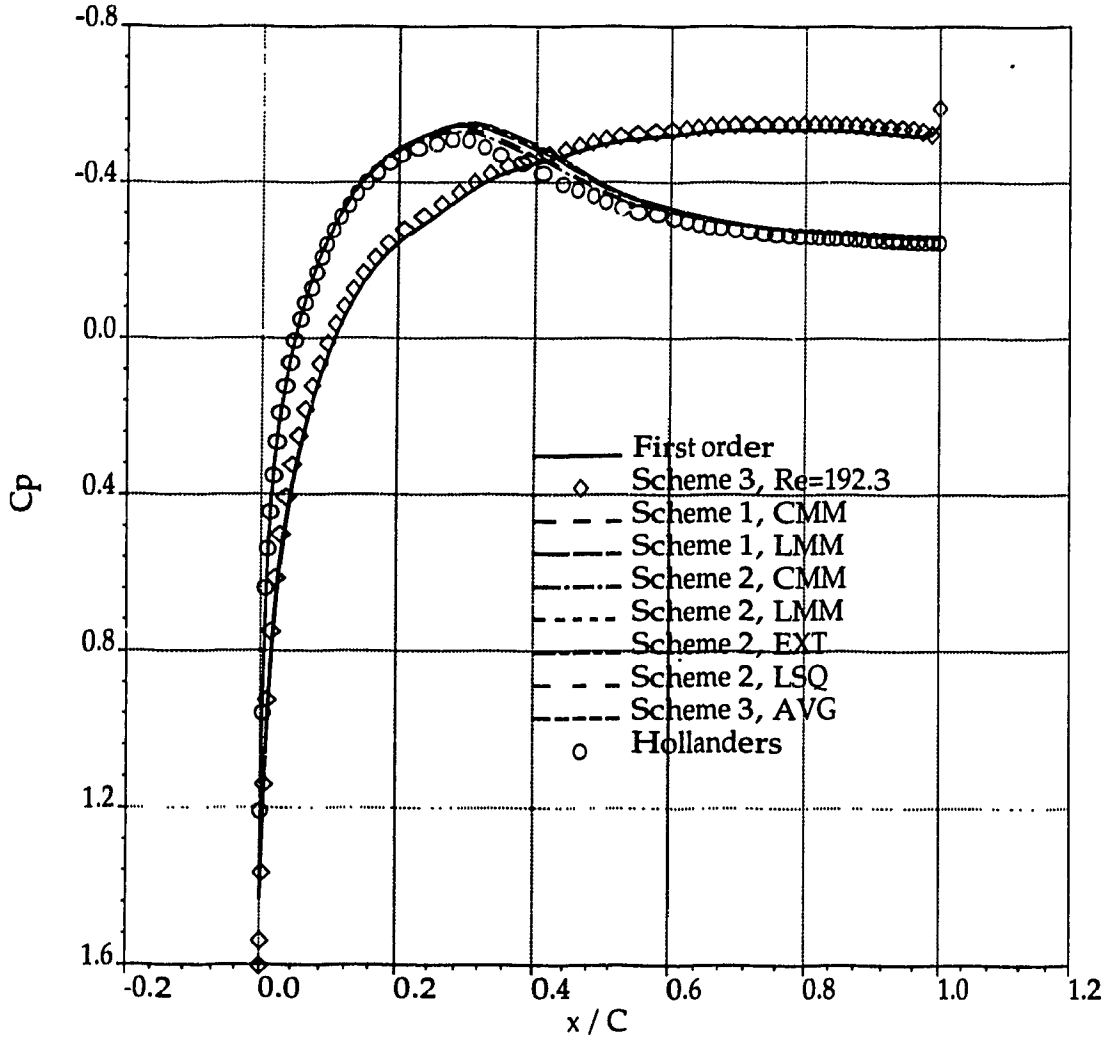

Figure 22. Cp distribution of the three second order schemes for transonic laminar viscous flow over a NACA0012 airfoil at M=0.9, Re=5,000 and $\alpha=0^{o}$.

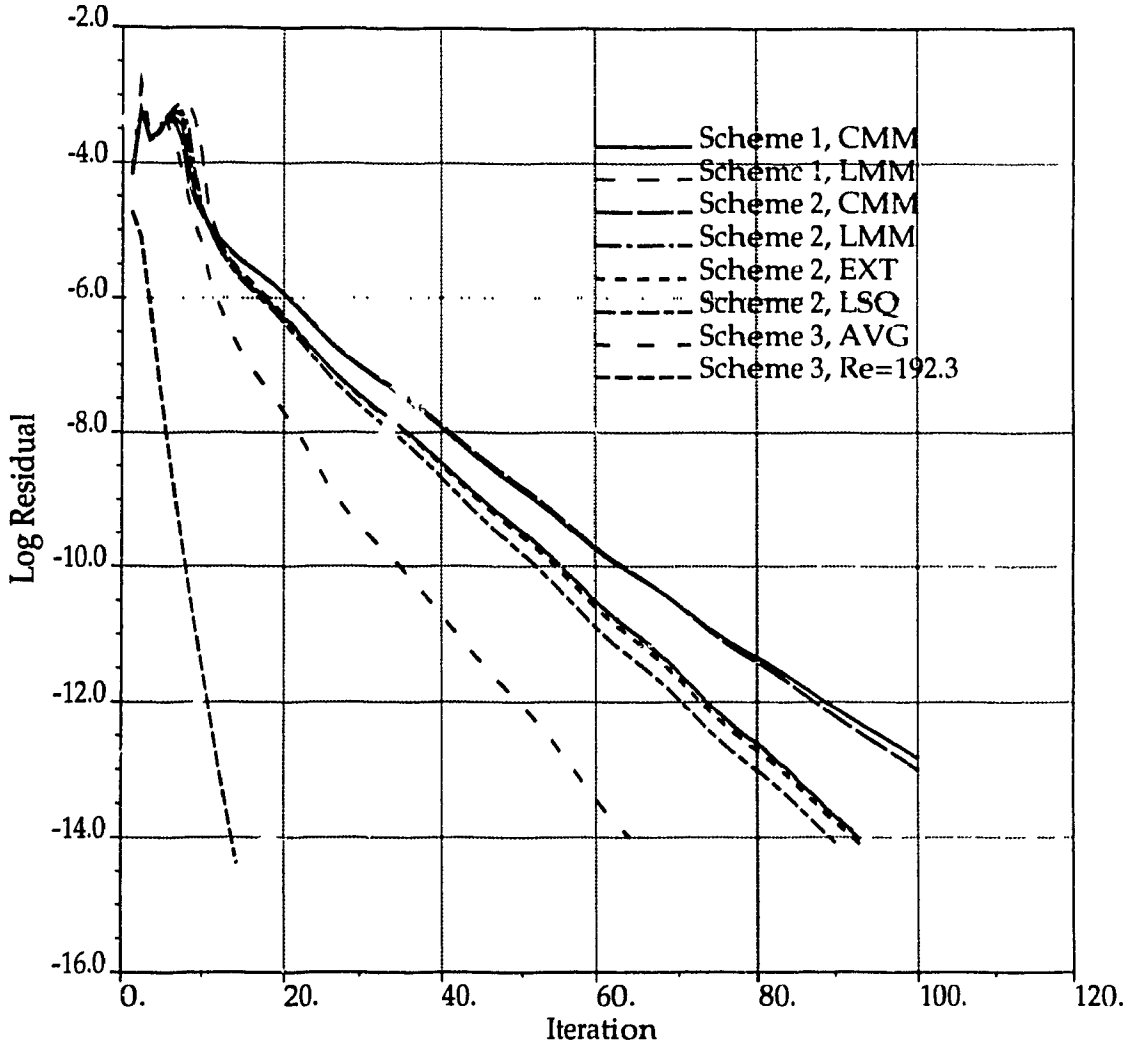

Figure 23. Convergence histories of the three second order schemes for transonic laminar viscous flow over a NACA0012 airfoil at M=0.9, Re=5,000 and a=0^{o}.