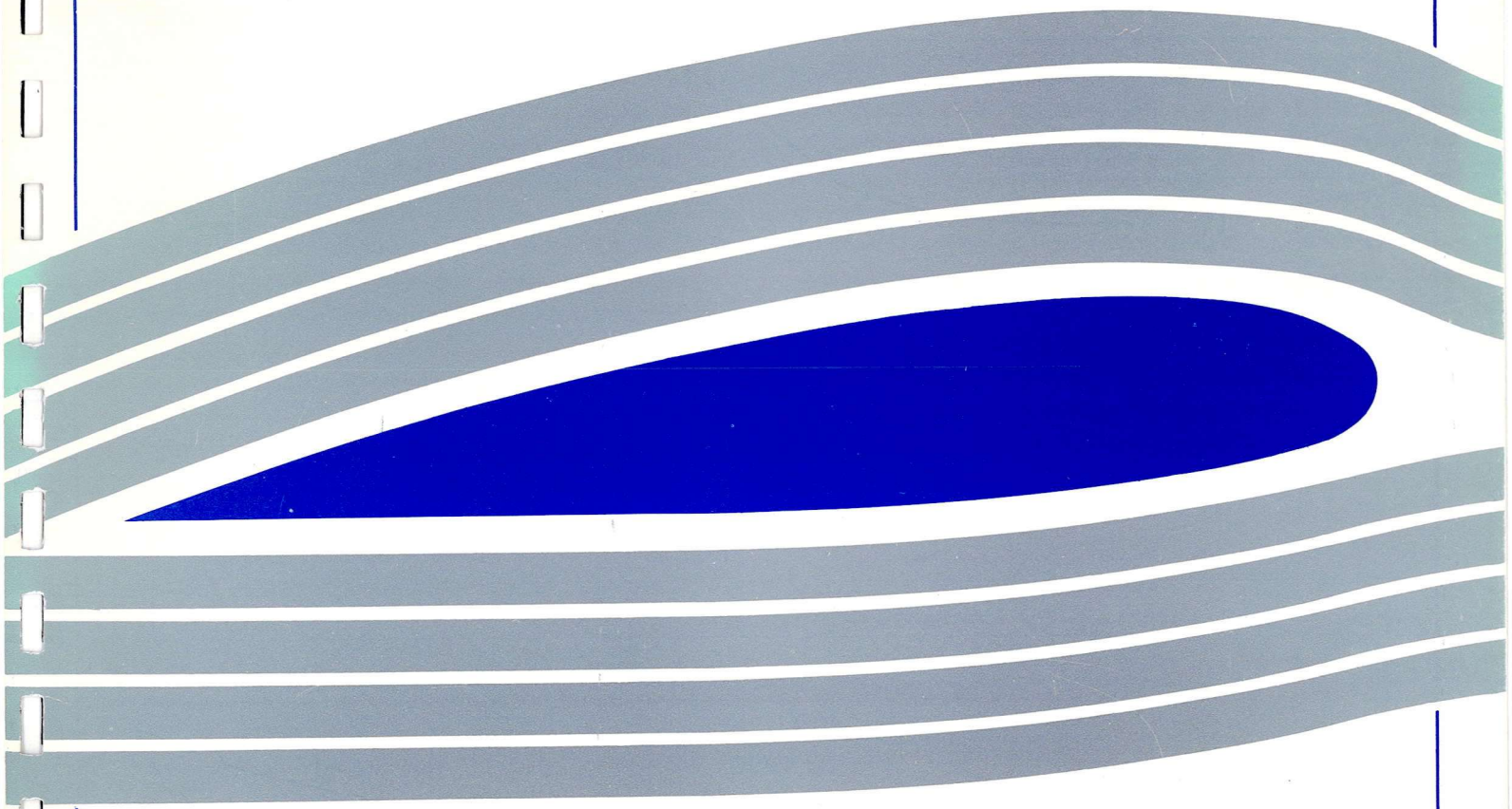
University of Glasgow
DEPARTMENT OF
AEROSPACE
ENGINEERING

Implicit High-Order Resolution of Supersonic Flow on Unstructured Grids

Y.F.Yao



Implicit High-Order Resolution of Supersonic Flow on Unstructured Grids

Y.F.Yao

Dept. of Aerospace Engineering
Aero. Report 9409
University of Glasgow

May 31,1994

ABSTRACTS

An upwinding-biased finite-volume implicit high-order technique has been implemented on unstructured grids for supersonic compressible flows. The method utilizes a Point-Gauss-Jacobi and a Point-Gauss-Seidel implicit scheme to improve the efficiency of computation. High-order spacial accuracy is also achieved by the use of the method of linear reconstruction of the variables proposed by Barth & Jesperson and the method of variable extrapolation MUSCL approach (Monotone Upstream-centred Schemes for Conservation Laws) of van Leer. The above techniques have been applied to the supersonic corner flow. Comparisons of the efficiency and accuracy between: explicit and implicit scheme; first-order and high-order schemes have been made.

CONTENTS

1.Introduction

Recently, the use of unstructured grid techniques associated with the finite-volume method for Computational Fluid Dynamics (CFD) calculations has become more widespread, due to the flexibility they afford in discretizing arbitrarily complex geometries (for example, a complete airplane), and also due to the possibilities they offer in resolving highly localized complex flow phenomena through the use of adaptive mesh refinement or mesh moving techniques. Here, a brief survey of the application of the unstructured grid technique to CFD is given.

In 1986, Jameson et al [1] reported some very positive results for the computation of inviscid flow over a complete aircraft on unstructured grids using a finite-element method. They used an explicit multi-stage Runge-Kutta method with central differencing and artificial viscosity. Mavriplis [2] employed a similar approach together with a multi-grid technique in order to obtain faster convergence. Morgan and Peraire [3] in 1987 used the Taylor-Galerkin method in conjuction with unstructured grids to deal with a wide range of CFD problems. Also in 1987, Peraire et al [4] developed an adaptive remeshing algorithm for the application of unstructured grids to CFD. Since then more attention has been paid to a combination of upwinding technique and unstructured grids. In 1987, Stoufflet et al [5] proposed an upwind scheme for the solution of Euler equations using unstructured grids in 3-D. Thareja et al [6] in 1988 developed an unstructured upwind scheme for the solution of Navier-Stokes equations. Barth & Jesperson [7] in 1989 provided a promising basis for the implementation of upwinding on unstructured grids. Batina [8] used van Leer's flux vector spliting method for the solution of Euler equations on unstructured grids. The main disadvantage of employing unstructured grids is the increased computational time. One method to improve this is to use multi-grid techniques [2][9]. The other is to adopt implicit schemes [10-12] to speed up the convergence. Thareja et al [10] reported on an upwind finite-element technique that uses cell-centred quantities and point implicit schemes. Batina [11] used implicit Gauss-Seidel relaxation scheme for unsteady aerodynamic analysis on unstructured meshes. Hwang and Lin [12] proposed locally implicit TVD schemes on triangular meshes. More recently Batina [13] gives the results of transonic flow around Boeing 747 airplane by application of implicit upwind schemes (PGS and PGJ) on unstructured meshes.

In the present paper, the implicit schemes, Point-Gauss-Jacobi and Point-Gauss-Seidel, are used associated with the Roe and Osher flux methods to improve the explicit Euler/N-S code on unstructured grids [14]. To obtain high-order spatial accuracy, both the linear reconstruction of variables [7] and the variable extrapolation methods [11] (MUSCL

approach) are used. To validate the present code we chose supersonic flow passing a compressible corner as there exists analytical solution for comparison. The results show obvious improvement on the convergence when using an
implicit scheme and the accuracy is also improved by using high-order schemes.

2.Unstructured Grid Generation

Generation of quality unstructured grids founds a basis for the success of flow calculations using this approach. In this work a two-dimensional unstructured grid generator, developed by Peraire using the advancing front technique, is employed to form a set of triangular elements over the whole flow domain. The approach consists of the construction of a background grid and the specification of a boundary condition. This is all required by the advancing front technique. A more detailed description of this mesh generation technique is given in Ref.[14].

3.Mathematical Model of the Euler/N-S Equations

3.1 Euler Equations in 2-D

The Euler equations in a Cartesian system of two spatial coordinates in conservation form can be expressed as:

$$\frac{\partial U}{\partial t}+\frac{\partial F_1}{\partial x}+\frac{\partial F_2}{\partial y}=0 \tag{3.1}$$

The expressions for the unknowns and fluxes are

$$U=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho\varepsilon\end{pmatrix}\qquad F_1=\begin{pmatrix}\rho u\\ \rho u^2+P\\ \rho uv\\ u(\rho\varepsilon+P)\end{pmatrix}\qquad F_2=\begin{pmatrix}\rho v\\ \rho vu\\ \rho v^2+P\\ v(\rho\varepsilon+P)\end{pmatrix} \tag{3.2}$$

where ρ,u,v,P and ε are the density, velocity components in Cartesian coordinates, pressure and specific total energy of the flow, respectively.

Pressure is related to other variables using perfect gas assumption

$$P=(\gamma-1)\rho\left[\varepsilon-\frac{1}{2}\left(u^2+v^2\right)\right] \tag{3.3}$$

where γ is the ratio of the specific heats, i.e $\gamma = Cp / Cv$

The speed of sound c is related to the other variables through

$$c^2=\gamma P/\rho \tag{3.4}$$

3.2 Navier-Stokes Equations in 2-D

The flow of a compressible heat conducting viscous fluid is governed by the full Navier-Stokes equations. These equations represent the conservation of mass, momentum and energy. The N-S equations in non-dimensional form can be expressed as

$$\frac{\partial U}{\partial t}+\frac{\partial F_1}{\partial x}+\frac{\partial F_2}{\partial y}=\frac{\partial G_1}{\partial x}+\frac{\partial G_2}{\partial y} \tag{3.5}$$

In the above equation, the definition of the vectors U and Fi (i=1,2) are the same as given by equation (3.2). The entries for the vectors of viscous fluxes, Gi (i=1,2) are

$$G_1=\frac{1}{Re_\infty}\cdot\begin{pmatrix}0\\ \tau_{xx}\\ \tau_{xy}\\ u\tau_{xx}+v\tau_{xy}-q_x\end{pmatrix}\qquad G_2=\frac{1}{Re_\infty}\cdot\begin{pmatrix}0\\ \tau_{yx}\\ \tau_{yy}\\ u\tau_{yx}+v\tau_{yy}-q_y\end{pmatrix} \tag{3.6}$$

where Re_∞ is the free stream Reynolds number based on the representative length L, i.e

$$Re_\infty=\frac{\rho_\infty u_\infty L}{\mu_\infty} \tag{3.7}$$

τ represents the stress tensor and q the heat flux vector, which are given by the constitutive equations for a Newtonian fluid

$$\begin{aligned}
\tau_{xx}&=2\mu\frac{\partial u}{\partial x}-\frac{2}{3}\mu\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\\
\tau_{xy}&=\tau_{yx}=\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\\
\tau_{yy}&=2\mu\frac{\partial v}{\partial y}-\frac{2}{3}\mu\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\\
q_x&=-k\frac{\partial T}{\partial x}=-\frac{1}{(\gamma-1)}\cdot\frac{\mu}{M_\infty^2\,Pr}\cdot\frac{\partial T}{\partial x}\\
q_y&=-k\frac{\partial T}{\partial y}=-\frac{1}{(\gamma-1)}\cdot\frac{\mu}{M_\infty^2\,Pr}\cdot\frac{\partial T}{\partial y}
\end{aligned} \tag{3.8}$$

where the coefficient of viscosity μ is calculated as

$$\mu=\left(\frac{1+s}{T+s}\right)\cdot T^{3/2}\qquad s=\frac{110.4}{T_\infty} \tag{3.9}$$

4. Solution Algorithm

The solution algorithm employed in the present paper is an implementation of the Finite-Volume method on unstructured grids.

Generally the compressible Navier-Stokes equations are written in the conservation form

$$\frac{\partial U}{\partial t}+\frac{\partial F_1}{\partial x}+\frac{\partial F_2}{\partial y}=\frac{\partial G_1}{\partial x}+\frac{\partial G_2}{\partial y} \tag{4.1}$$

where U is the vector of unknowns and Fi and Gi (i=1,2) denotes the inviscid and viscous fluxes respectively in the direction xi of a Cartesian coordinate system Ox1x2. (see equations (3.2),(3.6)).

The solution domain Ω is discretized by an assembly of triangular elements. Over a single element Ωe, the integral form of (4.1) is

$$\begin{aligned}\int_{\Omega_e}\frac{\partial U}{\partial t}d\Omega &= \int_{\Omega_e}\left(\frac{\partial G_1}{\partial x}+\frac{\partial G_2}{\partial y}-\frac{\partial F_1}{\partial x}-\frac{\partial F_2}{\partial y}\right)d\Omega \\ &= \int_{\Gamma_e} n_i(G_i-F_i)d\Gamma \qquad i=1,2\end{aligned} \tag{4.2}$$

by using the divergence theorem.

Where: n=(n1,n2) denotes the unit vector outward normal to the boundary Γe of control volume Ωe.

[See Figure 4.1]

Assuming a piecewise constant distribution of the unknowns Ue on each element Ωe, Eq(4.2) may be approximated in the form as

$$\Delta U_e = U_e^{n+1}-U_e^n=\frac{\Delta t_e}{\Omega_e}(F^I+G^V) \tag{4.3}$$

Where: U_e^n denotes the valus of Ue at time $t=t^n$

$\Delta t = t^{n+1}-t^n$ is the time step between t^{n+1} and t^n

F^I and G^V denotes the inviscid and viscous contributions respectively.

4.1 Upwinding Flux-Difference Scheme (inviscid contributions)

To evaluate the inviscid numerical flux F^I, two types of approximate Riemann solver

developed by Roe [15] and Osher [16] are applied locally at each interface between cells, assuming a local Riemann problem in the normal direction at interface.

The inviscid contributions F^I are given by

$$F^I = \int_{\Gamma_e} -n_i F_i d\Gamma = -\int_{\Gamma_e} F_n d\Gamma \tag{4.4}$$

and can be evaluated by summing the contributions from each individual element side Γes in turn. In this evaluation the normal flux Fn is replaced by a numerical flux $\tilde{F}_n$, so that

$$F^I = -\sum_{Se} \int_{\Gamma_{es}} \tilde{F}_n d\Gamma \tag{4.5}$$

4.1.1 Numerical Flux of Roe [15]

The numerical flux of Roe can be written in terms of two discrete Riemann states (left and right), with respect to an interface as:

$$F^I(U_L, U_R) = \frac{1}{2}\left[F^I(U_L) + F^I(U_R) - \left|A_{Roe}\right|(U_R - U_L)\right] \tag{4.6}$$

Where: A_{Roe} is the flux Jacobian evaluated using Roe's average fluid states. The absolute value symbols indicate that the absolute value of the eigenvalues were used to evalute A_{Roe}.

Matrix $\left|A_{Roe}\right|$ can be decomposed in terms of its eigenvectors and eigenvalues as

$$\left|A_{Roe}\right| = R|\Lambda|R^{-1} \tag{4.7}$$

Where: R, R^{-1} denote the right and left eigenvectors respectively. Λ is a diagonal matrix containing the eigenvalues λ_i of A_{Roe}. Details about the formula can be found in Ref.[14] and [17].

The mininum allowable value for λ_i is restricted according to the method proposed by Harten [18] and is such that:

$$\left|\lambda_i\right| = \begin{cases} \left|\lambda_i\right| & \left|\lambda_i\right| > \varepsilon_\lambda \\ 0.5\left(\lambda_i^2 / \varepsilon_\lambda + \varepsilon_\lambda\right) & \left|\lambda_i\right| \le \varepsilon_\lambda \end{cases} \tag{4.8}$$

where ε_λ is the eigenvalue limiter.

Explicit Scheme:

An explicit scheme results from an evaluation of the forms in equation (4.6) at time level n. Hence, the formulation using Roe's numerical flux will take the form

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{S_e}\left\{\frac{1}{2}\left[F_e^n + F_r^n - \left|A_{Roe}^n\right|(U_r^n - U_e^n)\right]\right\}\delta s_e \tag{4.9}$$

where the subscripts e and r denote the value at the current element and neighbouring element respectively, δs_e is the length of the side Γe_s . Details about the explicit scheme can be found in Ref.[14].

Implicit Scheme:

If the inviscid contributions are evaluated at time t^{n+1} , equation (4.6) leads to the implicit scheme

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{S_e}\left\{\frac{1}{2}\left[F_e^{n+1} + F_r^{n+1} - \left|A_{Roe}^{n+1}\right|(U_r^{n+1} - U_e^{n+1})\right]\right\}\delta s_e \tag{4.10}$$

Linearization of the equation for the values of the unknowns and fluxs at time level (n+1) in the terms of the time level (n) result in

$$U_e^{n+1} = U_e^n + \Delta U_e \tag{4.11a}$$

$$U_r^{n+1} = U_r^n + \Delta U_r \tag{4.11b}$$

$$F_e^{n+1} = F_e^n + A_e^n \Delta U_e \tag{4.11c}$$

$$F_r^{n+1} = F_r^n + A_r^n \Delta U_r \tag{4.11d}$$

Replacing the above expressions into equation (4.10), result in

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^n + A_e^n\Delta U_e + F_r^n + A_r^n\Delta U_r - \left|A_{Roe}^n\right|(U_r^n + \Delta U_r - U_e^n - \Delta U_e)\right]\right\}\delta s_e$$

It can be rearranged as

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^n + F_r^n - \left|A_{Roe}^n\right|(U_r^n - U_e^n)\right]\right\}\delta s_e$$

$$-\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[A_e^n\Delta U_e + A_r^n\Delta U_r - \left|A_{Roe}^n\right|(\Delta U_r - \Delta U_e)\right]\right\}\delta s_e \tag{4.13}$$

The first term on the right hand side of the above equation is equivalent to the right hand side of the explicit formulation given by equation (4.9). Also

$$\sum_{Se}\left[A_e^n\Delta U_e\right]\delta s_e = \sum_{Se}\Delta F_e\delta s_e = 0 \tag{4.14}$$

Denoting the right hand side of equation(4.9) by RHS_{exp} and using equation (4.14), then equation (4.13) can be written as

$$\Delta U_e = RHS_{exp} - \frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[\left(A_r^n - \left|A_{Roe}^n\right|\right)\Delta U_r\right]\right\}\delta s_e$$
$$-\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left[\frac{1}{2}\left|A_{Roe}^n\right|\Delta U_e\right]\delta s_e \qquad (4.15)$$

This equation can be rearranged as

$$\left[I + \frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left|A_{Roe}^n\right|\delta s_e\right]\Delta U_e = RHS_{exp} - \frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left[\left(A_r^n - \left|A_{Roe}^n\right|\right)\Delta U_r\right]\delta s_e \qquad (4.16)$$

The above system of equations can be solved in each time step, using a Point-Gauss-Jacobi procedure given by

$$\left[I + \frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left|A_{Roe}^n\right|\delta s_e\right]\Delta U_e^{n+1} = RHS_{exp}^n - \frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left[\left(A_r^n - \left|A_{Roe}^n\right|\right)\Delta U_r^n\right]\delta s_e \qquad (4.17)$$

Alternatively, one can use the latest available value for the neighbouring elements and arrive at a Point-Gauss-Seidel scheme. In this case, the linearization is only performed for the values and fluxes at the current element. That is to say equation (4.10) may be written as

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^{n+1} + F_r^* - \left|A_{Roe}^*\right|(U_r^* - U_e^{n+1})\right]\right\}\delta s_e \qquad (4.18)$$

Replacing from equation (4.11a),(4.11c) and (4.14) into the above equation and rearranging results in

$$\left[I + \frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left|A_{Roe}^*\right|\delta s_e\right]\Delta U_e^{n+1} = -\frac{\Delta t_e}{2\Omega_e}\sum_{S_e}\left[F_e^n + F_r^* - \left|A_{Roe}^*\right|\left(U_r^* - U_e^n\right)\right]\delta s_e \qquad (4.19)$$

where the terms denoted by an asterisk are evaluted using the latest available values of the variables.

4.1.2 Numerical Flux of Osher [16]

The numerical flux of Osher in terms of left and right states is defined as

$$F(U_L, U_R) = \frac{1}{2}\left[F(U_L) + F(U_R) - \int_{U_L}^{U_R}\left|A_{Osher}\right|dQ\right] \qquad (4.20)$$

where the fluxes at the right and left states are calculated in the same way as in Roe. The integration in the above expression is performed by the procedure given in Ref. [16][17].

Considering 2-D flow, there are four characteristic fields of which the two corresponding to $\lambda_{2,3}$ are identical. The invariant functions are:

For $\lambda_1 = U_n + c$

$$\Psi_2^1 = U_n - \frac{2c}{(\gamma-1)}$$

$$\Psi_3^1 = P/\rho^\gamma$$

$$\Psi_4^1 = V_t$$

For $\lambda_{2,3} = U_n$ (4.21)

$$\Psi_1^{2,3} = P$$

$$\Psi_4^{2,3} = U_n$$

For $\lambda_4 = U_n - c$ (4.22)

$$\Psi_1^4 = U_n + \frac{2c}{(\gamma-1)}$$

$$\Psi_2^4 = P/\rho^\gamma$$

$$\Psi_3^4 = V_t$$

(4.23)

In the above expressions, Un and Vt are the normal and tangential velocities to the cell side defined as

$$U_n = un_x + vn_y$$

$$V_t = -un_{y+vn_x}$$

(4.24)

The local speed of sound is given by equation (3.4).

The first and fourth characteristic fields are genuinely non-linear and the second and third are linearly degenerate. The path of integration in the state space is as shown in Figure 4.2.

By writing the invariant functions along each subpath, we obtain eight equations which can be solved to get the eight variables that define the value of intermediate points. They are

$$\rho_{i-2/3}^{\left(\frac{\gamma-1}{2}\right)} = \frac{\left[\frac{\gamma-1}{2}((U_n)_i - (U_n)_{i-1}) + c_i + c_{i-1}\right]}{c_{i-1}\left[1+\left(\frac{P_i}{P_{i-1}}\right)^{\frac{1}{2\gamma}}\left(\frac{\rho_{i-1}}{\rho_i}\right)^{\frac{1}{2}}\right]}\rho_{i-1}^{\left(\frac{\gamma-1}{2}\right)}$$

(4.25a)

$$\rho_{i-1/3}^{\left(\frac{\gamma-1}{2}\right)} = \frac{\left[\frac{\gamma-1}{2}((U_n)_i - (U_n)_{i-1}) + c_i + c_{i-1}\right]}{c_i\left[1+\left(\frac{P_{i-1}}{P_i}\right)^{\frac{1}{2\gamma}}\left(\frac{\rho_i}{\rho_{i-1}}\right)^{\frac{1}{2}}\right]}\rho_i^{\left(\frac{\gamma-1}{2}\right)} \tag{4.25b}$$

$$P_{i-2/3} = P_{i-1/3} = P_{i-1}\left(\frac{\rho_{i-2/3}}{\rho_{i-1}}\right)^{\gamma} = P_i\left(\frac{\rho_{i-1/3}}{\rho_i}\right)^{\gamma} \tag{4.25c}$$

$$(U_n)_{i-2/3} = (U_n)_{i-1} - \frac{2}{\gamma-1}(c_{i-1} - c_{i-2/3}) = (U_n)_{i-1/3} =$$

$$(U_n)_i + \frac{2}{\gamma-1}(c_i - c_{i-1/3})$$

$$(V_t)_{i-2/3} = (V_t)_{i-1} \tag{4.25d}$$

$$(V_t)_{i-1/3} = (V_t)_i \tag{4.25e}$$

(4.25f)

The sonic points (denoted by a prime) are determined by a similar procedure. There is no sonic point on the second subpath. For the first and third subpaths, there are

$$(U_n')_{i-2/3} = \frac{\gamma-1}{\gamma+1}\left((U_n)_{i-1} - \frac{2}{\gamma-1}c_{i-1}\right) \tag{4.26a}$$

$$\rho_{i-2/3}'^{\left(\frac{\gamma-1}{2}\right)} = \left(\frac{-(U_n')_{i-2/3}}{c_{i-1}}\right)\rho_{i-1}^{\left(\frac{\gamma-1}{2}\right)} \tag{4.26b}$$

$$P_{i-2/3}' = P_{i-1}\left(\frac{\rho_{i-2/3}'}{\rho_{i-1}}\right)^{\gamma} \tag{4.26c}$$

$$(V_t')_{i-2/3} = (V_t)_{i-1} \tag{4.26d}$$

$$(U_n')_{i-1/3} = \frac{\gamma-1}{\gamma+1}\left((U_n)_i - \frac{2}{\gamma-1}c_i\right) \tag{4.26e}$$

$$\rho_{i-1/3}'^{\left(\frac{\gamma-1}{2}\right)} = \left(\frac{-(U_n')_{i-1/3}}{c_i}\right)\rho_i^{\left(\frac{\gamma-1}{2}\right)} \tag{4.26f}$$

$$P_{i-1/3}' = P_i\left(\frac{\rho_{i-1/3}'}{\rho_i}\right)^{\gamma} \tag{4.26g}$$

$$(V_t')_{i-1/3} = (V_t)_i \tag{4.26h}$$

Having determined the intermediate and the sonic points, the integration can be carried out using the form in Ref.[16].

Explicit Scheme:

The explicit time stepping formulation can now be written as

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^n + F_r^n - \int_{U_e^n}^{U_r^n}\left|A_{Osher}\right|dQ\right]\right\}\delta s_e \tag{4.27}$$

Implicit Scheme:

The numerical flux of Osher in fully implicit form can be written as

$$F^{n+1}(U_L,U_R) = \frac{1}{2}\left[F^{n+1}(U_L) + F^{n+1}(U_R) - \int_{U_L^{n+1}}^{U_R^{n+1}}\left|A_{Osher}\right|dQ\right] \tag{4.28}$$

Linearization of the type given by equation (4.11a)--(4.11d) requires an evaluation of the Jacobian matrix A , which for this numerical flux proves to be a tedious procedure.

Another approach, which leads to a much simpler formulation, is used to determine the left hand side of the implicit system of equations by replacing the numerical flux of Osher with a flux vector splitting.

Considering the flux vector splitting scheme of Steger and Warming [19] , which can be expressed as

$$F(U_L,U_R) = F^+(U_L) + F^-(U_R) \tag{4.29}$$

For an implicit formulation, this equation can be linearized and written as

$$F^{n+1}(U_L,U_R) = F^n(U_L,U_R) + \left(\frac{\partial F^+(U_L)}{\partial U_L}\right)^n \Delta U_L + \left(\frac{\partial F^-(U_R)}{\partial U_R}\right)^n \Delta U_R \tag{4.30}$$

The Jacobian matrices in the above equation can be approximated by

$$\frac{\partial F^+(U_L)}{\partial U_L} = \left(\frac{\partial F(U_L)}{\partial U_L}\right)^+ = A_L^+ \tag{4.31a}$$

$$\frac{\partial F^-(U_R)}{\partial U_R} = \left(\frac{\partial F(U_R)}{\partial U_R}\right)^- = A_R^- \tag{4.31b}$$

Substituting these expressions into equation (4.30) results in the following linearization

$$F^{n+1}(U_L,U_R) = F^n(U_L,U_R) + A_L^+\Delta U_L + A_R^-\Delta U_R \tag{4.32}$$

The jacobian matrices in the above expressions are defined as

$$A^+ = R\Lambda^+R^- \tag{4.33a}$$

$$A^- = R\Lambda^-R^- \tag{4.33b}$$

Where Λ^+ and Λ^- are the diagonal matrices of positive and negative eigenvalues respectively, i.e

$$\lambda_i^+ = \max(0,\lambda_i)$$

$$\lambda_i^- = \min(0, \lambda_i)$$

The definition of R, R^- are the same as those in Roe's flux.

Now the term at time level n on the right hand side of the above equation (4.32) is replaced by the numerical flux of Osher in its explicit form. Hence the linearised implicit finite volume formulation will be given as

$$\Delta U_e = RHS_{\exp} - \frac{\Delta t_e}{\Omega_e}\sum_{Se}\left[A_e^+\Delta U_e + A_r^-\Delta U_r\right]\delta s_e \tag{4.34}$$

where the term $RHS_{\exp}$ represents the right hand side of equation (4.27).

Taking all the terms depending on ΔU_e to the left hand side results in the following Point-Gauss-Jacobi iterative procedure

$$\left[I + \frac{\Delta t_e}{2\Omega_e}\sum_{Se} A_e^+\delta s_e\right]\Delta U_e^{n+1} = RHS_{\exp}^n - \frac{\Delta t_e}{\Omega_e}\sum_{Se}\left[A_r^-\Delta U_r^n\right]\delta s_e \tag{4.35}$$

Similiar to that of the numerical flux of Roe, an alternative Point-Gauss-Seidel formulation can be obtained by using the latest available values (denoted by asterisk) to determine the fluxes at the neighbouring elements. In this case equation (4.34) is written as

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^n + F_r^* - \int_{U_e^n}^{U_r^*}|A_{Osher}|dQ\right]\right\}\delta s_e - \frac{\Delta t_e}{\Omega_e}\sum_{Se}\left[A_e^+\Delta U_e\right]\delta s_e \tag{4.36}$$

On rearranging this equation and factorising the terms containing ΔU_e can be written as

$$\left[I + \frac{\Delta t_e}{2\Omega_e}\sum_{Se} A_e^+\delta s_e\right]\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\sum_{Se}\left\{\frac{1}{2}\left[F_e^n + F_r^* - \int_{U_e^n}^{U_r^*}|A_{Osher}|dQ\right]\right\}\delta s \tag{4.37}$$

4.2 Central-Difference Scheme (viscous contributions)

Differing from the inviscid terms, which are discretised using the upwinding scheme, the viscous terms are always discretised using a central difference type scheme because it plays a parabolic or elliptic part in the compressible flow equations.

The definition of the viscous terms is given in equation (3.6), for a cell side s , the numerical viscous flux is calculated using the average value of the variables in the left and right elements, that is

$$u_s = 0.5(u_L + u_R)$$
$$v_s = 0.5(v_L + v_R)$$
$$T_s = 0.5(T_L + T_R) \tag{4.38}$$

The required values for μ is obtained by using Ts in equation (3.9). Hence the viscous contributions to equation (4.3) are

$$G_1^V = \frac{1}{\mathrm{Re}_\infty} \cdot \begin{pmatrix} 0 \\ (\tau_{xx})_s \\ (\tau_{xy})_s \\ u_s(\tau_{xx})_s + v_s(\tau_{xy})_s - (q_x)_s \end{pmatrix} \tag{4.39a}$$

$$G_2^V = \frac{1}{\mathrm{Re}_\infty} \cdot \begin{pmatrix} 0 \\ (\tau_{yx})_s \\ (\tau_{yy})_s \\ u_s(\tau_{yx})_s + v_s(\tau_{yy})_s - (q_y)_s \end{pmatrix} \tag{4.39b}$$

The normal viscous fluxes with respect to a side with outward normal vector are therefore written as

$$(G_1^V)_n = 0$$

$$(G_2^V)_n = \frac{1}{\mathrm{Re}_\infty}\left[(\tau_{xx})_s n_x + (\tau_{yx})_s n_y\right]$$

$$(G_3^V)_n = \frac{1}{\mathrm{Re}_\infty}\left[(\tau_{xy})_s n_x + (\tau_{yy})_s n_y\right]$$

$$(G_4^V)_n = \frac{1}{\mathrm{Re}_\infty}\left\{\left[u_s(\tau_{xx})_s + v_s(\tau_{xy})_s - (q_x)_s\right]n_x + \left[u_s(\tau_{yx})_s + v_s(\tau_{yy})_s - (q_y)_s\right]n_y\right\} \tag{4.40}$$

where subscripts denote an evaluation at the sides.

Referring to equation (3.8), the viscous stresses are determined from

$$(\tau_{xx})_s = 2\mu_s\left(\frac{\partial u}{\partial x}\right)_s - \frac{2}{3}\mu_s\left(\left(\frac{\partial u}{\partial x}\right)_s + \left(\frac{\partial v}{\partial y}\right)_s\right) \tag{4.41a}$$

$$(\tau_{xy})_s = (\tau_{yx})_s = \mu_s\left(\left(\frac{\partial u}{\partial y}\right)_s + \left(\frac{\partial v}{\partial x}\right)_s\right) \tag{4.41b}$$

$$(\tau_{yy})_s = 2\mu_s\left(\frac{\partial v}{\partial y}\right)_s - \frac{2}{3}\mu_s\left(\left(\frac{\partial u}{\partial x}\right)_s + \left(\frac{\partial v}{\partial y}\right)_s\right) \tag{4.41c}$$

It is clear that the evaluation of the viscous contributions to the right-hand side of (4.3) requires a knowledge of the first derivatives of quantities, such as the velocity components (u,v) and the temporature T.

4.2.1 Method 1: Arithematic average

The necessary first derivatives can be obtained by the same method as Ref.[14] in which the gradients are determined by Green's theorem along the path including side s [see Figure 4.3].

Based on the Green's theorem, one can obtain the gradient from

$$\int_\Omega \frac{\partial u}{\partial x} d\Omega = \oint_\Gamma u \cdot n_x d\Gamma \tag{4.42}$$

(Here we take a scalar variable u with respect to x at an element side s for example.)

Assuming a constant distribution of the gradient over this domain (n1→n4→n2→n3→n1), the left side becomes

$$\left(\frac{\partial u}{\partial x}\right)_s \cdot \Omega$$

where Ω is the area of this domain.

The right side integration is evaluated along the path s , which can be represented as 4-subpath. Assuming each variable is constant along each subpath and is replaced by an average value, i.e

$$\oint_\Gamma u \cdot n_x d\Gamma = \int_{N1}^{N4} (u \cdot \bar{n})_{14} ds + \int_{N4}^{N2} (u \cdot \bar{n})_{42} ds + \int_{N2}^{N3} (u \cdot \bar{n})_{23} ds + \int_{N3}^{N1} (u \cdot \bar{n})_{31} ds$$
$$u_{ij} = 0.5(u_i + u_j) \tag{4.43}$$

thus the gradient $\partial/\partial x$ is completely determined by writting

$$\left(\frac{\partial u}{\partial x}\right)_s = \frac{1}{2\Omega}\left[(u_1+u_4)(y_4-y_1)+(u_4+u_2)(y_2-y_4)+(u_2+u_3)(y_3-y_2)+(u_3+u_1)(y_1-y_3)\right] \tag{4.44}$$

For the gradient $\partial/\partial y$ using the similar procedure, we have

$$\left(\frac{\partial u}{\partial y}\right)_s = \frac{1}{2\Omega}\left[(u_1+u_4)(x_1-x_4)+(u_4+u_2)(x_4-x_2)+(u_2+u_3)(x_2-x_3)+(u_3+u_1)(x_3-x_1)\right] \tag{4.45}$$

In the above expression, it requires the knowledge of the value of variables at node points n3 and n4. As we can see later, this will also be needed for the high-order construction.

We can calculate this value simply from the average of flow variables for all elements surrounding the node, i.e

$$u_i = \frac{1}{L_i}\sum_{e=1}^{L_i} u_e \tag{4.46}$$

Unfortunately, for a strongly stretched grid, in a region where area of an element changes suddenly, there will arise some errors when using the above formulation. Hence we have atempted to use a
weighted average method instead of this simple average method.

4.2.2 Method 2: Weighted average

A more accurate gradient estimation at side s can be obtained by weighted average

$$\nabla f_s = \omega_L \nabla f_L + \omega_R \nabla f_R \tag{4.47}$$

where f is physical quantity (i.e u,v, or P);

and ω_L is the area ratio of right triangular (n1→n4→n2) to quadrilateral (n1→n4→n2→n3), and ω_R is defined similarly.

∇f_L and ∇f_R can be evaluated separately by the Green's theorem along its integral path, i.e for the left side integral path this is (n1→n2→n3→n1), and for right side integral path this is (n1→n4→n2→n1).

Also a more accurate node value estimation can be obtained using a weighted average (see Fig.4.4).

$$f_{node} = \sum_{j=1}^{J} f_j \omega_j = \frac{\sum_{j=1}^{J} [f_j (A - A_j)]}{\sum_{j=1}^{J} A_j}$$

$$A = \sum_{j=1}^{J} A_j \tag{4.48}$$

This implies that the smaller the area Aj (i.e the nearer the point fj to node), the greater it affects the value of that node.

The above expressions define the explicit evaluation of the viscous contributions. The complete formulation of the explicit scheme is obtained by combining the viscous numerical fluxes and the inviscid numerical fluxes into the right hand side of equation (4.3).

Implicit Scheme:

A point implicit time integration scheme can be obtained by linearising the viscous contributions as

$$G_S^{n+1} = G_S^n + B_S^n \Delta U_e \tag{4.49}$$

where B_S^n is the Jacobian matrix of the transformation.

Replacing the above linearization into the general finite volume formulation and rearranging the terms results in

$$\left\{LHS_{iv} - \frac{\Delta t_e}{\Omega_e}\sum_{Se} B_S^n \delta s_e\right\}\Delta U_e = RHS_{iv} + \frac{\Delta t_e}{\Omega_e}\sum_{Se} G_e^n \delta s_e \tag{4.50}$$

where LHSiv and RHSiv denote the inviscid contributions to the left hand side and right hand side respectively.

The inviscid contributions, depending on the type of the numerical flux (i.e Roe's or Osher's) are given in equations (4.17),(4.19) and (4.35),(4.37) respectively.

In order to calculate matrix B_S^n , we use another method called the variational recovery process to obtain the first derivatives. In this process the derivatives are represented in a piecewise linear manner over the computational domain, i.e for variable f , we have

$$f = \sum_e f_e P_e \qquad \frac{\partial f}{\partial x_i} = \sum_I \left.\frac{\partial f}{\partial x_i}\right|_I N_I \tag{4.51}$$

where Pe is the piecewise constant shape function associated with element e and Ni is the piecewise linear or bilinear shape function associated with node, I, with the nodes placed at the vertices of the elements. The nodal values of the derivatives are obtained from the integral statement

$$\int_\Omega \frac{\partial f}{\partial x_i} N_k d\Omega = \int_\Gamma n_i f N_k d\Gamma - \int_\Omega f \frac{\partial N_k}{\partial x_i} d\Omega \tag{4.52}$$

By inserting the approximation (4.51), the result is that

$$\left.\frac{\partial f}{\partial x_i}\right|_k = \frac{1}{(M_L)_k}\left[\int_\Gamma n_i f N_k d\Gamma - \sum_e \int_{\Omega_e} f_e \frac{\partial N_k}{\partial x_i} d\Omega\right] \tag{4.53}$$

where the summation appearing in this expression extends over those elements e which are associated with node k , and M_L denotes the standard lumped mass matrix. For a general mesh it is thus possible to write

$$\left.\frac{\partial f}{\partial x_1}\right|_k = f_{x1k} - \sum_e \tilde{b}_{ke} f_e$$

$$\left.\frac{\partial f}{\partial x_2}\right|_k = f_{x2k} - \sum_e \tilde{c}_{ke} f_e \tag{4.54}$$

where f_{x1k} and f_{x2k} denote the boundary terms and

$$\tilde{b}_{ke} = \frac{1}{(M_L)_k}\int_{\Omega_e} \frac{\partial N_k}{\partial x_1} d\Omega$$

$$\tilde{c}_{ke} = \frac{1}{(M_L)_k}\int_{\Omega_e} \frac{\partial N_k}{\partial x_2} d\Omega \tag{4.55}$$

Now consider the linearization of the viscous terms for element E. In particular, consider the contribution from a side s which has associated nodes M and N, then

$$\left(\frac{\partial f}{\partial x_1}\right)_S = \frac{1}{2}\left[f_{x1M} - \sum_{e\neq E}\tilde{b}_{Me}f_e - \tilde{b}_{ME}f_E + f_{x1N} - \sum_{e\neq E}\tilde{b}_{Ne}f_e - \tilde{b}_{NE}f_E\right] \tag{4.56a}$$

$$\left(\frac{\partial f}{\partial x_2}\right)_S = \frac{1}{2}\left[f_{x2M} - \sum_{e\neq E}\tilde{c}_{Me}f_e - \tilde{c}_{ME}f_E + f_{x2N} - \sum_{e\neq E}\tilde{c}_{Ne}f_e - \tilde{c}_{NE}f_E\right] \tag{4.56b}$$

and an evaluation at time t^{n+1} can be obtained in the form

$$\left(\frac{\partial f}{\partial x_1}\right)_S^{n+1} = \left(\frac{\partial f}{\partial x_1}\right)_S^{n} - b_{SE}\Delta f_E$$

$$\left(\frac{\partial f}{\partial x_2}\right)_S^{n+1} = \left(\frac{\partial f}{\partial x_2}\right)_S^{n} - c_{SE}\Delta f_E \tag{4.57}$$

where

$$b_{SE} = 0.5(\tilde{b}_{ME} + \tilde{b}_{NE})$$
$$c_{SE} = 0.5(\tilde{c}_{ME} + \tilde{c}_{NE}) \tag{4.58}$$

Elements of matrix Bs are determined by using equation (4.57) as adapted for velocity components and temperature and substituting the resulting expressions into equation (4.40).

Detailes of the derivation can be found in Ref. [17]. The result is that

$B_{11} = B_{12} = B_{13} = B_{14} = 0 \quad B_{21} = (\phi_{1S}u_{1E}^n + \phi_{2S}u_{2E}^n)/\rho_E^n$

$B_{22} = -\phi_{1S}/\rho_E^n \qquad B_{23} = -\phi_{2S}/\rho_E^n \qquad B_{24} = 0$

$B_{31} = (\phi_{3S}u_{1E}^n + \phi_{4S}u_{2E}^n)/\rho_E^n \qquad B_{32} = -\phi_{3S}/\rho_E^n \qquad B_{33} = -\phi_{4S}/\rho_E^n$

$B_{34} = 0$

$$B_{41} = -\gamma\phi_{7S}\left[(u_{1E}^2 + u_{2E}^2)^n - (\rho E)_E^n\right]/(\rho_E^n)^2 + u_{1E}^n\phi_{5S}/\rho_E^n + u_{2E}^n\phi_{6S}/\rho_E^n$$

$$B_{42} = -\phi_{5S}/\rho_E^n + \gamma\phi_{7S}u_{1E}^n/\rho_E^n \qquad B_{43} = -\phi_{6S}/\rho_E^n + \gamma\phi_{7S}u_{2E}^n/\rho_E^n$$

$$B_{44} = -\gamma\phi_{7S}/\rho_E^n \tag{4.59}$$

where

$$\phi_{1S} = \alpha(4n_1b_{SE}/3 + n_2c_{SE})$$
$$\phi_{2S} = \alpha(-2n_1c_{SE}/3 + n_2b_{SE})$$
$$\phi_{3S} = \alpha(n_1c_{SE} - 2n_2b_{SE}/3)$$
$$\phi_{4S} = \alpha(n_1b_{SE} + 4n_2c_{SE}/3)$$
$$\phi_{5S} = (u_{1S}\phi_{1S} + u_{2S}\phi_{3S})$$
$$\phi_{6S} = (u_{1S}\phi_{2S} + u_{2S}\phi_{4S})$$
$$\phi_{7S} = \alpha(n_1b_{SE} + n_2c_{SE})/\mathrm{Pr}$$
$$\alpha = \mu_S/\mathrm{Re} \tag{4.60}$$

In these expressions Re and Pr denote Reynolds and Prandtl numbers respectively.

Since the viscous terms on the right hand side of equation (4.50) are evaluated at time level n, the procedure is equivalent to a Point-Gauss-Jacobi iteration for the viscous terms. It also can use the most recent values to determine the viscous contributions to the right hand side and result in the Point-Gauss-Seidel scheme. For a P-G-S scheme, linearization given by equation (4.49) is replaced by

$$G_S^{n+1} = G_S^* + B_S^* \Delta U_e \tag{4.61}$$

where, as before, an asterisk represents an evaluation using the latest available values. Apart from using the latest values of variables, details of treatment will be similar to that of the P-G-J iteration, i.e the resulting equations can be obtained by substituting the superscript n with * for the elements surrounding the current element in equations (4.50),(4.57).

To avoid complexity one can chose B_S^n instead of B_S^*, so that the evaluation of the matrix Bs remains unchanged. Otherwise, in order to have a compatible Gauss-Seidel formulation, evaluation of gradients in G* must be re-calculated as soon as the relating unknowns are updated. That is

$$\nabla f_S^* = \omega_L \nabla f_L^* + \omega_R \nabla f_R^* \tag{4.62}$$

5.The Treatment of the Boundary Conditions

The treatment of the boundary conditions of a multi-dimensional flow can be performed by analogy with the one-dimensional case. The number and type of conditions at a boundary of a multi-dimensional domain are defined by the eigenvalue spectrum of the Jacobians associated with the normal to the boundary. This defines locally quasi-one-dimensional propagation properties.

5.1 Boundary Condition for the Euler Equations

All the boundary conditions used for the exterior boundary are based on the method of characteristics. For the wall boundary, both the method of characteristics and extrapolation from the interior flow field are used.

At the exterior boundary, we wish to minimize the reflection of outgoing disturbances. Consider the flow normal to this boundary. Assuming it to be locally one-dimensional, we introduce the fixed and extrapolated Riemann invariants according to 1-D Riemann

relations

$$R_\infty = q_\infty \vec{n} - 2c_\infty / (\gamma - 1)$$
$$R_e = q_e \vec{n} + 2c_e / (\gamma - 1) \qquad (5.1)$$

corresponding to incoming and outgoing characteristics. The normal velocity and local speed of sound may thus be determined by

$$q \cdot \vec{n} = 0.5(R_e + R_\infty)$$
$$c = 0.25(\gamma - 1)(R_e - R_\infty) \qquad (5.2)$$

Two other independent conditions are needed to complete the definition of the outer boundary condition.These are given by the values of tangential velocity and entropy. For an outer flow boundary these are extrapolated from the interior values, whereas for an inflow boundary they are set equal to their freestream values.

At the inner boundary, i.e a solid wall, the appropriate boundary conditions are the wallslip boundary condition, which means that the normal component of the velocity to the wall is zero. This can be implemented numerically in two ways, as follows.

(a) Strong Formulation

To specify the values of the unknowns, a set of imaginary elements is introduced inside the wall boundary. The values for the variables for these elements are set so that the average interface value satisfy the tangency condition. i.e Un=0

The values of the other two parameters (density and pressure) are taken to be the same as the values inside the domain.

(b) Weak Formulation

Using the velocity tangency condition in equations Fn

$$F_n = \begin{pmatrix} \rho U_n \\ \rho u U_n + P n_x \\ \rho v U_n + P n_y \\ U_n(\rho\varepsilon + P) \end{pmatrix} \qquad (5.3)$$

i.e Un=0, then the fluxes at the wall are obtained in the following expression

$$F_W = \begin{pmatrix} 0 \\ P_W n_x \\ P_W n_y \\ 0 \end{pmatrix} \qquad (5.4)$$

It is necessary then to determine the pressure at the wall. This also means only the pressure contribution remains at the walls.

Various methods can be applied in order to obtain the wall pressure.

Method 1: Characteristic Relations

Variables other than the normal velocity, in particular the tangential velocity, the pressure and the density, can be obtained from the interior domain by applying the Riemann Invariant. Through those relations we can find out the pressure at the wall.

Method 2: Extrapolation

This is a simple and efficient approach, whereby an extrapolation of generally 1st-order or 2nd-order, is applied from neighbouring elements to the wall. For the explicit scheme, the numerical fluxes of equation (5.4) can be imposed directly at the sides on the wall.

For the implicit scheme, however, further care is needed. The Jacobian matrix of the transformation Ue ---> Fw must also be used on the left hand side of the implicit equation system. It is

$$A_W = (\gamma - 1)\begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{u^2+v^2}{2}n_x & -un_x & -vn_x & -n_x \\ \frac{u^2+v^2}{2}n_y & -un_y & -vn_y & -n_y \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.5}$$

As an example the implicit formulation for an element adjacent to the wall using the numerical flux of Roe is explained. Equation (4.10) is now written as

$$\Delta Ue = -\frac{\Delta t_e}{\Omega_e}\left(\sum_{Se \neq W} \left\{ \frac{1}{2}\left[F_e^{n+1} + F_r^{n+1} - \left| A_{Roe}^{n+1} \right| (U_r^{n+1} - U_e^{n+1}) \right] \right\} \delta s_e + F_W^{n+1} \delta_W \right) \tag{5.6}$$

Using the linearizations

$$F_W^{n+1} = F_W^n + A_W^n \Delta U_e \tag{5.7}$$

it can be re-written as (in form of P-G-S)

$$\Delta U_e = -\frac{\Delta t_e}{\Omega_e}\left(\sum_{Se \neq W} \left\{ \frac{1}{2}\left[F_e^n + F_r^* - \left| A_{Roe}^* \right| (U_r^* - U_e^n) + A_e^n \Delta U_e + \left| A_{Roe}^* \right| \Delta U_e \right] \right\} \delta s_e + (F_W^n + A_W^n \Delta U_e)\delta_W \right) \tag{5.8}$$

The above equation, upon taking terms involving ΔUe , can be written as

$$\left[I + \frac{\Delta t_e}{\Omega_e} [\sum_{Se \neq W} (A_e^n + \left| A_{Roe}^* \right|) \delta s_e + 2 A_W^n \delta_W] \right] \Delta U_e =$$

$$- \frac{\Delta t_e}{\Omega_e} \left(\sum_{Se \neq W} 0.5 * \left[F_e^n + F_r^* - \left| A_{Roe}^* \right| (U_r^* - U_e^n) \right] \delta s_e + F_W^n \delta_W \right) \tag{5.9}$$

The implicit formulation for an element adjacent to wall using the numerical flux of Osher is carried out in a similar way.

5.2 Boundary Condition for the N-S Equations

The formulation of the exterior boundary is similar to that given for the Euler equations. For the inner boundary, i.e the solid wall, the boundary condition specific to the Navier-Stokes equations is the no-slip wall condition which means the relative velocity between the fluid and the solid wall is zero. Assuming a fixed wall, all the velocity components at the wall are taken to be zero. For an isothermal wall, the temperature is fixed at the wall temperature. For an adiabatic wall, the heat flux is zero. In this case the temperature at the boundary side is taken to be the same as the temperature at the adjacent element inside the domain. For the pressure, the boundary layer assumption $\partial P/\partial n = 0$, is employed. Other variables, in particular the density, can be determined from the equation of state.

6. High-Order Resolutions

High-order accurate evaluations of the numerical flux are not straight forward on unstructured grids though some successes have been reported. Here we use two method to construct high-order resolutions. One is called linear reconstruction of variables proposed by Barth & Jesperson, which is an extension of the MUSCL concept of van Leer [20] to unstructured grids. The other is directly using variable extrapolation (MUSCL) reported by Batina [21].

6.1 Linear Reconstruction of Variables

Details about this method can be found in Ref. [14][17]. Here we only make a brief explanation.

The high-order accuracy variable f over an arbitrary element can beobtained by linear

reconstruction (see Fig 6.1)

$$f(x,y) = f(x_e, y_e) + \nabla f \cdot \vec{r} \tag{6.1}$$

where r is the position vector of point (x,y) with respect to some reference point e .

Generally the linear reconstruction given by equation (6.1) may exhibit nonphysical oscillations in the form of overshoots or undershoots near the flow discontinuities. To prevent this a limiter is applied to the higher order correction term

$$f(x,y) = f(x_e, y_e) + \phi \nabla f \cdot \vec{r} \tag{6.2}$$

Normally during calculation, the centroid of the element is chosen as the reference point, the gradient vector is assumed to be constant over the element. The element limiter, ϕ , is determined in such a way that the value of f over the element does not exceed the extrema of the cell-averaged values of f in the surrounding elements.

6.2 Variable Extrapolation (MUSCL approach)

Similar to that used on a structured grid, the variable extrapolation, i.e MUSCL (Monotone Upstream-centred Scheme for Conservation Laws) approach, is also used to determine the resulting accuracy of the scheme. It was found [22] that use of the primitive variable $q = [\rho, u, v, P]^T$ in the extrapolation is more robust than the use of conserved variables Q .

For two given triangles j and k for example, and considering the diagram in Fig 6.2a

A k-parameter family of high-order schemes can be written as

$$q_S^L = q_j + \left\{ \frac{s_1}{4} [(1 - ks_1)\Delta_- + (1 + ks_1)\Delta_+] q \right\}_j \tag{6.3}$$

where $\Delta_+ = q_k - q_j \qquad \Delta_- = q_j - q_i$

$$q_S^R = q_k - \left\{ \frac{s_2}{4} [(1 + ks_2)\Delta_- + (1 - ks_2)\Delta_+] q \right\}_k \tag{6.4}$$

where $\Delta_+ = q_l - q_k \qquad \Delta_- = q_k - q_j$

In equation (6.3) and (6.4), qj and qk are the vectors of primitive variables at the centroids of triangles j and k, respectively. And qi ,ql the vector of primitive variables at the node i,l are determined by the weighted average of the flow variables in the triangles surrounding node i,l.

The parameter k controls a family of difference schemes by appropriately weighting Δ_+ and Δ_- . On structured meshes, it is easy to show that k=-1 corresponds to a full-upwind second order scheme, k=0 yields Fromm's scheme, and k=1 yields a central difference scheme. The value k=1/3 leads to a third order accuracy upwind-biased scheme.

The parameter s1,s2 serves to limit high-order terms in the extrapolation in order to avoid oscillatious in the solutions at discontinuities such as shock waves. According to van Albada et el [23], the limiting is implemented by locally modifying the difference values in the extrapolation to ensure monotone extrapolation as

$$s_1 = s_2 = \frac{2\Delta_+ q \Delta_- q + \delta}{(\Delta_+ q)^2 + (\Delta_- q)^2 + \delta} \tag{6.5}$$

where δ is a small number preventing division by zero in regions of null gradients.

On highly stretched meshes, the formula for Δ_+ (equation (6.3a)) is modified to be

$$\Delta_+ = [2a/(a+b)](q_k - q_j) \tag{6.6}$$

For Δ_- in equation (6.4b) is also modified to be

$$\Delta_- = [2b/(a+b)](q_k - q_j) \tag{6.7}$$

where a and b are the distances from the midpoint of an edge to the centroids of triangles j and k, respectively, as shown in Fig 6.2b.

This formula weights the flow variables in the extrapolation formula, differently to account for the streching of the mesh. For example, by substituting equation (6.6) into equation (6.3) and letting k=0, s1=1 yields

$$q_S^L = \frac{b}{a+b} q_j + \frac{a}{a+b} q_k \tag{6.8}$$

For the case shown in Fig 6.2b, this means more weight in calculation of q_S^L to the flow variables at centroid j than to the flow variables at centroid k, since b>a.

7. Numerical Results and Discussions

To validate the present codes, calculations were performed on typical supersonic corner flow tests. Definitions can be found in Fig 7.1. The deflection angle is 16°. Analytical solution to this problem can be obtained from elementary gas dynamics. The solution consists of two different regions of constant states which are separated by an oblique shock wave as is sketched in Fig 7.1. It can be seen that the flow remains supersonic

behind the shock wave. Therefore the flowfield is supersonic throughout the domain.

To illustrate the application of the mesh enrichment procedure, the numerical computation was performed on successively refined meshes. The number of elements in the refined meshes are 213 (coarse mesh), 1153(intermediate mesh) and 2390(fine mesh), respectively. Computations on the coarse mesh are not sufficiently accurate but may be used to localize the shock wave. The intermediate and fine mesh are defined with a larger number of elements and have been refined along the shock line, computed on the coarse mesh.

The coarse mesh is presented in Fig.7.2. There are 213 elements and 131 nodes in the flow domain. In the present calculations we use six different codes. They are Explicit-Roe Code, PGS-Roe Code, PGJ-Roe Code, Explicit-Osher Code, PGS-Osher Code and PGJ-Osher Code. Figure 7.3 gives the convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes. Figure 7.4 gives the convergence history of the Explicit-Osher , PGS-Osher and PGJ-Osher codes. It can be seen that the implicit method doubles the efficiency in convergence over the explicit code. During the calculation we also found the residual can not reduce than 0.5E-4 -- 1.E-5 when using single precision. For double precision, the residual can easily reach 1.E-14 on a coarse mesh (see Fig.7.5 and Fig.7.6). Fig.7.7 gives the convergence history of the Explicit -Roe code of 1st-order and high-order (MUSCL) methods. It can be seen that more iterations are needed to reach the convergence when using the high-order scheme. Fig.7.8 also gives the convergence history of the PGS-Roe code. Table 1 lists the CPU time per iteration of each code. All calculations are performed on the computer IRIS INDIGO XS workstation. Appendix A gives the flow results (including velocity field, contours of pressure, density and Mach number respecting).

The intermediate size mesh is presented in Fig.7.9. There are 1153 elements and 615 nodes in the flow domain. After finishing the calculation on a coarse mesh we find that in the flow domain there remains a region where physical parameters change rapidly. Thus in the intermediate mesh additional elements were placed around that region. As in the coarse mesh the six codes were implemented. Fig.7.10 gives the convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes. Figure 7.11 gives the convergence history of the Explicit-Osher, PGS-Osher and PGJ-Osher codes. All above codes used single precision. Fig.7.12 shows the convergence history of the PGS-Roe code using both single and double precision. Fig.7.13 shows the convergence history of the PGS-Roe code when using the high-order scheme (MUSCL method). Table 2 gives the CPU time per iteration of each code. All calculation are done again on the computer IRIS

INDIGO XS workstation. Appendix B gives the flow results (including velocity field, contours of pressure, density and Mach number).

The results on the intermediate mesh shows there exists a shock wave in the flow field. In the fine mesh more elements were placed along the shock line (Fig.7.14) in order to capture the shock wave more accurately. Also during the calculation on coarse mesh and intermediate mesh it was found that although the Osher scheme gives as good results as the Roe scheme but it takes nearly twice CPU time per iteration for the implicit scheme due to the fact that it needs to do integration and flux-vector splitting. Hence it was decided to adopt the Roe scheme in the fine mesh calculation. Fig.7.15 gives the convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes. It can be seen that the PGS-Roe code reaches a further improvement converged solution faster than the Explicit-Roe code and the PGJ-Roe code provide. Fig.7.16 gives the convergence history of Explicit-Roe code using both single and double precision. Fig.7.17 gives the convergence history of the PGS-Roe code using both single and double precision. To improve the calculation accuracy we use two high-order VAR and MUSCL schemes. Fig.7.18 gives the convergence history of PGS-Roe code using 1st-order ,VAR high-order and MUSCL high-order schemes. It can be seen that the residual does not decline further for the VAR method after it reaches 1.E-3, however the residual of the MUSCL method can reach 1.E-10. Fortunately both the high-order methods can give good results of the flow including the capture of the shock wave. Table 3 gives the CPU time per iteration of the code. All calculations are made on the computer IRIS INDIGO XS workstation. Appendix C gives the flow results (including velocity field, contours of pressure, density and Mach number).

To validate the Navier-Stokes code, we also select the same example as above. According to boundary layer theory the magnitude of the flow variable gradients in the direction of the flow is much smaller than in the direction normal to the flow. Hence in order to obtain a compatible spatial accuracy the mesh must be much finer in the direction normal to the flow in the vicinity of a no-slip wall. Numerical experiments have indicated that about 15-20 grid points are required in order to accurately represent the boundary layer profile.

In a structured grid it is possible to generate very stretched quadrilateral elements along the wall. But in the unstructured grid, difficulties are expended in the process of computation because of the very stretched triangle mesh.

In present test we use the same mesh as the Euler code. The calculation of the PGS-Roe (NS) code has been done on a fine mesh (Fig.7.19). Fig.7.20 gives the convergence

history of PGS-Roe (NS) code using 1st-order and MUSCL high-order schemes. It takes 2.872 second CPU time per iteration for the 1st-order PGS-Roe (NS) code using double precision. For the high-order (MUSCL method) it takes 6.0802 second CPU time per iteration on IRIS INDIGO XS. Appendix D gives the flow results (including velocity field, contours of pressure, density and Mach number).

Further work will involve using a viscous mesh to validate the Navier-Stokes code.

8. Concluding Remarks

In this paper we develop an implicit scheme (PGS and PGJ) on an unstructured grid to achievea more efficient code than an explicit approach using upwinding discretization techniques for the invisid terms. We also apply the high-order MUSCL scheme as used on a structured grid on an unstructured grid. The convergence rate is shown better than that of VAR method.

Through calculation we obtain the following conclusions:
(1) The convergence history improves markedly when using an implicit scheme instead an of explicit one. Although the implicit code takes a little more CPU time per iteration, the total improvement in efficiency is important.
(2) For single precision a residual of 0.5E-4 -- 1.E-5 can be reached. After using double precision the residual can reach 1.E-14.
(3) Both Roe and Osher schemes give satisfactory results.
(4) For the implicit scheme we suggest using Roe scheme because it takes less CPU time than that of Osher.
(5) For PGS and PGJ both methods can give nearly the same convergence rate on coarse and intermediate meshs. However on a fine mesh the PGS method convergences better than the PGJ method.
(6) Implementation of the high-order scheme improves the accuracy. Both VAR method and MUSCL method can produce the better results. The residual of VAR method can only reach 1.E-3, however the residual of MUSCL method can reach 1.E-10.
(7) The PGS-Roe (NS) code can also run successfully on a fine mesh with reasonable results.

Aknowledgement

The author would like to thank Prof.B.E.Richards for his successive supervisions and

discussions.
Many thanks to Mr.L.Dubuc for his helpful discussions. Without his previous work on explicit scheme,the author could not have made the same progress.
The author also wish to thank Dr. X.Xu and Dr. K.Badcock for their discussions.

References

[1] A.Jameson, T.J.Baker, N.P.Weatherhill "Calculation of Inviscid Transonic Flow over a Complete Aircraft" AIAA paper 86-0202, (1986)
[2] D.J.Mavriplis "Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes" AIAA paper 88-3707 (1988)
[3] K.Morgan, J.Peraire "Finite Element Methods for Compressible Flows" von Karman Institute for Fluid Dynamics Lecture Series 1987-04, (1987)
[4] J.Peraire, M.Vahdati, K.Morgan, O.C.Zienkiewicz "Adaptive Remeshing for Compressible Flow Computations" Journal of Comp. Phys., Vol. 72, pp 449-466, (1987)
[5] B.Stoufflet, J.Periaux, F.Fezoui, A.Dervieux "Numerical Simulation of 3-D Hypersonic Euler Flows Around Space Vehicles Using Adapted Finite Element" AIAA paper 87-0560 (1987)
[6] R.R.Thareja, J.R.Stewart, O.Hassan, K.Morgan, J.Peraire "A Point Implicit Unstructured Grid Solver for the Euler and Navier-Stokes Equations" AIAA paper 88-0036, (1988)
[7] T.J.Barth, D.C.Jesperson "The Design and Application of Upwind Schemes on Unstructured Meshes" AIAA paper 89-0366 (1989)
[8] J.T.Batina "Three-dimensional Flux-split Euler Scheme Involving Unstructured Dynamic Meshes" AIAA paper 90-1649, (1990)
[9] D.J.Mavripls A.Jameson "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes" AIAA J. Vol.28 No.8 Aug.1990 pp1415-1425
[10] Rajiv R.Thareja, James R.Stewart, Obey Hassan, K.Morgan and Jaime Peraire " A Point Implicit Unstructured Grid Solver for the Euler and Navier-Stokes Equations" Int. J. for Num. Methods in Fluids Vol.29, 405-425 (1989)
[11] J.T.Batina "Implicit Flux-Split Euler Schemes for Unsteady Aerodynamic Analysis Involving Unstructured Dynamic Meshes" AIAA J. Vol.29 No.11 Nov.1991 pp 1836-1843
[12] C.J.Hwang , J.L.Lin "Locally Implicit Total-Variation-Diminishing Scheme on Unstructured Triangular Meshes" AIAA J. Vol.29 No.10 Oct.1991 pp 1619-1626
[13] J.T.Batina "Implicit Upwind Solution Algorithm for Three-Dimensional

Unstructured Meshes" AIAA J. Vol.31 No.5 May 1993 pp 801-805
[14] L.Dubuc "Two-Dimensional Navier-Stokes Solver Using An Upwind Scheme on Unstructured Grids" Final Report, Sept.1992 Dept. of Aeorspace Engineering, University of Glasgow
[15] P.L.Roe "Approximate Riemann Solver, Parameter Vectors and Difference Scheme" J. of Comp.Physics Vol.43 pp 357-372,1981
[16] S.Osher, F.Solomon "Upwind Difference Scheme for Hyperbolic Systems of Conservation Laws" Maths. of Computation Vol.38, pp339-374, 1982
[17] S.Soltani "An Upwind Scheme for the Equations of Compressible Flow on Unstructured Grids" Thesis submitted for the degree of Doctor of philosophy of the University of London, July 1991
[18] A.Harton "High Resolution Scheme for Hypersonic Conservation Laws" DOE/ER/03077-175 Courant Mathematics and Computing Laboratory, NYU 1982
[19] J.L.Steger R.F.Warming "Flux Vector Splitting of the Invisid Gas Dynamics Equations with Application to Finite Difference Methods" J. of Comp. Phys. Vol.40 pp 263-293, 1981
[20] B. van Leer "Towards an Ultimate Conservative Differencing Scheme IV:A New Approcach to Numerical Convection" J. of Comp.Phys. Vol.23 pp276-299 1977
[21] J.T.Batina "Implicit Flux-Split Euler Scheme for Unsteady Aerodynmic Analysis Involving Unstructured Dynamic Meshes" AIAA J. Vol.29 No.11 1991 pp 1836-1843
[22] Qin.N "A Comparative Study of Two Upwind Schemes As Applied to Navier-Stokes Equations for Resolving Boundary Layers in Viscous Hypersonic Flow" GU Aero. Report 9120, 1991
[23] B.van Leer , G.D. van Albada and W.W.Roberts "A Comparative Study of Computational Methods in Cosmic Gas Dynamics" Astronomy and Astrophysics, 108,1982

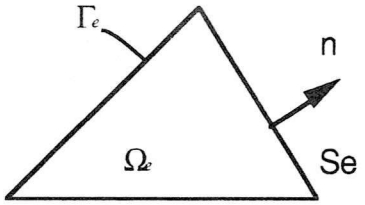


Figure 4.1
Notations for the control volume

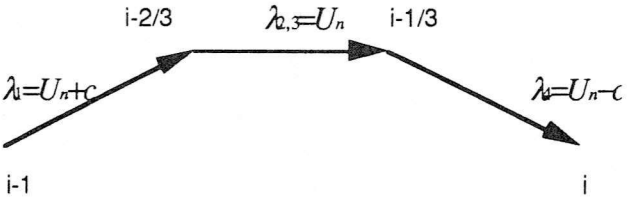


Figure 4.2
path of integration for Osher's flux

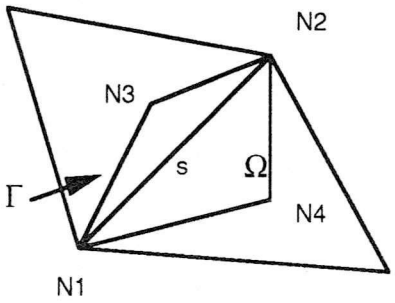


Figure 4.3
Definition of the integration path

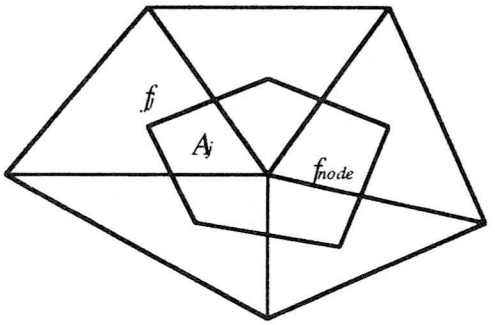


Figure 4.4
weighted average method

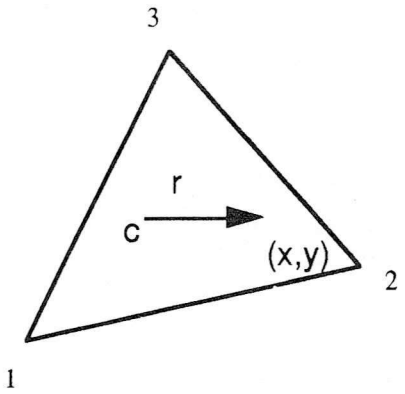


Figure 6.1
linear representation over an element

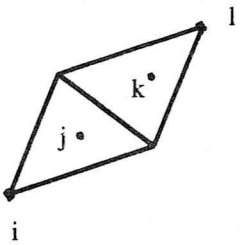


Figure 6.2a
Centroids and nodes

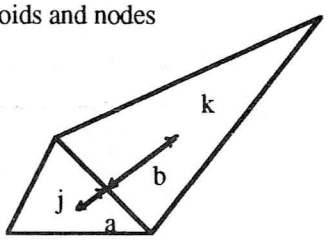


Figure 6.2b
Distances between centroids and midpoint of edge

Code	CPU time per iteration		
	1st-order,sp	1st-order,dp	2nd-order,dp
Explicit-Roe	0.0872 sec	0.1062 sec	0.2137 sec
PGS-Roe	0.1385 sec	0.1657 sec	0.2922 sec
PGJ-Roe	0.1497 sec	--	--
Explicit-Osher	0.1187 sec	--	--
PGS-Osher	0.2589 sec	--	--
PGJ-Osher	0.2638 sec	--	--

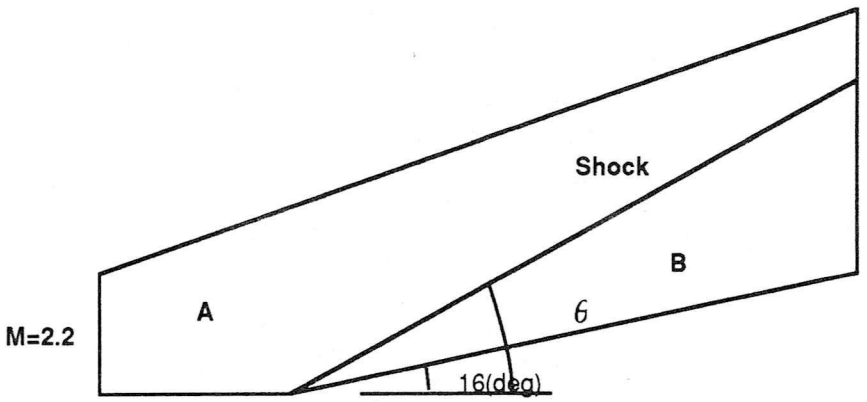
Table 1 : CPU time for different codes
Mesh: Coarse Mesh
Computer: IRIS INDIGO XS

Code	CPU time per iteration		
	1st-order,sp	1st-order,dp	2nd-order,dp
Explicit-Roe	0.42 sec	0.5547 sec	1.3982 sec
PGS-Roe	0.72 sec	0.8744 sec	1.822 sec
PGJ-Roe	0.78 sec	--	--
Explicit-Osher	0.63 sec	--	--
PGS-Osher	1.36 sec	--	--
PGJ-Osher	1.41 sec	--	--

Table 2 : CPU time for different codes
Mesh: Intermediate Mesh
Computer: IRIS INDIGO XS

Code	CPU time per iteration		
	1st-order,sp	1st-order,dp	2nd-order,dp
Explicit-Roe	0.95 sec	1.087 sec	3.377 sec
PGS-Roe	1.32 sec	1.7784 sec	4.074 sec MUSCL 4.104 sec VAR
PGJ-Roe	1.76 sec	--	--
Explicit-Osher	--	--	--
PGS-Osher	--	--	--
PGJ-Osher	--	--	--

Table 3 : CPU time for different codes
Mesh: Fine Mesh
Computer: IRIS INDIGO XS



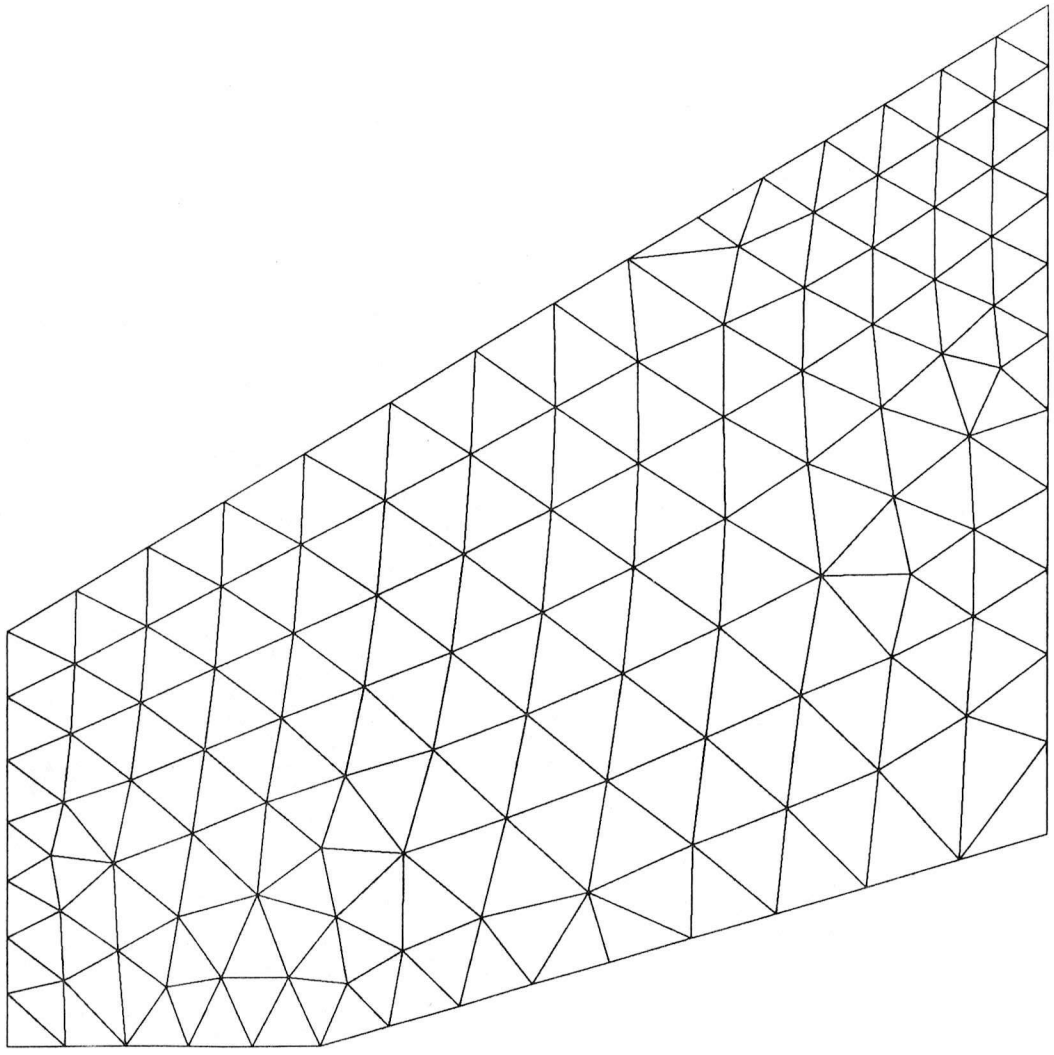
δ=16(deg) **angle of flow deflection across an oblique shock-wave**

θ=42.5(deg) **shock-wave angle measured from uptream flow direction**

Region A
M=2.2
P=0.1475
rho=1.0

Region B
M=1.58
P=0.3554
rho=1.8382

Figure 7.1
Definition of the supersonic flow past a compression corner



UNSTRUCTURED GRID

Model: Compression Corner Flow
Code : Euler Equation
Explicit scheme, PGS scheme and PGJ scheme
Flux : Roe and Osher
Order : 1st-order and high-order
Mesh : Coarse Mesh
213 elements
131 nodes
47 boundary points
Free stream condition :

$M_{\inf} = 2.2$

$T_{\inf} = 120K$

$T_w = 300K$

Figure 7.2 Unstructured grid (coarse mesh)

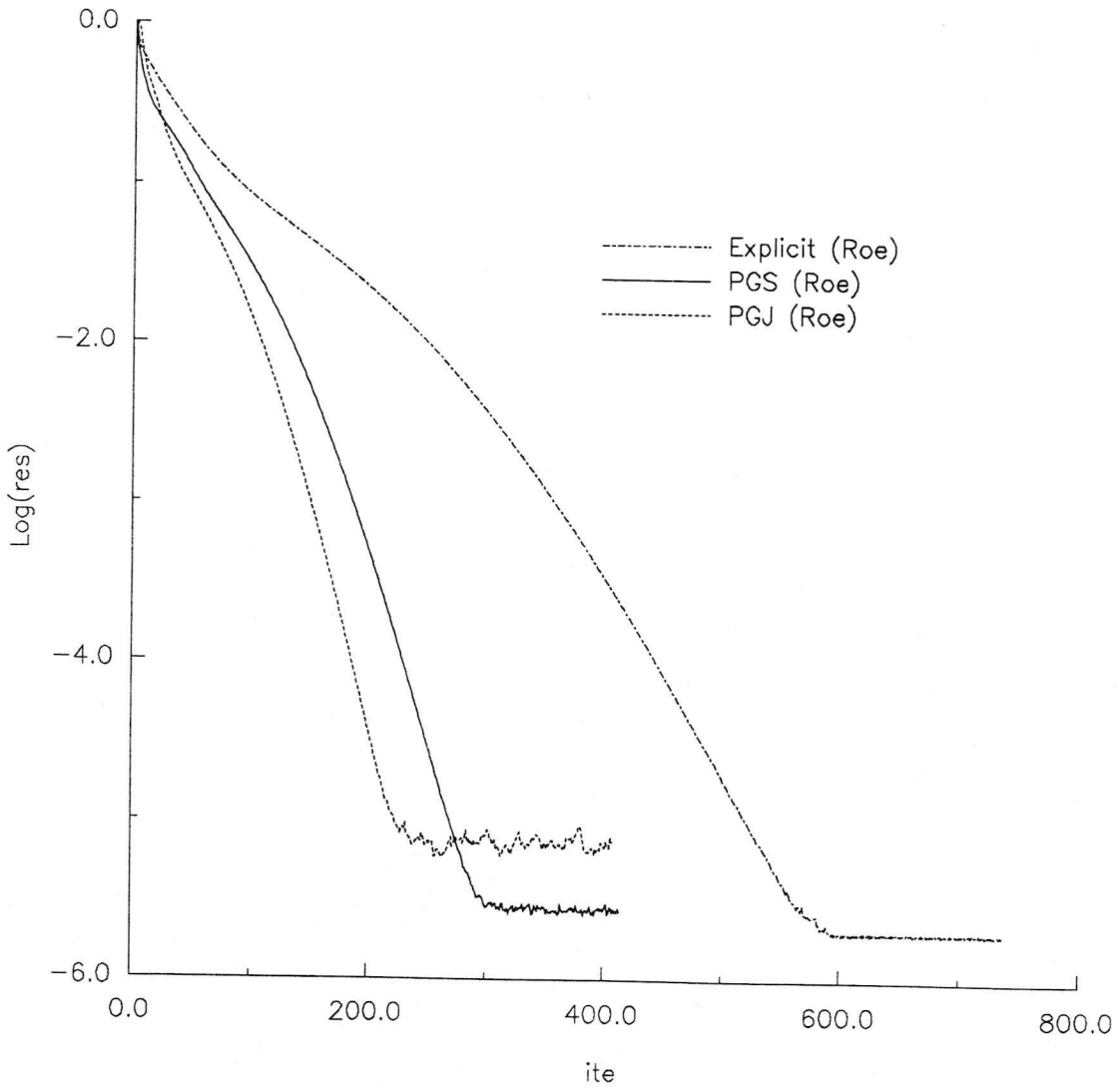


Figure 7.3 Convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes on coarse mesh using single precision

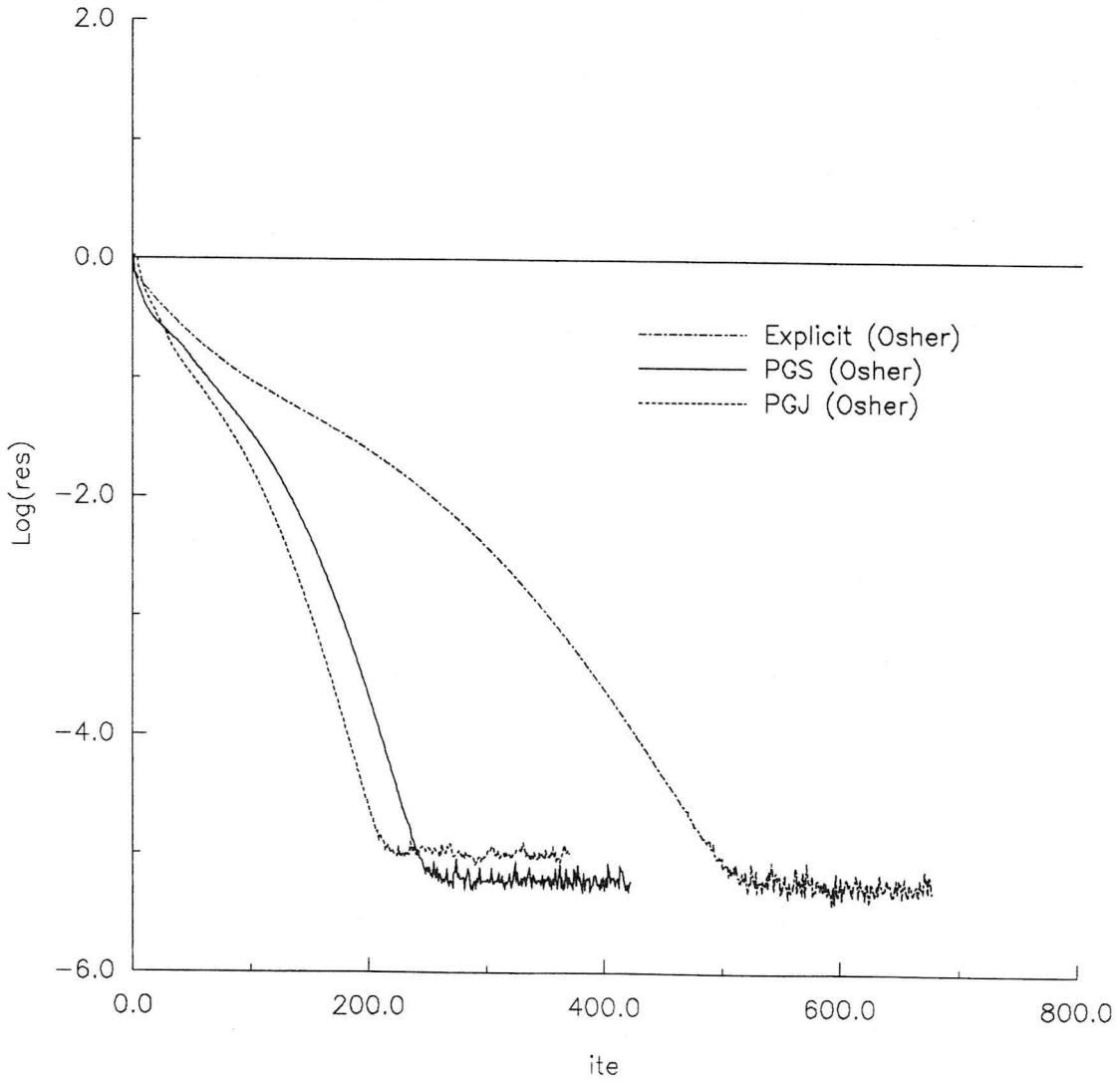


Figure 7.4 Convergence history of the Explicit-Osher, PGS-Osher and PGJ-Osher codes on coarse mesh using single precision

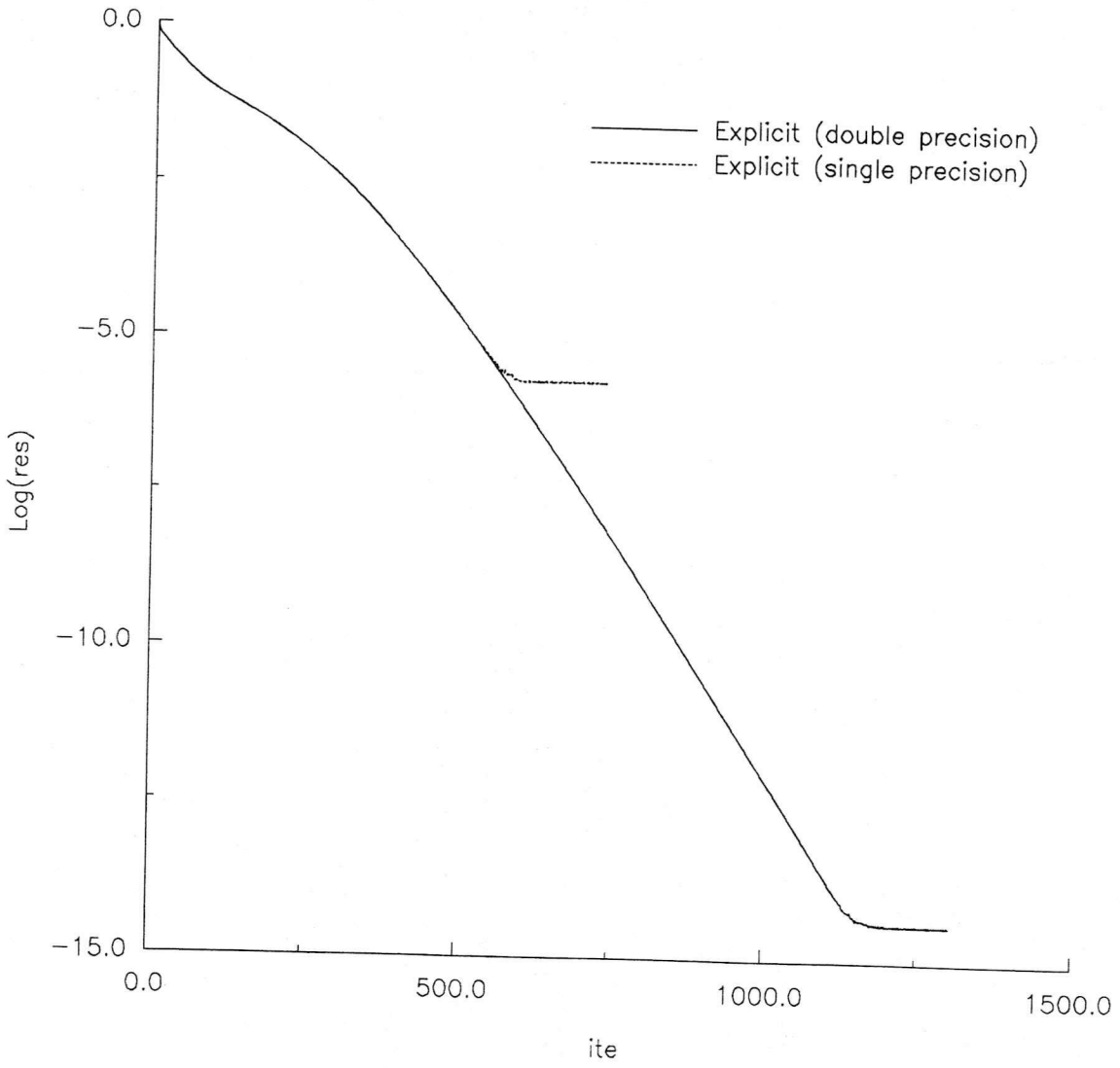


Figure 7.5 Convergence history of the Explicit-Roe code on coarse mesh using single and double precision

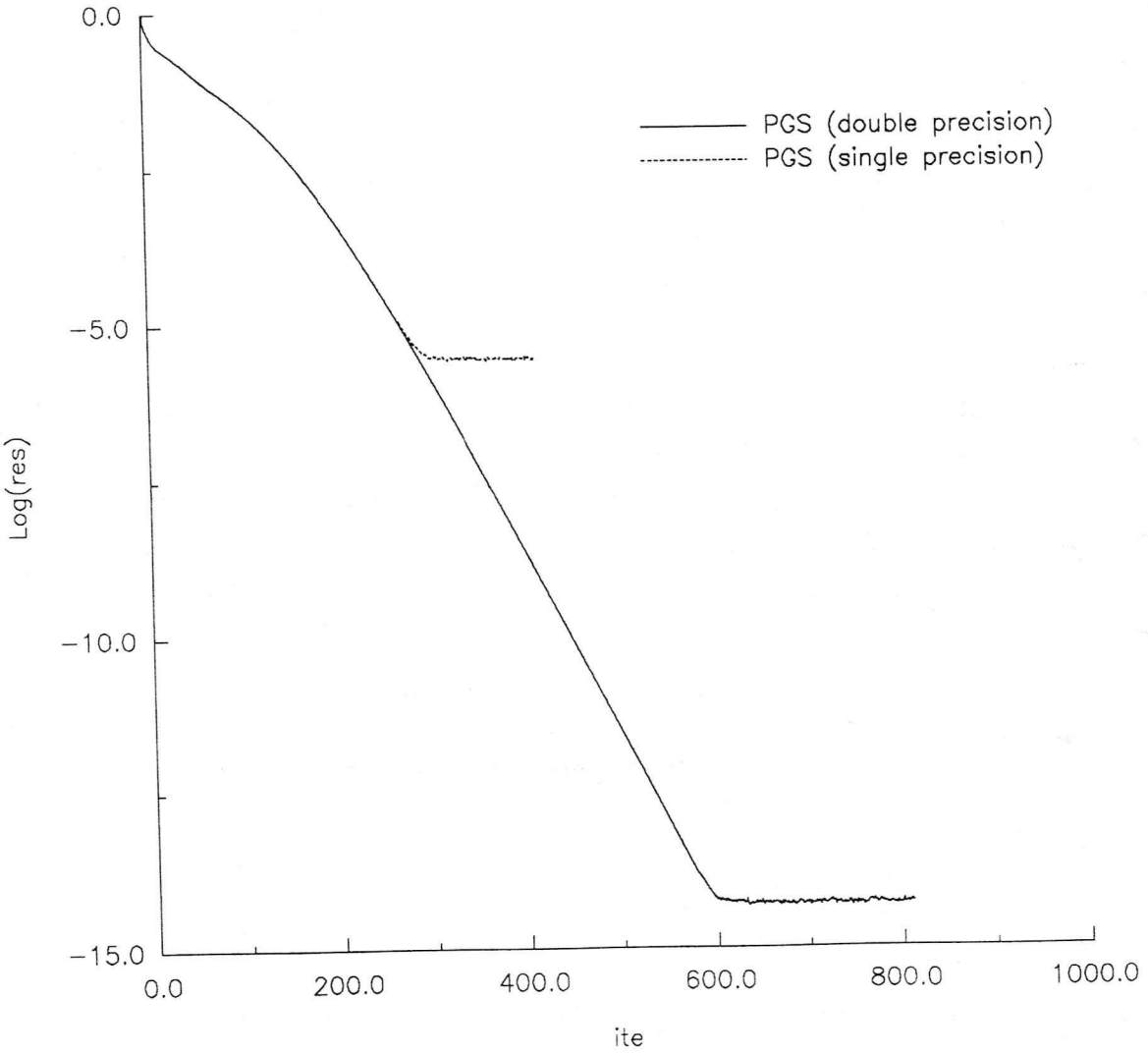


Figure 7.6 Convergence history of the PGS-Roe code on coarse mesh using single and double precision

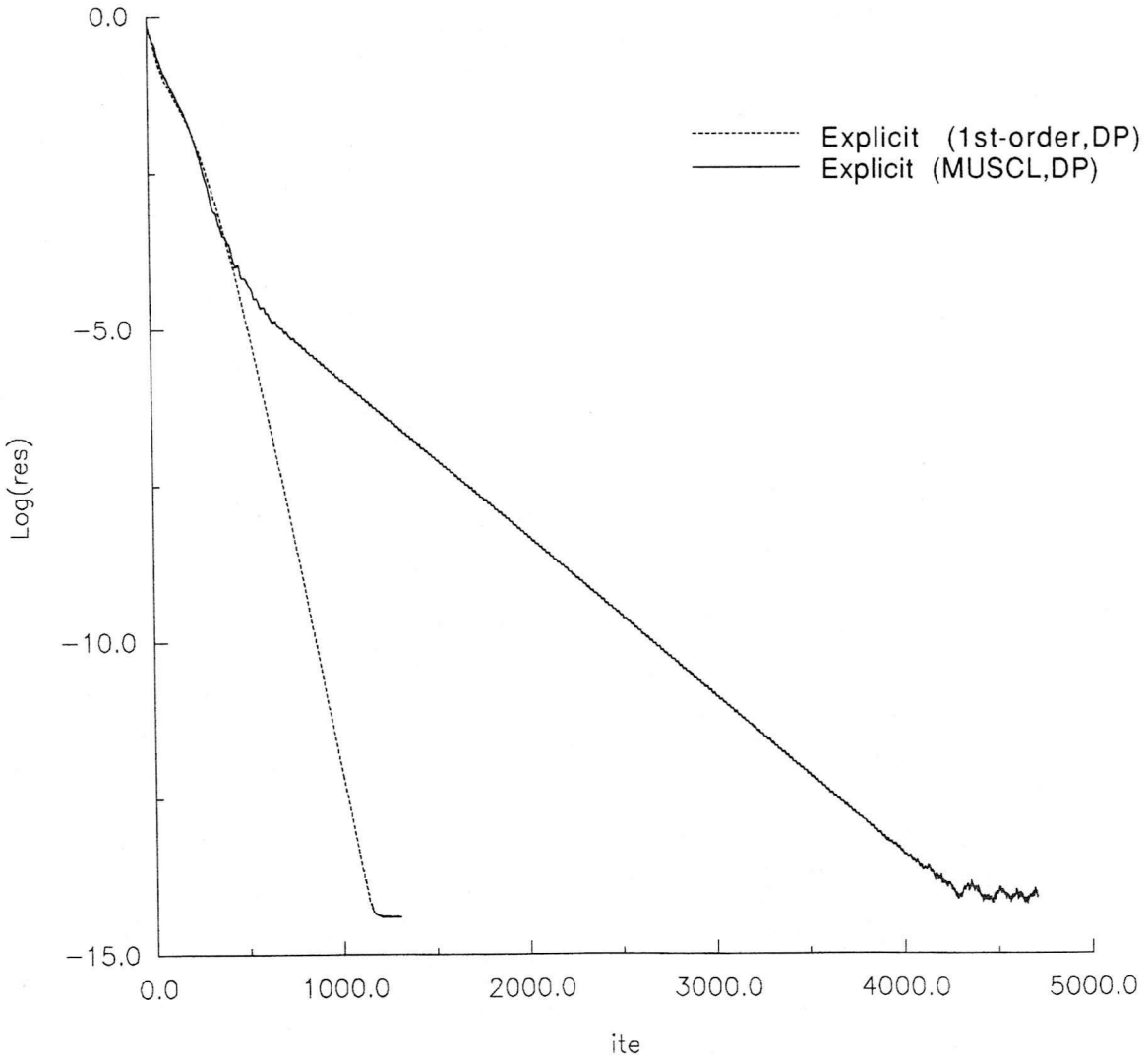


Figure 7.7 Convergence history of the Explicit-Roe code on coarse mesh
using 1st-order double precision and high-order(MUSCL) double precision

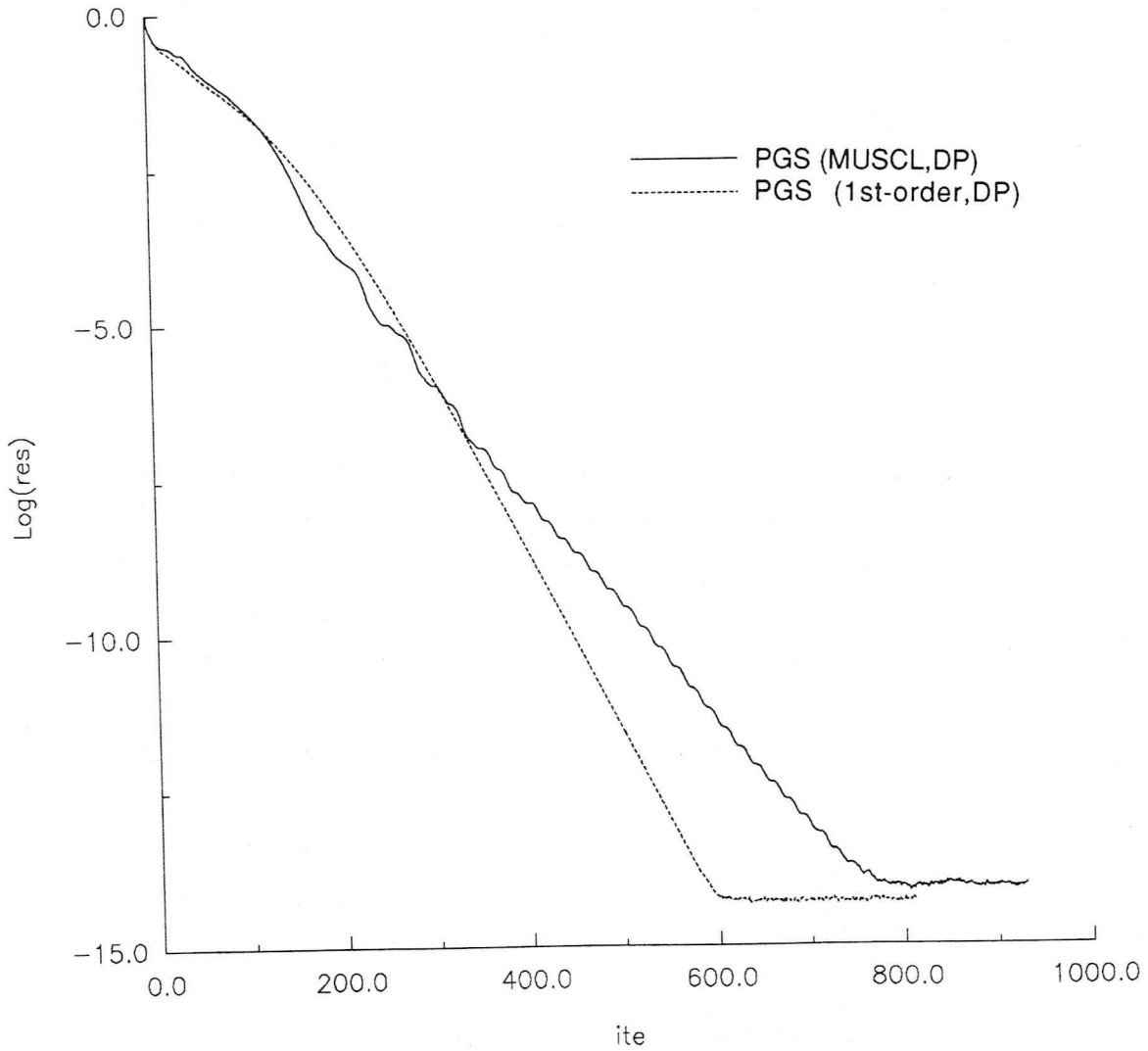
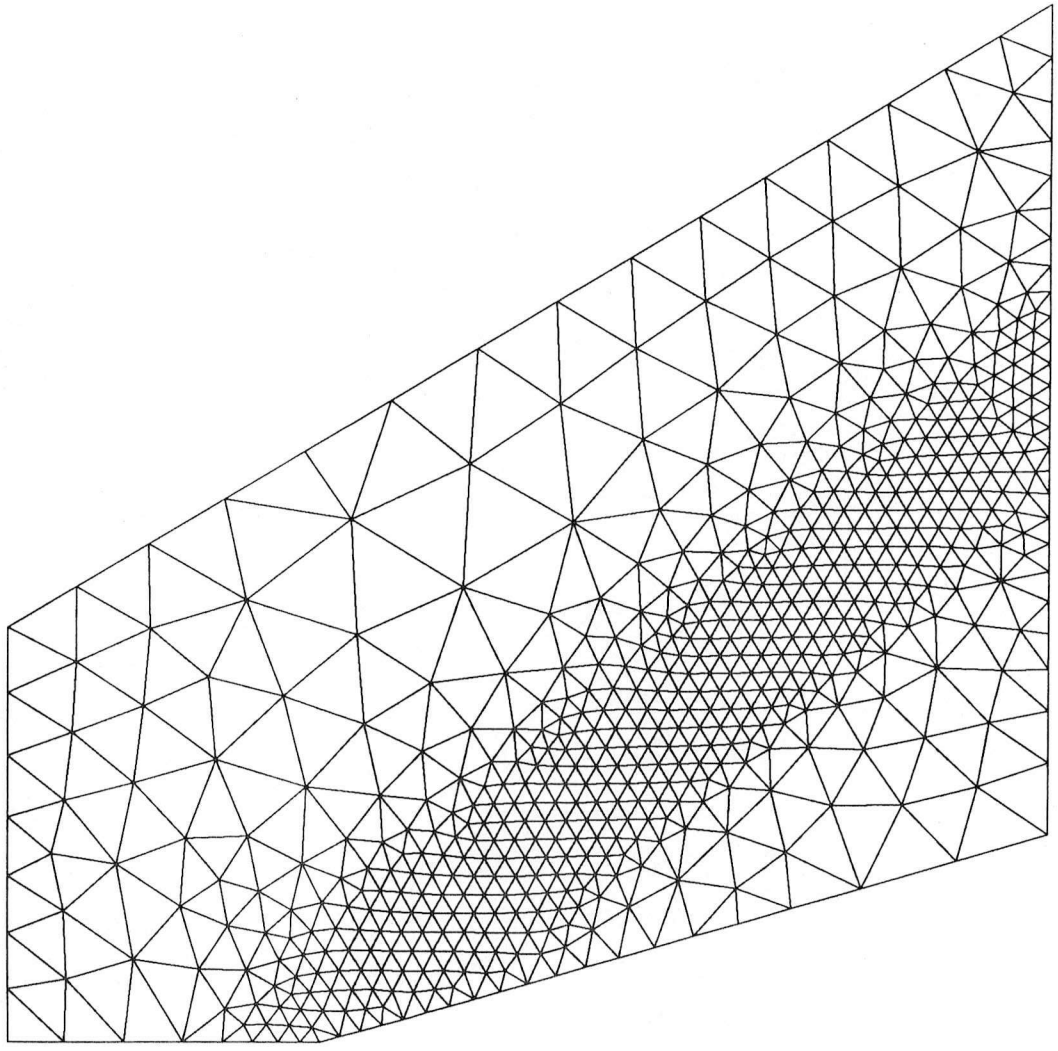


Figure 7.8 Convergence history of the PGS-Roe code on coarse mesh
using 1st-order double precision and high-order(MUSCL) double precision



UNSTRUCTURED GRID

Model: Compression Corner Flow
Code : Euler Equation
Explicit scheme, PGS scheme and PGJ scheme
Flux : Roe and Osher
Order : 1st-order and high-order
Mesh : Intermediate Mesh
1153 elements
615 nodes
75 boundary points
Free stream condition :

$M_{\inf} = 2.2$

$T_{\inf} = 120K$

$T_w = 300K$

Figure 7.9 Unstructured grid (intermediate mesh)

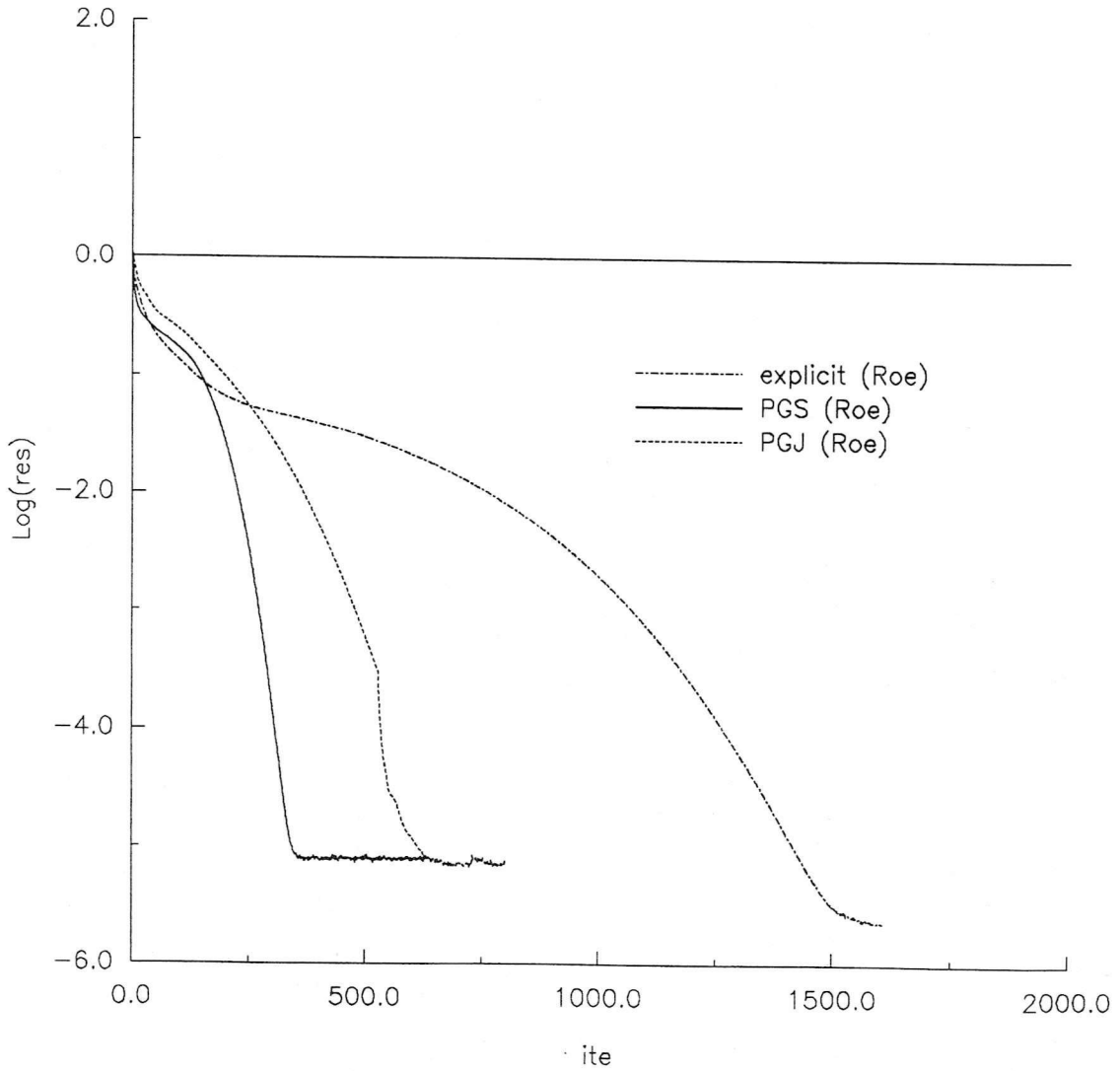


Figure 7.10 Convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes on intermediate mesh using single precision

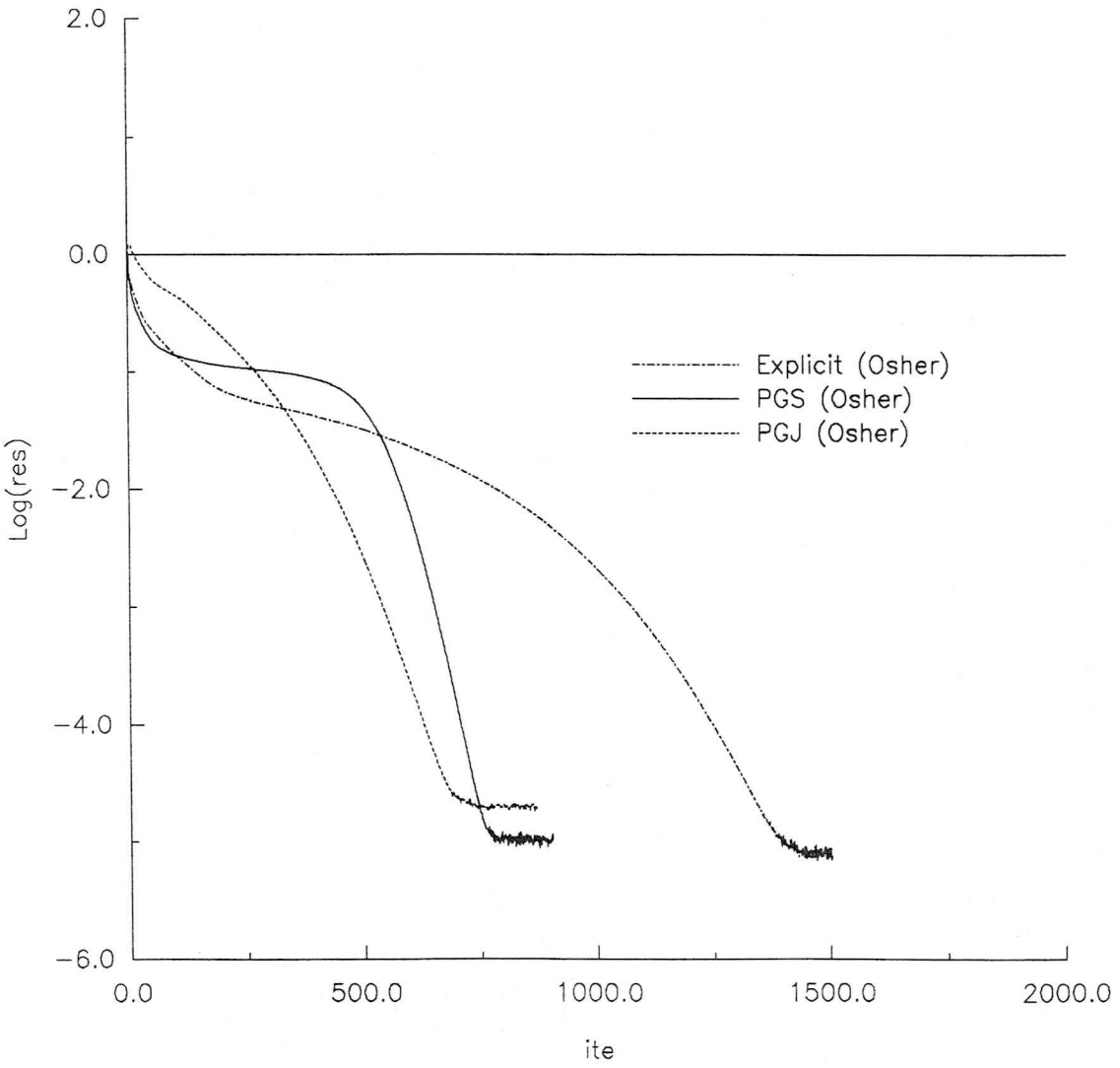


Figure 7.11 Convergence history of the Explicit-Osher, PGS-Osher and PGJ-Osher codes on intermediate mesh using single precision

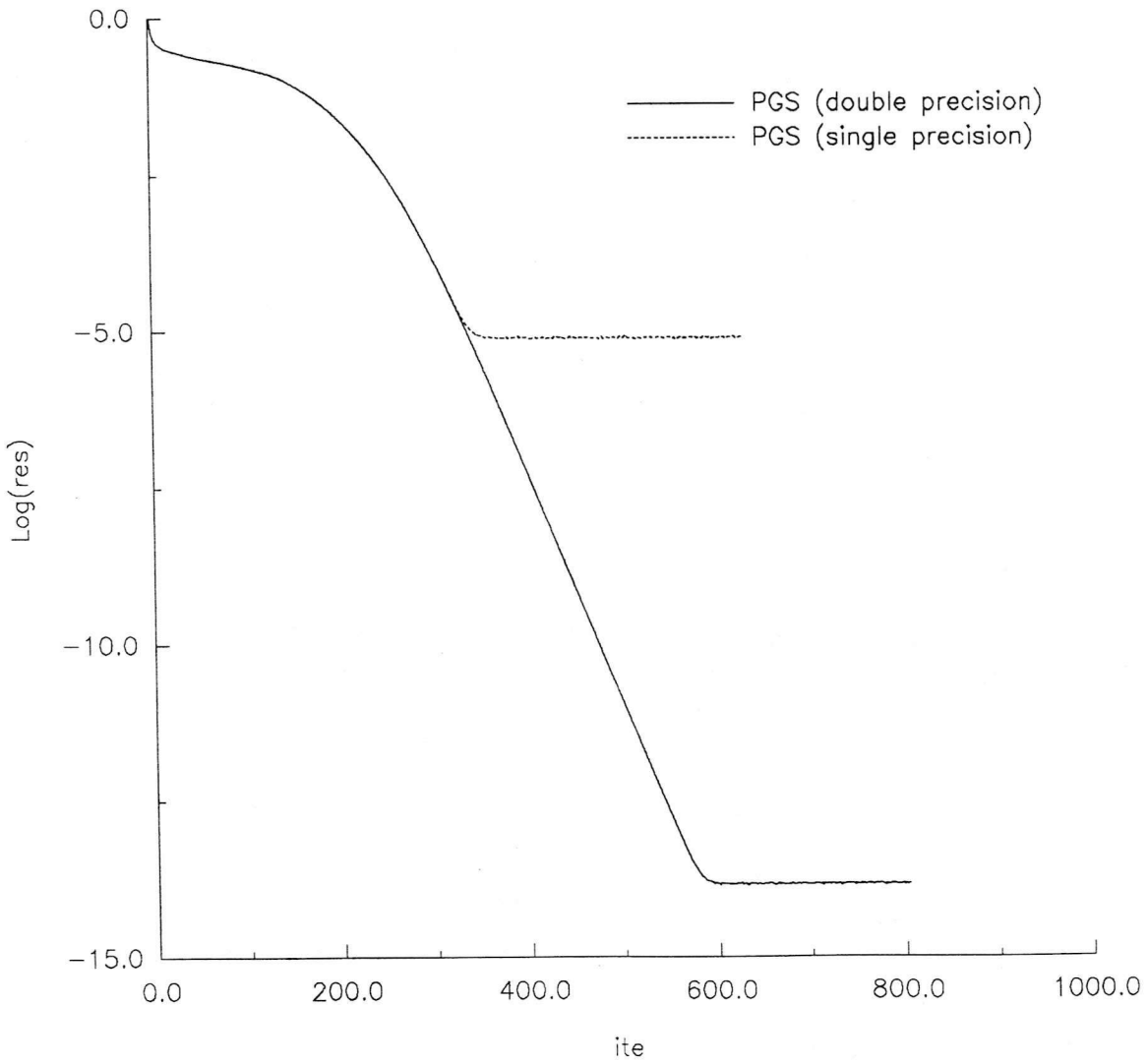


Figure 7.12 Convergence history of the PGS-Roe code on intermediate mesh using single and double precision

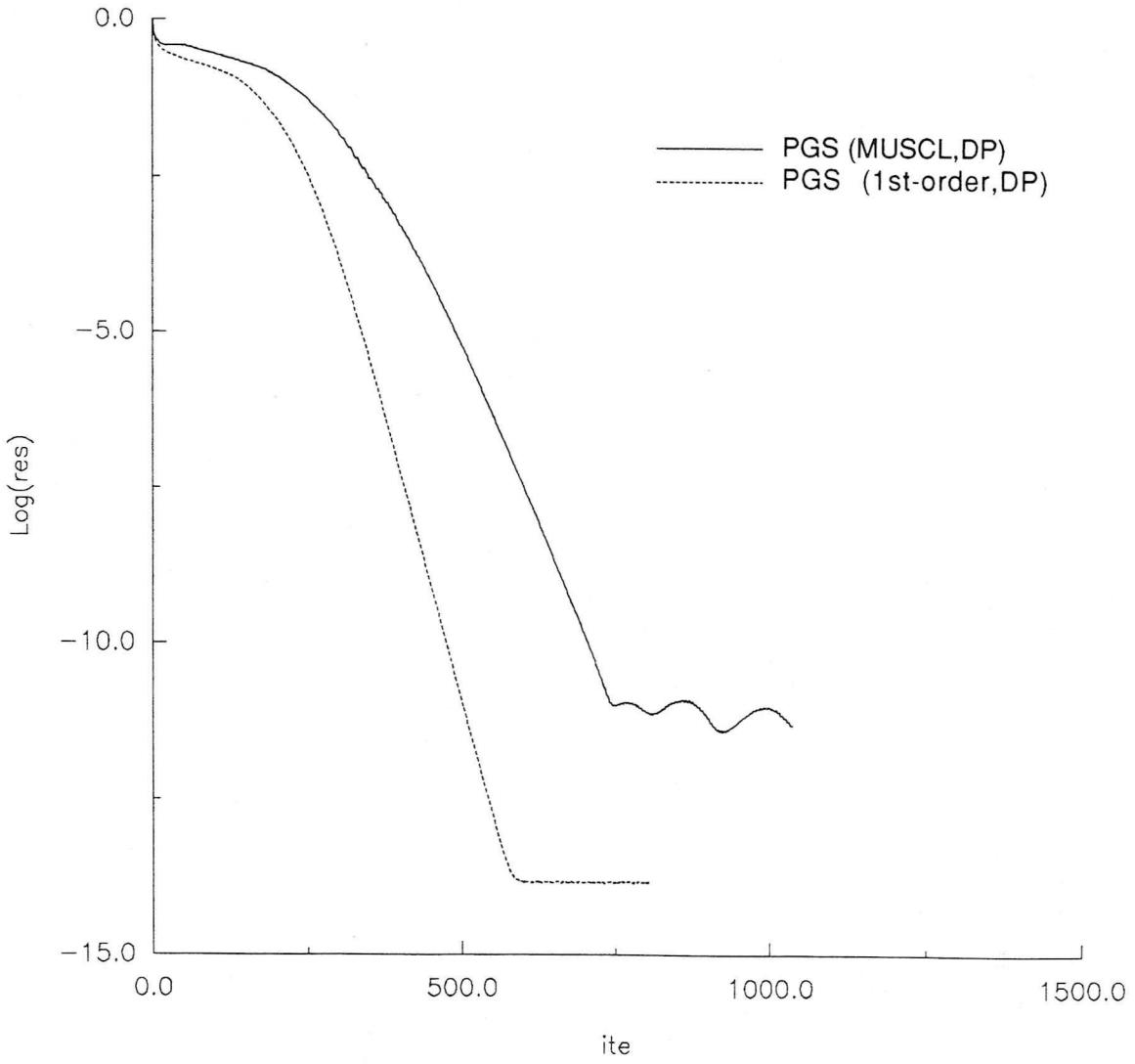
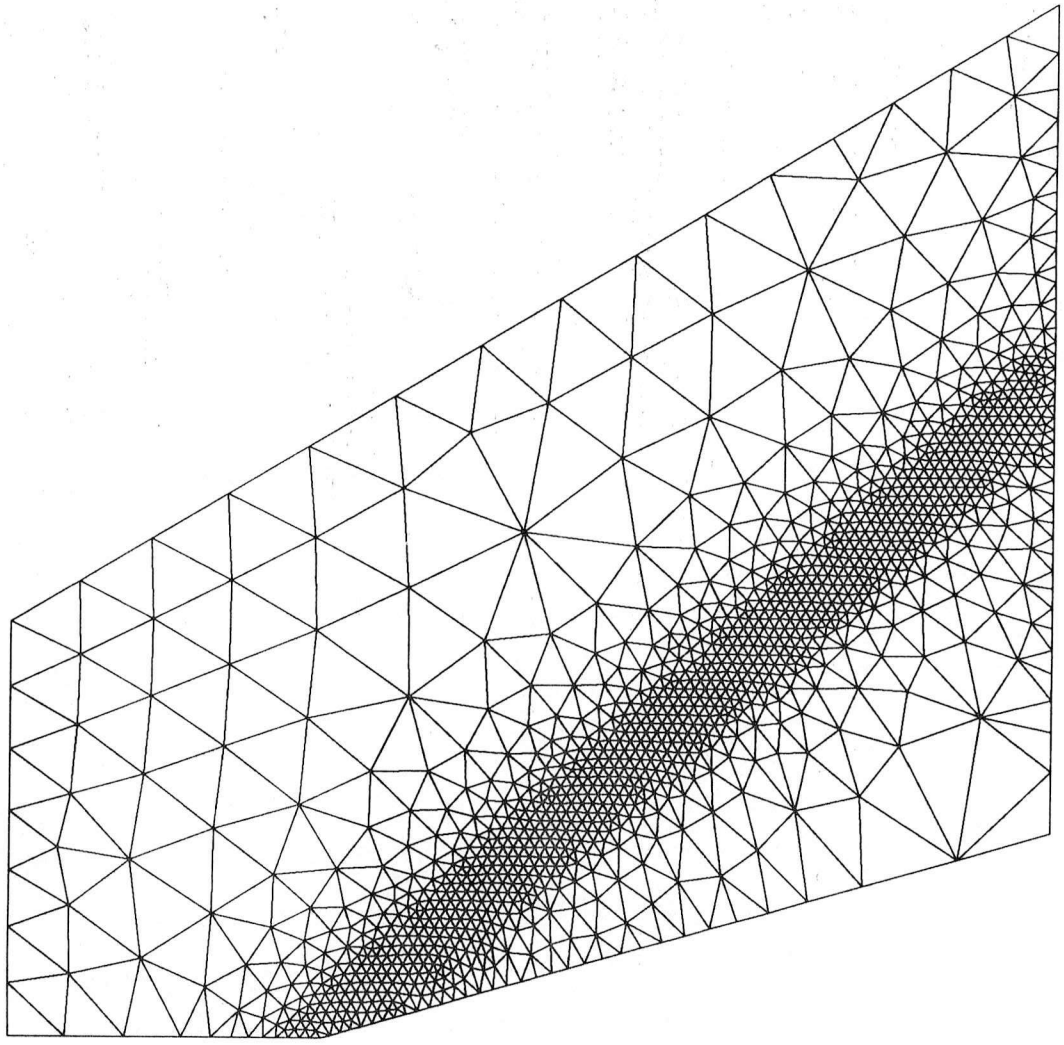


Figure 7.13 Convergence history of the PGS-Roe code on intermediate mesh using 1st-order double precision and high-order(MUSCL) double precision



UNSTRUCTURED GRID

Model: Compression Corner Flow
Code : Euler Equation
Explicit scheme, PGS scheme and PGJ scheme
Flux : Roe
Order : 1st-order and high-order
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{\inf} = 2.2$

$T_{\inf} = 120K$

$T_w = 300K$

Figure 7.14 Unstructured grid (fine mesh)

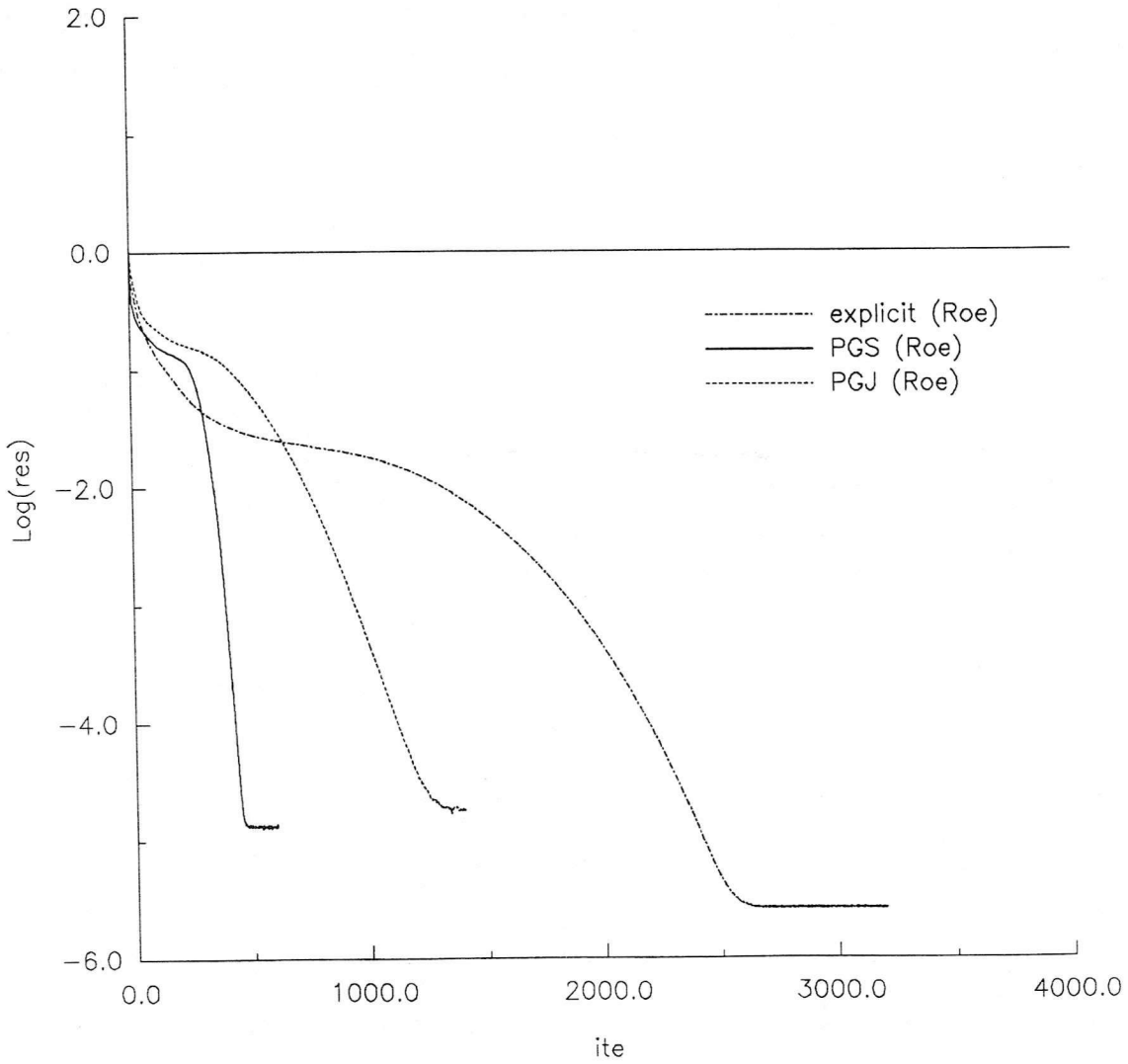


Figure 7.15 Convergence history of the Explicit-Roe, PGS-Roe and PGJ-Roe codes on fine mesh using single precision

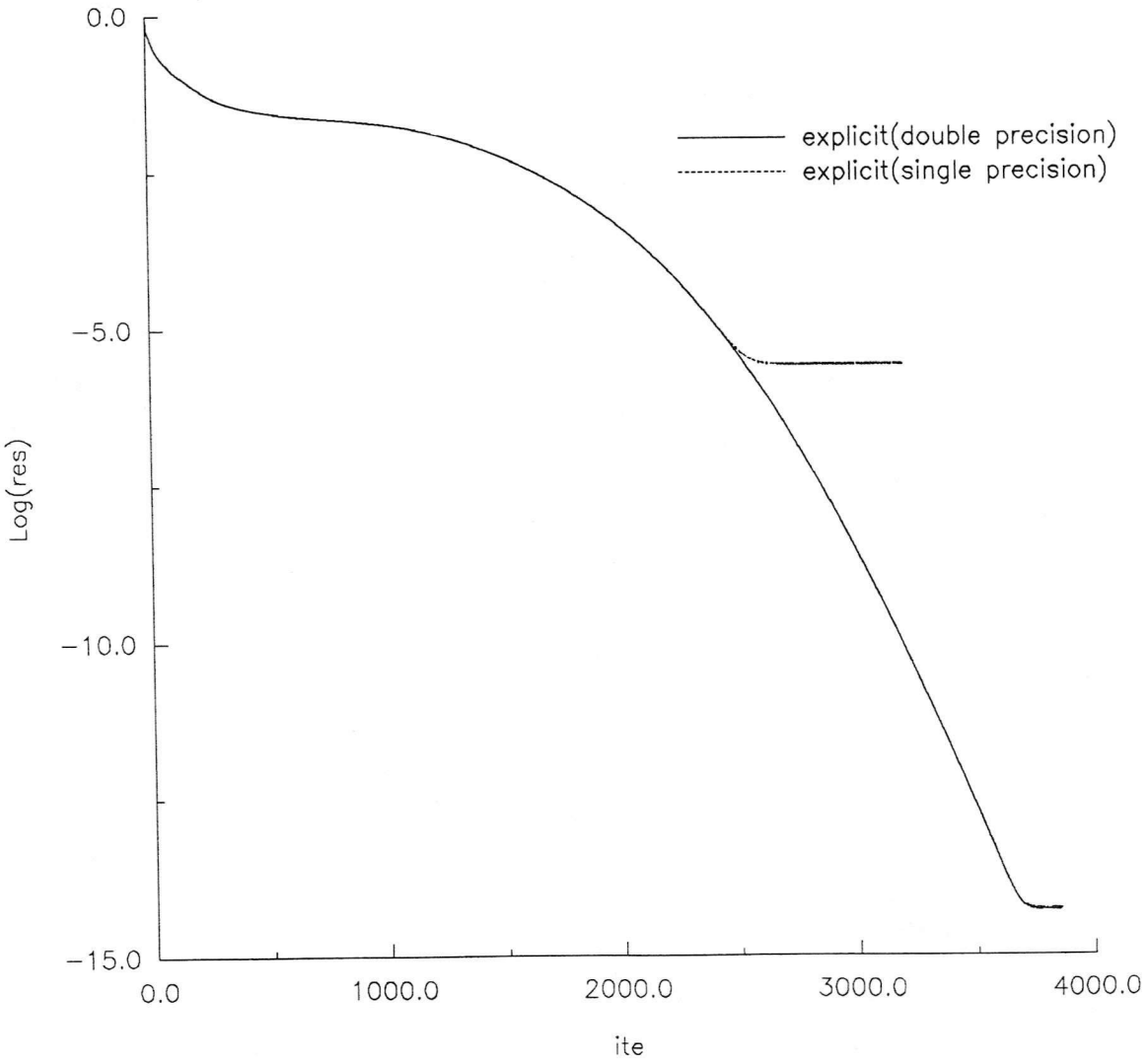


Figure 7.16 Convergence history of the Explicit-Roe code on fine mesh using single and double precision

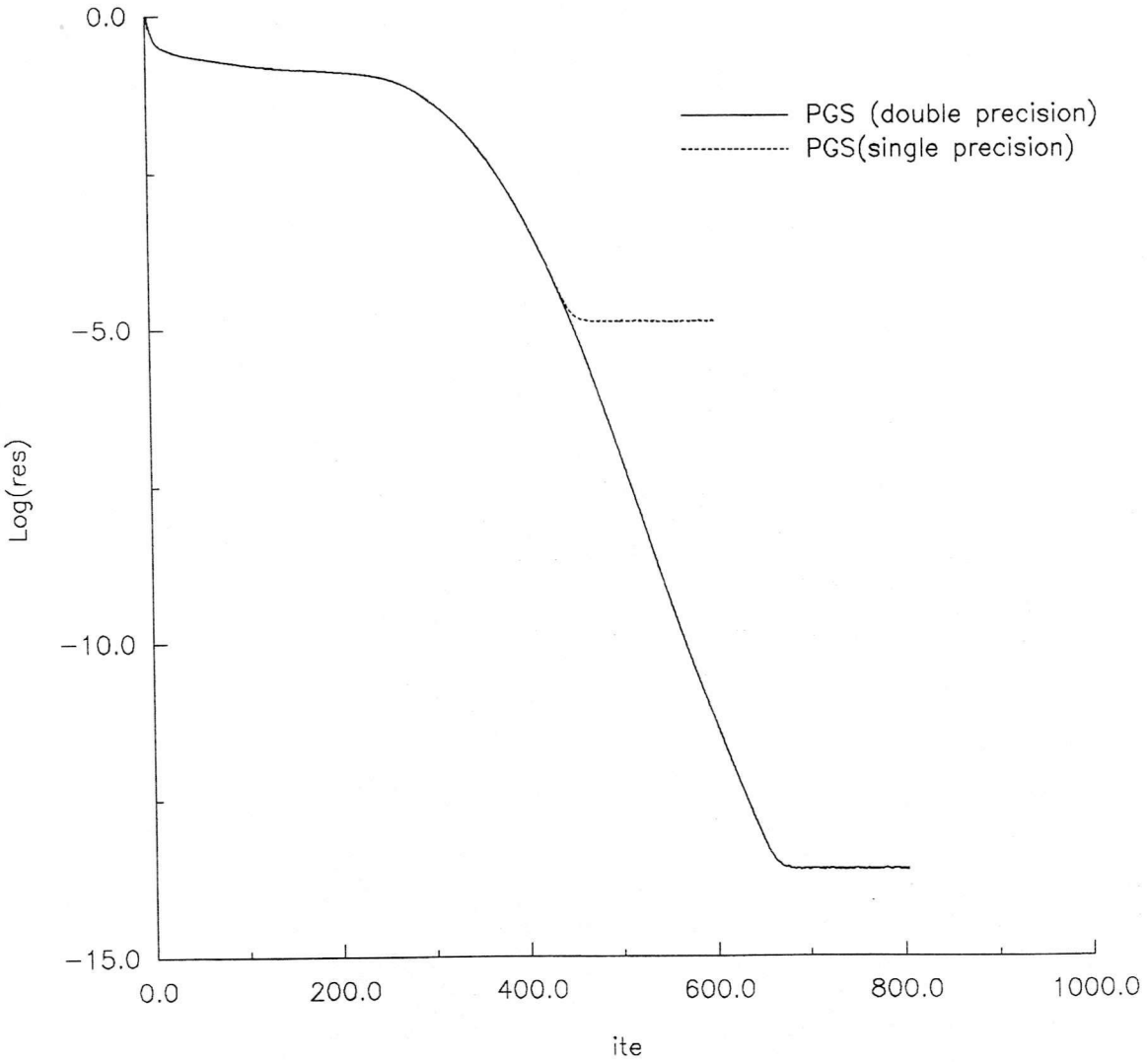


Figure 7.17 Convergence history of the PGS-Roe code on fine mesh using single and double precision

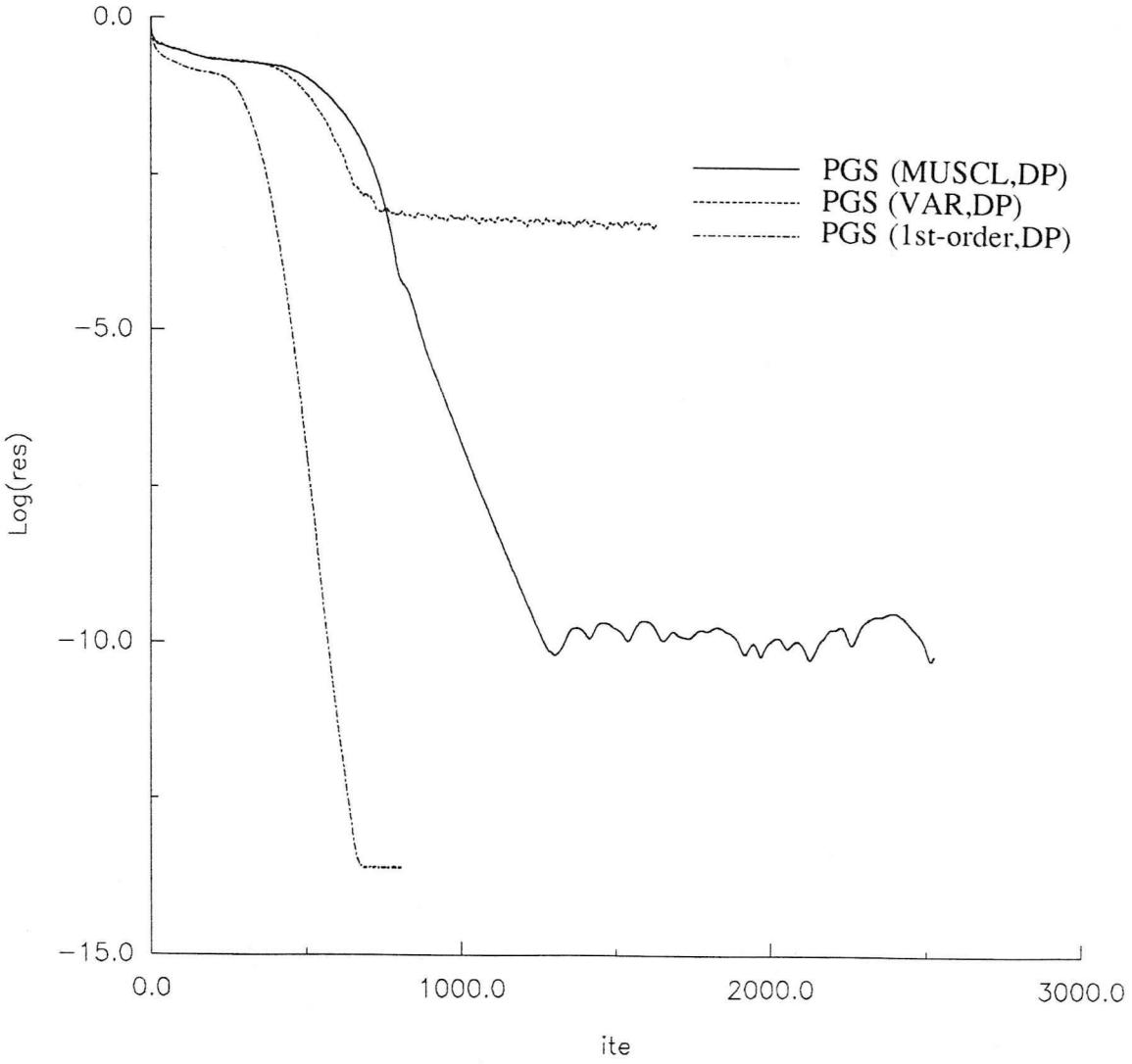
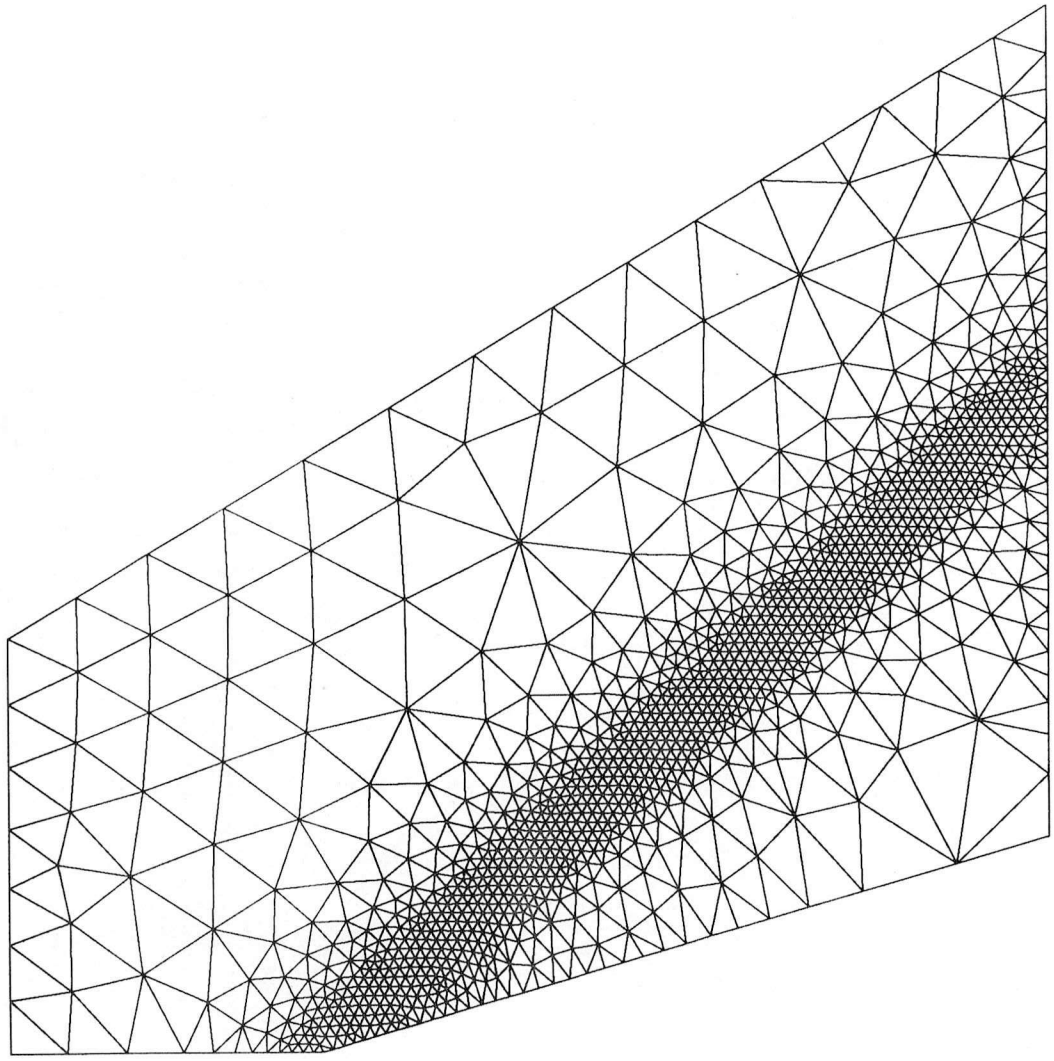


Figure 7.18 Convergence history of the PGS-Roe code on fine mesh using 1st-order double precision and high-order(MUSCL and VAR) double precision



UNSTRUCTURED GRID

Model: Compression Corner Flow
Code : Navier-Stokes Equation
PGS scheme
Flux : Roe
Order : 1st-order and high-order
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{\inf} = 2.2$

$T_{\inf} = 300K$

$T_w = 300K$

$\mathrm{Re}_\infty = 0.65E+07$

Figure 7.19 Unstructured grid (fine mesh)

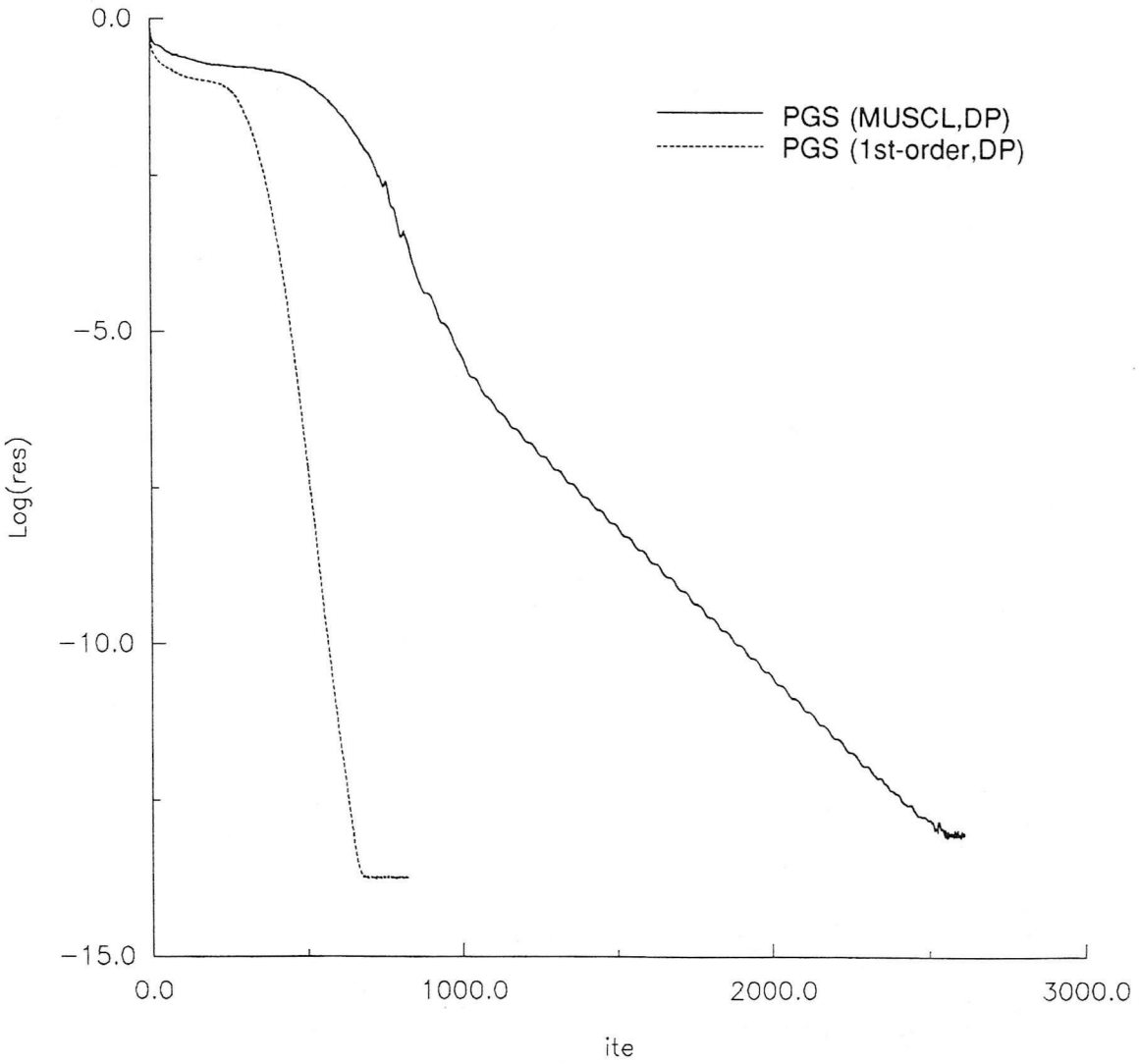


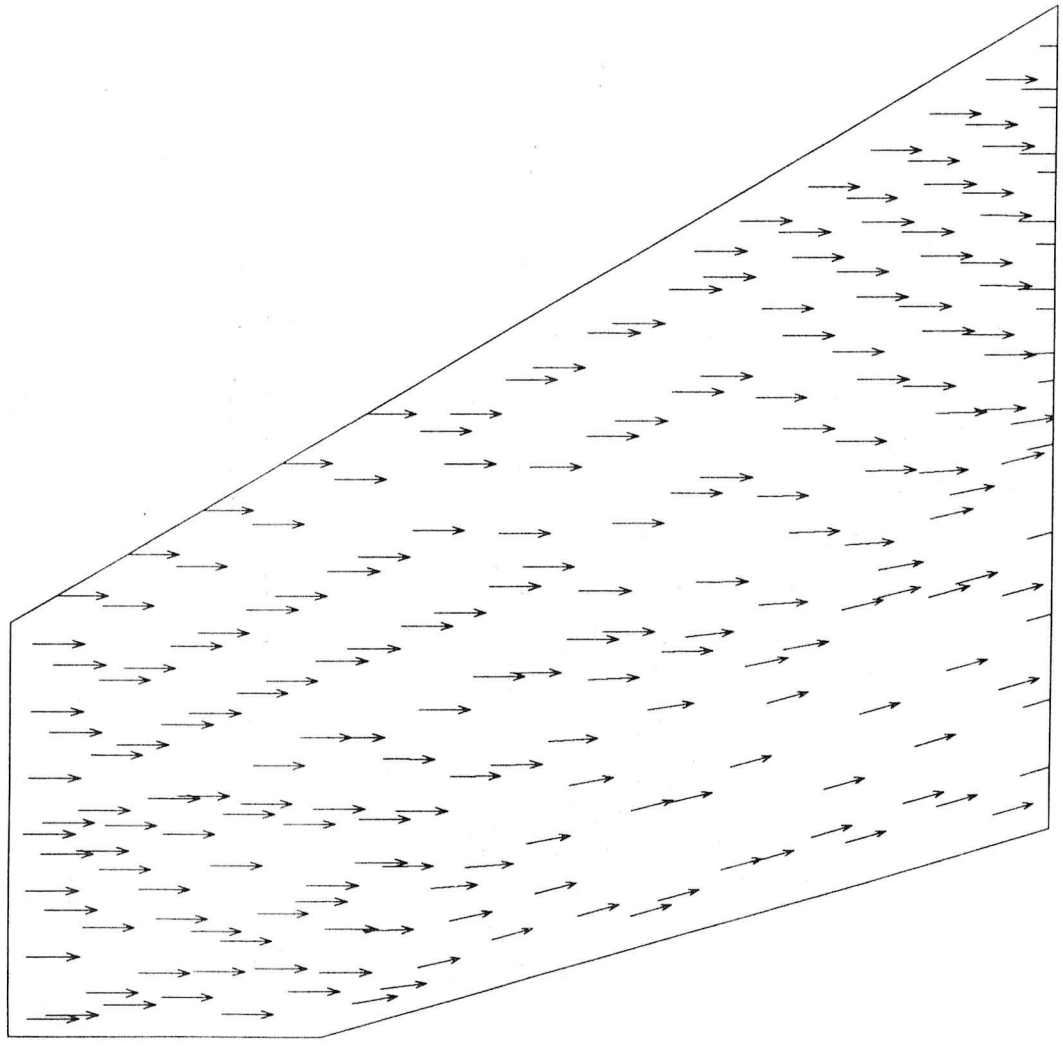
Figure 7.20 Convergence history of the PGS-Roe (Navier-Stokes) code on fine mesh using 1st-order double precision and high-order(MUSCL) double precision

APPENDIX A

SUPERSONIC COMPRESSION CORNER FLOW

COARSE MESH

INVICID CASE



FLOW FIELD

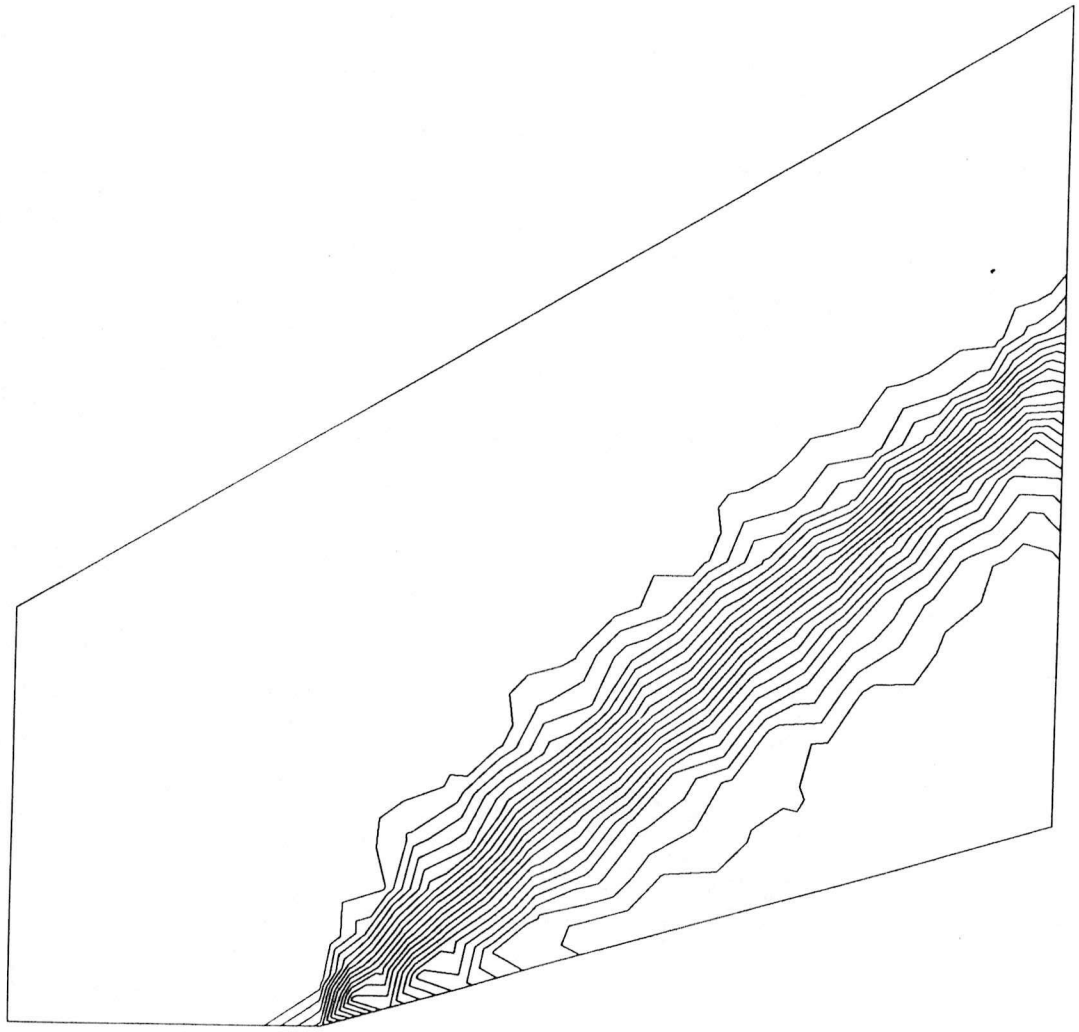
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Coarse Mesh
213 elements
131 nodes
47 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure A.10.1 Compression corner flow (coarse mesh) -- Velocity field



PRESSURE CONTOURS

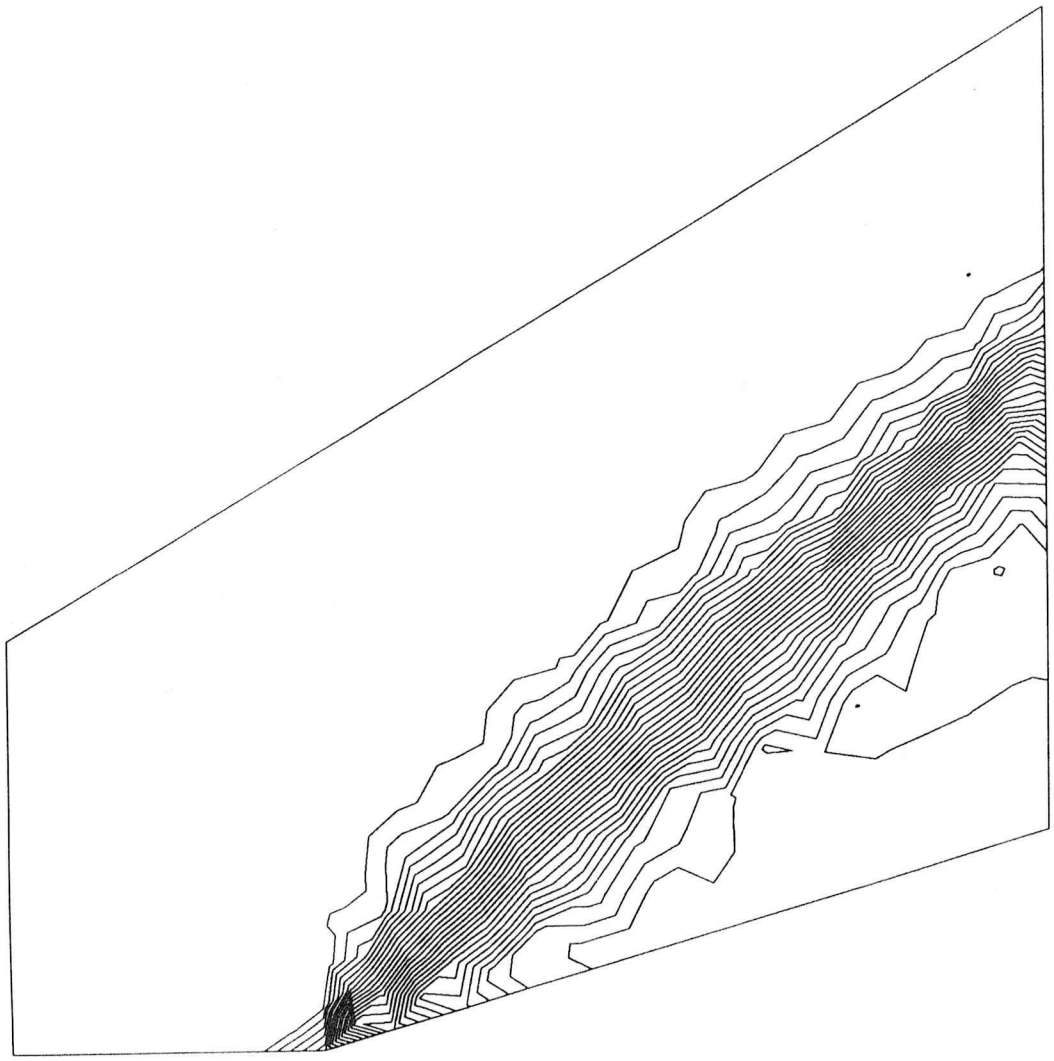
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Coarse Mesh
213 elements
131 nodes
47 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure A.10.2 Compression corner flow (coarse mesh) -- Pressure contours



DENSITY CONTOURS

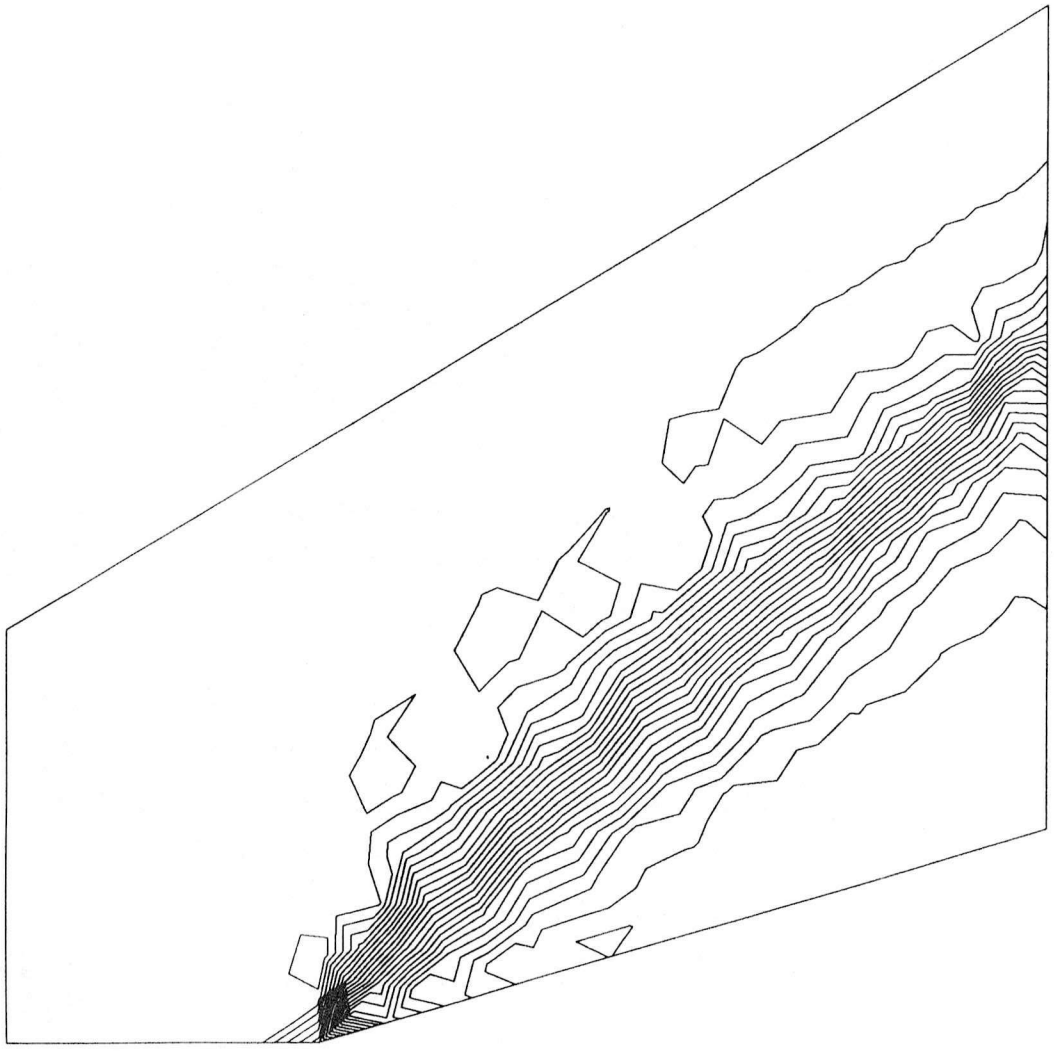
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Coarse Mesh
213 elements
131 nodes
47 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_{w} = 300K$

Figure A.10.3 Compression corner flow (coarse mesh) -- Density contours



MACH NUMBER CONTOURS

Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order , MUSCL
Mesh : Coarse Mesh
213 elements
131 nodes
47 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

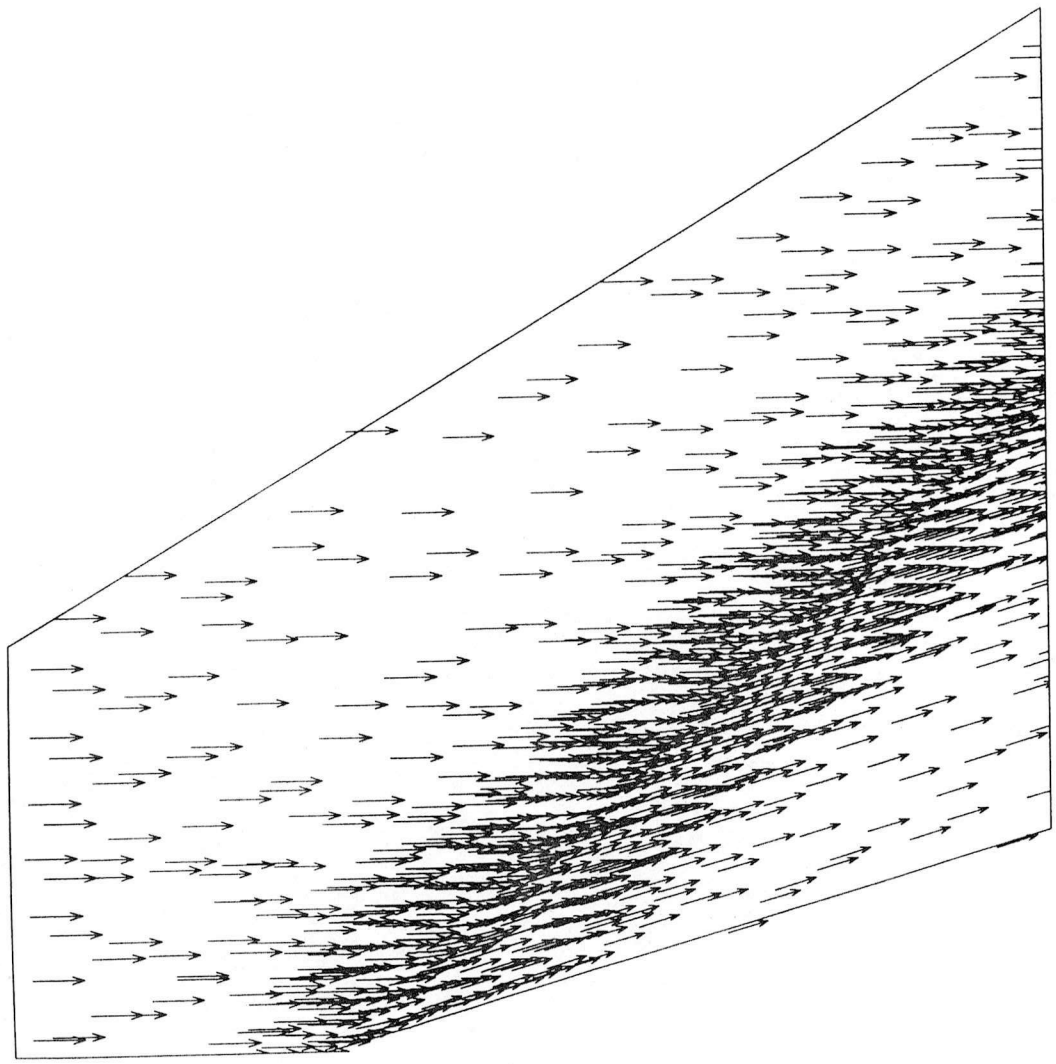
Figure A.10.4 Compression corner flow (coarse mesh) -- Mach number contours

APPENDIX B

SUPERSONIC COMPRESSION CORNER FLOW

INTERMEDIATE MESH

INVICID CASE



FLOW FIELD

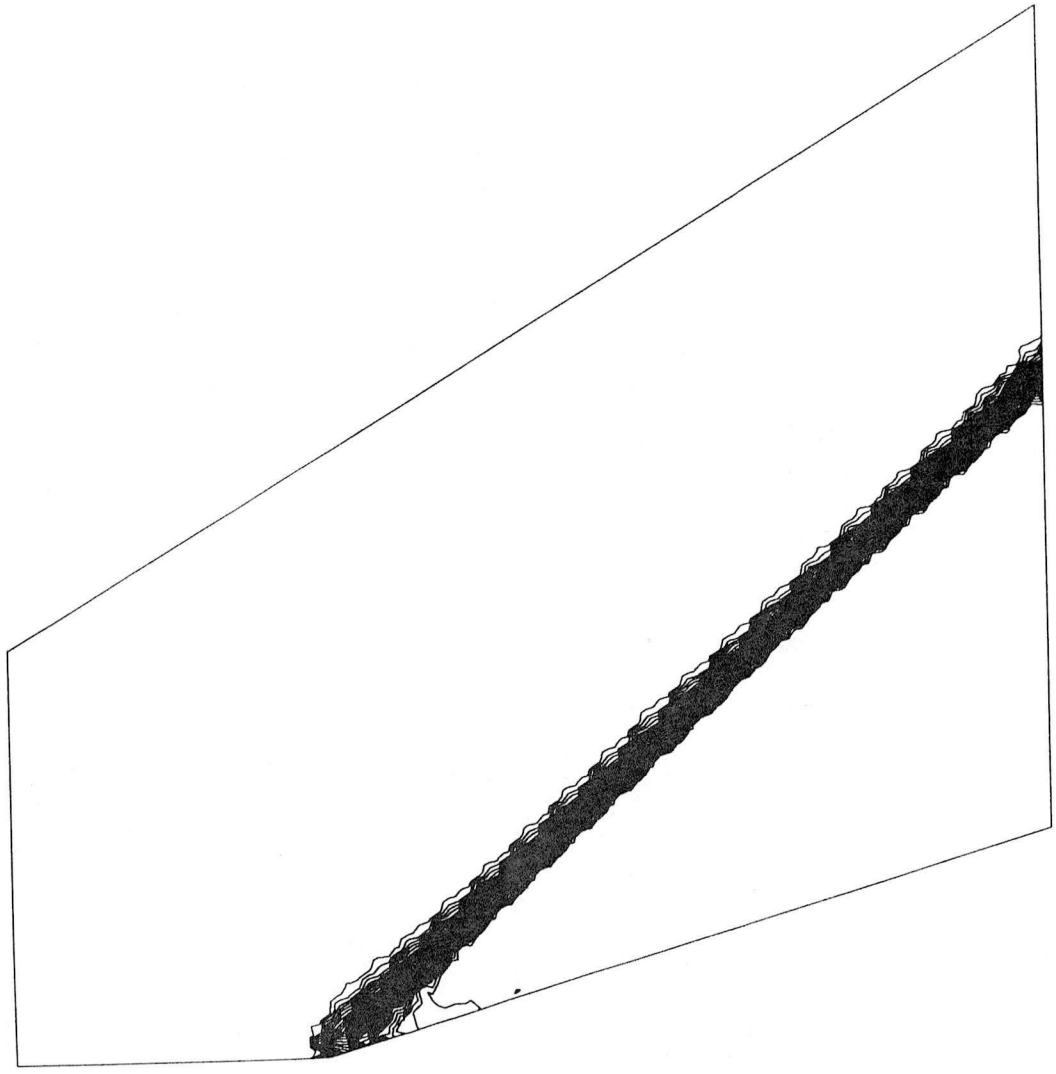
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Intermediate Mesh
1153 elements
615 nodes
75 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure B.8.1 Compression corner flow (intermediate mesh) -- Velocity field



PRESSURE CONTOURS

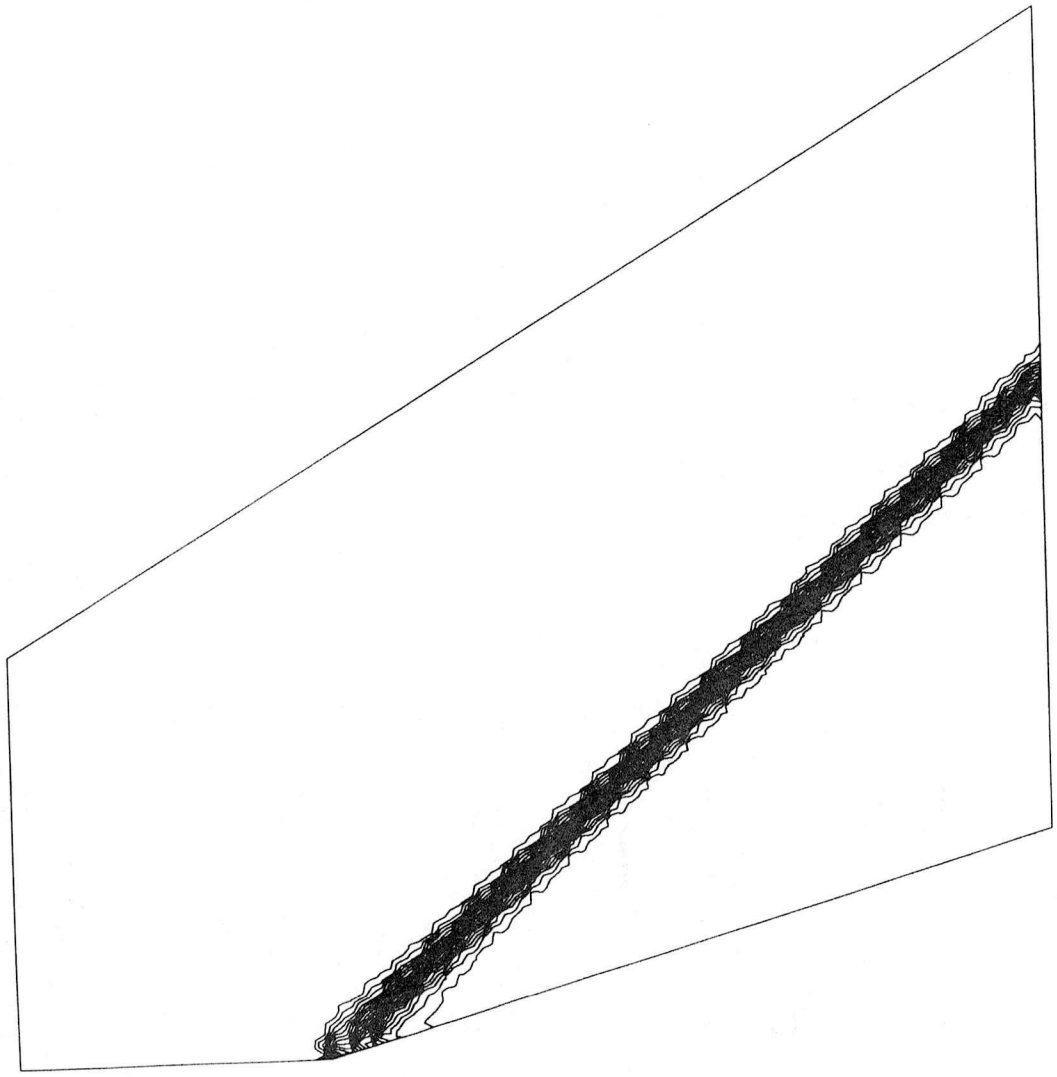
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Intermediate Mesh
1153 elements
615 nodes
75 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure B.8.2 Compression corner flow (intermediate mesh) -- Pressure contours



DENSITY CONTOURS

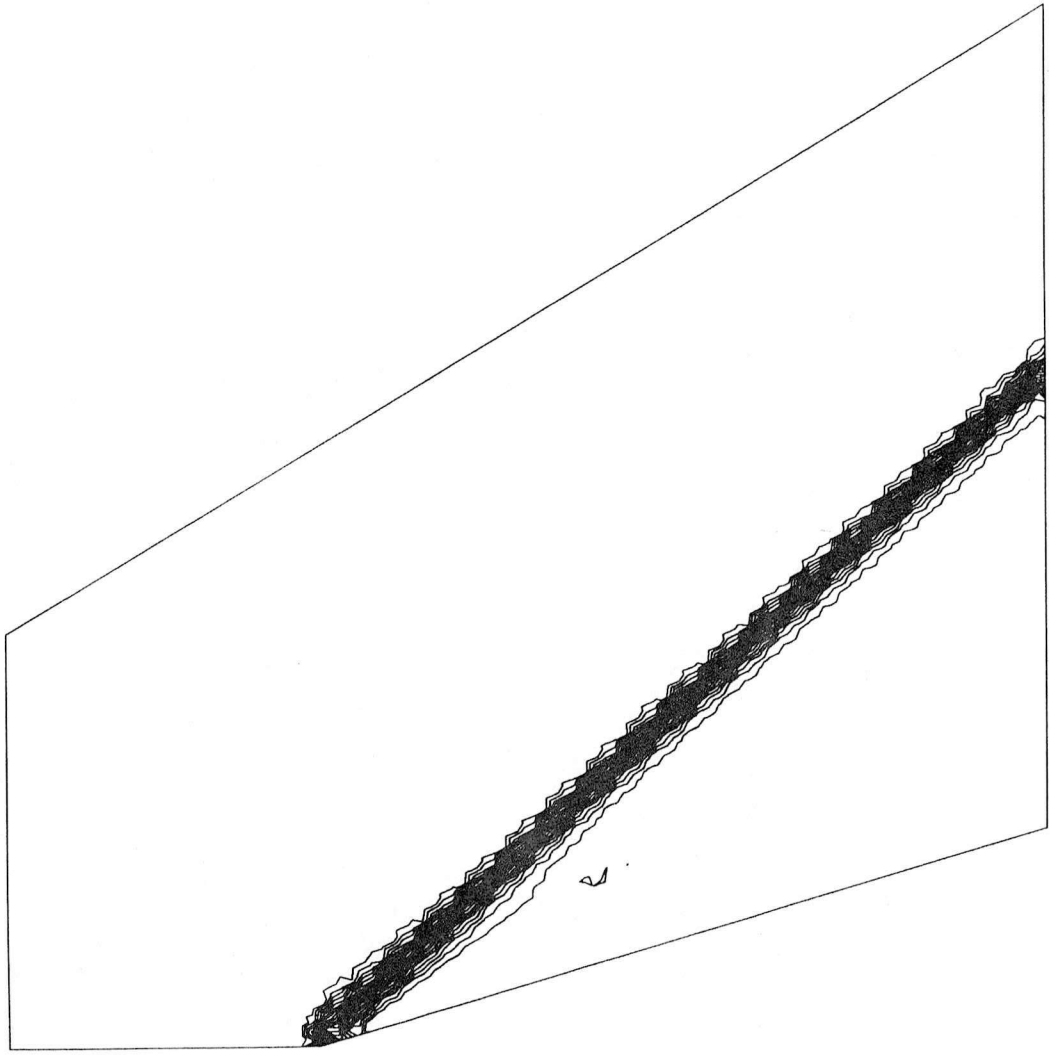
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Intermediate Mesh
1153 elements
615 nodes
75 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure B.8.3 Compression corner flow (intermediate mesh) -- Density contours



MACH NUMBER CONTOURS

Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Intermediate Mesh
1153 elements
615 nodes
75 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure B.8.4 Compression corner flow (intermediate mesh) -- Mach number contours

APPENDIX C

SUPERSONIC COMPRESSION CORNER FLOW

FINE MESH

INVICID CASE

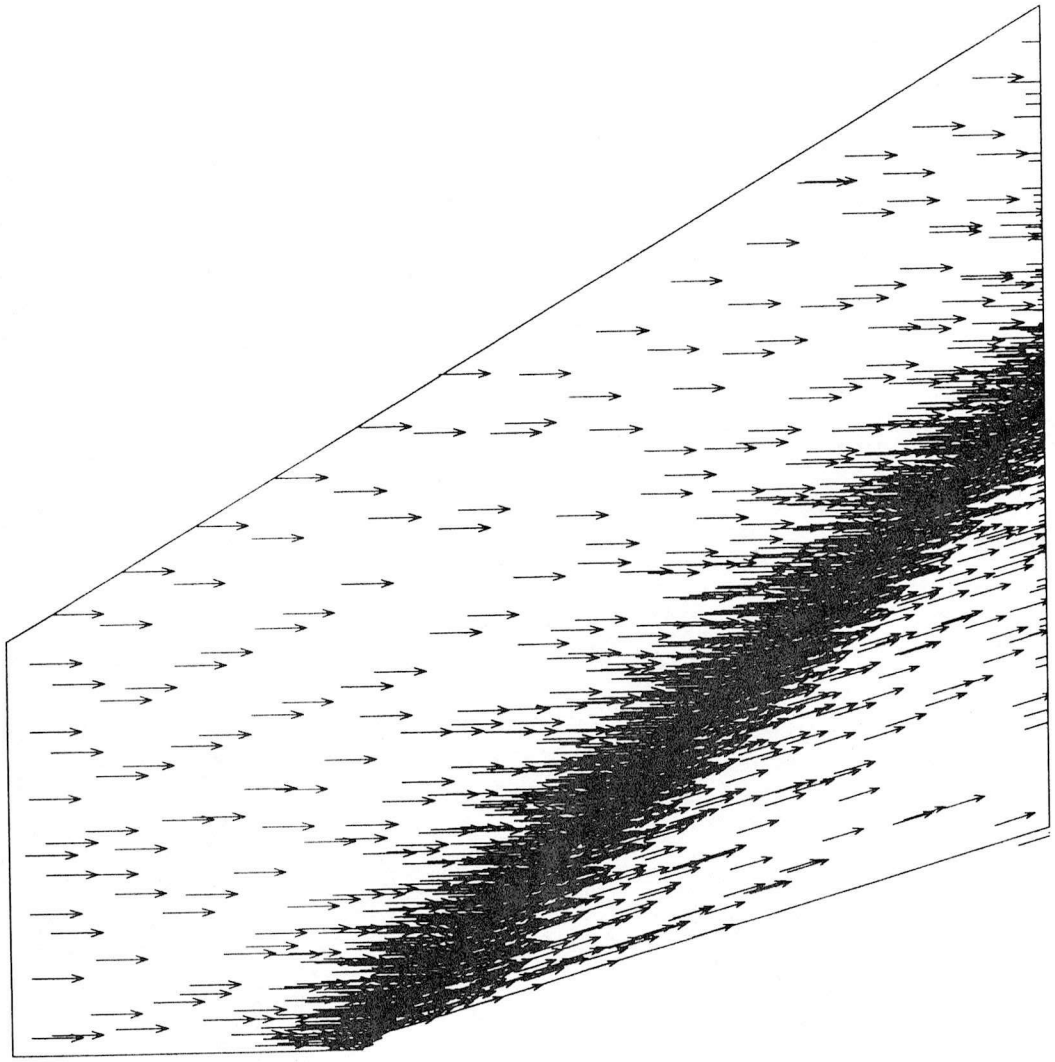


Figure C.6.1 Compression corner flow (fine mesh) -- Velocity field

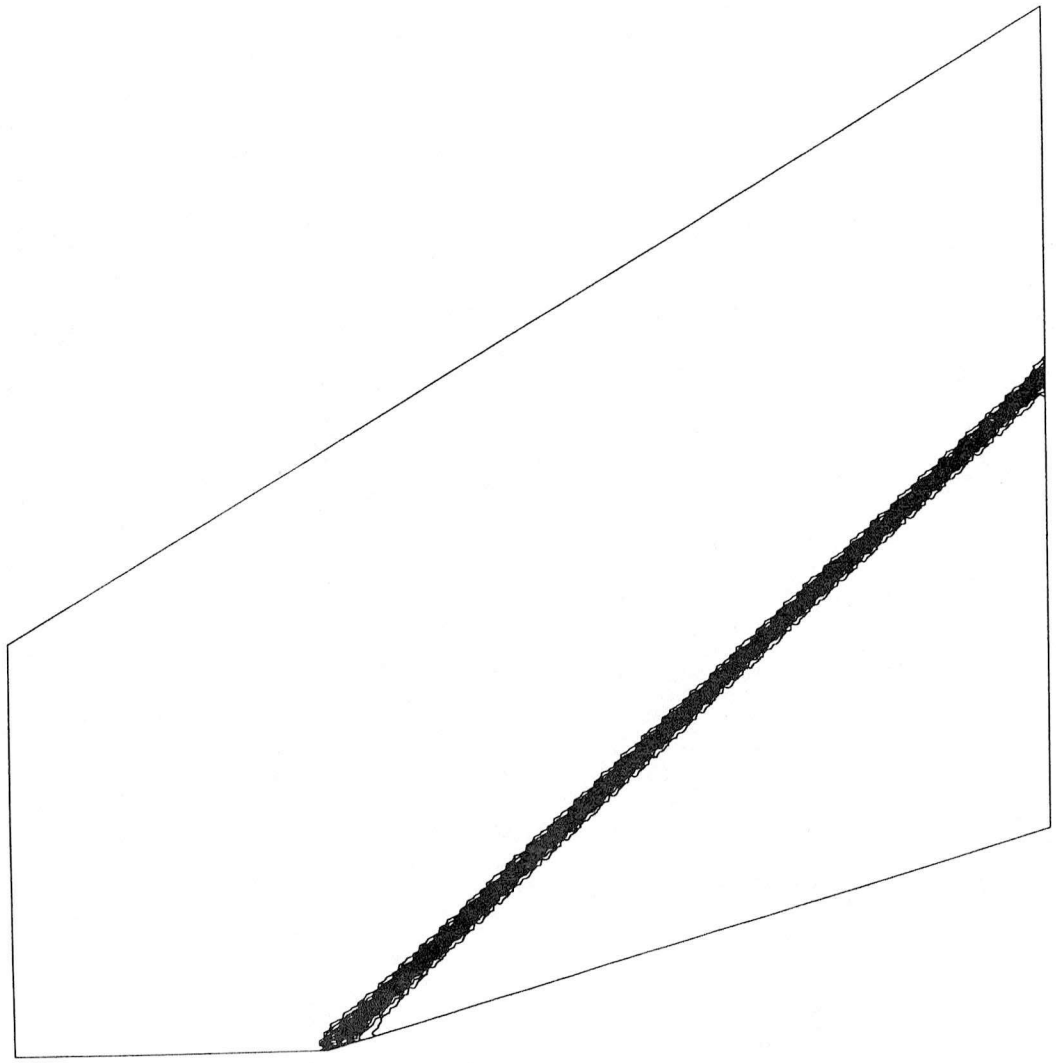
FLOW FIELD

Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_{w} = 300K$



PRESSURE CONTOURS

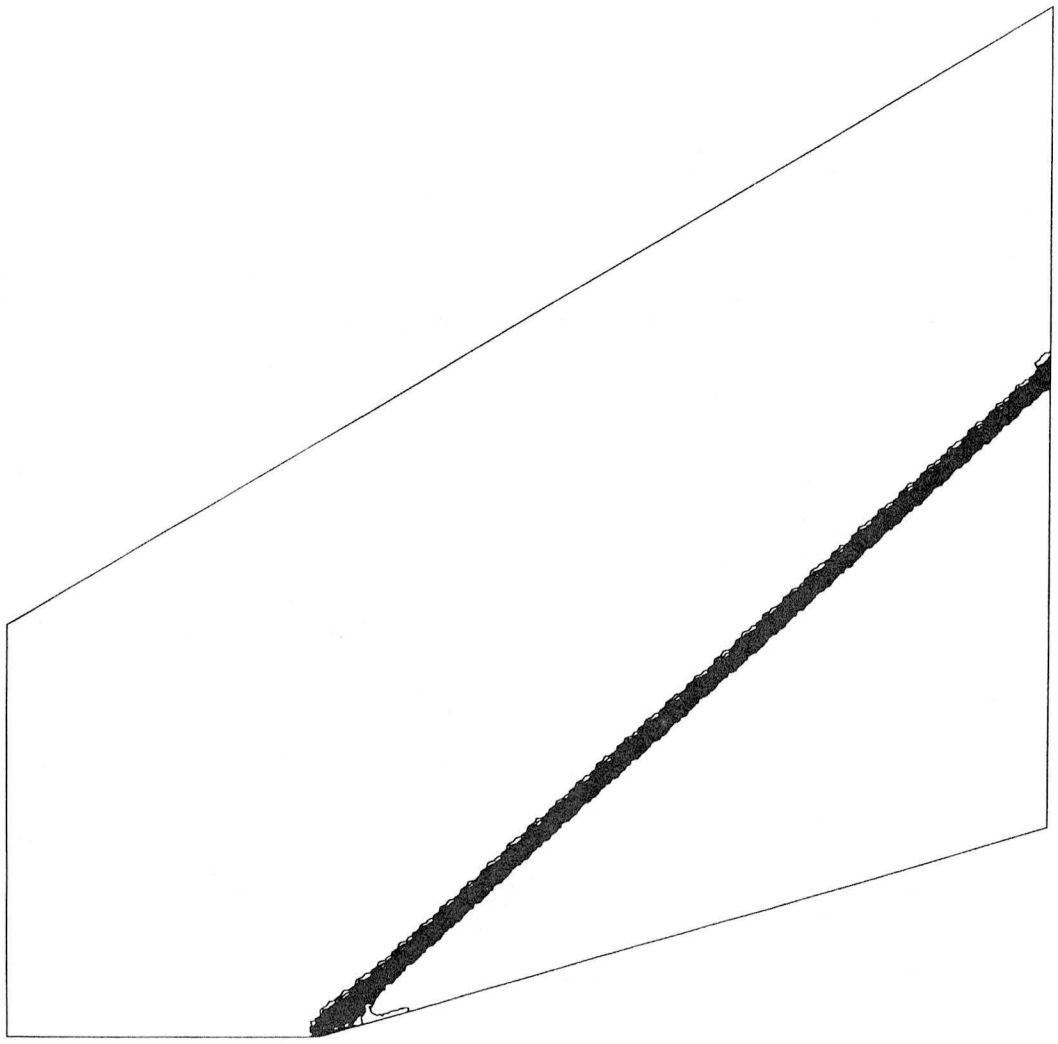
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_{w} = 300K$

Figure C.6.2 Compression corner flow (fine mesh) -- Pressure contours



DENSITY CONTOURS

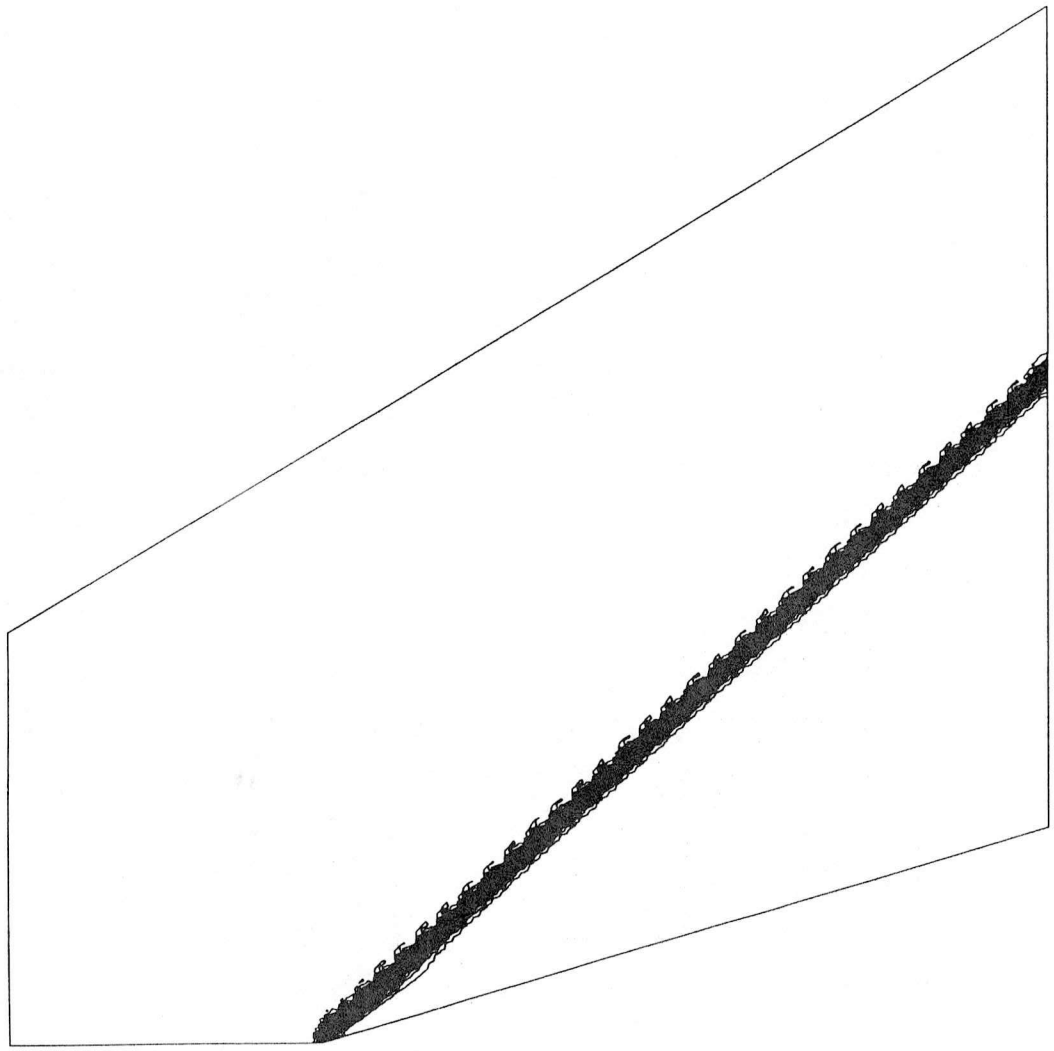
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure C.6.3 Compression corner flow (fine mesh) -- Density contours



MACH NUMBER CONTOURS

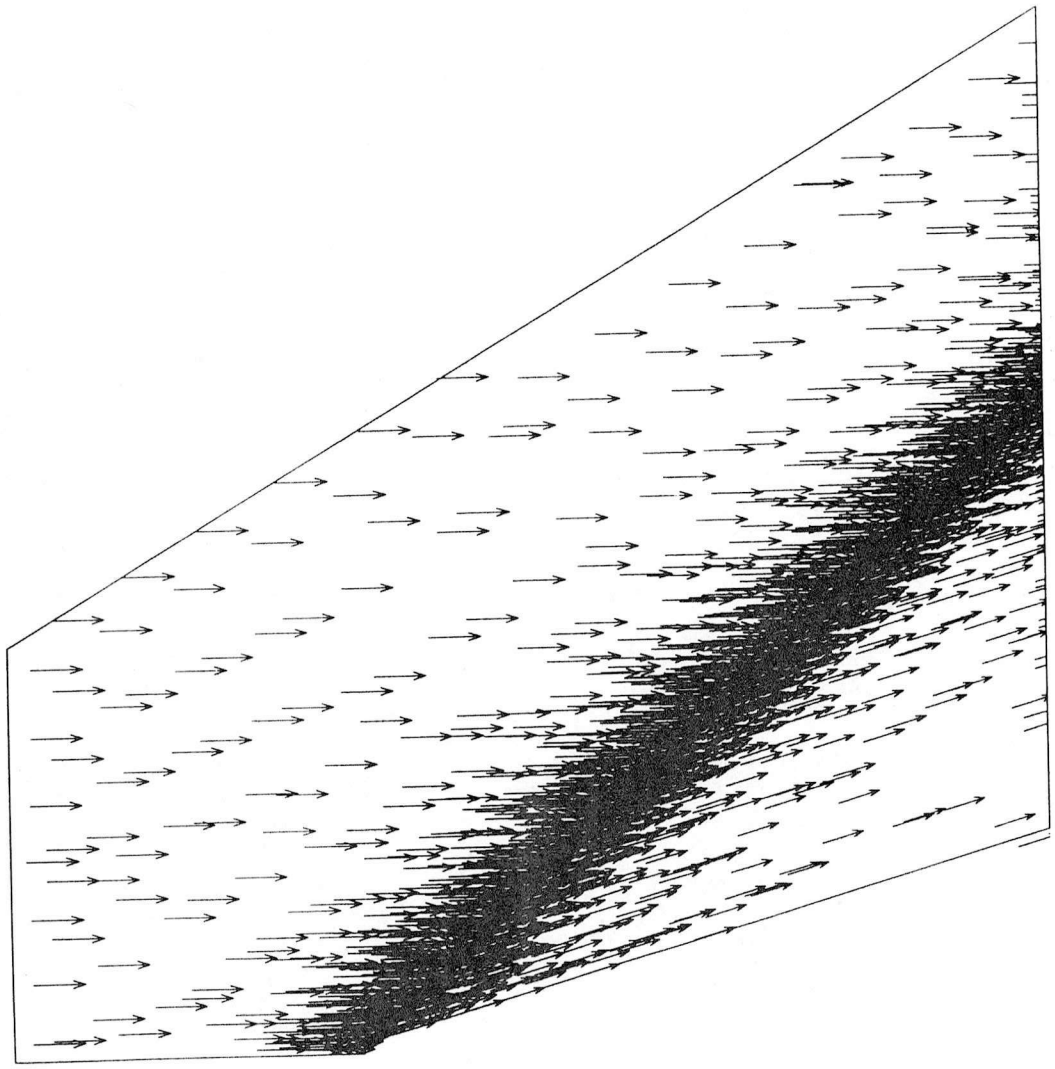
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Ooder : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_{w} = 300K$

Figure C.6.4 Compression corner flow (fine mesh) -- Mach number contours



FLOW FIELD

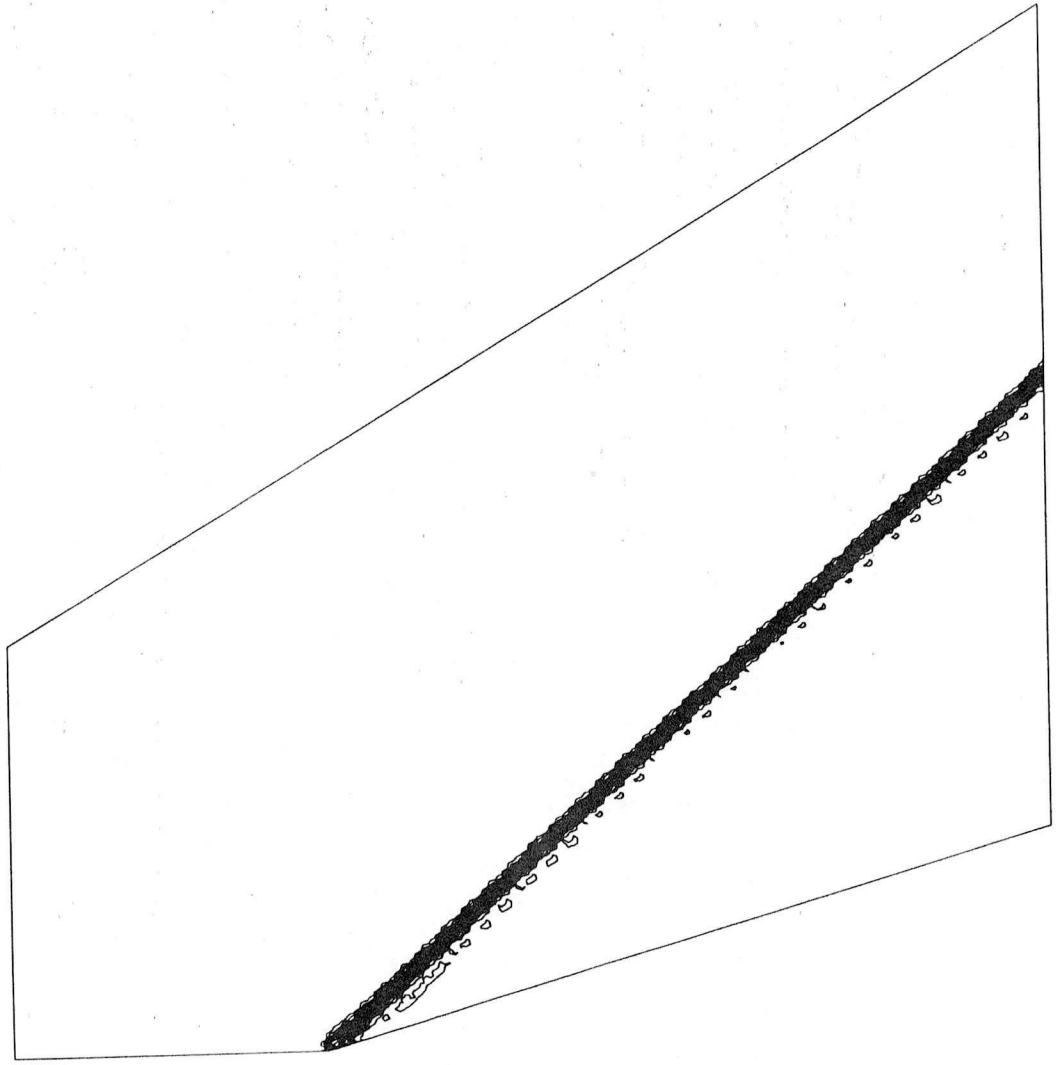
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, VAR
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure C.7.1 Compression corner flow (fine mesh) -- Velocity field



PRESSURE CONTOURS

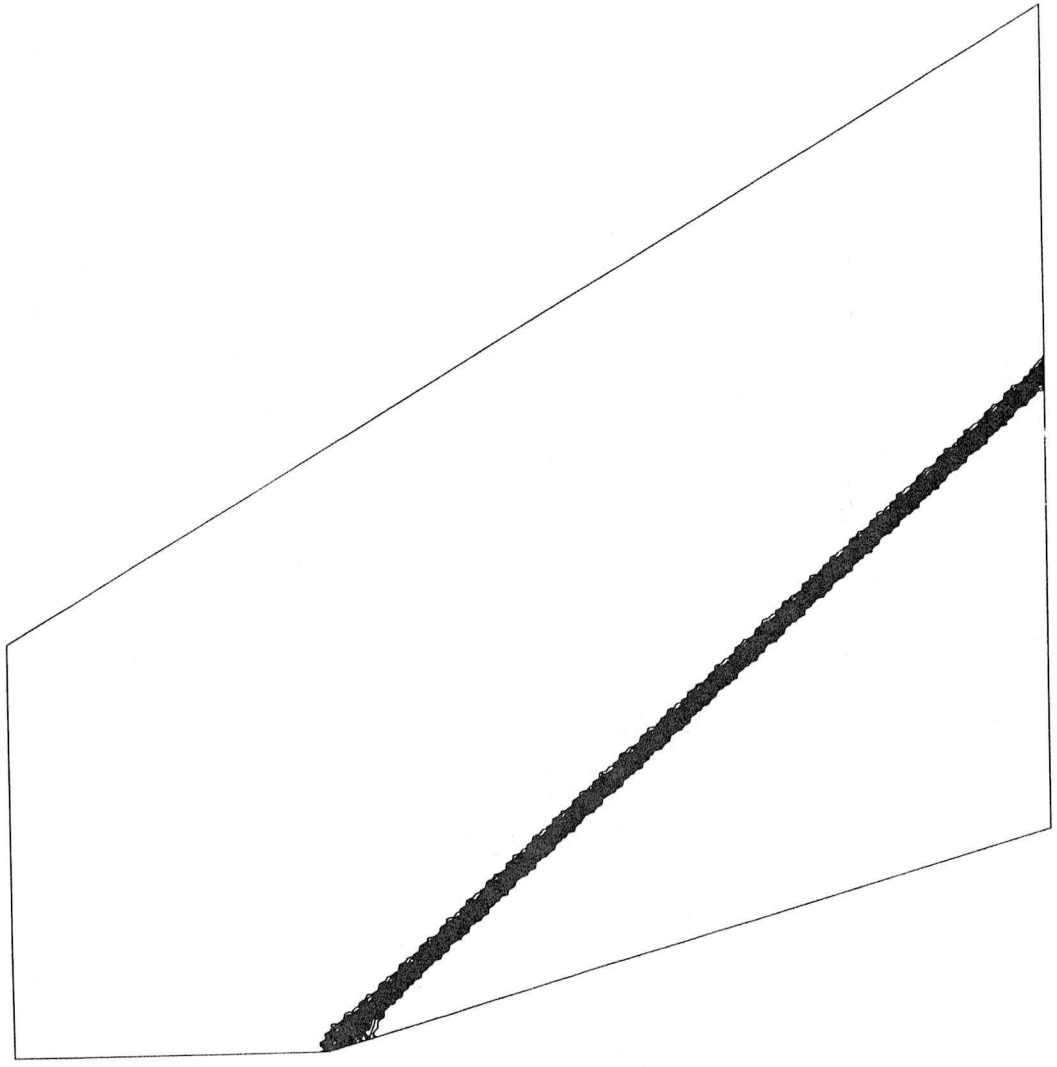
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, VAR
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure C.7.2 Compression corner flow (fine mesh) -- Pressure contours



DENSITY CONTOURS

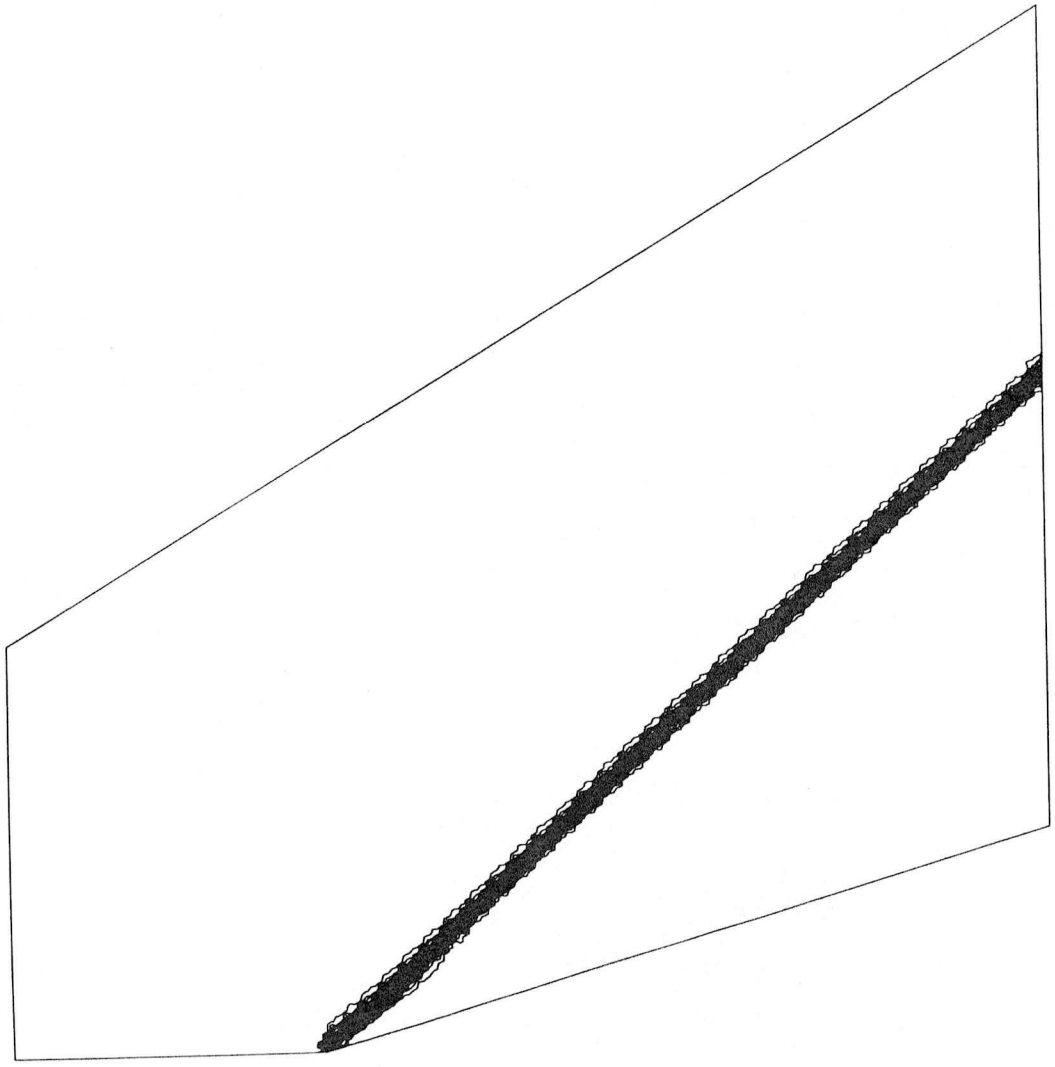
Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, VAR
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure C.7.3 Compression corner flow (fine mesh) -- Density contours



MACH NUMBER CONTOURS

Model: Compression Corner Flow
Code : Euler Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, VAR
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 120K$

$T_w = 300K$

Figure C.7.4 Compression corner flow (fine mesh) -- Mach number contours

APPENDIX D

SUPERSONIC COMPRESSION CORNER FLOW

FINE MESH

VISCOUS CASE

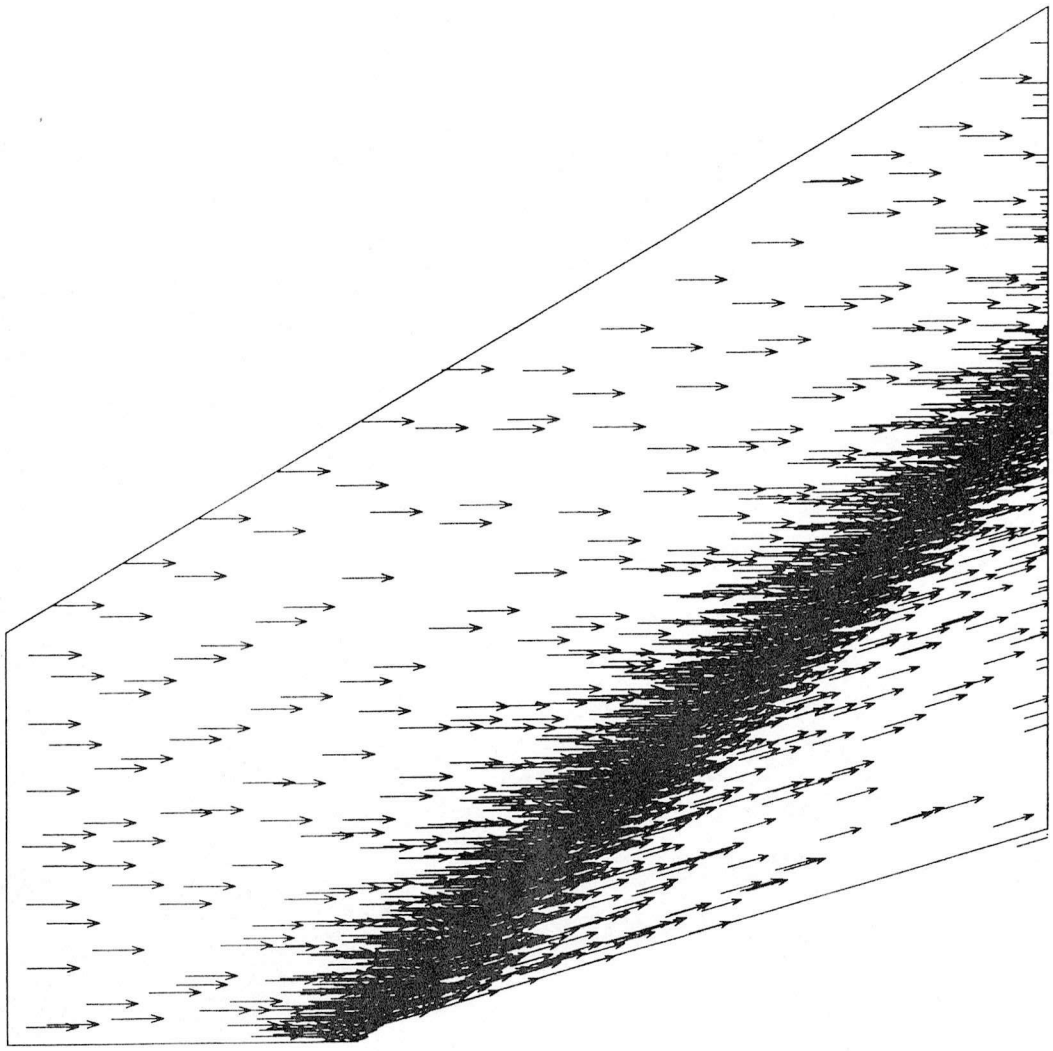


Figure D.2.1 Compression corner flow (fine mesh) -- Velocity field

FLOW FIELD

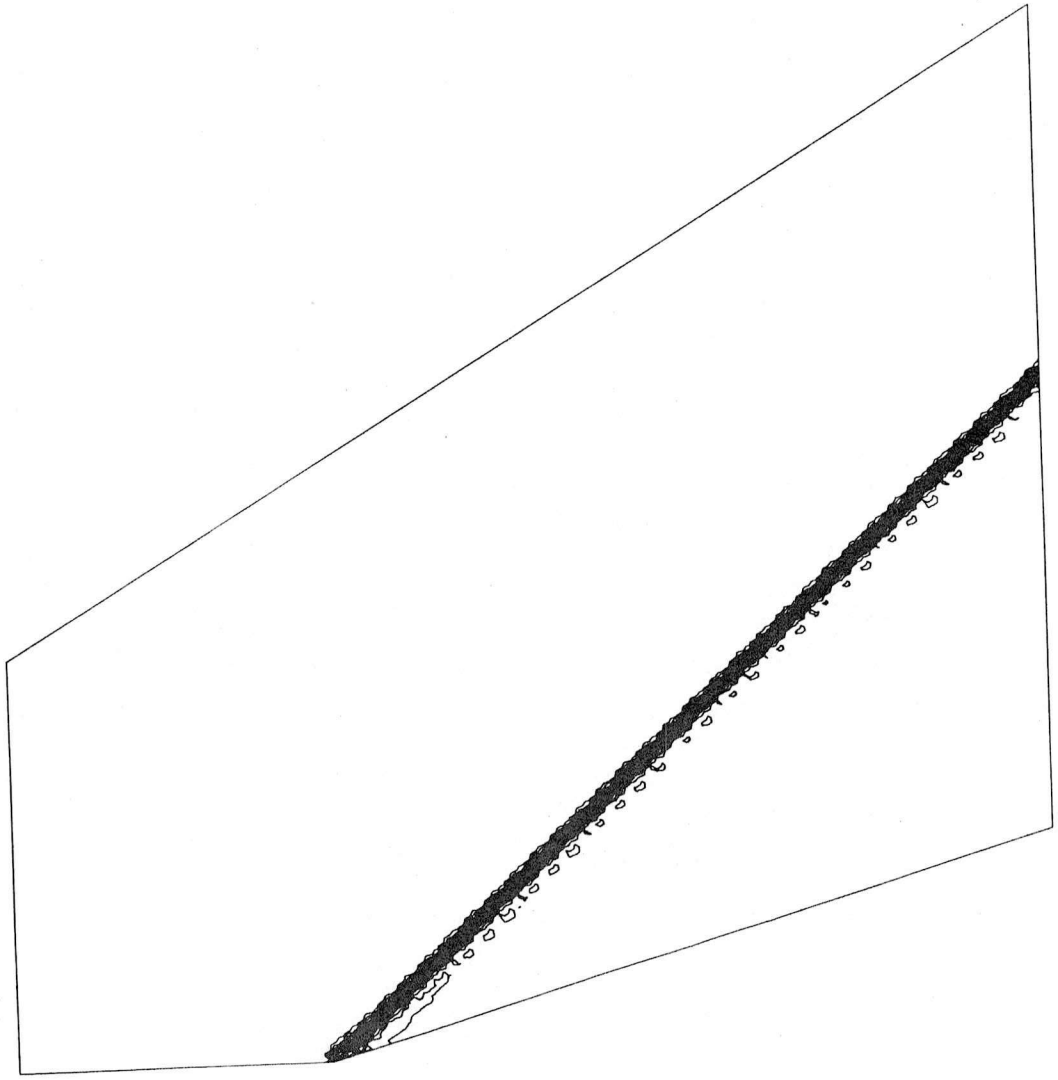
Model: Compression Corner Flow
Code : Navier-Stokes Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 300K$

$T_w = 300K$

$\mathrm{Re}_\infty = 0.65E+07$



PRESSURE CONTOURS

Model: Compression Corner Flow
Code : Navier-Stokes Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

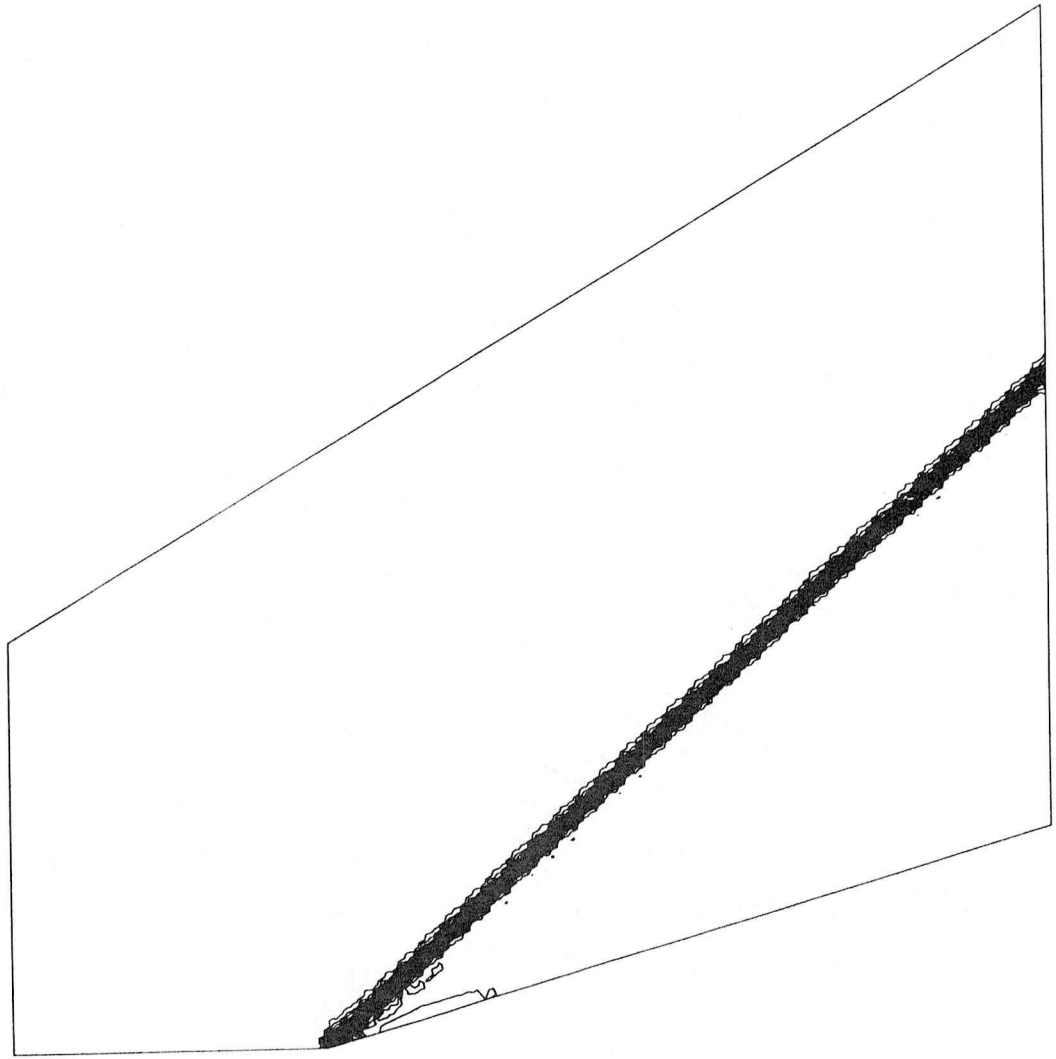
$M_{inf} = 2.2$

$T_{inf} = 300K$

$T_w = 300K$

$\mathrm{Re}_\infty = 0.65E+07$

Figure D.2.2 Compression corner flow (fine mesh) -- Pressure contours



DENSITY CONTOURS

Model: Compression Corner Flow
Code : Navier-Stokes Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

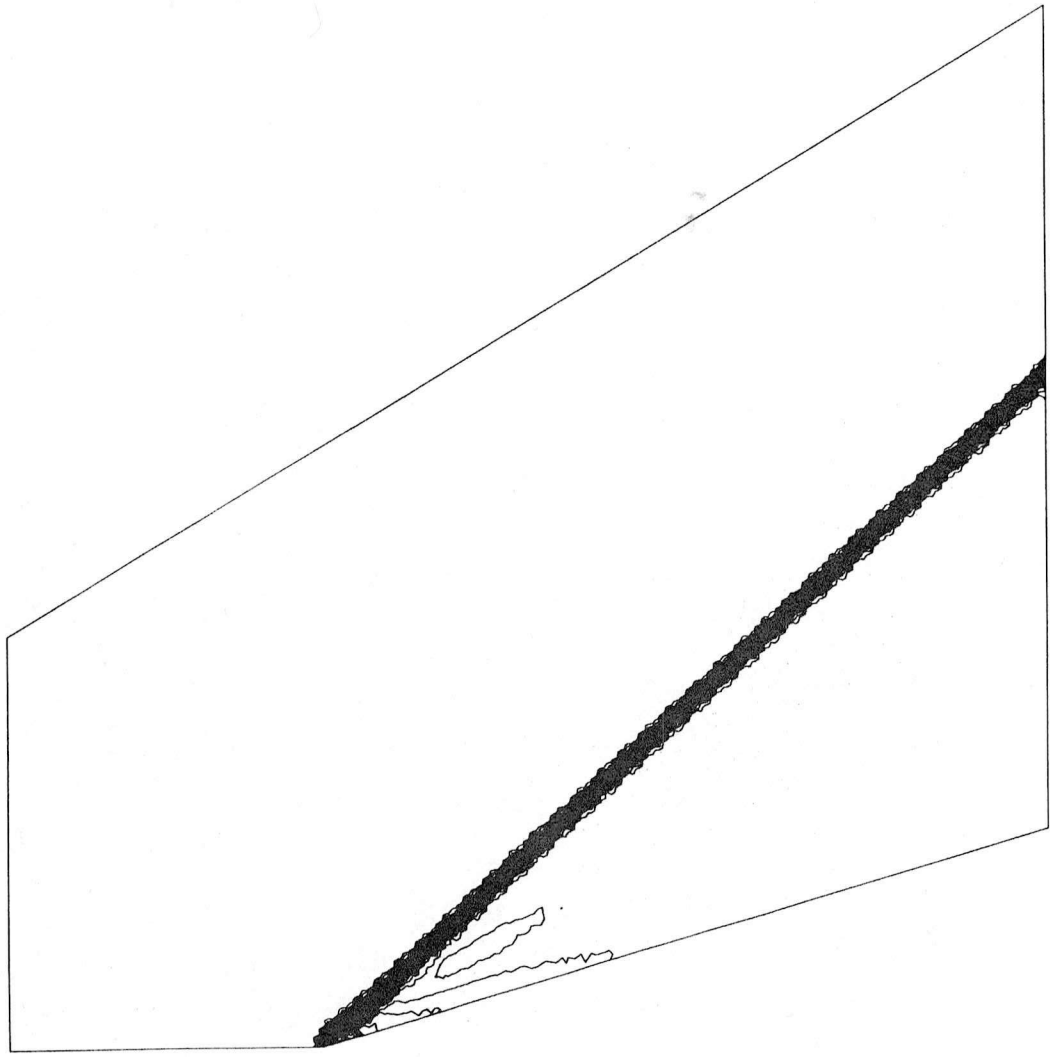
$M_{inf} = 2.2$

$T_{inf} = 300K$

$T_w = 300K$

$\mathrm{Re}_\infty = 0.65E+07$

Figure D.2.3 Compression corner flow (fine mesh) -- Density contours



MACH NUMBER CONTOURS

Model: Compression Corner Flow
Code : Navier-Stokes Equation
PGS scheme, double precision
Flux : Roe
Order : high-order, MUSCL
Mesh : Fine Mesh
2390 elements
1250 nodes
108 boundary points
Free stream condition :

$M_{inf} = 2.2$

$T_{inf} = 300K$

$T_w = 300K$

$Re_\infty = 0.65E+07$

Figure D.2.4 Compression corner flow (fine mesh) -- Mach number contours