SIERRA/Premo-A New General Purpose Compressible Flow Simulation Code

Thomas M. Smith[1]
Curtis C. Ober[2]
Alfred A. Lorber[3]
Sandia National Laboratories[4]
Albuquerque, NM, 87185

Abstract

This paper reports on the progress of a new compressible flow simulation code being developed at Sandia National Laboratories. The code called Premo is a CFD module that is part of a much larger multi-mechanics code framework called SIERRA. The goal of the Premo project is to deliver a general purpose CFD capability for designers and analysts of aerodynamics of flight vehicles. SIERRA provides unstructured mesh data services common to many computational mechanics codes. Utilizing the framework allows for rapid development of physics modules. In addition, the SIERRA framework provides an interface that will make it possible to conduct multi-physics simulations between physics modules to solve such problems as aero-thermal vehicle heating and aero elasticity in the future. Current code capabilities are demonstrated by solving one-, two- and three-dimensional flow problems.

Introduction

In order to meet the needs of the Stockpile Stewardship Program and the Accelerated Strategic Computing Initiative (ASCI), a new code development project is underway at Sandia National Laboratories. SIERRA/Premo is the name given to this new compressible flow simulation code. The code is a module of a larger multi-mechanics framework called SIERRA [Edw01]. The goal of the project is to provide a production simulation tool to be used to analyze compressible flows of aerodynamic vehicles. Flow regimes of interest range from transonic all the way up to hypersonic, steady-state and transient.

Premo is written in the C++ language and builds upon the object-oriented design in SIERRA to provide a flexible, maintainable easy to use and robust CFD code package. The SIERRA framework provides data services for application modules which include; I/O, domain decomposition and parallel process management, mesh adaptivity, load balancing, multiphysics code coupling and nonlinear and linear solver libraries. Fast prototyping new models is an important objective of the project. This is made possible by utilizing the framework data services. Developers of modules are able to concentrate on writing algorithms specific to the physics, thus reducing the amount of work.

The strategy for development has been to write the spatial discretization and numerical algorithms kernel and then add functionality. The final version of Premo will include solvers for a wide range of governing equations such as Euler and Navier-Stokes, Reynolds averaged Navier-Stokes and the equations for chemically reacting species. Arbitrary body motion and relative body motion simulations will also be possible.

In the next section, the formulation is described followed by computational results that demonstrate current capabilities. In the current state of development, hexahedral or tetrahedral element meshes can be used. One of the goals of the project is to extend the code to handle mixed-element meshes.

Formulation

This section describes the numerical formulation used to approximate solutions to the governing equations on unstructured meshes. The governing equations are discretized using a finite-volume formulation. The main aspects of the numerical scheme are discussed including; variable reconstruction, advection, diffusion, boundary conditions and time integration.

Governing Equations

Restricting the discussion to a single component ideal gas, the governing equations are the Navier-Stokes equations written in integral form

$$\int_{\Omega} \frac{\partial \mathbf{W}}{\partial t} dV + \oint_{\Gamma} \left(\mathbf{F}_e^{\alpha} + \mathbf{F}_v^{\alpha} \right) \bullet \mathbf{n} dS = 0 \, .$$

1 Senior Member of Technical Staff, Member AIAA, E-mail: tmsmith@sandia.gov
2 Principal Member of Technical Staff, Member AIAA, E-mail: ccober@sandia.gov
3 Senior Member of Technical Staff, E-mail: aalorbe@sandia.gov
4 Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

The volume of the domain is represented by Ω and is bounded by the surface Γ. The equations describe the evolution of the state vector, **W,** containing mass, momentum and energy,

$$\mathbf{W}=\begin{Bmatrix}\rho\\ \rho u_1\\ \rho u_2\\ \rho u_3\\ \rho E\end{Bmatrix}\ \mathbf{F}_e^\alpha=\begin{Bmatrix}\rho u_\alpha\\ \rho u_\alpha u_1+\delta_{\alpha 1}p\\ \rho u_\alpha u_2+\delta_{\alpha 2}p\\ \rho u_\alpha u_3+\delta_{\alpha 3}p\\ (\rho E+p)u_\alpha\end{Bmatrix}\ \mathbf{F}_v^\alpha=\begin{Bmatrix}0\\ \tau_{1\alpha}\\ \tau_{2\alpha}\\ \tau_{3\alpha}\\ q_\alpha-u_\beta\tau_{\alpha\beta}\end{Bmatrix}.$$

The inviscid flux vector is $\mathbf{F}_e^\alpha$, and the viscous flux vector is $\mathbf{F}_v^\alpha$. Greek subscripts $(\alpha,\beta=1,2,3)$ imply coordinate directions (x,y,z) and $\delta_{\alpha\beta}$ is the Kronecker delta. This system of equations is closed by an equation-of-state, $p=\rho RT$ and initial and boundary conditions. In the above system, u_α is the velocity component in the α coordinate direction, p is the pressure, ρ is the mass density, $E=e+u_\alpha u_\alpha/2$, is the total energy per unit mass (summation on α implied), T is the temperature and R is the gas constant. For a single component gas, internal energy is defined, $e=C_vT=p/\left[(\gamma-1)\rho\right]$, C_v is the specific heat at constant volume and γ is the ratio of specific heats. It is also convenient to introduce the specific enthalpy, $h=e+p/\rho$ and the total enthalpy per unit mass as $H=h+(u^2+v^2+w^2)/2=E+p/\rho$.

The primitive state vector is defined as $\mathbf{U}=[\rho,u,v,w,p]^T$. A change in variables from conservative to primitive is done by the operation $\mathbf{W}=[\mathbf{M}]\mathbf{U}$ where $[\mathbf{M}]$ is given by,

$$[\mathbf{M}]=\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ u & \rho & 0 & 0 & 0\\ v & 0 & \rho & 0 & 0\\ w & 0 & 0 & \rho & 0\\ \frac{q^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma-1}\end{bmatrix}.$$

The viscous stress tensor has the form

$$\tau_{\alpha\beta}=\mu\left(\frac{\partial u_\alpha}{\partial x_\beta}+\frac{\partial u_\beta}{\partial x_\alpha}\right)-\frac{2}{3}\mu\delta_{\alpha\beta}\frac{\partial u_\gamma}{\partial x_\gamma}$$

and the heat-flux vector is

$$q_\alpha=-\kappa\frac{\partial T}{\partial x_\alpha}.$$

Constitutive equations are required to specify the dynamic viscosity $\mu(T)$, and the heat conduction coefficient $\kappa(T)$.

Boundary conditions for Euler equations are $\mathbf{u}\bullet\mathbf{n}=0$ on impermeable surfaces and symmetry planes and subsonic/supersonic inflow/outflow. The slip condition results in the following specification of the fluxes,

$$\mathbf{F}_e^{wall}=\begin{Bmatrix}0\\ p\hat{n}_x\\ p\hat{n}_y\\ p\hat{n}_z\\ 0\end{Bmatrix}. \qquad (1.1)$$

Subsonic/supersonic inflow/outflow boundary conditions are based the characteristics of the Euler equations [Whi93]. For Navier-Stokes equations, in addition to the Euler equation boundary conditions, no-slip velocity, $\mathbf{u}=0$, and either constant temperature or adiabatic $\frac{\partial T}{\partial n}=0$ conditions are prescribed on solid surfaces. Here, $\mathbf{u}=u_\alpha\hat{i}_\alpha$ is the velocity vector and $\mathbf{n}$ is a unit vector normal to the surface. Specification of the no-slip and constant temperature conditions requires a change in variables from conservative to primitive via $\mathbf{U}=[\mathbf{M}]^{-1}\mathbf{W}$.

Spatial Discretization

The SIERRA framework defines a computational domain by a mesh of finite elements. Data describing the mesh are; nodes, edges, faces, elements and the related connectivities. State vectors are defined at nodes. Control-volumes are constructed for each node and bounding control-surfaces around each control-volume. Control-volumes and control-surfaces are defined by the median dual of the element mesh. The dual mesh is constructed by subdividing elements with sub-control surfaces that intersect element surface centroids, edge midpoints and the element centroid. For example, an eight-node hexahedral when sub-divided, contains eight sub-control volumes and twelve sub-control surfaces. In two dimensions, the sub-control surfaces are line segments joining element centroids with edge midpoints. Examples are shown by dashed lines in Figures 1 and 2. The sum of sub-control volumes (*scv*) sharing a common node define a control-volume (CV) for node *i*

$$\Delta V_i = \sum_{j \in N(i)} scv_j$$

where *N(i)* is the set of sub-control volumes sharing node *i*. This construction results in non-overlapping, space filling control-volumes. The summation of sub-control surfaces (*scs*) sharing an edge (distinguished by the edge node pair, *ij*) define a bounding control-surface. An area vector S_{ij}^{α}, is defined as the sum of projected areas of all the sub-control surfaces that share edge *ij*, normal to the α co-ordinate direction,

$$\mathbf{S}_{ij} = S_{ij}^{\alpha} = \sum_{k \in E(ij)} scs_k^{\alpha}$$

where the *k*th *scs* touches edge *ij* and *E(ij)* is the set of sub-control surfaces touching edge *ij*. We require that the summation of oriented areas be identically zero,

$$\sum_{j \in NE(i)} \sum_{\alpha} S_{ij}^{\alpha} = 0$$

where the set of edges that contain node *i* is *NE(i)*. The control-surface normal vector is defined as

$$\mathbf{n}_{ij} = n_{ij}^{\alpha} = \frac{S_{ij}^{\alpha}}{\left| S_{ij}^{\alpha} \right|},$$

and is oriented in the direction from node *i* to node *j*. We also define area and normal vectors $\mathbf{S}_i = S_i^{\alpha}$, $\mathbf{n}_i = n_i^{\alpha}$ on external boundary surfaces.

Cell-Centered Versus Node-Centered Spatial Discretization

The discretization described above results in a node-centered scheme (i.e., control-volumes are constructed around mesh nodes). The flux balance equations for each control-volume are constructed by looping over edges and external surface nodes. Alternatively, a cell-centered spatial discretization may be constructed by treating each element of the mesh as a control-volume. In this case, the unknown variables would be associated with the centroid of the element and surface integrals would be calculated on element faces. The control-volume would equal the element volume. There are efficiency issues that arise as a result of choosing cell-centered vs. node-centered discretization. The main issue is the number of control-volume faces that must be processed in the cell-centered case vs. the number of edges that must be processed in the node-centered case. Barth [Bar91] showed that there are twice as many faces as edges for a given tetrahedral mesh in three dimensions. A significant amount of the computational cost associated with finite volume schemes comes from the flux calculations. Therefore, the edge-based scheme requires less effort than the cell-centered scheme. For, a hexahedral mesh, the difference in effort is much less significant.

Edge-based schemes have a distinct advantage when the mesh is heterogeneous. In this case, the algorithms for cell-centered schemes must adapt locally to the element topology, while the edge-based algorithms are transparent to the element topology.

One important drawback of the node-centered discretization is that it is difficult to extend to higher than second order accuracy. This is due to the requirement for increased connectivity and more accurate quadrature.

Flux Quadrature

The semi-discrete form of the governing equations require approximations to volume and surface integrals. Let $\bar{\mathbf{W}}$ represent the volume average of $\mathbf{W}$. Volume integrals are approximated by

$$\bar{\mathbf{W}}_i \Delta V_i = \int_{V_i} \mathbf{W} dV .$$

Consider a node labeled *i* that shares an edge with node *j*. Now let $\mathbf{F}_{ij}^{\alpha} = \mathbf{F}_{eij}^{\alpha} + \mathbf{F}_{vij}^{\alpha}$ be the total flux vector evaluated at the midpoint on edge *ij*. Approximating the surface integrals with the midpoint rule yields the semi-discrete equation for $\bar{\mathbf{W}}_i$

$$\frac{\partial \bar{\mathbf{W}}_i}{\partial t} \Delta V_i + \sum_{j \in NE(i)} \mathbf{F}_{ij} \bullet \mathbf{n}_{ij} \left| \mathbf{S}_{ij} \right| = 0, \quad \forall i \in \Omega(V)$$

$$\frac{\partial \bar{\mathbf{W}}_i}{\partial t} \Delta V_i + \sum_{j \in NE(i)} \mathbf{F}_{ij} \bullet \mathbf{n}_{.ij} \left| \mathbf{S}_{ij} \right| + \mathbf{F}_i \bullet \mathbf{n}_i \left| \mathbf{S}_i \right| = 0, \quad \forall i \in \Gamma(S) .$$

Note that algebraic conservation (excluding inflow/outflow boundary fluxes) is assured by recognizing that the fluxes crossing control surfaces cancel exactly

$$\mathbf{F}_{ij} \bullet \mathbf{n}_{ij} \left| \mathbf{S}_{ij} \right| = -\mathbf{F}_{ji} \bullet \mathbf{n}_{ji} \left| \mathbf{S}_{ji} \right| .$$

Variable Reconstruction and Limited Gradients

Higher-order upwind based control-volume schemes are generally flux limited or gradient limited. A detailed discussion of each is given by LeVeque [LeV92]. In either case, reconstruction is necessary to represent unknown data at control surface using nodal data. In the present case, gradient limiting is used to achieve higher-order. One convenience that comes as a

result of gradient limiting is that the gradients of primitive variables are reused to construct the viscous terms. Evaluation of the flux function $\mathbf{F}_{ij}^{\alpha}$, requires that the state data be reconstructed at control surfaces (edge midpoints). State variables within the CV are assumed to vary continuously. At control-surfaces, variables are discontinuous. In order to resolve discontinuities a Riemann solver is used. The Riemann solver requires the definition of a left (-) and right (+) state at each interface. A piece-wise constant approximation of the left state (of any primitive variable *u*) is obtained from the state at node *i*, and the right state from node *j*

$$\begin{aligned} u_{ij}^{-} &= u_i \\ u_{ij}^{+} &= u_j \end{aligned} .$$

Higher-order spatial accuracy is achieved by a multi-dimensional MUSCL extrapolation of the variables to the interface [van77] [Bar89]. A piece-wise linear approximation is obtained from

$$\begin{aligned} u_{ij}^{-} &= u_i + \frac{1}{2}\nabla u_i \bullet \Delta \mathbf{r}_{ij} \\ u_{ij}^{+} &= u_j - \frac{1}{2}\nabla u_j \bullet \Delta \mathbf{r}_{ij} \end{aligned}$$

where ∇u_i is the averaged gradient of *u* in control volume *i*, $\Delta \mathbf{r}_{ij} = \left(\mathbf{r}_j - \mathbf{r}_i\right)$ is the distance from node *i* to node *j*. We note that this particular choice of reconstruction is equivalent to Fromm's scheme which is part of a two-parameter family of reconstructions. A limiter function, $\Phi_i \in [0,1]$, limits ∇u_i such that u_{ij}^{-} is not a local minimum or maximum. To calculate Φ_i, the minimums and maximums of the nodal values compared to its neighbors are computed

$$\begin{aligned} u_i^{\max} &= \max_{j \in NE(i)} (u_i, u_j) \\ u_i^{\min} &= \min_{j \in NE(i)} (u_i, u_j) \\ \Delta_1^{\max} &= u_i^{\max} - u_i \\ \Delta_1^{\min} &= u_i^{\min} - u_i \\ \Delta_2 &= u_{ij} - u_i \end{aligned} .$$

The limit function must enforce the condition, $u_i^{\min} \le u(x^{\alpha}) \le u_i^{\max}$, everywhere within the control volume. The extrapolated values are calculated without the limiter and the edge value ϕ_{ij}, is defined such that

$$\phi_{ij} = \begin{Bmatrix} \min\left(1, \dfrac{\Delta_1^{\max}}{\Delta_2}\right), & \text{if } \Delta_2 > 0 \\ \min\left(1, \dfrac{\Delta_1^{\min}}{\Delta_2}\right), & \text{if } \Delta_2 < 0 \\ 1, & \text{if } \Delta_2 = 0 \end{Bmatrix}.$$

Finally, the limiter for control-volume *i* is defined as the minimum of all the edge limiter values, $\Phi_i = \min_{j \in NE(i)} \left(\phi_{ij}\right)$.

The final reconstruction becomes

$$\begin{aligned} u_{ij}^{-} &= u_i + \frac{1}{2}\Phi_i \nabla u_i \bullet \Delta \mathbf{r}_{ij} \\ u_{ij}^{+} &= u_j - \frac{1}{2}\Phi_j \nabla u_j \bullet \Delta \mathbf{r}_{ij} \end{aligned}$$

This limiter was developed by Barth and Jespersen [Bar89]. When applied to steady flows, this limiter can produce "*chatter*" in the results. Modifications that delay the limiter until steep gradients are encountered have been discussed in Hosangadi et al. [Hos96]. Venkatakrishnan [Ven95] has proposed modifications that replace the discontinuous limit function with a smoothly varying function. While the new limiter does not strictly enforce monotinicity of the reconstructed fluxes, it has shown better convergence properties than the Barth-Jesperson limiter

$$\phi_{ij} = \begin{Bmatrix} \dfrac{1}{\Delta_2}\dfrac{\left((\Delta_1^{\max})^2 + \varepsilon^2\right)\Delta_2 + 2\Delta_2^2\Delta_1^{\max}}{(\Delta_1^{\max})^2 + 2\Delta_2^2 + \Delta_1^{\max}\Delta_2 + \varepsilon^2} & \text{if } \Delta_2 > 0 \\ \dfrac{1}{\Delta_2}\dfrac{\left((\Delta_1^{\min})^2 + \varepsilon^2\right)\Delta_2 + 2\Delta_2^2\Delta_1^{\min}}{(\Delta_1^{\min})^2 + 2\Delta_2^2 + \Delta_1^{\min}\Delta_2 + \varepsilon^2}, & \text{if } \Delta_2 < 0 \\ 1, & \text{if } \Delta_2 = 0 \end{Bmatrix}$$

in some cases. The parameter $\varepsilon = (Kh)^3$, is a function of grid spacing and *K* is a small value (~0.1). We note that Aftosmis et. al [Aft95] has proposed limiting only in the control-surface normal direction, thereby reducing artificial dissipation tangent to the face. We have not implemented that here,

$$\begin{aligned} \nabla u_i^{\Phi} &= \Phi_i \nabla u_i^n + \nabla u_i^t \\ \nabla u_i^n &= \left(\nabla u_i \bullet \mathbf{n}_{ij}\right)\mathbf{n}_{ij} \\ \nabla u_i^t &= \nabla u_i - \nabla u_i^n \end{aligned} .$$

In the case of non-isotropic limiting, the final reconstruction becomes,

$$u_{ij}^{-} = u_i + \frac{1}{2}\nabla u_i^{\Phi} \bullet \Delta \mathbf{r}_{ij} .$$

Green-Gauss gradient reconstruction
The control volume averaged gradient is calculated using one form of the divergence theorem

$$\int_V \nabla u dV = \oint_S u \mathbf{n} dS .$$

A discrete form is defined as,

$$\begin{aligned} \nabla u_i &= \frac{1}{\Delta V_i}\oint_S u\mathbf{n}dS \approx \frac{1}{\Delta V_i}\sum_{j\in NE(i)}\frac{1}{2}\left(u_i+u_j\right)S_{ij}^{\alpha} \\ &= \frac{1}{\Delta V_i}\sum_{j\in NE(i)}\frac{1}{2}\left(u_j-u_i\right)S_{ij}^{\alpha} \end{aligned} \qquad (1.2)$$

This formula is equivalent to midpoint quadrature on hexahedral meshes and is equivalent to trapezoid quadrature on tetrahedral meshes. On Cartesian meshes, this produces the canonical stencil for a second-order central-difference approximation to the first derivative. It is also an edge-based formula and, in a slightly modified form, is used on the boundaries. The Green-Gauss gradient evaluation on simplexes has the property that linear variation within the cell is reconstructed exactly regardless of shape of the element. In order to recover this linear preserving (LP) property on non-simplexes, a different approach will be necessary.

Least-squares gradient reconstruction
Least-squares reconstruction preserves exact linear gradients on arbitrary meshes. The least-squares system of equations for the gradient at node *i* are formed by applying a Taylor's series expansion on each edge containing node *i* and neglecting higher-order-terms, $u_j = u_i + \nabla u_i \bullet \Delta \mathbf{r}_{ij}$.

On hexahedral element meshes, distance-two node pairs that share an element must be included in the gradient evaluation. In Premo this is done by looping over hexahedral elements and defining virtual edges between nodes of opposite corners and looping over quadrilateral faces and defining virtual edges between opposite nodes. Haselbacher and Blazek [Has00] have derived and summarized the three-dimensional formulation of the un-weighted least-squares for arbitrary meshes,

$$\begin{bmatrix} \Delta x_1 & \Delta y_1 & \Delta z_1 \\ \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \vdots & \vdots & \vdots \\ \Delta x_{NE(i)-1} & \Delta y_{NE(i)-1} & \Delta z_{NE(i)-1} \\ \Delta x_{NE(i)} & \Delta y_{NE(i)} & \Delta z_{NE(i)} \end{bmatrix} \begin{Bmatrix} (u_x)_i \\ (u_y)_i \\ (u_z)_i \end{Bmatrix} = \begin{Bmatrix} u_1 - u_i \\ u_2 - u_i \\ \vdots \\ u_{NE(i)-1} - u_i \\ u_{NE(i)} - u_i \end{Bmatrix}$$

.

This over-determined system of linear equations is solved using a Gram-Schmidt process. The three gradient components can be written as a weighted sum of differences between CV *i* and its neighbors

$$\begin{aligned} \left(u_x\right)_i &= \sum_{j\in NE(i)} W_{ij}^{x}\left(u_j-u_i\right) \\ \left(u_y\right)_i &= \sum_{j\in NE(i)} W_{ij}^{y}\left(u_j-u_i\right) \\ \left(u_z\right)_i &= \sum_{j\in NE(i)} W_{ij}^{z}\left(u_j-u_i\right) \end{aligned}$$

where subscripts (*x*,*y*,*z*) denote differentiation and superscripts denote component. The set of edges containing node *i* now contains the set of virtual edges as well. The weights are defined by geometric quantities

$$\begin{aligned} W_{ij}^{x} &= \alpha_{ij,1} - \frac{r_{12}}{r_{11}}\alpha_{ij,2} + \psi\alpha_{ij,3} \\ W_{ij}^{y} &= \alpha_{ij,2} - \frac{r_{23}}{r_{22}}\alpha_{ij,3} \\ W_{ij}^{z} &= \alpha_{ij,3} \end{aligned}$$

where $\Delta(\bullet)_{ij} = (\bullet)_j - (\bullet)_i$, and

$$\alpha_{ij,1} = \frac{\Delta x_{ij}}{r_{11}^2},\ \alpha_{ij,2} = \frac{1}{r_{22}^2}\left(\Delta y_{ij} - \frac{r_{12}}{r_{11}}\Delta x_{ij}\right)$$

$$\alpha_{ij,3} = \frac{1}{r_{33}^2}\left(\Delta z_{ij} - \frac{r_{23}}{r_{22}}\Delta y_{ij} + \psi\Delta x_{ij}\right)$$

where

$$\psi = \frac{r_{12}r_{23} - r_{13}r_{22}}{r_{11}r_{22}}$$

and

$$r_{11} = \sqrt{\sum_{j\in NE(i)} \left(\Delta x_{ij}\right)^2},\ r_{12} = \frac{1}{r_{11}} \sum_{j\in NE(i)} \Delta x_{ij}\Delta y_{ij}$$

$$r_{13} = \frac{1}{r_{11}} \sum_{j\in NE(i)} \Delta x_{ij}\Delta z_{ij},\ r_{22} = \sum_{j\in NE(i)} \left(\Delta y_{ij}\right)^2 - r_{12}^2$$

$$r_{23} = \frac{1}{r_{22}}\left(\sum_{j\in NE(i)} \Delta y_{ij}\Delta z_{ij} - r_{12}r_{13} \right)$$

$$r_{33} = \sqrt{\sum_{j\in NE(i)} \left(\Delta z_{ij}\right)^2 - \left(r_{13}^2 + r_{23}^2\right)}$$

Implementation of the least-squares gradient reconstructions requires that six floating point values $\left(r_{11}, r_{22}, r_{33}, r_{12}, r_{13}, r_{23}\right)$ be stored at each node *i*. Luo et al. [Luo95] present results using a weighted least-squares gradient reconstruction.

Inviscid Fluxes

The upwind advection scheme used in Premo is the approximate Riemann solver of Roe [Roe81] [Hir90]. Spatial accuracy has been extended to higher-order through MUSCL extrapolation as discussed earlier. Selmin [Sel92] has shown that the Roe scheme provides the minimum amount of artificial dissipation required to ensure monotinicity. A summary of Roe's scheme is given here. The advection scheme has been formulated as a one-dimensional Roe scheme in the direction normal to each control surface. The presentation follows closely that of Whitaker [Whi93]. The inviscid flux function is written as,

$$\mathbf{F}_{ij} = \frac{1}{2}\left[\mathbf{F}\left(\mathbf{U}^+, \mathbf{n}_{ij}\right) + \mathbf{F}\left(\mathbf{U}^-, \mathbf{n}_{ij}\right) - \left|\widetilde{\mathbf{A}\bullet\mathbf{n}}_{ij}\right|\left(\mathbf{U}^+ - \mathbf{U}^-\right) \right] \quad (1.3)$$

where $\tilde{\mathbf{A}} = \left(\tilde{\mathbf{A}}^x, \tilde{\mathbf{A}}^y, \tilde{\mathbf{A}}^z\right)$ is the vector of flux Jacobian matrices and the edge identifier *ij*, has been removed from the arguments for clarity. The absolute value of the matrix indicates that it was constructed from the absolute value of the eigenvalue matrix. The tilde indicates Roe averaging. The dissipation term is written compactly as

$$\left|\widetilde{\mathbf{A}\bullet\mathbf{n}}\right|\left(\mathbf{U}^+ - \mathbf{U}^-\right) = \left|\Delta\mathbf{F}_1\right| + \left|\Delta\mathbf{F}_{2,3,4}\right| + \left|\Delta\mathbf{F}_5\right|$$

where,

$$\left|\Delta\tilde{\mathbf{F}}_1\right| = \left|\tilde{\lambda}^1\right|\left(\frac{\Delta p - \tilde{\rho}\tilde{c}\Delta\tilde{U}}{2\tilde{c}^2} \right)\begin{bmatrix} 1 \\ \tilde{u} - \tilde{c}\hat{n}_x \\ \tilde{v} - \tilde{c}\hat{n}_y \\ \tilde{w} - \tilde{c}\hat{n}_z \\ \tilde{H} - \tilde{c}\tilde{U} \end{bmatrix}$$

$$\left|\Delta\tilde{\mathbf{F}}_{2,3,4}\right| = \left|\tilde{\lambda}^{2,3,4}\right|\left[\left(\Delta\rho - \frac{\Delta p}{\tilde{c}^2} \right)\begin{pmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ \tilde{q}^2/2 \end{pmatrix} + \tilde{\rho}\begin{pmatrix} 0 \\ \Delta u - \hat{n}_x\Delta\tilde{U} \\ \Delta v - \hat{n}_y\Delta\tilde{U} \\ \Delta w - \hat{n}_z\Delta\tilde{U} \\ \tilde{u}\Delta u + \tilde{v}\Delta v + \tilde{w}\Delta w - \tilde{U}\Delta\tilde{U} \end{pmatrix} \right]$$

$$\left|\Delta\tilde{\mathbf{F}}_5\right| = \left|\tilde{\lambda}^5\right|\left(\frac{\Delta p + \tilde{\rho}\tilde{c}\Delta\tilde{U}}{2\tilde{c}^2} \right)\begin{bmatrix} 1 \\ \tilde{u} + \tilde{c}\hat{n}_x \\ \tilde{v} + \tilde{c}\hat{n}_y \\ \tilde{w} + \tilde{c}\hat{n}_z \\ \tilde{H} + \tilde{c}\tilde{U} \end{bmatrix}$$

In the above equations the eigenvalues are $\tilde{\lambda}^l = \left(\tilde{U} - \tilde{c}, \tilde{U}, \tilde{U}, \tilde{U}, \tilde{U} + \tilde{c}\right)$, $\Delta u = u^+ - u^-$, $\tilde{q}^2 = \tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2$, $\Delta\tilde{U} = \Delta\tilde{u}\hat{n}_x + \Delta\tilde{v}\hat{n}_y + \Delta\tilde{w}\hat{n}_z$ and $\tilde{U} = \tilde{u}\hat{n}_x + \tilde{v}\hat{n}_y + \tilde{w}\hat{n}_z$.

Roe averages denoted with a tilde are defined as

$$\tilde{\rho} = \sqrt{\rho^-\rho^+}$$

$$\tilde{u} = \frac{u^- + u^+\left(\rho^+/\rho^-\right)^{1/2}}{1 + \left(\rho^+/\rho^-\right)^{1/2}}$$

$$\tilde{H} = \frac{H^- + H^+\left(\rho^+/\rho^-\right)^{1/2}}{1 + \left(\rho^+/\rho^-\right)^{1/2}}$$

and

$$\tilde{c} = \sqrt{(\gamma - 1)\left(\tilde{H} - \tilde{q}^2/2\right)}.$$

An *"entropy fix"* (that prevents the artificial dissipation from vanishing at the sonic points) is enforced through a lower bound on the eigenvalues [Luo94]. This is similar to the function proposed by Liou and Van Leer [Lio88] and is given by

$$\tilde{\lambda}^l = \begin{Bmatrix} \tilde{\lambda}^l, & \tilde{\lambda}^l > \varepsilon \\ \dfrac{\left(\tilde{\lambda}^l\right)^2 + \varepsilon^2}{2\varepsilon}, & \tilde{\lambda}^l < \varepsilon \end{Bmatrix}$$

$$\varepsilon = K \max\left[\left(\lambda_j^l - \lambda_i^l\right), 0\right]$$

and K is a small number.

Viscous Fluxes

There are different methods for calculating the viscous terms. A representative term in the viscous fluxes is $\nabla\bullet(\mu\nabla u)$. A straight-forward way to evaluate this term is to apply the Green-Gauss theorem to the average nodal gradients

$$\nabla\bullet(\mu\nabla u) = \oint_S \mu\nabla u\bullet\mathbf{n}dS \approx \frac{1}{V_i}\sum_{j\in NE(i)} \bar{\mu}\overline{\nabla u}\bullet\mathbf{S}_{ij}$$

where $\overline{\nabla u} = \dfrac{\nabla u_i + \nabla u_j}{2}$ and $\bar{\mu} \approx \dfrac{\left(\mu(T_i) + \mu(T_j)\right)}{2}$.

Unfortunately, this produces a stencil that skips nearest neighbor nodal values. In one dimension assuming uniform spacing $i\Delta x$, the stencil for the second derivative would be,

$$\frac{\partial^2 u}{\partial^2 x} = \frac{1}{4(\Delta x)^2}\left(u_{i+2} - 2u_i + u_{i-2}\right)$$

which is equivalent to a second-order accurate approximation to the second derivative of u on a stencil that is $2\Delta x$! Therefore, a more accurate and compact support algorithm will be sought. In the CHAD code, O'Rourke et al. [ORo99] calculate the stresses at the control surface using a correction to the Green-Gauss gradient. This correction stabilizes the computation of the viscous terms,

$$(\nabla u)_{ij} = \frac{(\nabla u)_i + (\nabla u)_j}{2} + \left(u_j - u_i - \frac{(\nabla u)_i + (\nabla u)_j}{2}\bullet\Delta\mathbf{r}_{ij}\right)\frac{\Delta\mathbf{r}_{ij}}{\left|\Delta\mathbf{r}_{ij}\right|^2} \quad (1.4)$$

The second term on the right hand side of Eq. (1.4) produces the canonical stencil on regular meshes. The first and third term add a correction for non-orthogonal meshes. The average cell gradient is evaluated using the third expression in (Eq. (1.2)) for hexahedral meshes. This has no effect on the calculation of gradients for interior control volumes and requires no special treatment or additional boundary surface quadrature at nodes lying on external boundary surfaces. However, tetrahedral meshes do require boundary surface quadrature.

Boundary Conditions

Nodes exist on external boundary surfaces in the edge-based scheme. These surfaces are treated as control surfaces constructed from the dual of the external element faces. Each sub-control surface is assigned to a boundary node and contributes to the flux balance for that control volume. In this way, different boundary conditions sharing a common node are handled in a straight-forward manner (e.g., corner nodes). In addition, the option exists for piece-wise linear reconstruction of state variables at sub-control surface integration points (centroids) using all element face nodes and bilinear interpolation. Unlike interior nodes, boundary-surface fluxes are assembled by looping over element faces instead of edges. An inner loop over surface nodes is then done to calculate the residuals.

Boundary conditions required for Euler equation solutions include the flow tangency and inflow/outflow conditions. These boundary conditions are implemented in a weak sense as flux conditions and contribute directly to the residuals. Flow tangency is enforced by using Eq. (1.1). Supersonic/subsonic inflow/outflow are prescribed using the Roe flux function and the definition of the free-stream primitive state. The left state is given by the node data and the right state by the free stream data. No-slip and constant temperature conditions are enforced indirectly on the nodes by relaxing the primitive nodal value to the specified value. For steady-state simulations this results in the exact values that were specified. An option exists to extrapolate the Riemann invariants at inflow/outflow boundaries in order to specify the right state of the Riemann flux function [Chi85].

Time Integration

The semi-discrete equations are written as a system of initial value problems and solved for transient and steady-state from time $n\Delta t$ to $(n+1)\Delta t$

$$\frac{\partial\bar{\mathbf{W}}_i}{\partial t} = -\frac{1}{\Delta V_i}\sum_{j\in NE(i)} \mathbf{F}_{ij}\bullet\mathbf{n}_{ij}\left|\mathbf{S}_{ij}\right|, \quad \forall i\in\Omega(V)$$

$$\frac{\partial\bar{\mathbf{W}}_i}{\partial t} = -\frac{1}{\Delta V_i}\sum_{j\in NE(i)} \mathbf{F}_{ij}\bullet\mathbf{n}_{.ij}\left|\mathbf{S}_{ij}\right| - \frac{1}{\Delta V_i}\mathbf{F}_i\bullet\mathbf{n}_i\left|\mathbf{S}_i\right|, \quad \forall i\in\Gamma(S)$$

$$\frac{\partial\mathbf{W}_i}{\partial t} = \mathbf{R}\left(\mathbf{W}^n\right).$$

A multi-stage low-storage Runge-Kutta (RK) algorithm has been implemented that provides 2nd-order accuracy of transient simulations,

$$\mathbf{W}^{(0)} = \mathbf{W}^{(n)}$$
$$\mathbf{W}^{(1)} = \mathbf{W}^{(0)} + \alpha_1 \frac{\Delta t}{\Delta V}\mathbf{R}\left(\mathbf{W}^{(0)}\right)$$
$$\mathbf{W}^{(2)} = \mathbf{W}^{(0)} + \alpha_2 \frac{\Delta t}{\Delta V}\mathbf{R}\left(\mathbf{W}^{(1)}\right)$$
$$\mathbf{W}^{(3)} = \mathbf{W}^{(0)} + \alpha_3 \frac{\Delta t}{\Delta V}\mathbf{R}\left(\mathbf{W}^{(2)}\right)$$
$$\mathbf{W}^{(4)} = \mathbf{W}^{(0)} + \alpha_4 \frac{\Delta t}{\Delta V}\mathbf{R}\left(\mathbf{W}^{(3)}\right)$$
$$\mathbf{W}^{(n+1)} = \mathbf{W}^{(4)}$$

The four coefficients are; $\alpha_1 = 1/4,\ \alpha_2 = 1/3,\ \alpha_3 = 1/2,$ and $\alpha_4 = 1.0$.

While this method is efficient for the time accurate Euler equations at high Mach number, RK suffers from a stability restriction making it inefficient for most viscous and steady-state solutions. Development of a fully-implicit matrix-free Newton-Krylov (MFNK) method (similar to that developed by Luo et. al [Luo98] [Luo01]) is underway and will be reported on in the future.

Results

Results will be presented for six inviscid and one viscous flow. In the current state of development, Premo solves the three-dimensional equations. One- and two-dimensional flows are solved by applying symmetry boundary conditions in the y and z, or z directions respectively. All the meshes were generated using the CUBIT code [Owe02] developed at Sandia National Laboratories expect for the airfoil mesh which was converted to unstructured format from a structured mesh.

Shock-Tube Problem

The shock-tube problem is similar to the problem proposed by Sod [Sod78]. A one-dimensional domain spanning from x=-0.5 to x=0.5 is discretized with 101 nodes. At x=0, a high pressure, high density region is separated from a low pressure, low density region by a diaphragm. The left state is defined by $p_0 = 1.0$, $\rho_0 = 1.0$, $u_0 = 0$, and the right state is defined as $p_1 = 0.25$, $\rho_1 = 0.25$, $u_1 = 0$ and γ = 1.4. At time t=0, the diaphragm is removed. A shockwave propagates from left to right followed by a contact discontinuity and an expansion fan propagates from right to left. The exact solution is the solution to the Rankine-Hugoniot relations. Figure 3 presents a comparison of the Premo predictions with the analytic results. Predictions are represented by symbols, analytic solutions are represented by solid lines. The comparison is quite good.

2D Supersonic Wedge

Flow over a 10 degree wedge in two-dimensions was also simulated. The inflow conditions were Ma=2.5, p=101,325 N/m^2, T=300 K, and γ=1.4. The mesh contained 60x50x3 nodes. Uniform spacing in the x-direction was used. Figure 4 presents pressure contours of the flow for both the first-order and second-order schemes. The pressure and density rise in both cases match the theoretical result to within 2% everywhere behind the shock except near the leading edge of the wedge where errors increase to ~5-8%. The theoretical shock angle is predicted to within 1% by the first-order scheme and to within 10% by the second-order scheme on this very coarse mesh.

2D Shock Reflection

Figure 5 shows a snapshot of a transient calculation of shock impingement. This produces a self-similar solution. The Mach number was 1.7, the wedge angle 25 degrees. The domain was 0.25 m x 0.165 m and the mesh contained 300x150x2 nodes. This problem is discussed in Toro [Tor99]. The shock impinges on the wedge and a reflection occurs. Density contours reveal the shock location, slip surface and triple point. Depressions in the contours mid-way up the curved reflection are due to lingering errors in the specification of properties within the shock in the initial conditions. This issue is discussed by Toro.

3D Supersonic Sphere-Cone

The flow over a three-dimensional sphere-cone geometry was computed. Only one half of the domain was computed. A symmetry plane was placed at y=0. The mesh contained 99,000 nodes. The flow conditions were as follows; Ma=4, p=101,325 N/m^2, T=300 K, angle-of-attack was 20 degrees. A snapshot from the solution is presented in Figure 6. Flow is from right to left. Flooded contours of pressure reveal the shock location. The black lines are velocity streamlines. A high pressure stagnation region behind the detached shock is also present. The shock standoff distance was compare with experimental correlations and found to be in very good agreement. For Ma=4 and sphere radius equal to 0.103 m, the standoff distance is ~0.018m [Lie57].

2D Supersonic Circular Arc Channel

This problem was first proposed by Ni [Ni82]. The domain was a two-dimensional channel with dimensions 3x1. A 4% thick bump, of length one, was positioned in the center of the channel. The Mach number was 1.4. Two meshes were used to solve this problem. The first mesh was made up of 3,200

hexahedral elements with one element in the z-direction. The mesh contained 6,666 nodes or 3,333 in the x-y plane. The second mesh used 19,777 tetrahedral elements with approximately two elements in the z-direction and a total of 5,523 nodes. The two meshes are shown in Figures 7a and 7b. While an attempt to maintain approximately equal spacing was made, these two meshes are significantly different.

Three cases were run on the hexahedral mesh; Green-Gauss (GG) gradient reconstruction with the Barth-Jespersen limiter (BJ), Least-Squares (LS) gradient reconstruction with the BJ limiter and first-order upwind. Two cases were run on the tetrahedral mesh; LS gradient reconstruction with the BJ limiter and first-order upwind. Pressure contours are shown in Figures 7c-7g. It is evident that the linear reconstruction schemes provide much better shock resolution compared to the first-order schemes. Comparisons of the pressure coefficient between the three second-order cases are presented on Figure 8a and for the two first-order cases in Figure 8b. The initial and reflected shocks are predicted to be in the same location while the peak pressures differ between the hexahedral and tetrahedral simulations.

2D NACA 0012 Airfoil

This is a classic problem for Euler solvers. A 129x33x2 node O-mesh with chord length equal to one and far-field boundary at a radius of 25 chord lengths was used. A portion of the mesh near the airfoil is shown in Figure 9a. Two cased were run. Both used GG gradient reconstruction with either the BJ or the Venkatrakrishnan limiter. The Mach number was 0.8 and the angle-of-attack was 1.25 degrees. Four orders of magnitude reduction in the L1 and L2 density norms were observed at steady-state. Pressure contours are shown in Figures 9b and 9c for the two cases and the pressure coefficients for both cases are shown in Figure 9d. No significant differences can be discerned between the two cases.

2D Laminar Boundary Layer

The final problem that will be presented was chosen to test the implementation of the viscous terms in the Navier-Stokes equations. A flat-plate boundary layer has been computed and compared to the Blasius solution. A stretched Cartesian mesh was generated with a buffer zone preceding the flat plate. The mesh is shown in Figure 10a. The buffer zone was necessary to damp transient waves as they approached the inflow boundary. The buffer zone was 0.1x0.05 m containing 10x80x2 nodes. The mesh above the flat plate was also 0.1x0.05 m and contained 100x80x2 nodes. The Reynolds number based on plate length was 62,500 and the Mach number was 0.5. The smallest mesh spacing normal to the wall was 5.0×10^{-5} m. Comparisons between predicted *u* and *v* velocity components with the Blasius solution at three locations; x/L=1/4, 1/2, and 3/4, where L is the plate length are presented in Figures 10b and 10c. The second-order scheme with GG gradient reconstruction and no limiter was used for this case. The simulation was run using first-order time stepping. The duration to steady-state was t=0.00229 sec. The agreement with theory for both *u* and *v* is very good. The only significant discrepancy is in *v* near the outflow boundary.

Discussion

A new compressible flow solver has been written as a module for the Sandia National Laboratories ASCI applications framework SIERRA. The code uses state-of-the-art spatial discretization. One-, two-, and three-dimensional solutions to the Euler equations have been presented. Both steady-state and transient flows have been simulated. A two-dimensional solution to the Navier-Stokes equations has also been presented. These results demonstrate current capabilities of the code. A rigorous verification of the Euler and Navier-Stokes equations, both subsonic and supersonic, has been conducted by Roy, Smith and Ober [Roy02] using the method of Manufactured Solutions. Their results demonstrate second-order accuracy on uniform meshes. Work is continuing to improve subsonic boundary conditions. In addition, work has begun on the implicit solver and parallelization of the code.

An important goal of this project is to produce a general-purpose, state-of-the-art CFD simulation tool that is flexible, maintainable, easy to use and robust. Spatial algorithms have been chosen based on accuracy, robustness and flexibility. These algorithms will form the kernel of the general-purpose code. Rapid development of Premo was aided by the SIERRA framework which provided many services common to computational mechanics codes thus freeing the developers to concentrate on developing algorithms specific to the physics module.

Future developments will include solvers for a wide range of governing equations such as Reynolds Averaged Navier-Stokes, chemically reacting Euler and Navier-Stokes as well as reduced models such as parabolized Navier-Stokes and thin-layer Navier-Stokes. Arbitrary body motion, relative body motion and coupled physics are also planned.

Acknowledgements

The first author would also like to acknowledge Srinivasan Arunajatesan and Andreas Haselbacher for their helpful comments.

References

[Aft95] Aftosmis, M., Gaitonde, D. and Tavares, T.S. (1995)
Behavior of Linear Reconstruction Techniques on Unstructured Meshes, AIAA Journal, vol. 33, no. 11, pp. 2038-2049.

[Bar91] Barth, T. (1991)
A 3-D upwind Euler solver for unstructured meshes, AIAA paper 91-1548-CP.

[Bar89] Barth, T.J. and Jespersen, D.C. (1989)
The design and application of upwind schemes on unstructured meshes, AIAA paper 89-0366, Jan.

[Chi85] Chima, R. V. (1985)
Inviscid and viscous flows in cascades with an Explicit Multiple-grid algorithm, AIAA Journal, vol. 23, no. 10, pp. 1556-1563.

[Edw01] Edwards, H. C. and J. R. Stewart (2001)
SIERRA : A Software Environment for Developing Complex Multiphysics Applications,
Proceedings of First MIT Conference on Computational Fluid and Solid Mechanics,
Elsevier Scientific, K. J. Bathe, ed., June 2001.

[Has00] Haselbacher, A. and Blazek, J. (2000)
Accurate and efficient discretization of Navier-Stokes equations on mixed grids, AIAA Journal, Vol. 38, No. 11, pp. 2094-2102.

[Hos96] Hosangadi, A., Lee, R.A., York, B.J., Sinha, N. and Dash, S. (1996)
Upwind unstructured scheme for three-dimensional combusting flows, Journal of Propulsion and Power, vol. 12, no. 3, pp. 494-502.

[Hir90] Hirsch, C. (1990)
Numerical Computation of Internal and External Flows, Volume 2, chapter 16, John Wiley & Sons.

[LeV92] LeVeque, R.J. (1992)
Numerical Methods for Conservation Laws, 2nd ed., Birkhauser Verlag, pp. 183-187.

[Lie57] Liepmann, H.W. and Roshko, A. (1957)
Elements of Gas Dynamics, pp. 105,
Galcit Aeronautical Series, Wiley, New York.

[Lio88] Liou, M.-S. and van Leer, B. (1988)
Choice of Implicit and Explicit operators for the upwind differencing method,
AIAA paper AIAA-1988-0624, Reno, NV, Jan.

[Luo95] Luo, H., Baum, J.D. and Lohner, R. (1995)
An improved finite volume scheme for compressible flows on unstructured grids,
AIAA paper AIAA-95-0348, Reno, NV, Jan.

[Luo94] Luo, H., Baum, J.D. and Lohner, R. (1994)
Edge-based finite element scheme for the Euler equations, AIAA Journal, Vol. 32, No. 6, pp. 118-1190.

[Luo98] Luo, H., Baum, J.D. and Lohner, R. (1998)
A fast, matrix-free implicit method for compressible flows on unstructured grids, Journal of Computational Science, Vol. 146, pp. 664-690.

[Luo01] Luo, H., Baum, J.D. and Lohner, R. (2001)
An accurate, fast, matrix-free implicit method for computing unsteady flows on unstructured grids,
Computers & Fluids, Vol. 30, pp. 137-159.

[Ni82] Ni, R.-H. (1982)
A Multiple-Grid Scheme for Solving the Euler Equations, AIAA Journal, vol. 20, no. 11, pp. 1565-1571.

[Oro99] O'Rourke, P.J., Zhang, S. and Sahota, M.S. (1999)
A parallel, unstructured-mesh methodology for device-scale combustion calculations, Oil & Gas Science and Technology-Review, Vol. 54, No. 2, pp. 169-173.

[Owe02] Steven J. Owen ed. (2002)
"Cubit 7.0 Users Manual", Sandia National Laboratories, SAND94-1100 Rev. 4/2002
URL:http://endo.sandia.gov/cubit/help-version7/cubithelp.html

[Roe81] Roe, P.L. (1981)
Approximate Riemann solvers, parameter vectors and difference schemes, Journal of Computational Physics, Vol. 43, pp. 357-372.

[Roy02] Roy, C., Ober, C. and Smith, T. (2002)
Verification of a compressible CFD code using the method of manufactured solutions, AIAA paper AIAA-2002-3110, St. Louis, MO, June.

[Saa86] Saad, Y. and Schultz, M. (1986)
GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal of Scientific and Statistical Computing, Vol. 7, pp. 856-869.

[Sel92] Selmin, V. and Formaggia, L. (1992)
Simulation of hyperbolic flows on unstructured grids,
International Journal for Numerical Methods in Engineering, vol. 34, pp. 569-606.

[Sod78] Sod, G.A. (1978)
A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws,
Journal of Computational Physics, vol. 27, pp. 1-31.

[Tor99] Toro, E.F. (1999)
Riemann Solvers and Numerical Methods for Fluid Dynamics, 2nd Ed., Springer-Verlag.

[van77] van Leer, B. (1977)
Towards the ultimate conservative difference scheme III. Upstream-centered finite-difference schemes for ideal compressible flow,
Journal of Computational Physics, vol. 23, pp. 263-275.

[Ven95a] Venkatakrishnan, V. (1995)
Convergence to steady state solutions of the Euler Equations on unstructured grids with limiters, Journal of Computational Physics, Vol. 118, pp. 120-130.

[Whi93] Whitaker, D.L. (1993)
Three-Dimensional unstructured grid Euler computations using a fully-implicit, upwind method, AIAA paper 93-3337.

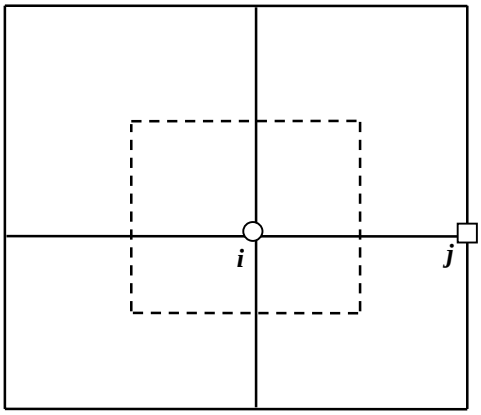


Figure 1. Quadrilateral element patch and dual mesh.

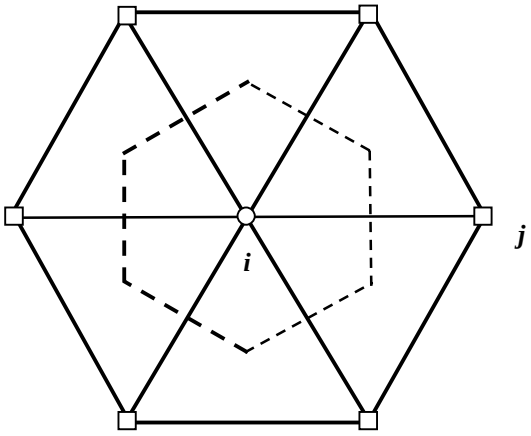


Figure 2. Triangle element patch and dual mesh.

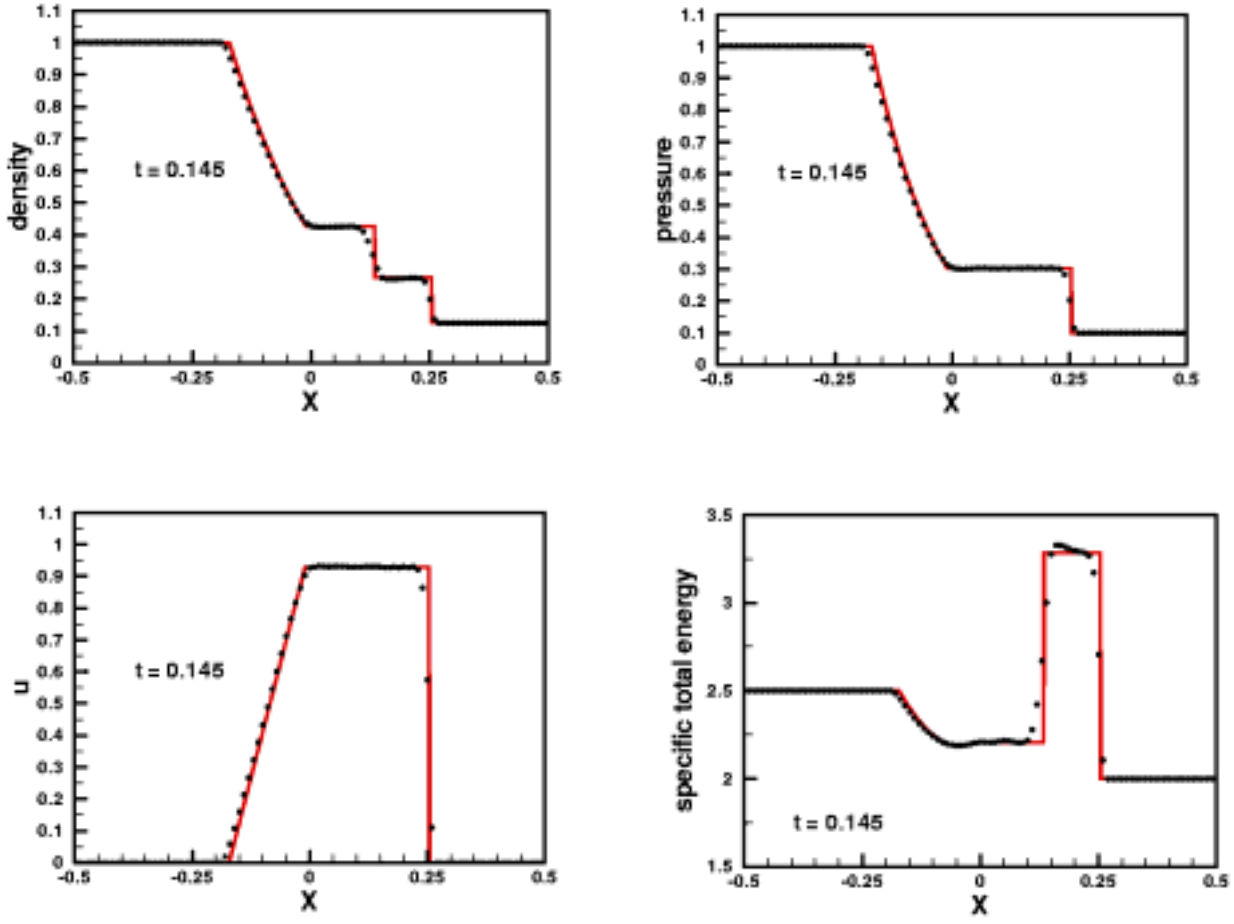


Figure 3 One-Dimensional shock-tube profiles.

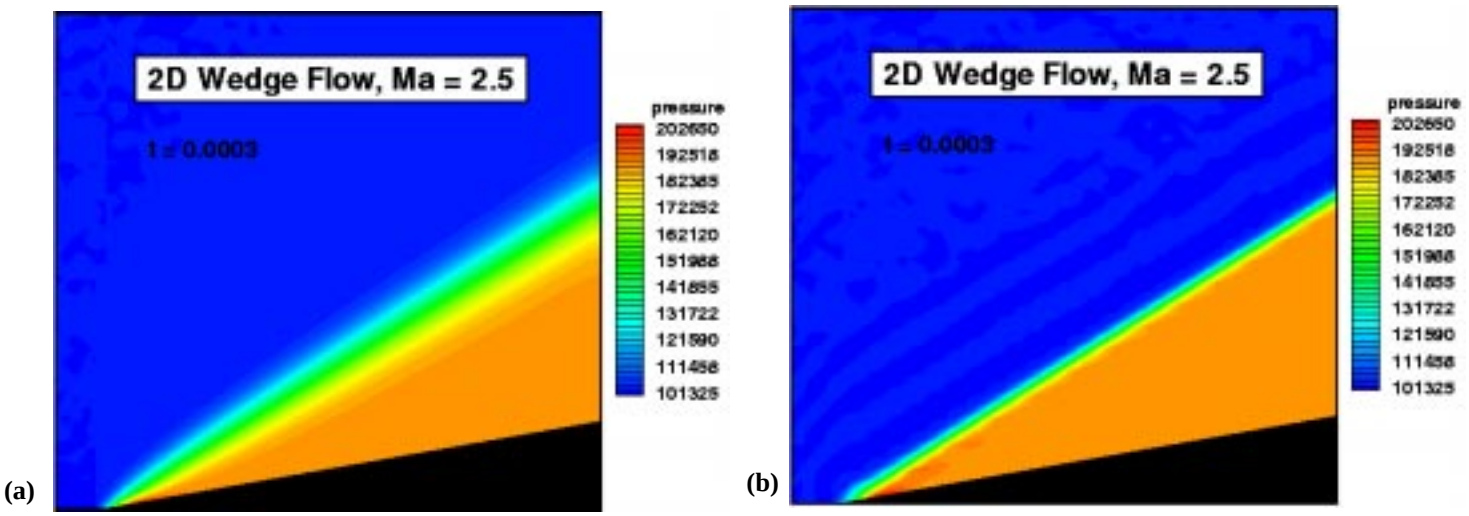


Figure 4. Flow over a two-dimensional wedge, a) 1st-order and b) 2nd-order scheme.

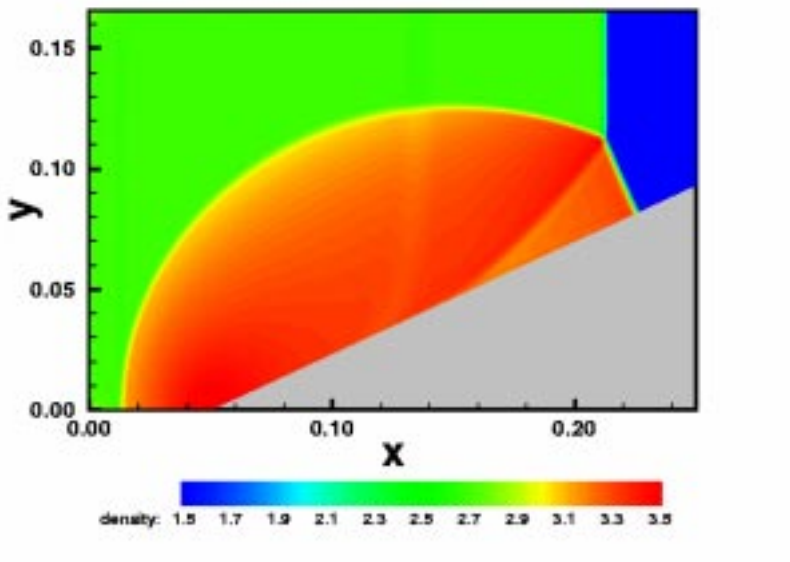


Figure 5. Shock impinging on a two-dimensional wedge.

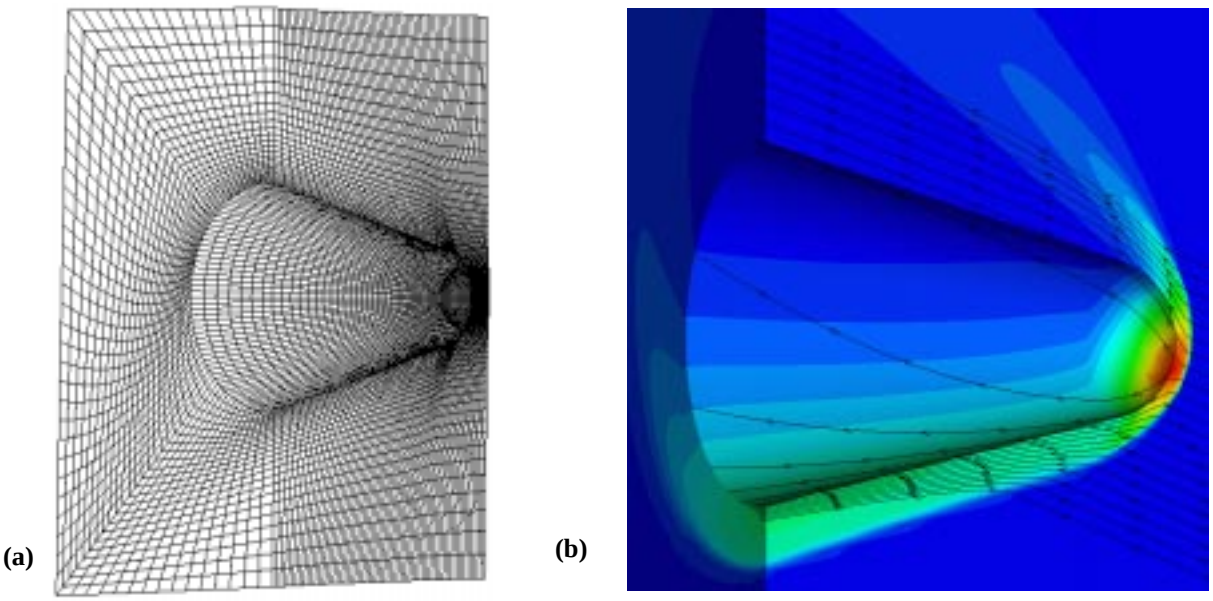


Figure 6. Supersonic flow over three-dimensional sphere-cone at angle of attack; a) mesh and b) pressure contours and stream lines.

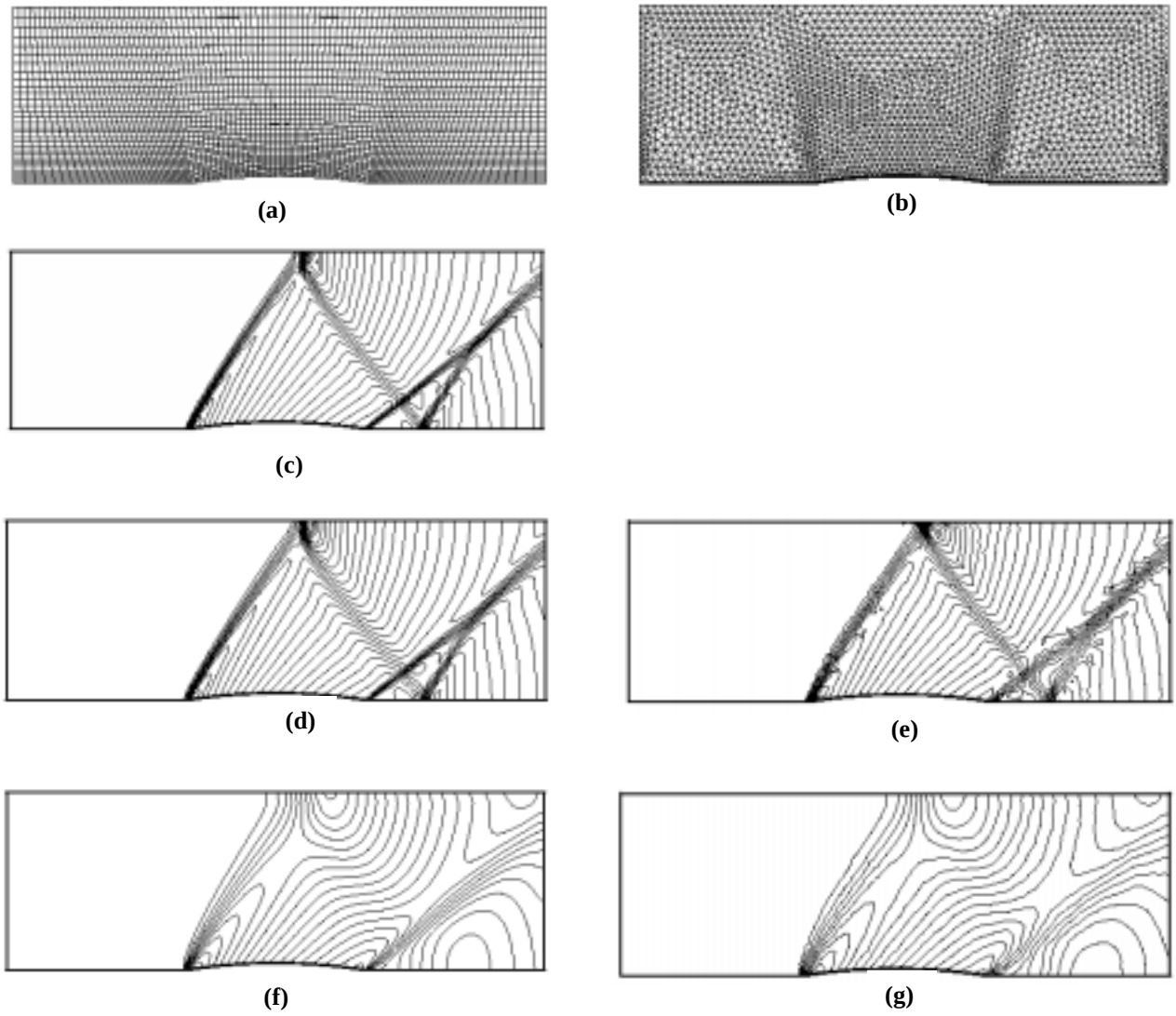
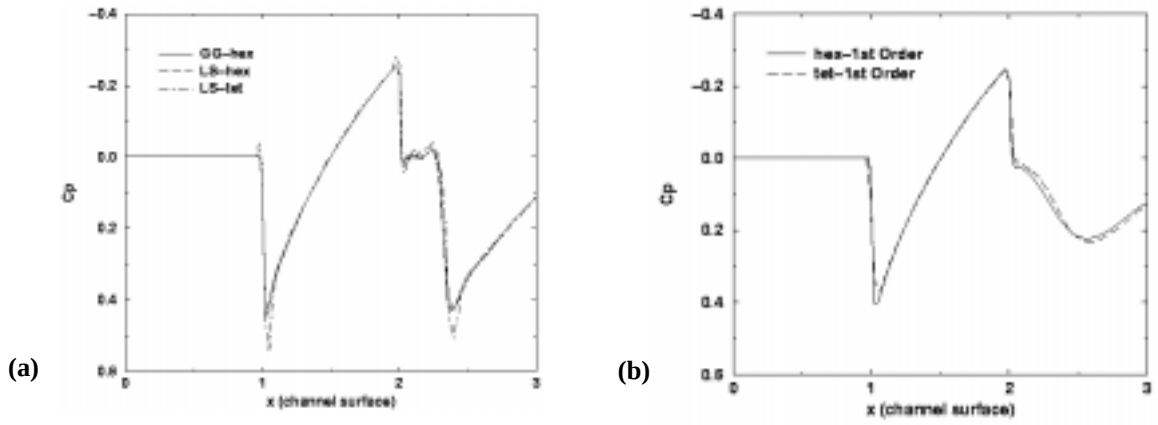


Figure 7. Supersonic flow in a circular arc channel; a) hexahedral mesh, b) tetrahedral mesh, c) GG-hex, d) LS-hex, e) LS-tet, f) 1st order-hex, and g) 1st order-tet pressure contours.



(a) (b)

Figure 8. Pressure coefficient on channel surface with circular arc; a) 2nd-order schemes, b) 1st-order schemes.

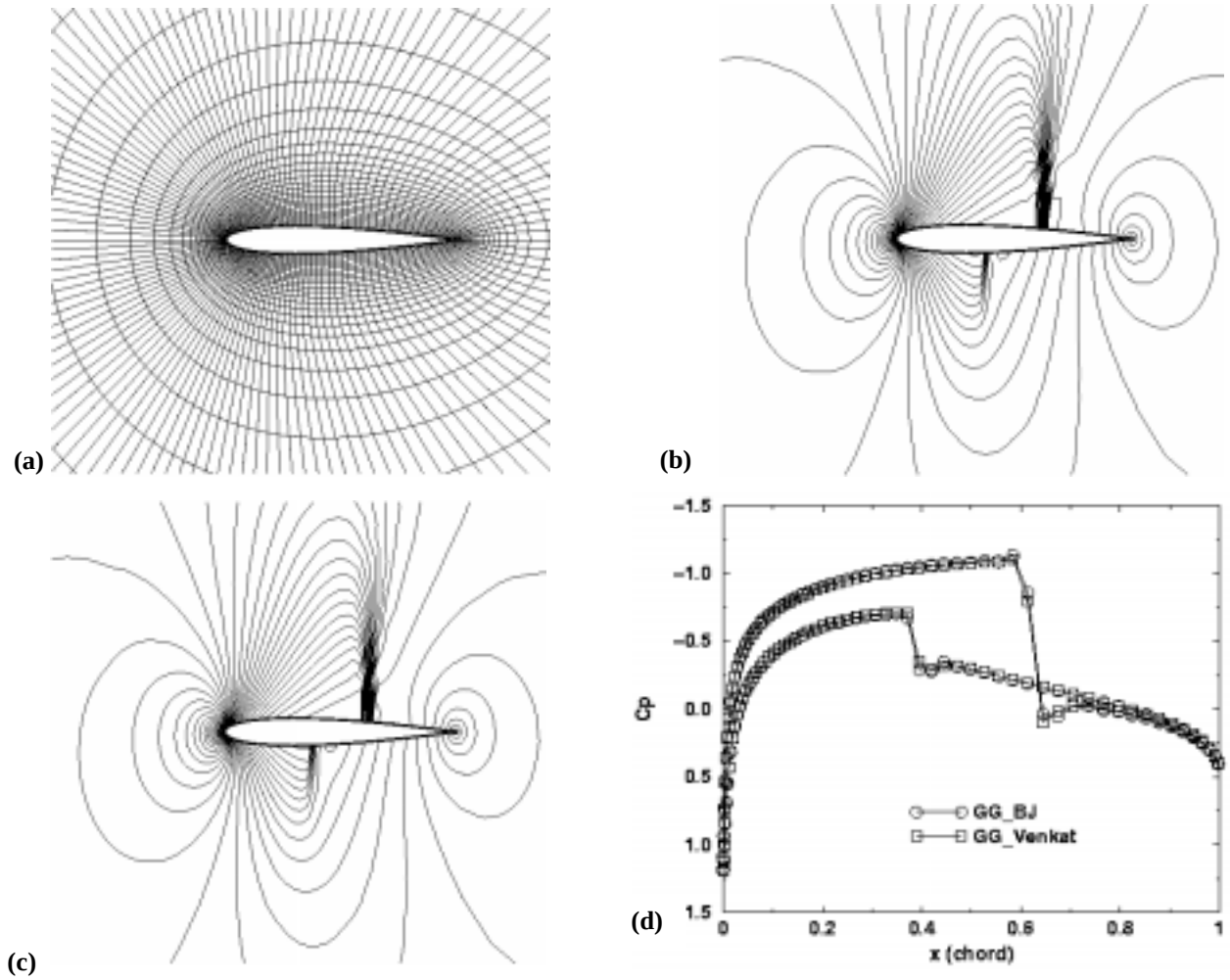


Figure 9. NACA 0012 airfoil at angle-of-attack; a) O-mesh, b) GG-BJ pressure contours, c) GG-Venkat pressure contours and d) pressure coefficient.

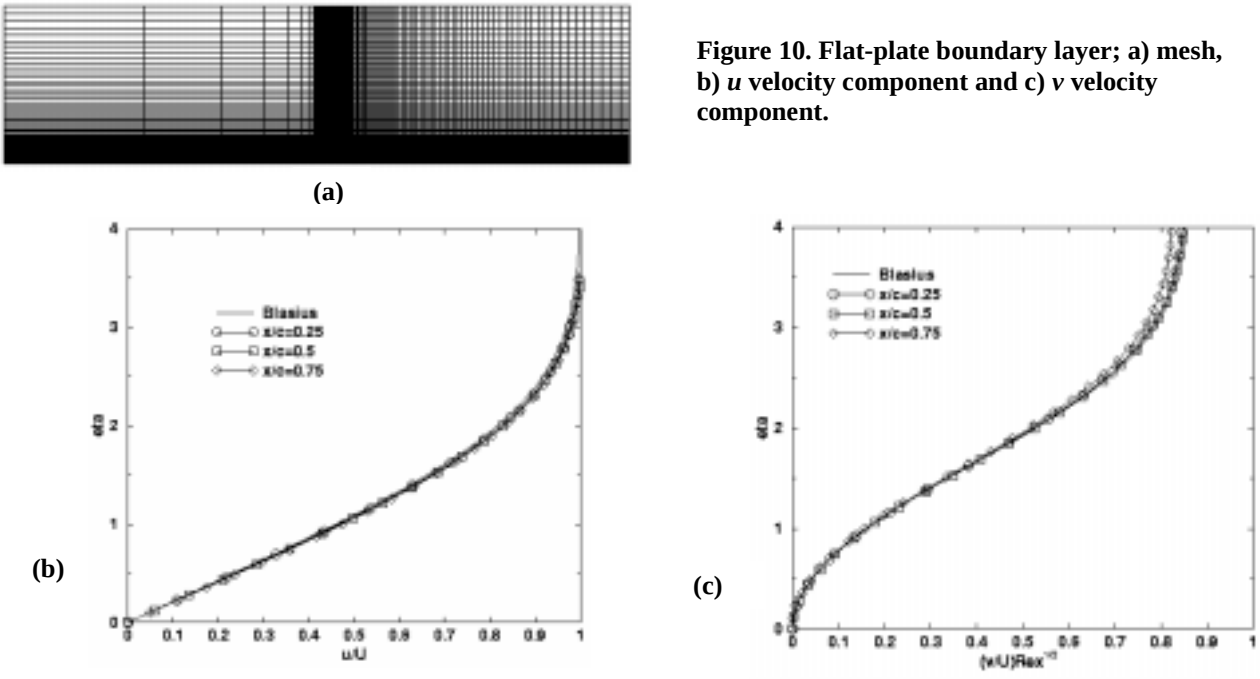


Figure 10. Flat-plate boundary layer; a) mesh, b) *u* velocity component and c) *v* velocity component.