NASA-CR-194233

IN-34-CR
183139
p. 37

Final Report

Multigrid Solution of Internal Flows Using Unstructured Solution Adaptive Meshes

SBIR 1987 Phase II

NASA Lewis Contract
Number NAS3-25785
SBIR- 01.01.3800
Fluent Inc. TN-035
November 1992
Release Date: 9-16-92

Wayne A. Smith
Kenneth R. Blake
Fluent Inc.
Lebanon NH 03766

Contents

Introduction

This is the final report of the NASA Lewis SBIR Phase II Contract Number NAS3–25785, *Multigrid Solution of Internal Flows Using Unstructured Solution Adaptive Meshes.* The objective of this project, as described in the Statement of Work, is as follows

> The objective of this SBIR Phase II project is to develop and deliver to NASA a general three-dimensional Navier-Stokes code using unstructured solution-adaptive meshes for accuracy and multigrid techniques for convergence acceleration. The code will primarily be applied, but not necessarily limited, to high speed internal flows in turbomachinery.

The objective of the SBIR program, as described in the NASA Lewis Research Center Annual Report at the time this project began, is as follows

> The Small Business Innovations Research (SBIR) Program is a congressionally mandated program aimed at the commercialization of innovative concepts by small business. These innovations are in general areas of research based on the needs of the U.S. Government.

This contract has successfully achieved the objectives of both this contract and of the SBIR program. On June 1, 1992, Fluent Inc. introduced a new Computational Fluid Dynamics package, called **Rampant**, into the industrial and research markets. **Rampant** is an unstructured, solution-adaptive code that can solve inviscid, laminar, and turbulent; compressible and incompressible flows. The initial development stages of **Rampant** were made possible by

Chapter 1

Overview

The licensed copy of the **Rampant** package provided with the contract deliverable includes a copy of the **Rampant User's Guide**. The User's Guide provides a comprehensive description of Rampant's capabilities and operation. This report does not, therefore, include a detailed description of the mechanics of operating the program, but instead concentrates on describing the numerical techniques and test results that are pertinent to the specifications of this contract.

1.1 Program Capabilities

Rampant has the following general capabilities:

Grid Generation

- 2D unstructured triangular mesh generation.
- 3D unstructured triangular surface mesh generation.
- 3D unstructured tetrahedral volume mesh generation.
- Can read triangular surface meshes from a variety of CAD packages, and can then mesh the interior with tetrahedra.
- Can read tetrahedral volume meshes from a variety of CAD packages.

- Arbitrary slices. Planar, spherical, cylindrical, etc.
- Iso-surface extraction. Extract surfaces of constant value.
- Iso-line extraction. Extract lines of constant value on a given surface.
- Surface integration. Pressure and viscous forces. Normal and shear forces. Lift and drag coefficients. Heat transfer.
- XY plotter. Residual histories, surface plots, over-plot data from file.
- Generate Tecplot and Cray MPGS format visualization files.

1.2 The Solution Process

The process of solving a flow problem with **Rampant** consists of the following primary steps

1. Generate a geometric surface description
2. Generate a surface mesh.
3. Generate a volume mesh.
4. Setup the problem to be solved. This includes setting boundary conditions, flow parameters, and solution parameters.
5. Solve the flow.
6. Analyze the flow field.
7. Adapt the grid.

These processes are described in the following chapters.

- Turbulent Kinetic Energy

$$\frac{\partial}{\partial t}\iiint \rho k\, dV + \iint \rho k\, \boldsymbol{v}_i d\boldsymbol{A}_i - \iint \frac{\mu_e}{\sigma_k}\frac{\partial k}{\partial i}\, d\boldsymbol{A}_i = \iiint (P - \rho\epsilon)\, dV \quad (2.4)$$

- Turbulent Dissipation

$$\frac{\partial}{\partial t}\iiint \rho\epsilon\, dV + \iint \rho\epsilon\, \boldsymbol{v}_i d\boldsymbol{A}_i - \iint \frac{\mu_e}{\sigma_\epsilon}\frac{\partial \epsilon}{\partial i}\, d\boldsymbol{A}_i = \iiint \frac{\epsilon}{k}(C_1 P - C_2\rho\epsilon)\, dV \quad (2.5)$$

Euler Equations

For inviscid flow, the momentum and energy equations reduce to

- Mass

$$\frac{\partial}{\partial t}\iiint \rho \boldsymbol{v}_j\, dV + \iint \rho \boldsymbol{v}_j\, \boldsymbol{v}_i d\boldsymbol{A}_i + \iint p\delta_{ij}\, d\boldsymbol{A}_i = \iiint \rho g_j\, dV \quad (2.6)$$

- Momentum

$$\frac{\partial}{\partial t}\iiint \rho E\, dV + \iint \rho E\, \boldsymbol{v}_i d\boldsymbol{A}_i + \iint p\boldsymbol{v}_i\, d\boldsymbol{A}_i = 0 \quad (2.7)$$

2.1.2 Rotating Cartesian Formulation

For rotating domains, the equations are formulated in terms of absolute velocity in a frame of reference rotating with angular velocity $\boldsymbol{\Omega}$ about the origin. The formulation is valid for $\boldsymbol{\Omega} = \boldsymbol{\Omega}(t)$. The relative velocity, $\boldsymbol{w}$, is defined as

$$\boldsymbol{w} \equiv \boldsymbol{v} - \boldsymbol{\Omega} \times \boldsymbol{x} \quad (2.8)$$

Euler Equations

- Mass

$$\frac{\partial}{\partial t}\iiint \rho\, dV + \iint \rho\, \boldsymbol{w}_i d\boldsymbol{A}_i = 0 \quad (2.9)$$

where

$$\begin{aligned}\Phi &= \nabla \boldsymbol{v} : (\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^{\mathrm{T}}) - 2/3(\nabla \cdot \boldsymbol{v})^2 \\ &= \nabla \boldsymbol{v}_{ji}(\nabla \boldsymbol{v}_{ji} + \nabla \boldsymbol{v}_{ij}) - 2/3(\nabla \boldsymbol{v}_{ii})^2 \\ &= \nabla \boldsymbol{v}_{xi}(\nabla \boldsymbol{v}_{xi} + \nabla \boldsymbol{v}_{ix}) + \\ &\quad \nabla \boldsymbol{v}_{yi}(\nabla \boldsymbol{v}_{yi} + \nabla \boldsymbol{v}_{iy}) + \\ &\quad \nabla \boldsymbol{v}_{zi}(\nabla \boldsymbol{v}_{zi} + \nabla \boldsymbol{v}_{iz}) - 2/3(\nabla \boldsymbol{v}_{ii})^2\end{aligned} \tag{2.15}$$

and for a Cartesian coordinate system, $\nabla \boldsymbol{v}$ is given simply by

$$\nabla \boldsymbol{v}_{ij} = \partial \boldsymbol{v}_i / \partial \boldsymbol{x}_j \tag{2.16}$$

The k-ϵ model parameters are

$$\begin{aligned}\mu_e &= \mu_{\mathrm{laminar}} + \mu_t \\ \mu_t &= \rho C_\mu k^2/\epsilon \\ \sigma_k &= 1.0 \\ \sigma_\epsilon &= 1.3 \\ C_1 &= 1.44 \\ C_2 &= 1.92\end{aligned}$$

For laminar flow,

$$\begin{aligned}\mu_e &= \mu_{\mathrm{laminar}} \\ \mu_t &= 0\end{aligned}$$

2.1.4 Equation of State

For compressible flow, the ideal gas equation of state is used to relate pressure and temperature to the conserved variables.

$$p = \rho R T \tag{2.17}$$

where

$$\begin{aligned}R &= c_p - c_v \\ \gamma &= c_p/c_v\end{aligned}$$

becomes

$$\begin{aligned}\iint \rho \boldsymbol{v} \cdot d\boldsymbol{S} &= \sum_{\text{faces}} \iint \rho \boldsymbol{v} \cdot d\boldsymbol{A} \\ &\approx \sum_{\text{faces}} \iint \bar{\rho} \tilde{\boldsymbol{v}} \cdot d\boldsymbol{A} \\ &= \sum_{\text{faces}} \bar{\rho} \tilde{\boldsymbol{v}} \cdot \boldsymbol{A} \qquad (2.21)\end{aligned}$$

where the face-averaged value of velocity is given by

$$\tilde{\boldsymbol{v}} = \frac{\overline{\rho \boldsymbol{v}}}{\bar{\rho}} \qquad (2.22)$$

and the components of the area vector, $\boldsymbol{A}$,

$$\boldsymbol{A} = \iint d\boldsymbol{A} \qquad (2.23)$$

are the area projections of the face.

The remaining integrals are discretized analogously, and the Euler equations reduce to the following system of ordinary differential equations

$$\frac{d}{dt}(V\tilde{\boldsymbol{W}}) + \sum_{\text{faces}} \bar{\boldsymbol{F}} \cdot \boldsymbol{A} = 0 \qquad (2.24)$$

where

$$\tilde{\boldsymbol{W}} = \begin{bmatrix} \tilde{\rho} \\ \tilde{\rho v_x} \\ \tilde{\rho v_y} \\ \tilde{\rho v_z} \\ \tilde{\rho e} \end{bmatrix}, \quad \bar{\boldsymbol{F}} = \begin{bmatrix} \bar{\rho} \quad \tilde{v} \\ \tilde{\rho v_x}\tilde{v} + \bar{p}\hat{\imath} \\ \tilde{\rho v_y}\tilde{v} + \bar{p}\hat{\jmath} \\ \tilde{\rho v_z}\tilde{v} + \bar{p}\hat{k} \\ \bar{\rho e} \quad \tilde{v} + \bar{p}\tilde{v} \end{bmatrix}$$

2.2.2 Added Dissipation

The finite volume scheme described above is analogous to to a central difference scheme on a structured grid. The use of central differences ensures that the scheme is second-order accurate for "smooth" meshes, but allows the solution to decouple at odd-even points.

PRECEDING PAGE BLANK NOT FILMED

where κ_2 and κ_4 are constants, for example, 2 and 0.05.

The application of the first-order dissipation is governed by the TVD-like switch of Turkel and Swanson [TSVW91] which is sensitive to second differences of pressure

$$\tilde{\nu} = \frac{\left| \sum_{\text{faces}} (\tilde{p}_+ - \tilde{p}_-) \right|}{\sum_{\text{faces}} |(\tilde{p}_+ - \tilde{p}_-)| + \sum_{\text{faces}} (\tilde{p}_+ + \tilde{p}_-)} \tag{2.31}$$

This switch also turns off the fourth difference dissipation in the neighborhood of a shock where it can be destabilizing.

For multigrid calculations, the blended second/fourth-difference dissipation is replaced on the coarse grid levels with a simpler second-difference dissipation. The dissipation is only first-order accurate, but accuracy is not necessary on the coarse grids and the second-difference dissipation is more stable and cheaper to compute. It is equivalent to setting

$$\epsilon_2 = \kappa_{2c} \tag{2.32}$$

$$\epsilon_4 = 0 \tag{2.33}$$

where κ_{2c} is typically 0.5.

The face-based dissipation operator, $\mathbf{D}$, results in added flux-like terms that are treated in the same way as the physical Euler fluxes. This results in a discretized equation of the form

$$\frac{d}{dt}(V\tilde{W}) + \sum_{\text{faces}} (\bar{F} \cdot A - \mathbf{D}\tilde{W}) = 0 \tag{2.34}$$

Since the facial dissipation operator, $\mathbf{D}$, is a flux-like term, it is set to zero along external boundary faces to maintain global conservation.

2.2.3 Divergence Theorem

Extensive use is made of a form of the divergence theorem (or Green's Lemma) in calculating derivatives and metric quantities. The divergence theorem is given by

$$\iiint \nabla \cdot F \, dV = \iint F \cdot dA \tag{2.35}$$

PRECEDING PAGE BLANK NOT FILMED

A standard set of coefficients for a four-stage scheme is 1/4, 1/3, 1/2, and 1.

This type of multistage scheme is, at best, only second-order accurate in time. For steady solutions, however, it is usually preferred over more accurate Runge-Kutta schemes because it requires the solution to be stored at only two time levels. Higher-order temporal accuracy is not required since the scheme is used simply as a means of advancing the solution towards its final value. A classical higher-order Runge-Kutta scheme could be implemented if the increase in time accuracy is deemed to warrant the increase in storage requirement. The classical fourth-order Runge-Kutta method is given by

$$\begin{aligned}
\tilde{W}^{(0)} &= \tilde{W}_t \\
\tilde{W}^{(1)} &= \tilde{W}^{(0)} - 1/2\Delta t \mathbf{R}\tilde{W}^{(0)} \\
\tilde{W}^{(2)} &= \tilde{W}^{(0)} - 1/2\Delta t \mathbf{R}\tilde{W}^{(1)} \\
\tilde{W}^{(3)} &= \tilde{W}^{(0)} - \Delta t \mathbf{R}\tilde{W}^{(2)} \\
\tilde{W}^{(4)} &= \tilde{W}^{(0)} - 1/6\Delta t(\mathbf{R}\tilde{W}^{(0)} + 2\mathbf{R}\tilde{W}^{(1)} + 2\mathbf{R}\tilde{W}^{(2)} + \mathbf{R}\tilde{W}^{(3)}) \\
\tilde{W}_{t+\Delta t} &= \tilde{W}^{(4)}
\end{aligned}$$

2.2.5 Implicit Residual Smoothing

The maximum time step can be further increased by increasing the support of the scheme through implicit averaging of the residuals with their neighbors. The residuals are filtered through a Laplacian smoothing operator.

$$R_i' = R_i + \epsilon \sum (R_j' - R_i') \tag{2.38}$$

The resulting set of equations can be solved by using Jacobi iteration

$$R_i'^{(m)} = \frac{R_i + \epsilon \sum R_j'^{(m-1)}}{1 + \epsilon \sum 1} \tag{2.39}$$

Two Jacobi iterations are usually sufficient to allow doubling the time step with a value of $\epsilon = 0.5$.

PRECEDING PAGE BLANK NOT FILMED

interpolation and relaxation operators. A sample coarse-grid hierarchy is shown in Figure 3.

The problem that remains is the solution of the governing equations on the coarse grids composed of irregularly-shaped cells. Since the discretized equations presented in the previous sections place no restrictions on the number of faces a cell is composed of, this actually poses little additional problem. There is a loss of accuracy when the finite volume scheme is used on the irregular coarse-grid cells. However, the accuracy of the multigrid solution is determined solely by the finest grid and is therefore not affected by the coarse-grid discretization. Also, a simple, more stable second-difference dissipation can be used on the coarse grids since they do not affect the accuracy of the fine grid solution. Higher-order terms such as viscous stresses and turbulence coefficients are "frozen" at their fine grid values and hence need not be calculated on the coarse grids.

One further note of interest is that although the coarse-grid cells look very irregular, the discretization cannot "see" the jaggedness in the cell faces. The discretization uses only the area projections of the cell faces and therefore each group of "jagged" cell faces separating two cells is equivalent to a single straight line connecting the endpoints of the jagged segment. Although not currently implemented, this fact can be used to decrease the cost of the coarse-grid iterations by reducing each group of face flux calculations to a single flux calculation using the group's net area projection.

2.3.2 Multigrid Cycle

A multigrid cycle can be defined as a recursive procedure that is applied at each grid level as it moves through the grid hierarchy. The traversal of the hierarchy is governed by three parameters, β_1, β_2, and β_3, as follows:

1. β_1 *smoothings*, or relaxation sweeps, are performed at the current grid level to reduce the high wave number components of the error. In the present case, the smoothing procedure is the multistage scheme with local time stepping, residual smoothing, etc. The high wave number components of error should be reduced to the point that the remaining error is expressible on the next coarser mesh without significant aliasing.

In the current implementation, interpolation of corrections is performed simply by setting the correction in a fine-grid cell equal to the correction in its associated coarse-grid cell. Following this, the interpolated corrections are smoothed with one pass of a Laplacian smoother (as for implicit residual smoothing) to reduce its high wave number components. After being smoothed, the corrections are then added to the fine-grid solution.

This flow of control can be expressed in psuedo-code as

```
multigrid_cycle(n)
{
  for i=1 to $\beta_1$
    multistage_iteration(n);
  if (n $\neq$ coarsest_grid_level)
    {
      compute_residuals(n);
      restrict_solution(n,n+1);
      restrict_residuals(n,n+1);
      for i=1 to $\beta_2$
        multigrid_cycle(n+1);
      interpolate_correction(n+1,n);
      for i=1 to $\beta_3$
        multistage_iteration(n);
    }
  }
```

The parameter that each of the functions calls is the grid level for which the result is computed, beginning with 0 on the finest grid and increasing by 1 for each successively coarser grid. The highest level of control consists simply of performing multigrid cycles at the finest grid level until some convergence criterion is met.

2.4 Boundary Conditions

A variety of boundary conditions are available in Rampant. Boundary conditions are specified for groups of boundary faces that are identified during the grid generation phases.

For rotating domains, pressure is specified at the "hub", and a radial pressure distribution is computed using the radial equilibrium condition

$$\frac{\partial p}{\partial r} = \frac{\rho V_\theta^2}{r} \tag{2.45}$$

where V_θ is averaged circumferentially. Circumferential averaging is performed by dividing the interval from the minimum radius (hub) to the maximum radius (shroud) into some number (e.g., 64) of evenly spaced "bins". Each boundary face is then associated with the bin that corresponds to its radial position, and averaged values are computed for each bin. A pressure for each bin can then be computed by marching out from the hub and applying the radial equilibrium condition to each bin. The pressure applied at each boundary face is then simply taken as the pressure in the associated bin.

Supersonic Inlet

All quantities are specified.

Supersonic Outlet

All quantities are extrapolated.

2.4.2 Far-Field Boundary Conditions

For external compressible flow such as that about aircraft configurations, far-field (or free-stream) boundary conditions are usually used. Free stream Mach number, static pressure, static density, and flow direction are prescribed and boundary values are computed in the standard way using Rieman invariants (see, for example, [Jam83]).

- maximization of the minimum angles in the grid cells, the production of the most equilateral mesh for a given distribution of vertices (equiangular property);
- simple criteria used to drive the triangulation process, either the max-min angle, the circle (2D) or sphere (3D), or the Voronoi neighbor criterion;
- advantageous environment for mesh refinement provided by specific implementations of the technique;
- flexibility gained by decoupling the grid point generation from the process of establishing connections between the points.

The present approach for generating the interior mesh composed of either triangular (2D) or tetrahedral (3D) cells was proposed independently by Bowyer [Bow81] and Watson [Wat81]. The scheme performs a nucleus of operations, casting vertices into an existing Delaunay triangulation and using efficient search procedures and the circle (2D) or sphere (3D) criteria to reconstruct the grid. The additional benefits of this scheme, besides possessing the desirable Delaunay properties, is that the procedure is very amenable to grid refinement and iterative node generation. The disadvantages of this particular scheme include the necessity to use high precision mathematics in certain situations and the need to introduce additional logic to preserve boundary integrity. The basic steps in the method are described below:

1. Generate a list of vertex locations. This list may include both boundary and interior vertices. In the present application of the scheme, an initial mesh is generated using only the boundary vertices and the final interior mesh is produced by using an iterative node generation scheme, similar to that suggested by Holmes and Snyder [HS88]. The iterative refinement scheme interrogates the existing mesh and locates points to improve the resolution, smoothness and aspect ratios of the cells.
2. Create an initial temporary or virtual mesh that encloses the entire domain of interest. The present approach uses the triangulation of a quadrilateral (2D with 4 vertices) or hexahedron (3D with 8 vertices).
3. Initially, introduce a vertex into the virtual mesh; and subsequently, into the existing triangulation from previous operations.

Adequately resolving the salient features of the solution field requires appropriately clustering of the vertices. A higher density of grid points is required in regions where the geometry or solution vector is more complex. Although solution-adaptive grid refinement relaxes this requirement, it is still necessary to resolve the basic features of the computational field since the adaption parameters are computed from this initial field. Unlike techniques that use mappings to generate the mesh, unstructured grid generation techniques permit a different number of points on each of its boundaries. Therefore, the user can produce very dense vertex distributions on one boundary and extremely coarse distributions on opposing boundaries. In addition, the regions may have numerous sides and even include embedded regions.

Smooth variations in the sizes of the cells increases the accuracy of the numerical analysis. In particular, most numerical analysis techniques discretely represent the various terms in the mathematical expressions being solved by approximations whose accuracy degrades in the presence of rapid fluctuations in cell size.

The aspect ratios and skewnesses of the cells can influence the accuracy and convergence characteristics of the numerical integration technique. The aspect ratios for triangles or tetrahedrons in the present study are defined by comparing the circumscribing radii with that of an optimal cell. An equilateral cell would have 0 skewness and a totally flat cell would have a skewness value of 1. A highly skewed cell decreases accuracy and slows convergence.

In addition to the accuracy and convergence requirements of the numerical analysis technique, the robustness of the interior mesh generation technique is influenced by the quality of the boundary mesh. Smoothness is particularly critical in regions where surfaces intersect or where surfaces are in close proximity. For instance, the faces defining the bottom of a vehicle should not have dramatically different sizes than the faces defining the ground.

Finally, since the purpose of the grid is to discretize the domain for numerical analysis, it is difficult to define the mesh quality requirements without some knowledge of the solution field. For example, highly skewed cells are tolerable in regions that contain no significant gradients. Nevertheless, the prime objective of the grid generation task is to cluster vertices in order to resolve salient features of the problem while producing smooth variations of low-aspect-ratio cells.

a decrease in node density on another area. The area from which nodes are moved usually has to be relatively near to the area into which they are moved (for not-trivial geometries), and this is not the place you tend to want to remove nodes from; you'd usually like to take them from somewhere far from the "action".

The obvious advantage of an unstructured mesh for solution adaption is that the node connectivity can change. This makes local node insertion and deletion possible. Less restricted node movement is also made possible because the node connectivity can be changed if a cell becomes too skewed.

3.3.1 Point Insertion

Rampant provides point insertion by allowing new points to be added at the midpoint of any edge. An edge is simply the line segment connecting two nodes.

In two dimensions, an (interior) edge has a triangle on either side of it. When a new node is added at the mid-point of the edge, the original edge is removed and four new edges are added; two that connect the new node to the endpoints of the original edge, and two that connect the new node to the opposite vertices of the two adjacent triangular cells. Nodes are added on exterior boundaries in the same way, except that there is only one triangle to connect to (i.e., only three new edges). Periodic boundaries are logically the same as interior edges, but the implementation is more complicated because of the "cut" in computational space at the boundary; the insertion algorithm must wrap across the cut.

In three dimensions, an edge has a "ring" of tetrahedra that surround it. As in 2D, when a new point is added along the central edge, each of the cells attached to it are divided into two, and the edges are reconnected. In 3D, the implementation becomes more complicated due to the variable number of cells attached to an edge and the intricate ways in which these rings of cells intersect solid boundaries, wrap across rotational periodic boundaries, and combinations thereof.

Although any edge of a triangle or tetrahedron can be split in this manner, care must be taken in choosing the edge to be split so as not to create skewed cells. This is accomplished in Rampant by requiring the edge that is split to be the longest edge in each of the cells affected. If there is any longer

3.3.4 Solution-Adaptive Control

Solution-adaptive mesh refinement is provided simply by controlling the point insertion and deletion procedures with information from the computed flow field. An adaption criterion is defined and applied to each of the cells in the domain. Depending on the value of the adaption function in the cell, the cell is marked for refinement, coarsening, or no change. A second pass is made through the domain, and in any cell marked for refinement, a new node is added at the mid-point of the longest edge in the cell. This may require refining neighboring cells first if the longest edge in the current cell is not also the longest edge in all the cells that contain it. After all cells marked for refinement have been refined, a third pass is taken through the domain and an attempt is made to coarsen any cell marked for coarsening. As described above, this is only implemented in 2D, and can only happen for certain cell configurations. In practice, the inability of the code to perform arbitrary coarsening is not much of a problem for steady solutions. It would be a significant problem for unsteady solutions where, for example, a shock might move across the domain leaving behind it a trail of small elements.

There are two forms of adaption criterion available in Rampant. The first is one based on flow gradients. The adaption function is generated by first choosing a flow quantity such as pressure, density, Mach number, entropy, etc. Then node values of this function are computed by taking the average of the cell values in each of the cells that contain that node as a vertex. The value of the adaption criterion in each cell is then computed by subtracting the cell value of the function from the average of the three (or four in 3D) node values in that cell. This essentially results in an undivided Laplacian of the flow quantity. Two thresholds are then prescribed; any cell whose adaption value (magnitude) is below the lower threshold is marked for coarsening, and any cell whose adaption value is above the upper threshold is marked for refinement.

The second adaption criteria is available for turbulent flows at walls and is based on the value of y^+ in the cell. Again, two thresholds are specified; any cell whose y^+ is less than the lower threshold is marked for coarsening, and any cell whose y^+ is greater than the upper threshold is marked for refinement. This provide a very easy way of generating a mesh with the proper spacing at walls. For the wall functions used in the solver, the first cell should have a y^+ less than 500 for the log-law to apply. In order to

Chapter 4

Results

4.1 Turbine Vane Cascade

The first test case is a two-dimensional cascade of core turbine vanes. The three-dimensional annular geometry and experimental flow conditions for this case are described by Goldman and Seasholtz [GS82].

Figure 1 demonstrates the two-dimensional unstructured grid generation procedure. First the geometry was defined in the PreBFC geometry modeler. The geometry consists of one curve for the inlet, one curve for the outlet, one curve for the periodic boundary, and three curves for the blade. Since finer spacing was desired at the trailing edge (and later on the upper surface), the blade was defined with three curves—upper surface, lower surface, and trailing edge—as a way to provide some simple grid spacing control. A single curve with non-uniform node spacing parameters could have been used (as is typical for a structured grid), but that degree of control was not opted for in this case. The disparities in the node spacing where the curves meet are easily smoothed away with a command in PreBFC that simply adds a few points to the end of the curve with the larger spacing.

The three panels show the grid in its initial state, an intermediate state, and the final state. The initial grid is formed by tessellating the initial boundary points. This grid is a valid (constrained) Delaunay triangulation of the nodes; it simply has cells with very poor quality. To improve the quality of the elements, new points are inserted into the grid until the prescribed

4.2 Annular Turbine Cascade

Next, a 3D inviscid computation was performed for the same geometry. The three-dimensional annular cascade of core turbine vanes is described by Goldman and Seasholtz [GS82]. A very coarse mesh was used:

9303	cells
19805	faces
2497	nodes
16763	interior faces
200	outlet faces
644	periodic faces
516	wall faces (blade)
671	wall faces (hub)
813	wall faces (shroud)
198	inlet faces

The memory required for the Euler solver was 184 bytes/cell, 28 bytes/face, and 24 bytes/node, for a combined cell-based requirement of 250 bytes per cell.

Contours of Mach number for the inviscid solution are shown in Figures 12 and 13. As expected, the results correspond directly to the previous 2D calculation.

4.3 Laminar Flat Plate

In order to examine the accuracy of the viscous stresses computed on the triangular grids, a laminar flat plate calculation was performed and compared to the Blasius similarity solution. Figure 14 shows the results for data taken at 5 locations along the plate.

The plate starts at x=0 and extends to the outlet at x=10. The exact (Blasius) solution's drag for the entire plate is 66.6 and the computed value was 64.4. The pressure residual converged 3 orders of magnitude in roughly 3500 iterations. The grid had 16608 cells, 25425 faces, and 8818 nodes.

The second panel shows all five data lines (the first two lines are indistinguishable at that scale), and the fourth panel shows a close-up of just the first

52333	cells
109938	faces
12493	nodes
974	inlet faces
1862	wall faces (shroud)
1048	wall faces (hub)
5819	wall faces (blade)
2587	periodic faces
841	outlet faces
96807	interior faces

A (partially converged) solution was obtained on this starting mesh, and then the mesh was iteratively refined at solid boundaries as the solution was advanced until the value of y^+ in each cell was less than 500. The resulting mesh consisted of the following:

180907	cells
370672	faces
37878	nodes
1112	inlet faces
4370	wall faces (shroud)
1134	wall faces (hub)
10249	wall faces (blade)
2894	periodic faces
851	outlet faces
350062	interior faces

Each iteration took 249 seconds on an SGI 4D/340VGX workstation. Approximately 1000 iterations would be required to converge the solution on this mesh if the flow were reset to uniform conditions. The memory usage was 240 bytes/cell, 28 bytes/face, and 100 bytes/node for a combined cell-based requirement of 318 bytes per cell and a total of 58 Mbytes of grid and solution storage. The program code, graphics, interface, etc. add an overhead of about 6 Mbytes to the size of the process.

The inlet stagnation properties were set at
$p_0 = 101325$ N/m2
$\rho_0 = 1.226$ kg/m3

and the outlet static pressure at the hub was set at
$p_{\text{hub}} = 101558 \text{ Pa} = 1.0023 * p_0$

of radial position. The open circles are experimental values, and the filled circles are computational values. Computed total pressure and temperature are consistently a few percent lower than experimental values, and the computed flow angle is consistently higher than the experimental values. This consistent behavior indicates that the discrepancies are due in large part to the different effective operating conditions for the two cases caused by the reduced blockage effects of the computed result.

Bibliography

[Bow81] A. Bowyer. "Computing Dirichlet Tessellations". *The Computer Journal*, 24(2):162–166, 1981.

[Bra79] Achi Brandt. Multi-level adaptive computations in fluid dynamics. Technical Report AIAA-79-1455, AIAA, Williamsburg, VA, 1979.

[Dir50] G.L. Dirichlet. "Uber die Reduction der Positiven Quadratischen Formen Mit Drei Undestimmten Ganzen Zahlen". *Z. Reine Angew. Math*, 40(3):209–227, 1850.

[GS82] Louis J. Goldman and Richard G. Seasholtz. Laser anemometer measurements in an annular cascade of core turbine vanes and comparison with theory. NASA Technical Paper 2018, NASA Lewis Research Center, 1982.

[HS88] D.G. Holmes and D.D Snyder. "The Generation of Unstructured Triangular Meshes Using Delaunay Triangulation". In S. Sengupta, J. Hauser, P.R. Eiseman, and J.F. Thompson, editors, *Numerical Grid Generation in Computational Fluid Mechanics '88*, pages 643–652. Pineridge Press Limited, 1988.

[Jam83] Antony Jameson. Solution of the Euler equations for two dimensional transonic flow by a multigrid method. MAE Report 1613, Princeton University, June 1983.

[JST81] Antony Jameson, W. Schmidt, and Eli Turkel. Numerical solution of the Euler equations by finite volume methods using runge-kutta time-stepping schemes. Technical Report AIAA-81-1259, AIAA 14th Fluid and Plasma Dynamics Conference, Palo Alto, California, June 1981.

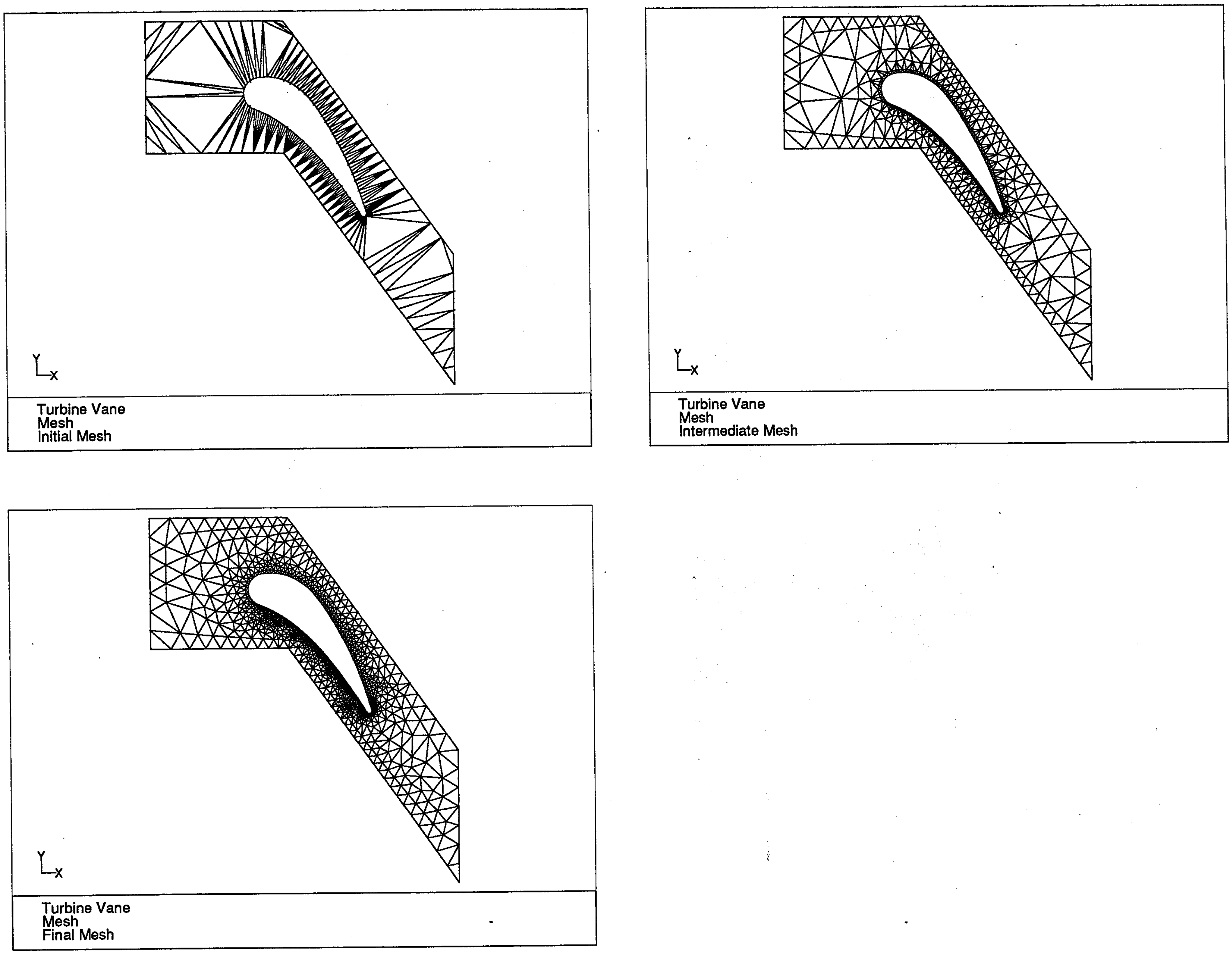

Y
X
Turbine Vane
Mesh
Initial Mesh
Y
X
Turbine Vane
Mesh
Intermediate Mesh
Y
X
Turbine Vane
Mesh
Final Mesh

Figure 1

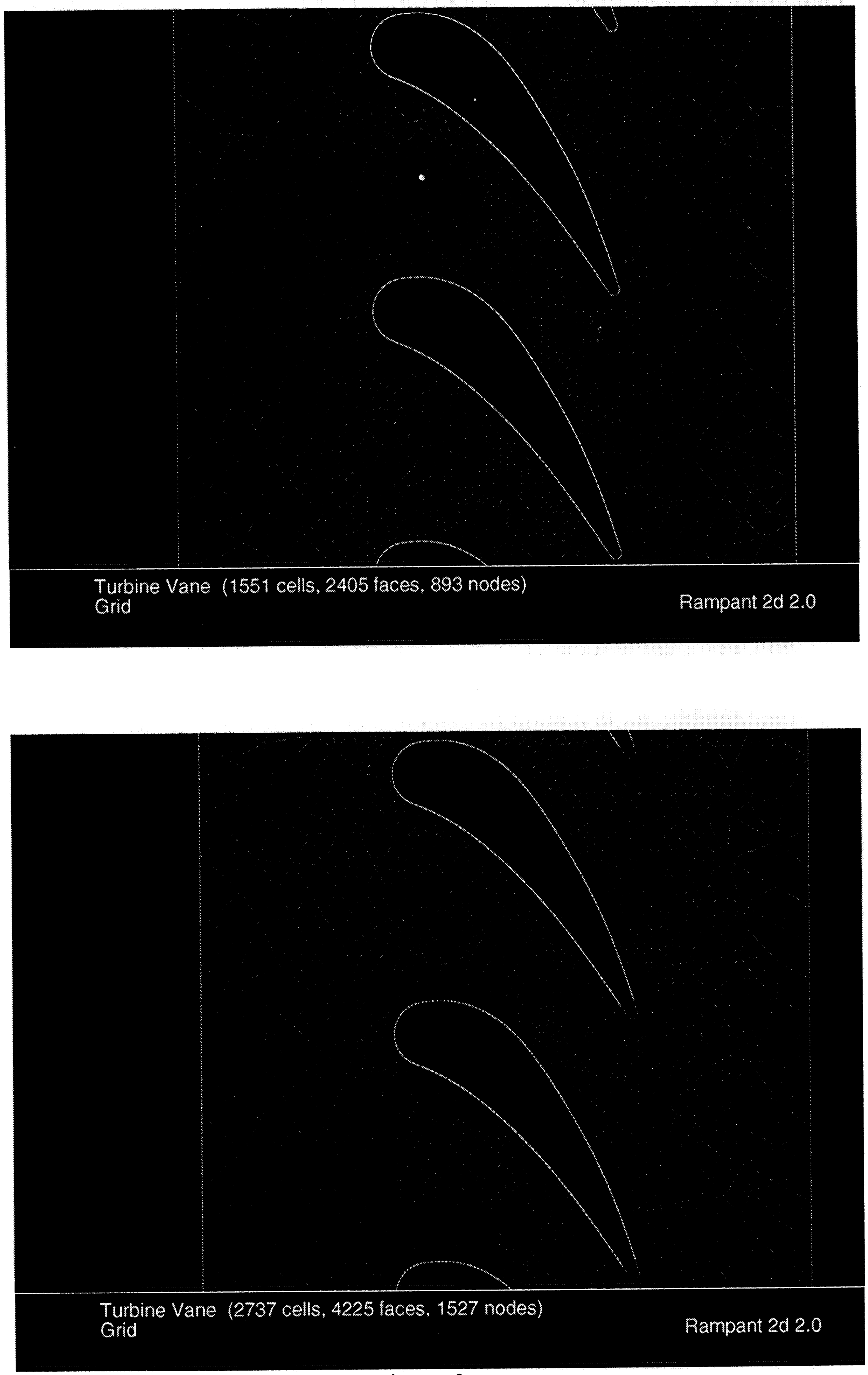
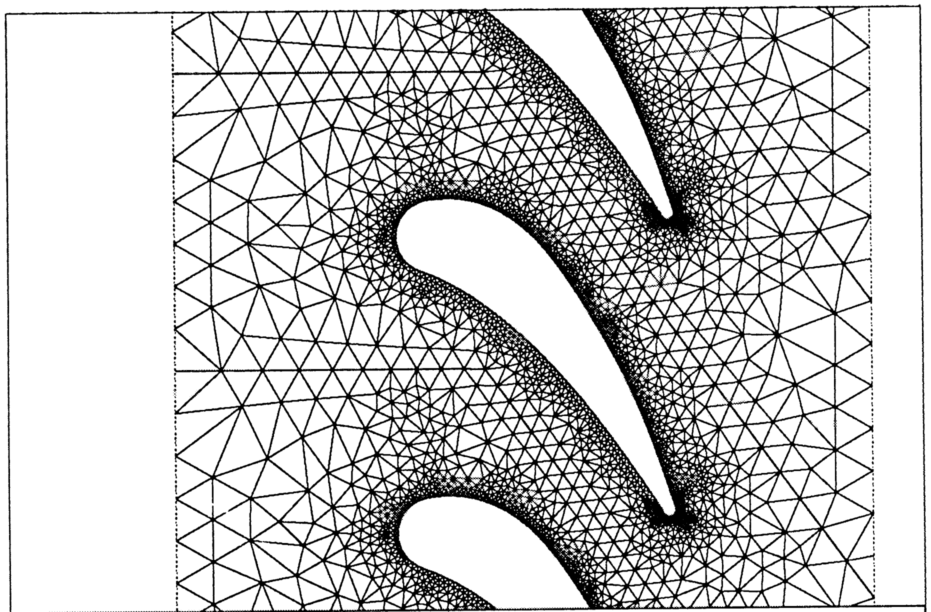
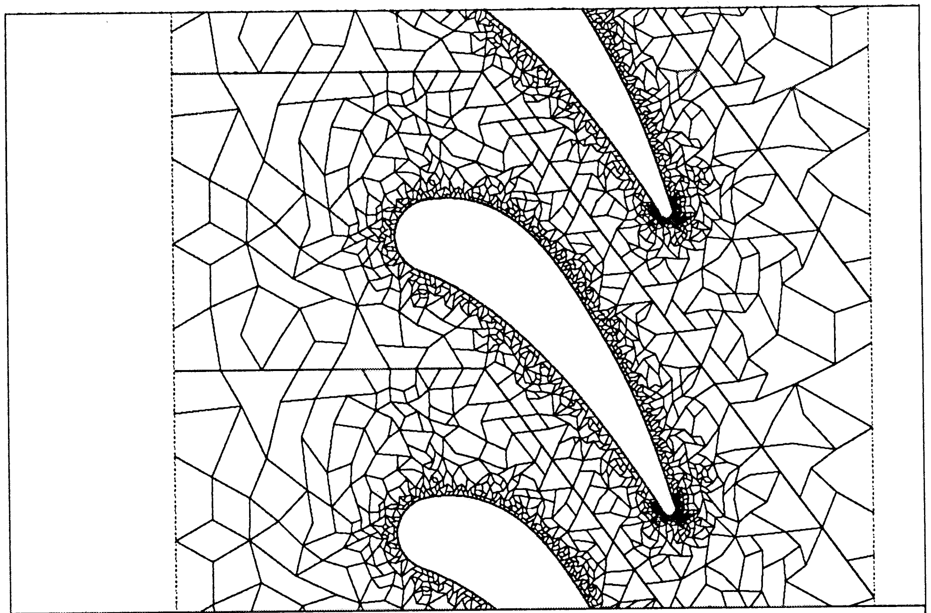
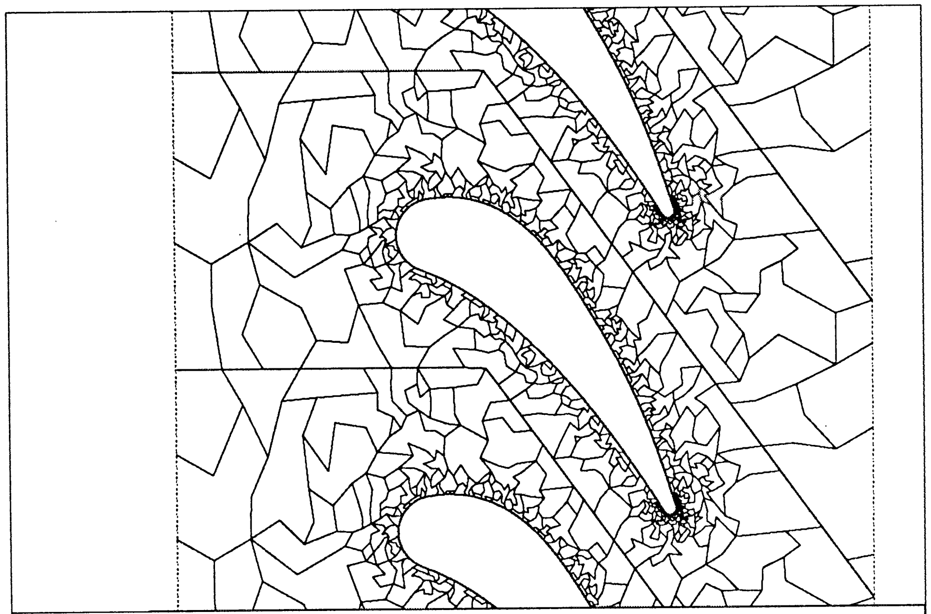

Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Grid
Rampant 2d 2.0
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid
Rampant 2d 2.0

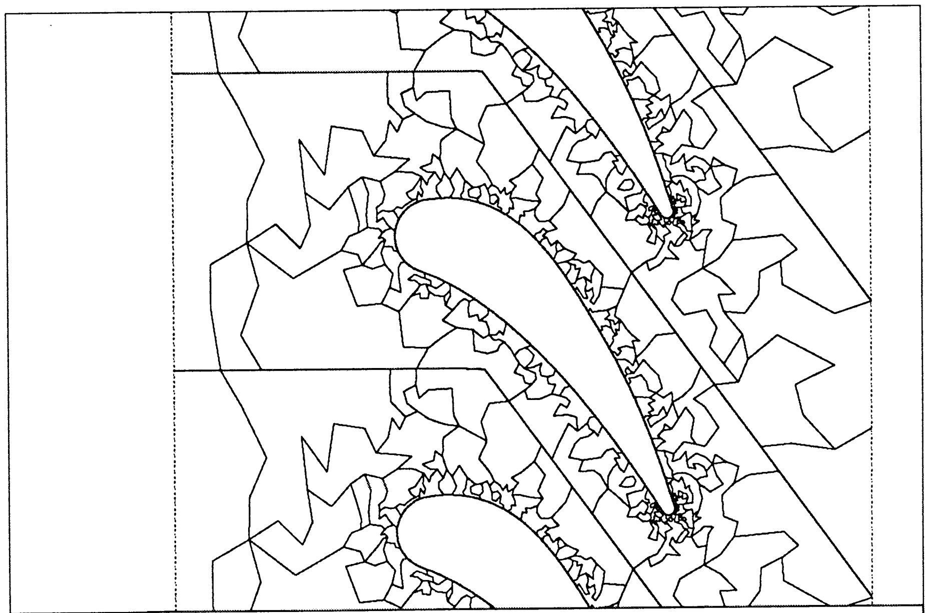
Figure 2

Figure 3


Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid Level 0
Rampant 2d 2.0


Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid Level 1
Rampant 2d 2.0


Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid Level 2
Rampant 2d 2.0


Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid Level 3
Rampant 2d 2.0

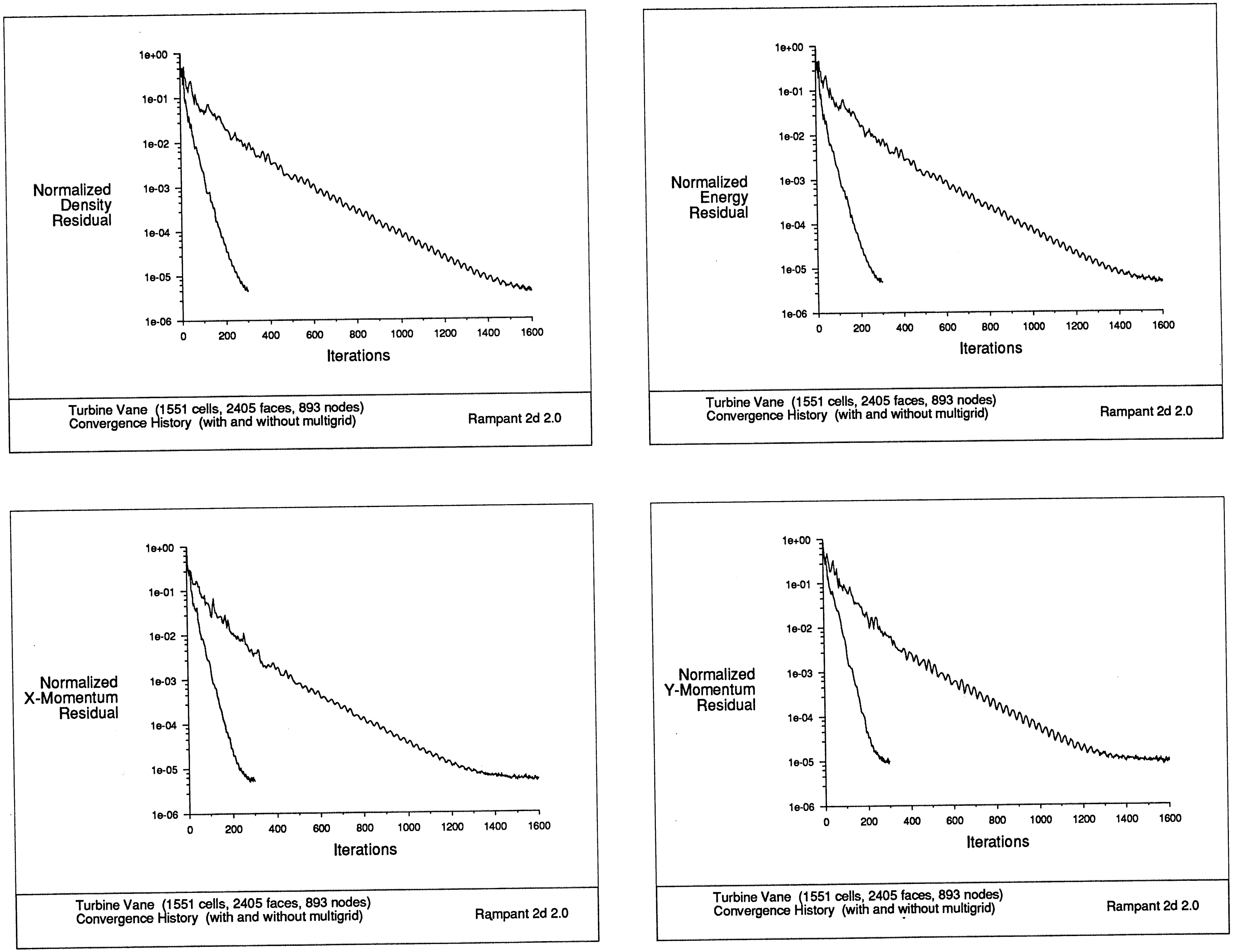
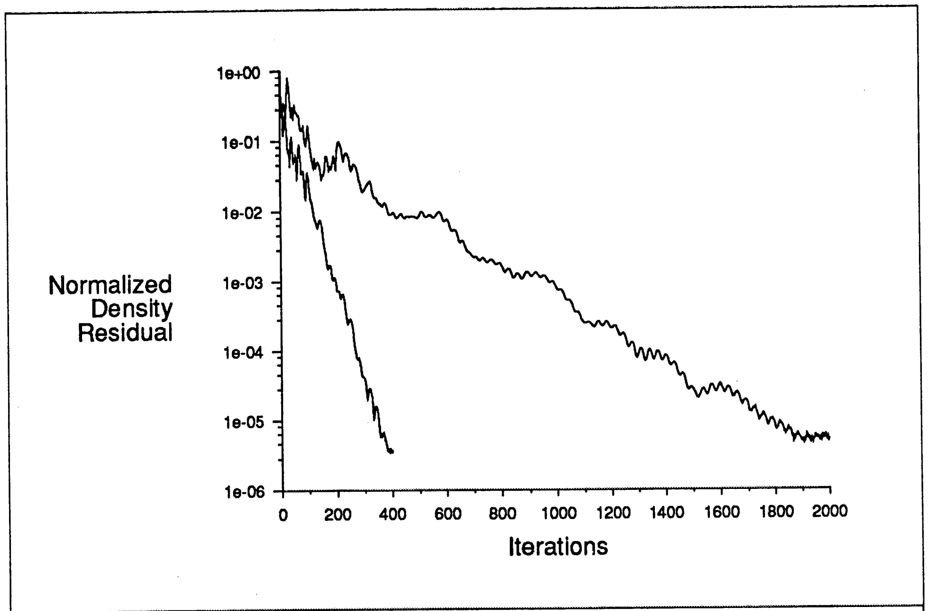

Normalized Density Residual
1e+00
1e-01
1e-02
1e-03
1e-04
1e-05
1e-06
0
200
400
600
800
1000
1200
1400
1600
Iterations
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Convergence History (with and without multigrid)
Rampant 2d 2.0
Normalized Energy Residual
Normalized X-Momentum Residual
Normalized Y-Momentum Residual

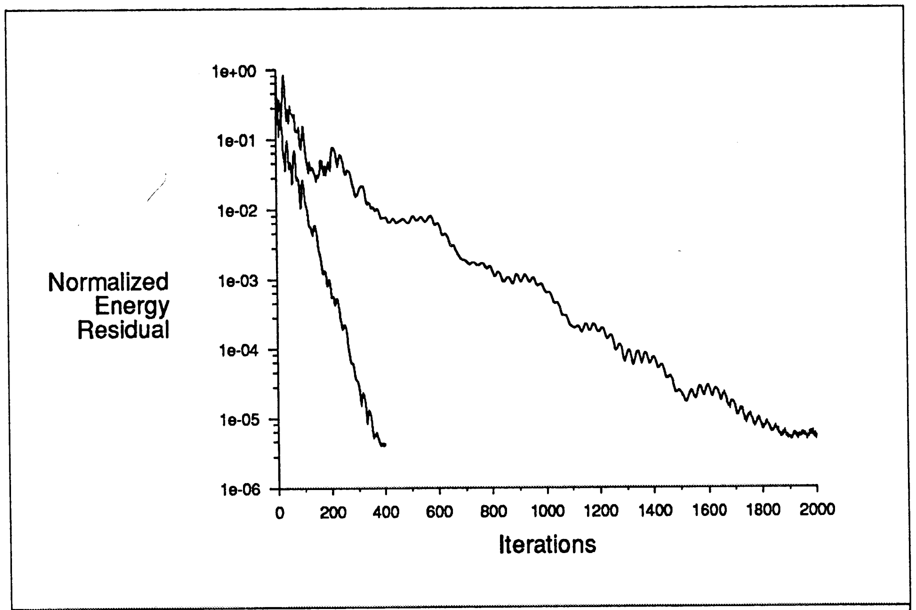
Figure 4

Figure 5



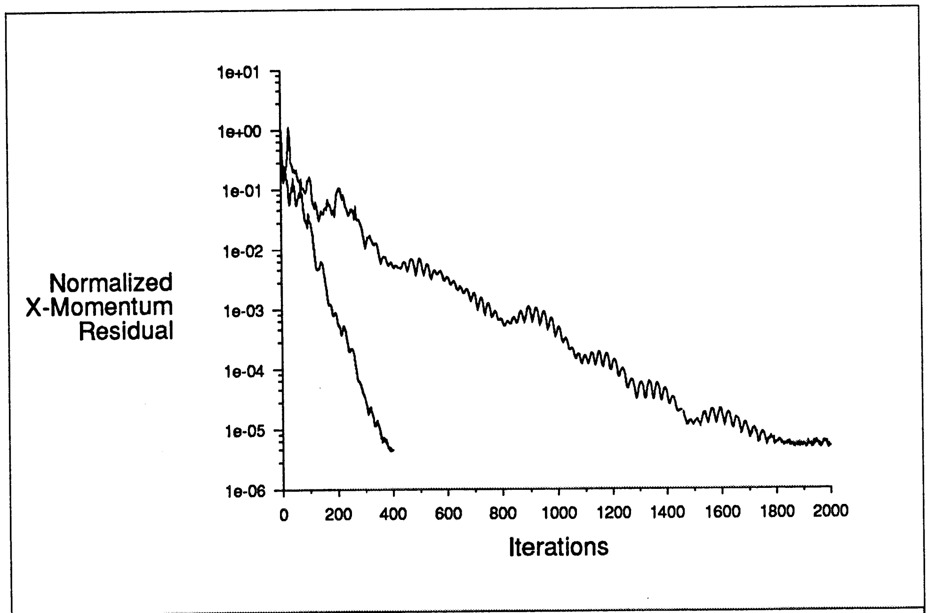
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Convergence History (with and without multigrid)

Rampant 2d 2.0



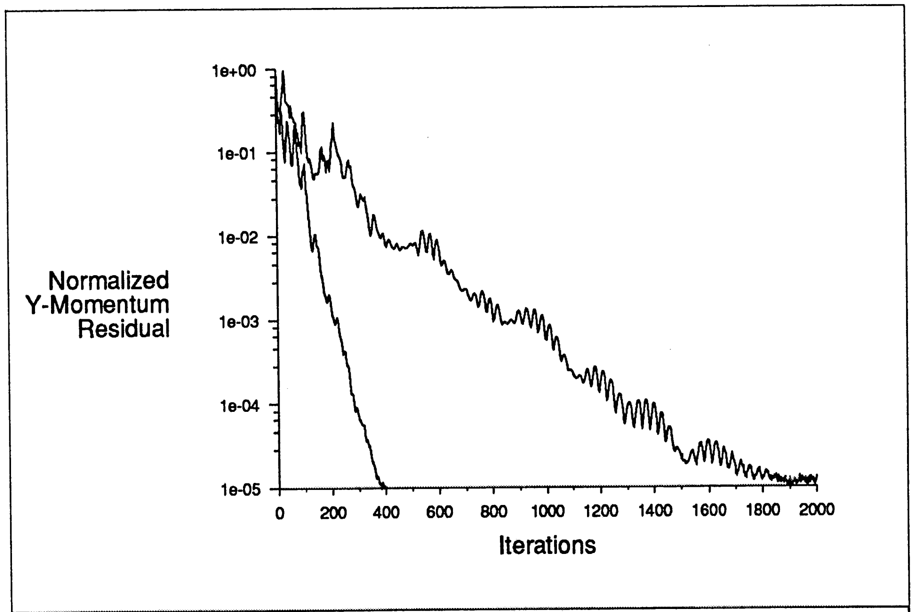
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Convergence History (with and without multigrid)

Rampant 2d 2.0



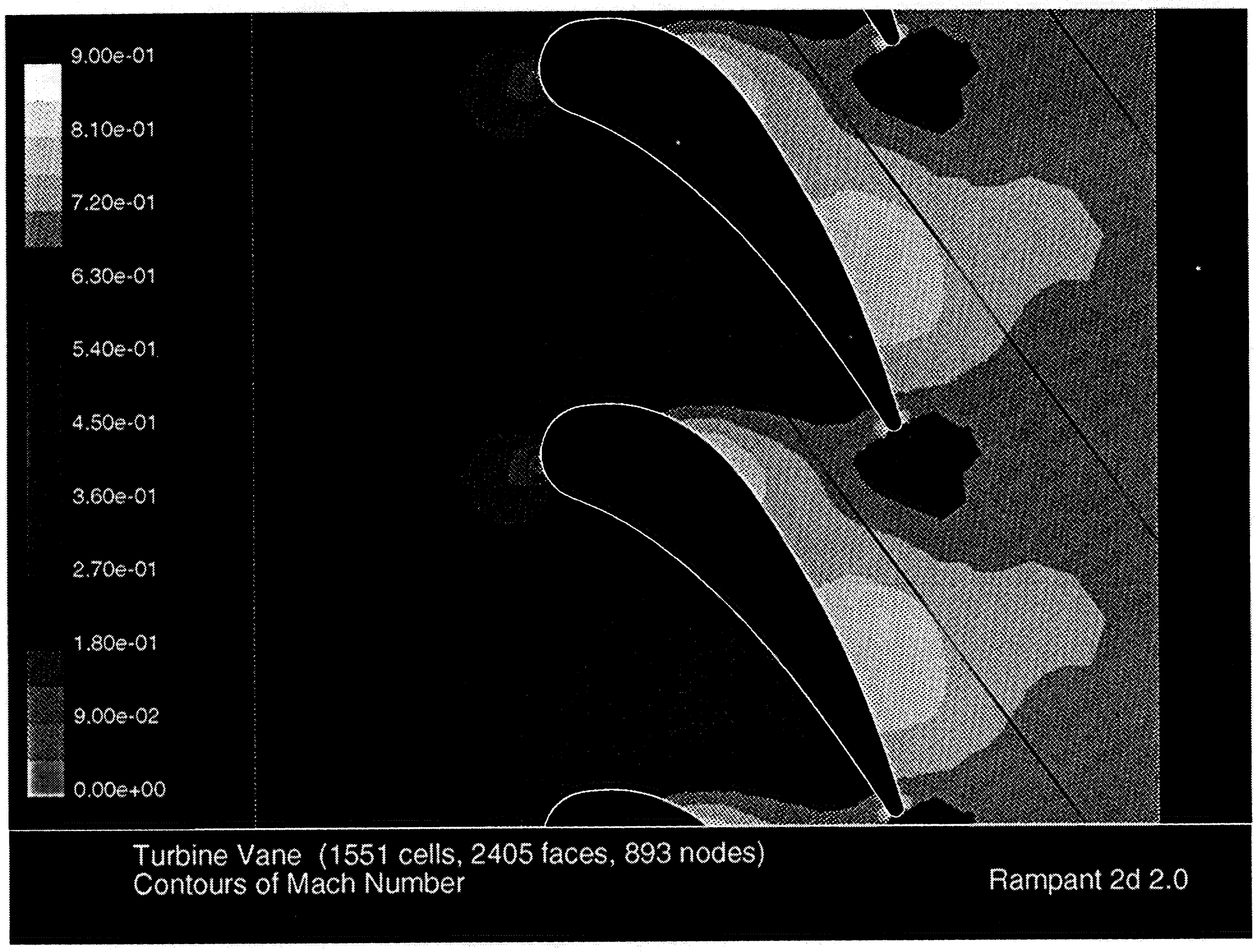
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Convergence History (with and without multigrid)

Rampant 2d 2.0



Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Convergence History (with and without multigrid)

Rampant 2d 2.0


9.00e-01
8.10e-01
7.20e-01
6.30e-01
5.40e-01
4.50e-01
3.60e-01
2.70e-01
1.80e-01
9.00e-02
0.00e+00
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Contours of Mach Number
Rampant 2d 2.0

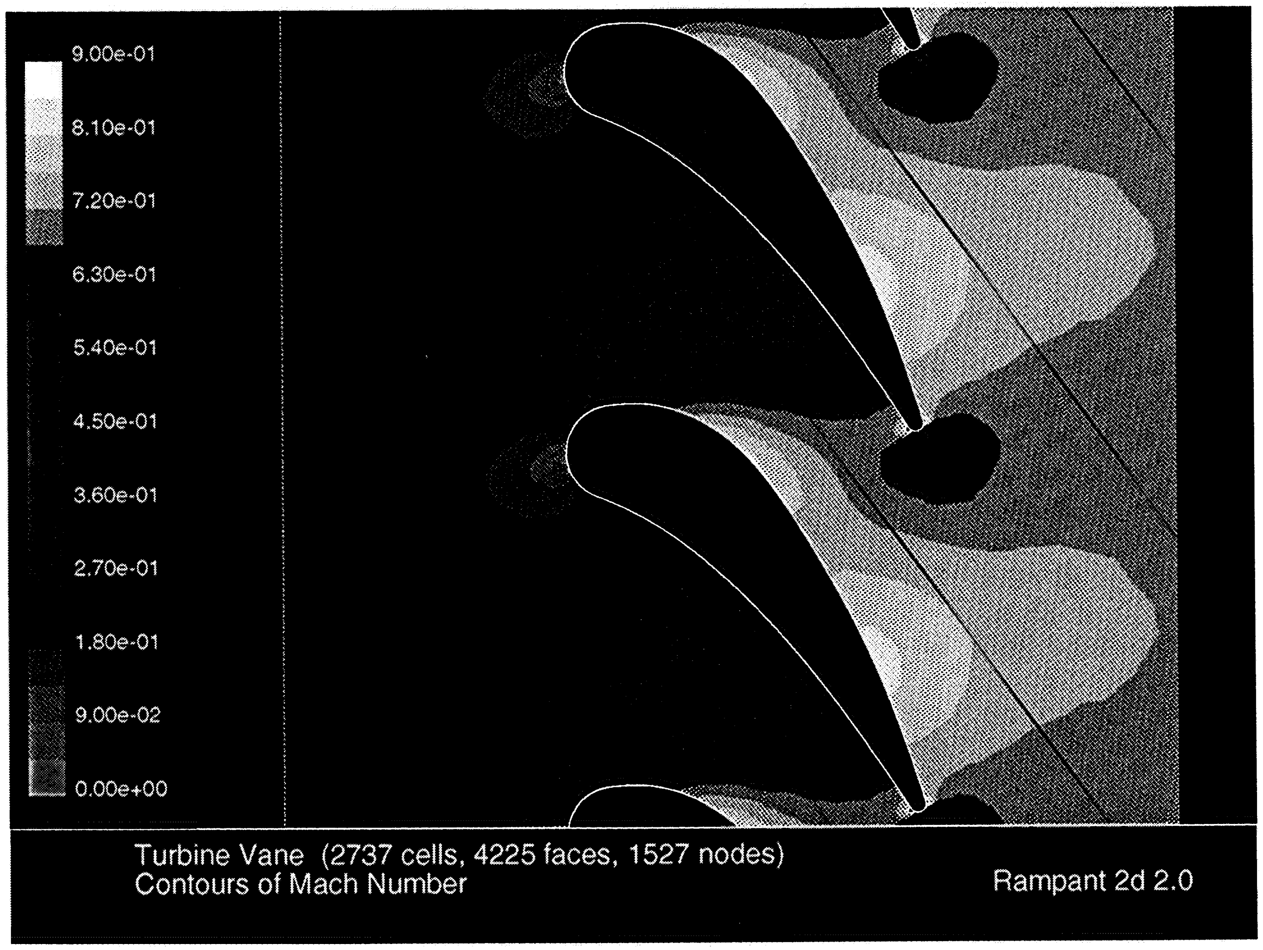
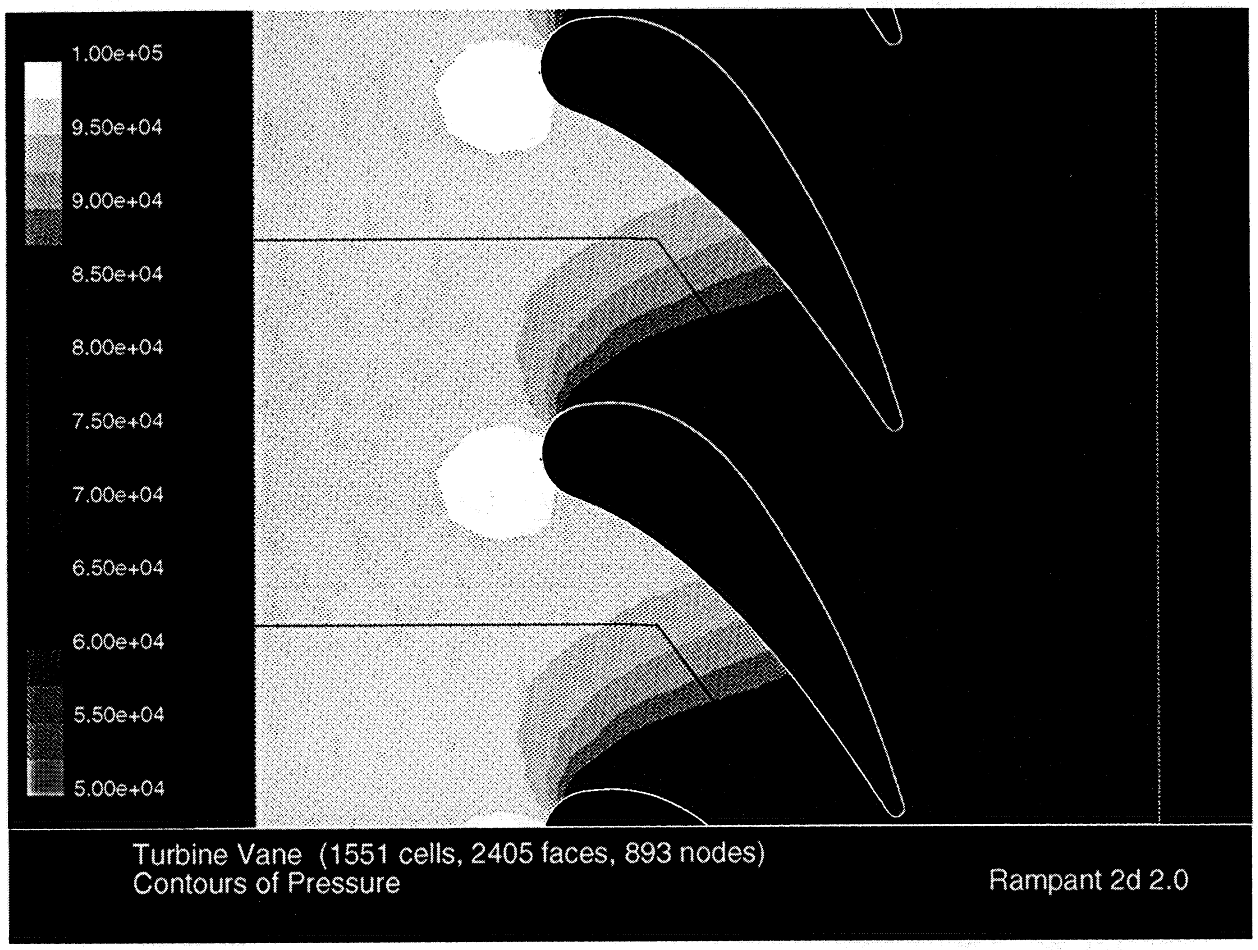

9.00e-01
8.10e-01
7.20e-01
6.30e-01
5.40e-01
4.50e-01
3.60e-01
2.70e-01
1.80e-01
9.00e-02
0.00e+00
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Contours of Mach Number
Rampant 2d 2.0

Figure 6


1.00e+05
9.50e+04
9.00e+04
8.50e+04
8.00e+04
7.50e+04
7.00e+04
6.50e+04
6.00e+04
5.50e+04
5.00e+04
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Contours of Pressure
Rampant 2d 2.0

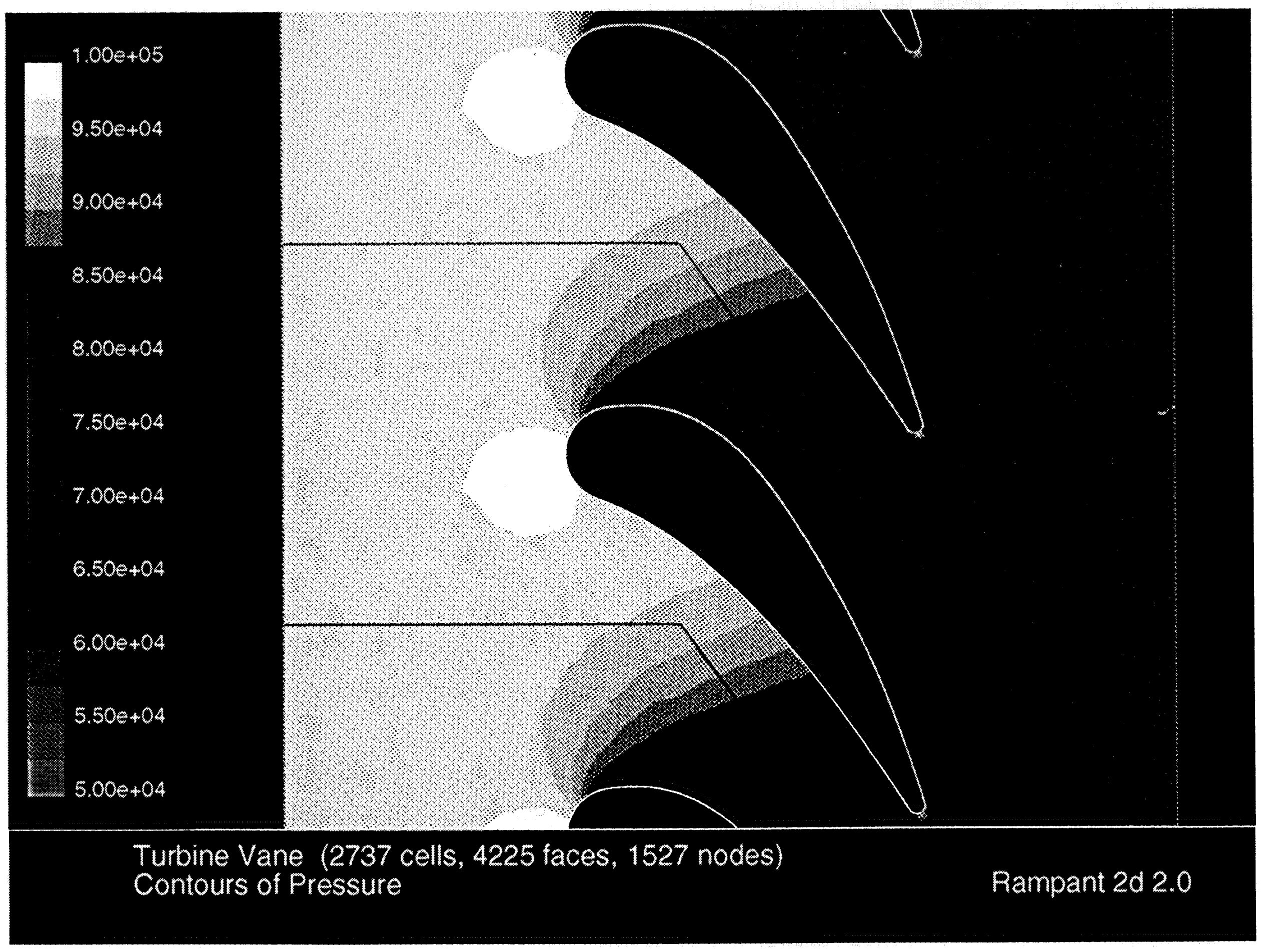
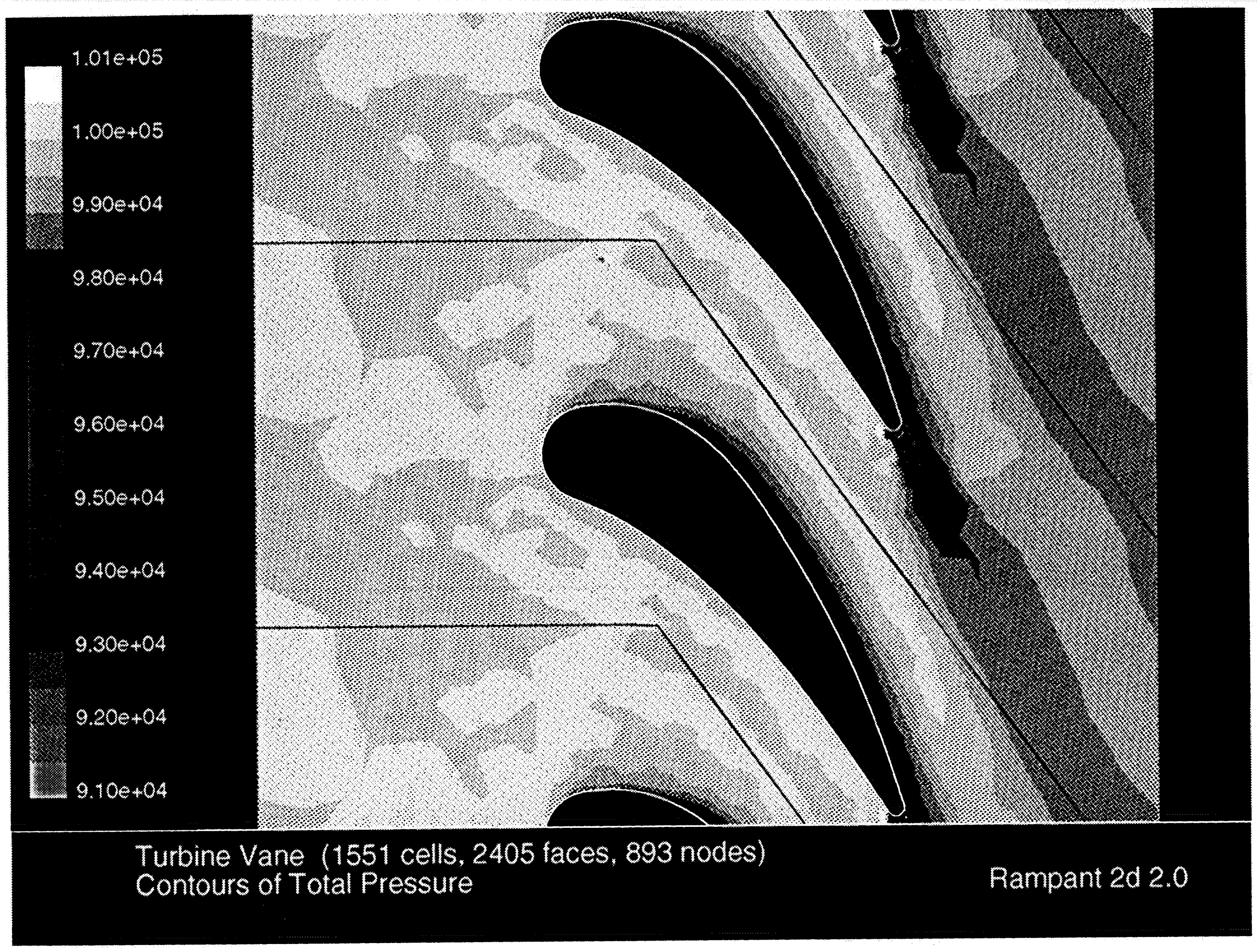

1.00e+05
9.50e+04
9.00e+04
8.50e+04
8.00e+04
7.50e+04
7.00e+04
6.50e+04
6.00e+04
5.50e+04
5.00e+04
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Contours of Pressure
Rampant 2d 2.0

Figure 7


1.01e+05
1.00e+05
9.90e+04
9.80e+04
9.70e+04
9.60e+04
9.50e+04
9.40e+04
9.30e+04
9.20e+04
9.10e+04
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Contours of Total Pressure
Rampant 2d 2.0

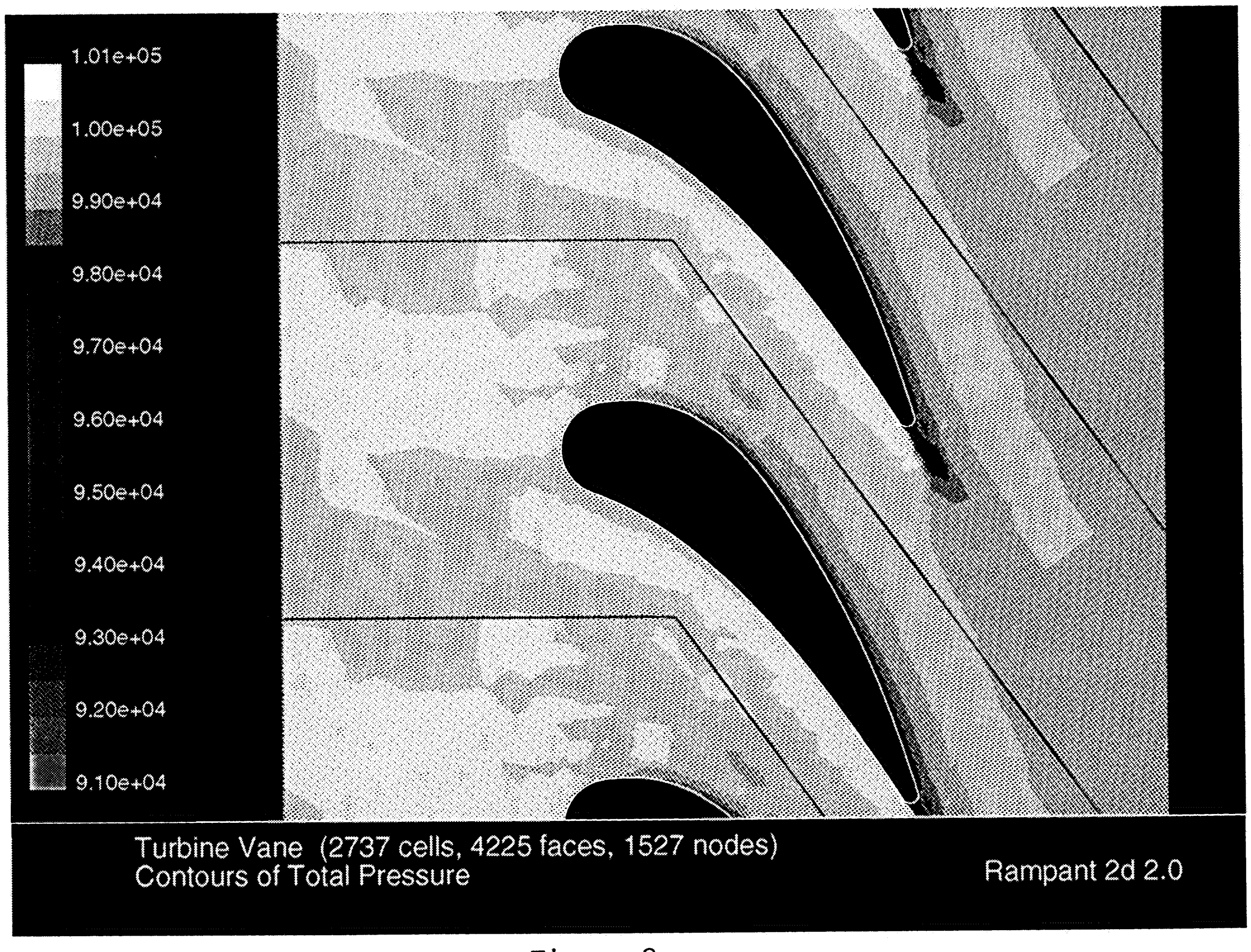
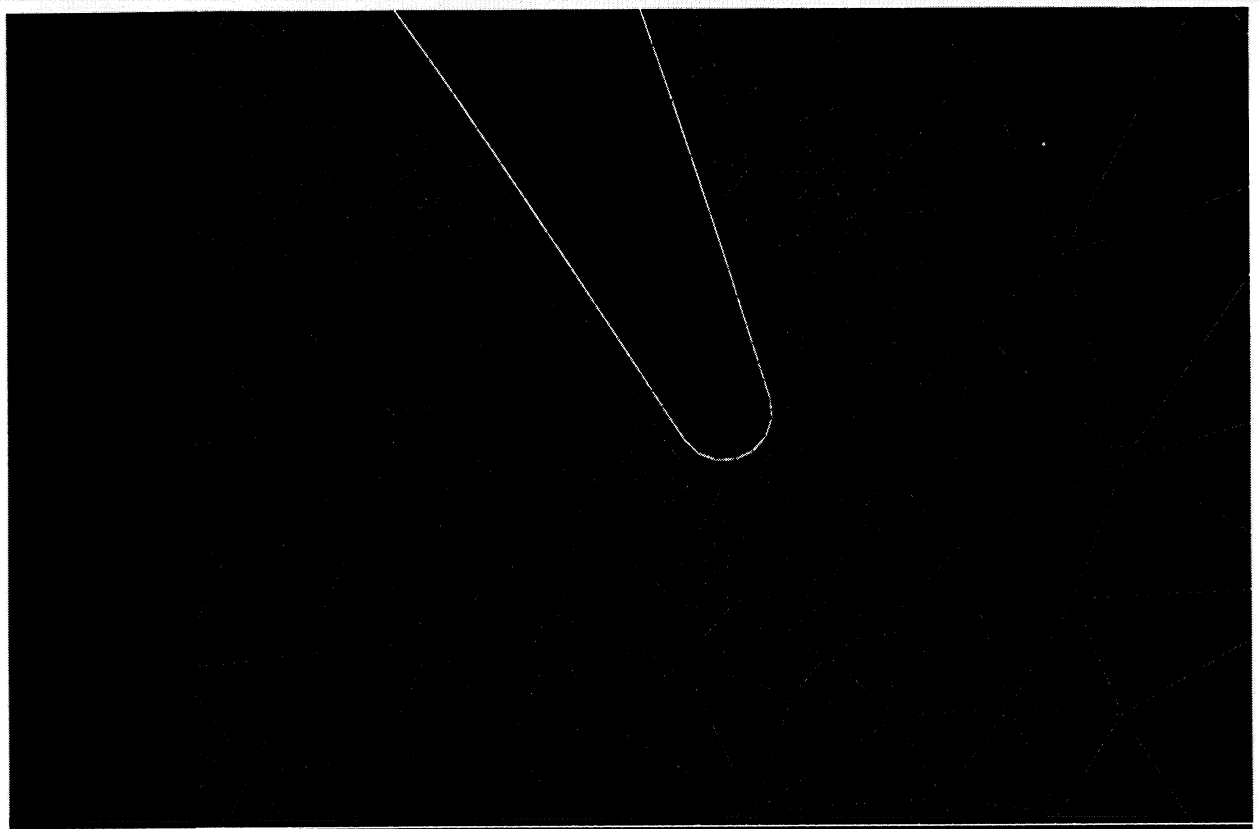

1.01e+05
1.00e+05
9.90e+04
9.80e+04
9.70e+04
9.60e+04
9.50e+04
9.40e+04
9.30e+04
9.20e+04
9.10e+04
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Contours of Total Pressure
Rampant 2d 2.0

Figure 8


Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Grid
Rampant 2d 2.0

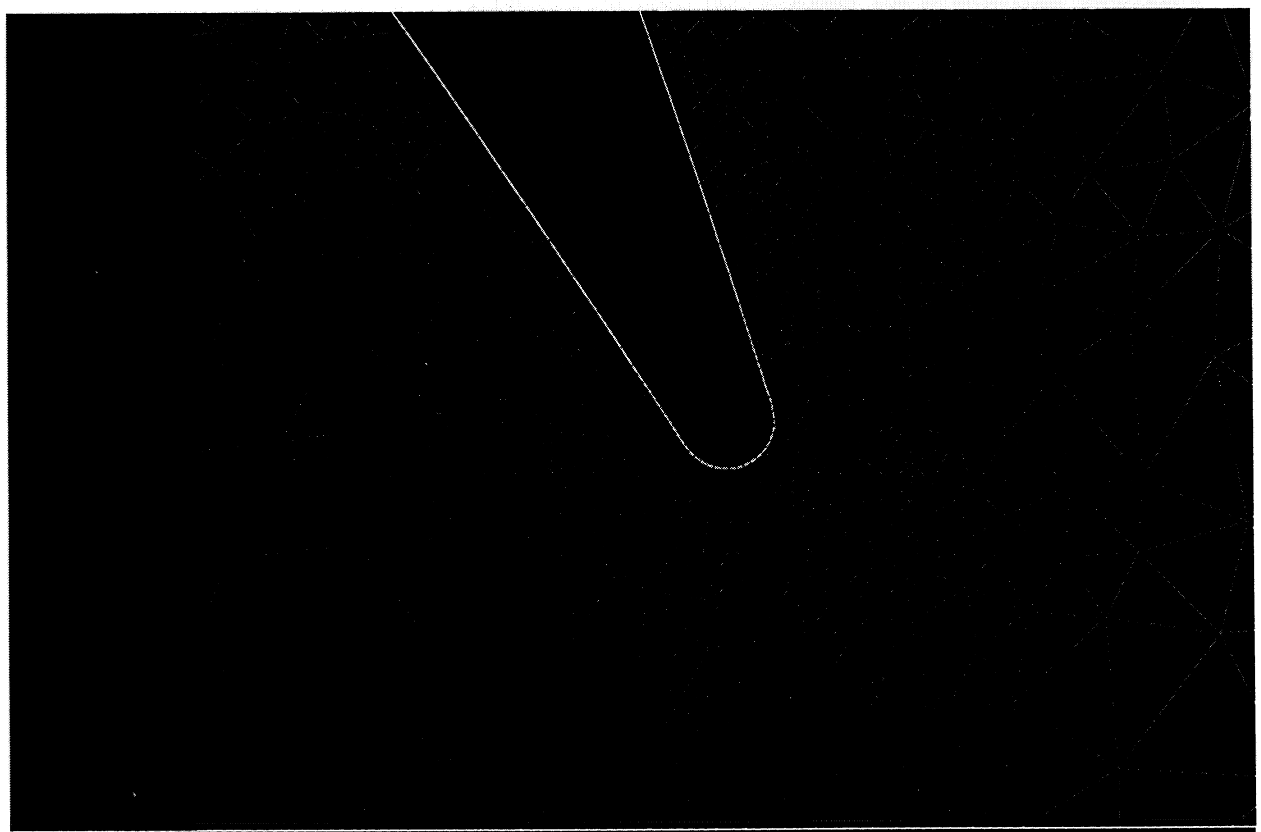
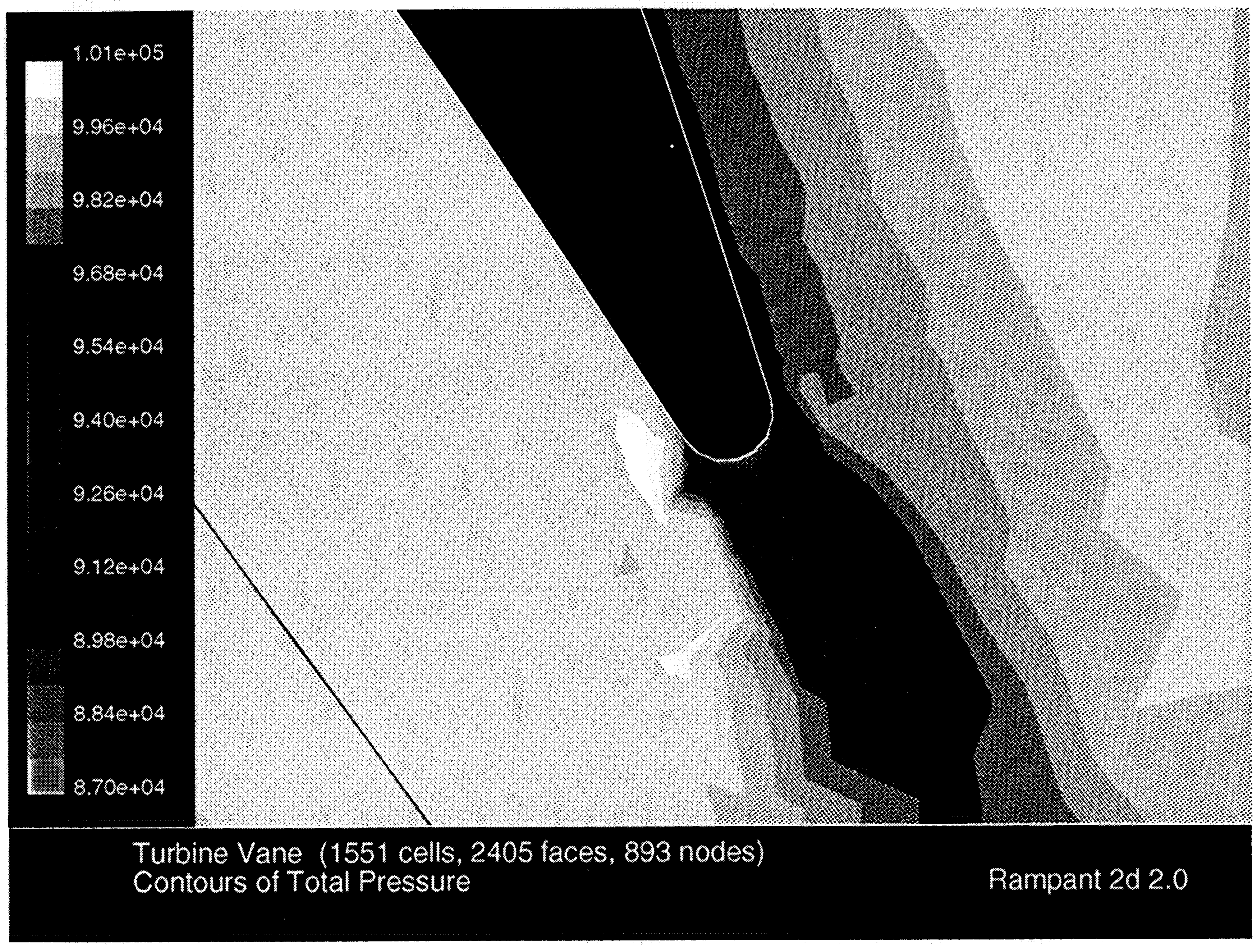

Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Grid
Rampant 2d 2.0

Figure 9


1.01e+05
9.96e+04
9.82e+04
9.68e+04
9.54e+04
9.40e+04
9.26e+04
9.12e+04
8.98e+04
8.84e+04
8.70e+04
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Contours of Total Pressure
Rampant 2d 2.0

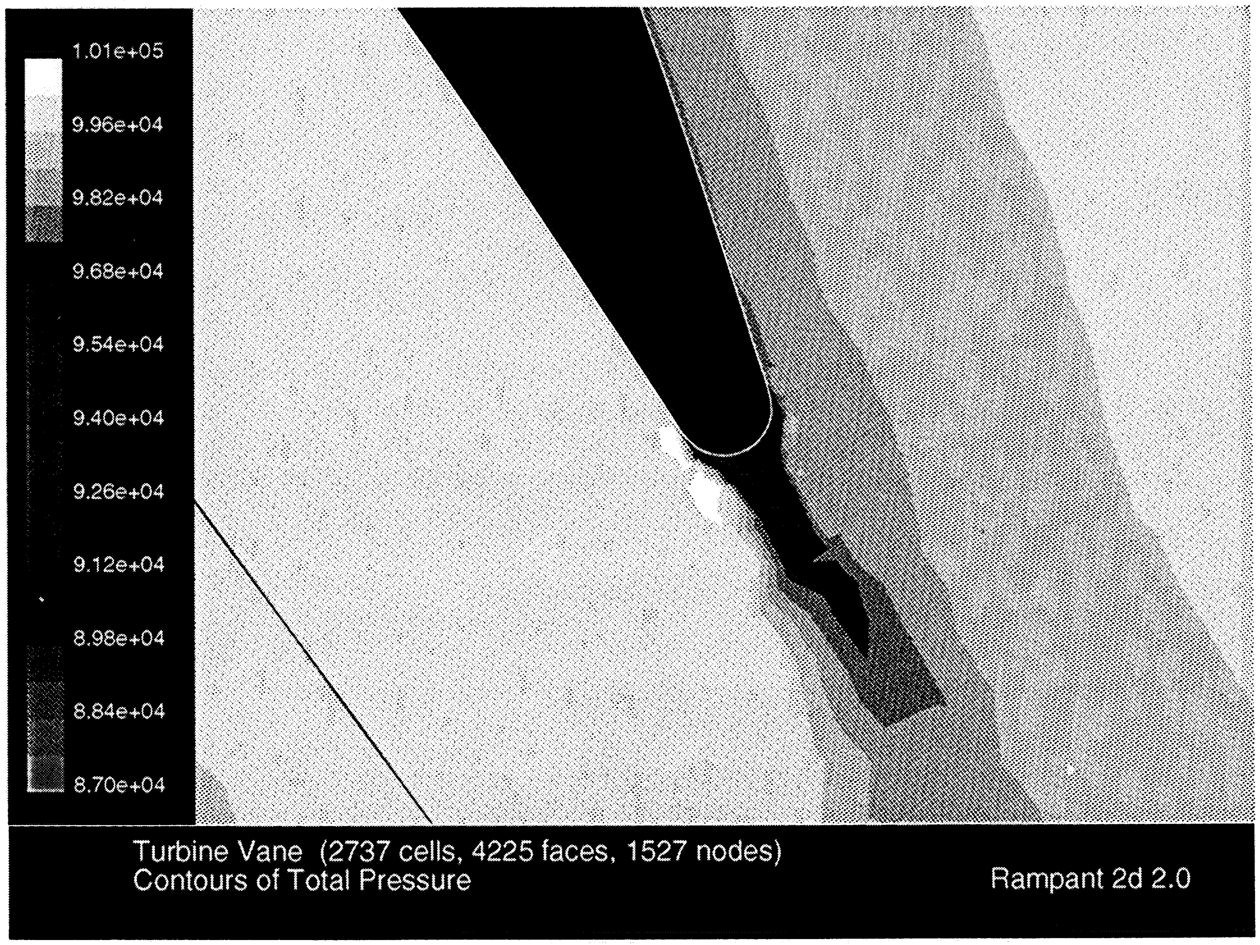
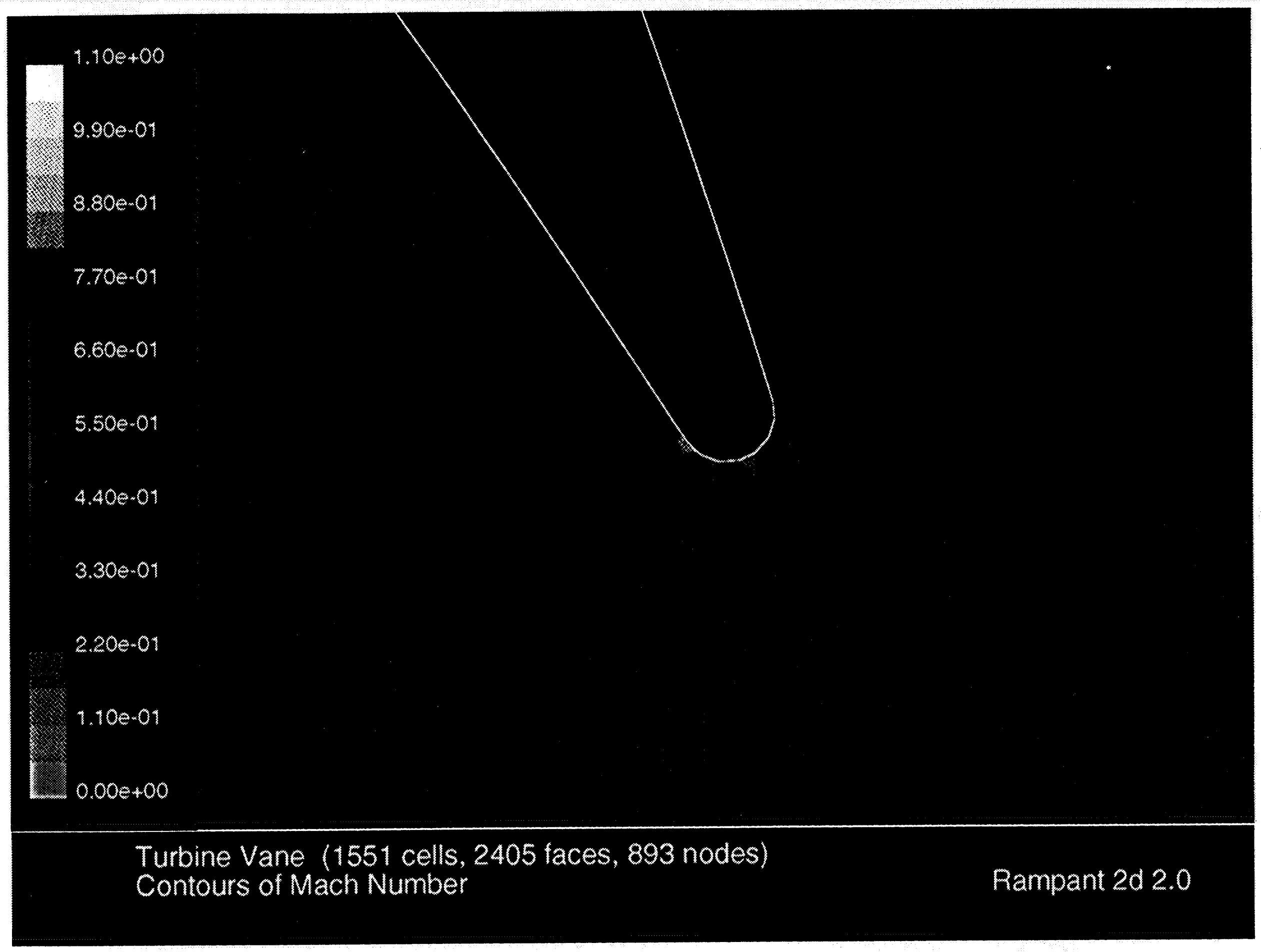

1.01e+05
9.96e+04
9.82e+04
9.68e+04
9.54e+04
9.40e+04
9.26e+04
9.12e+04
8.98e+04
8.84e+04
8.70e+04
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Contours of Total Pressure
Rampant 2d 2.0

Figure 10


1.10e+00
9.90e-01
8.80e-01
7.70e-01
6.60e-01
5.50e-01
4.40e-01
3.30e-01
2.20e-01
1.10e-01
0.00e+00
Turbine Vane (1551 cells, 2405 faces, 893 nodes)
Contours of Mach Number
Rampant 2d 2.0

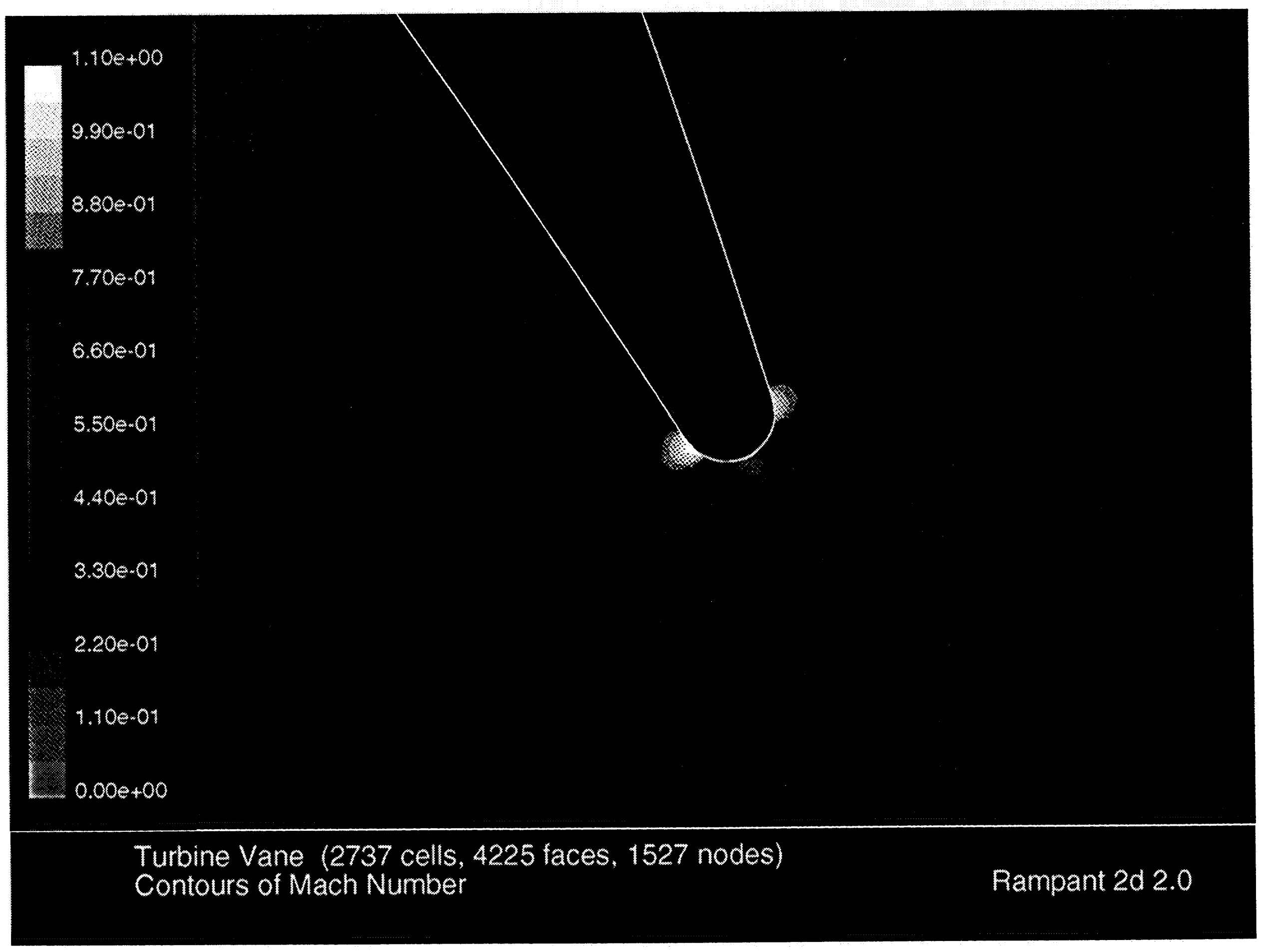
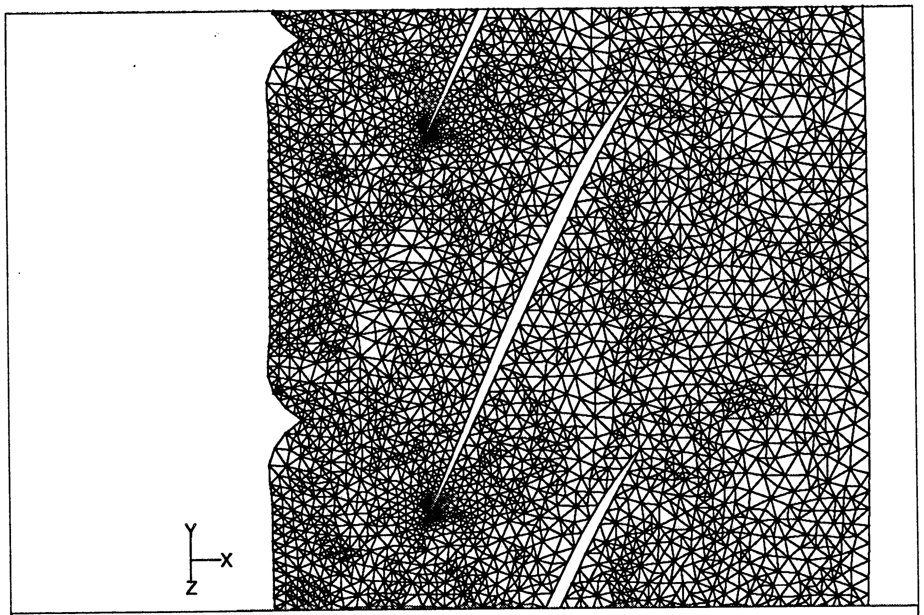

1.10e+00
9.90e-01
8.80e-01
7.70e-01
6.60e-01
5.50e-01
4.40e-01
3.30e-01
2.20e-01
1.10e-01
0.00e+00
Turbine Vane (2737 cells, 4225 faces, 1527 nodes)
Contours of Mach Number
Rampant 2d 2.0

Figure 11

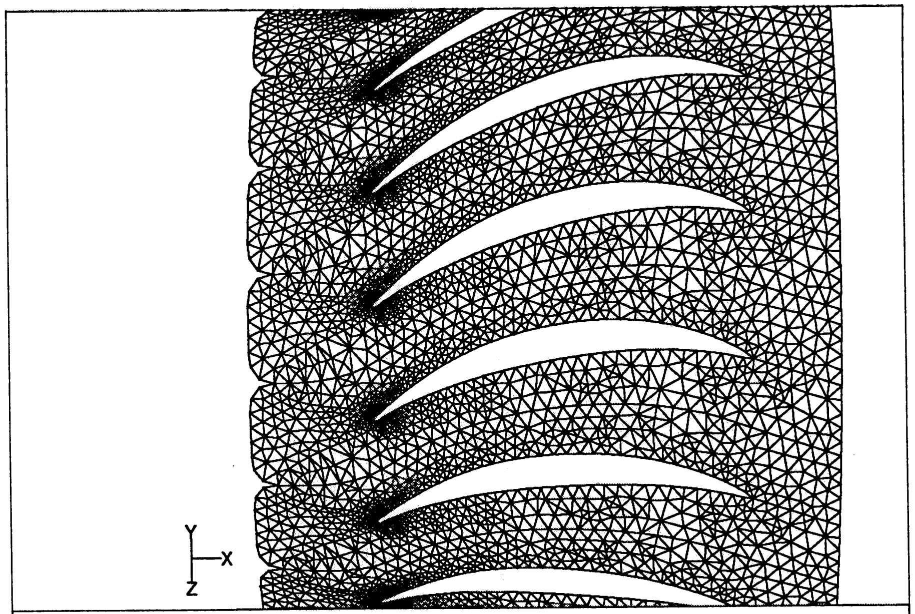
Figure 20

Figure 22



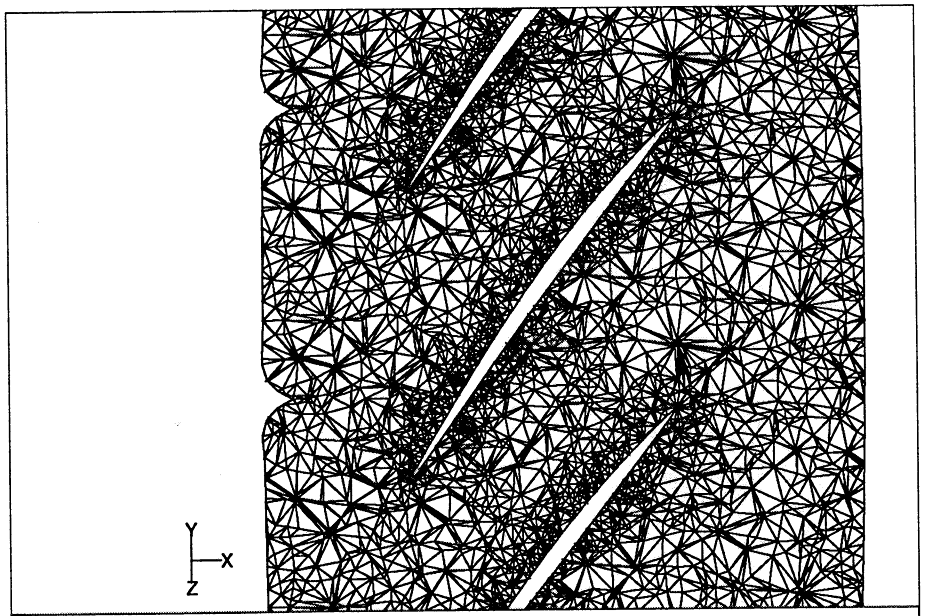
Rotor 67 (0% span)
Surface Grid

Rampant 3drke 2.0



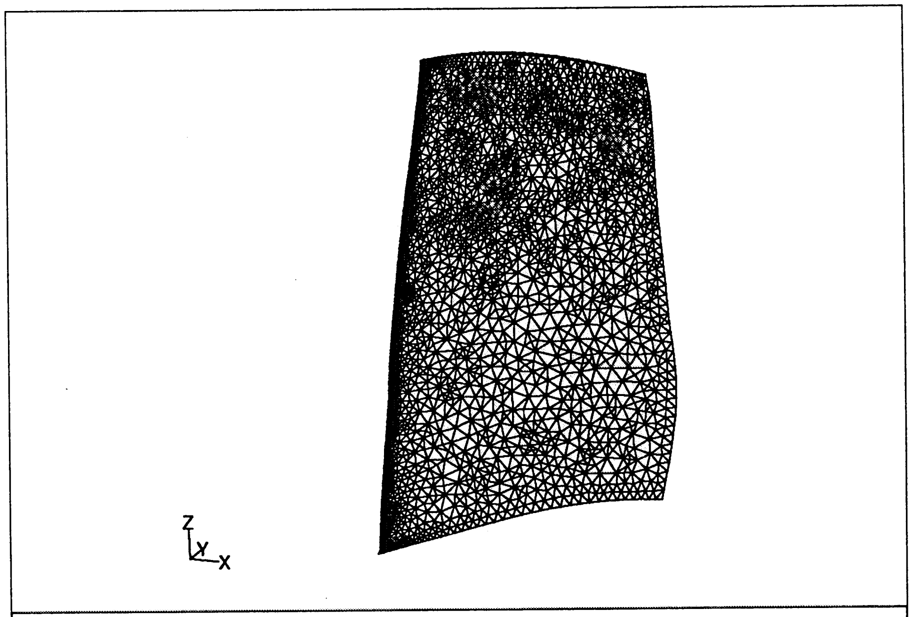
Rotor 67 (100% span)
Surface Grid

Rampant 3drke 2.0



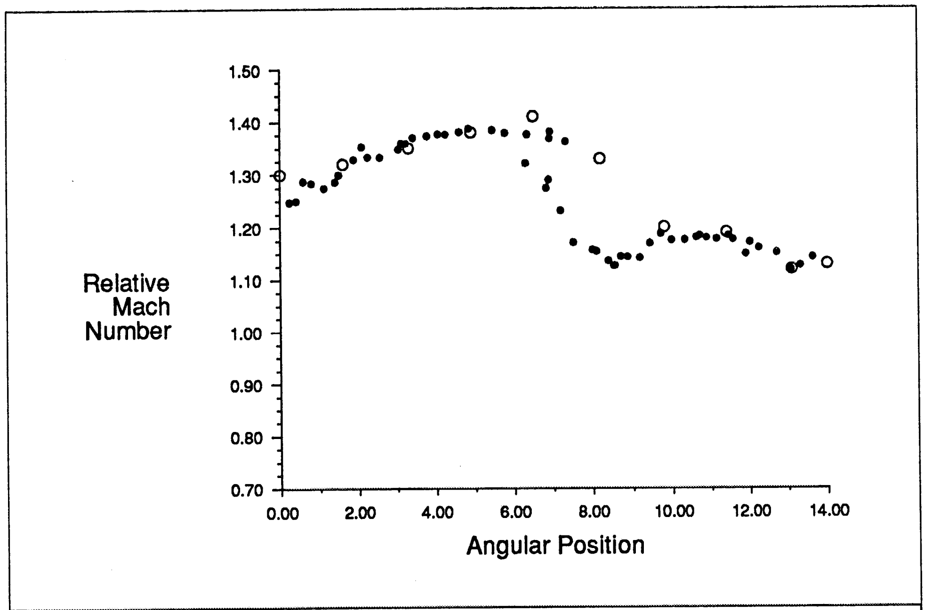
Rotor 67 (30% span)
Grid Slice

Rampant 3drke 2.0

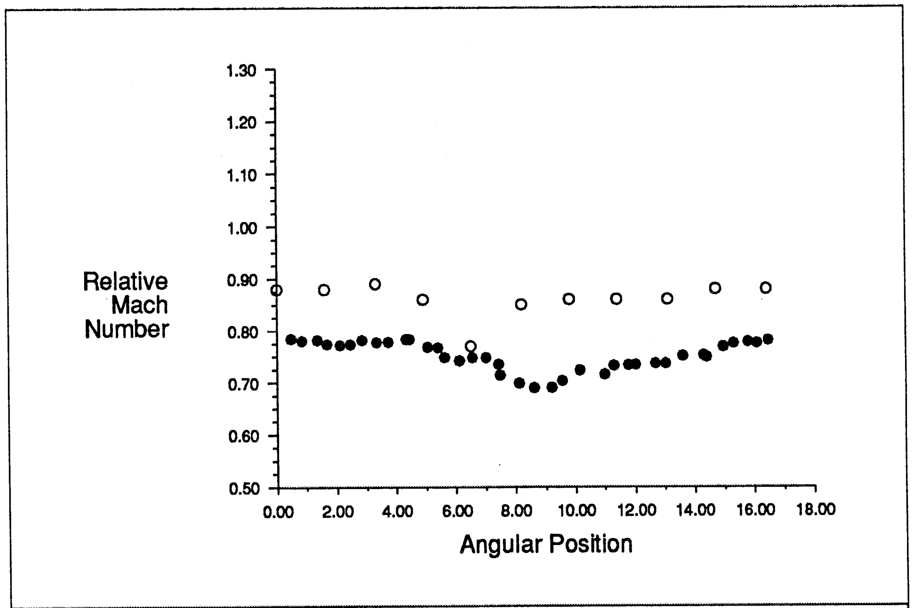


Rotor 67 (blade surface)
Surface Grid

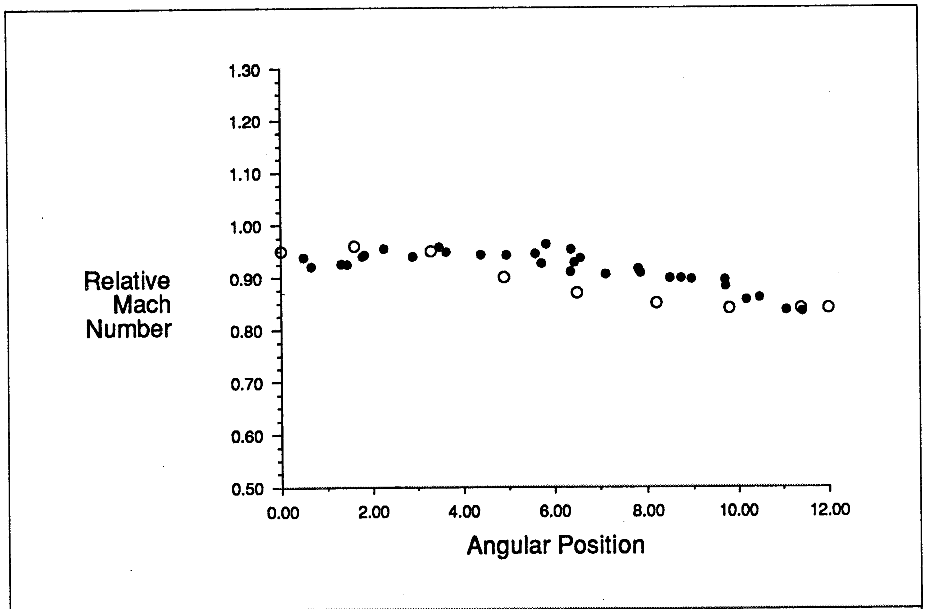
Rampant 3drke 2.0



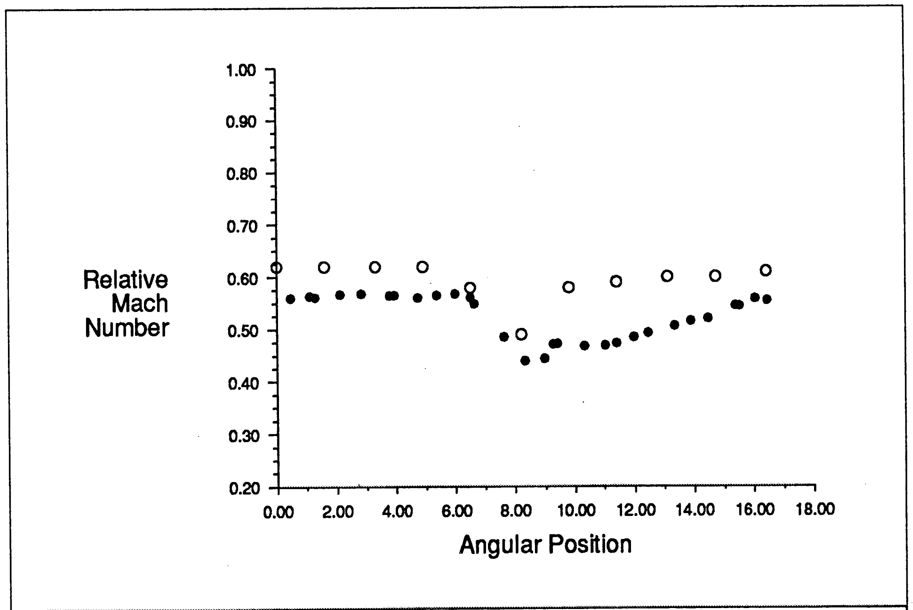
Rotor 67, 30% Span, 25% Chord
Open Symbols - Experiment
Filled Symbols - Rampant
Blade-to-blade distribution
Rampant 3drke 1.0



Rotor 67, 30% Span, 136% Chord
Open Symbols - Experiment
Filled Symbols - Rampant
Blade-to-blade distribution
Rampant 3drke 1.0



Rotor 67, 70% Span, 30% Chord
Open Symbols - Experiment
Filled Symbols - Rampant
Blade-to-blade distribution
Rampant 3drke 1.0
Mon Sep 14 1992



Rotor 67, 70% Span, 121% Chord
Open Symbols - Experiment
Filled Symbols - Rampant
Blade-to-blade distribution
Rampant 3drke 1.0

Figure 23