AIAA'89

AIAA-89-0080

An Adaptive Embedded Mesh Procedure for Leading-Edge Vortex Flows

Kenneth G. Powell
Michael A. Beer
Glenn W. Law

Department of Aerospace Engineering
University of Michigan
Ann Arbor, Michigan

27th Aerospace Sciences Meeting
January 9-12, 1989/Reno, Nevada

An Adaptive Embedded Mesh Procedure for Leading-Edge Vortex Flows

Kenneth G. Powell*
Michael A. Beer†
Glenn W. Law†
Department of Aerospace Engineering
The University of Michigan

Abstract

A procedure for solving the conical Euler equations on an adaptively refined mesh is presented, along with a method for determining which cells to refine. The solution procedure is a central-difference cell-vertex scheme. The adaptation procedure is made up of a parameter on which the refinement decision is based, and a method for choosing a threshold value of the parameter. The refinement parameter is a measure of mesh-convergence, constructed by comparison of locally coarse- and fine-grid solutions. The threshold for the refinement parameter is based on the curvature of the curve relating the number of cells flagged for refinement to the value of the refinement threshold. Results for three test cases are presented. The test problem is that of a delta wing at angle of attack in a supersonic free-stream. The resulting vortices and shocks are captured efficiently by the adaptive code.

Introduction

One of the primary challenges of computational fluid dynamics is that of disparate length scales. Due to nonlinearities in the equations of motion for a fluid, even the flow about a geometrically simple body can lead to localized high-gradient regions. The Euler equations can give rise to shocks, vortices, slip surfaces and contact discontinuities. More grid points are needed to resolve these high-gradient regions than would be necessary to resolve low-gradient regions. Since added grid points are expensive, both in terms of memory and computation, it is desirable to place the points where they are needed, and, conversely, to avoid placing them where they do not add to the quality of the solution. This is the basis of adaptive-mesh schemes.

There are two basic approaches in adaptive-mesh schemes. In one, the number of grid points is held constant, and the grid points are resditributed so that high resolution is provided where necessary. The redistribution may be done in any of several ways. Methods that have been used include:

- a method based on first and second derivatives of physical quantities (see, for example, [1]);
- a method based on a spring-mass analogy where the spring constants depend upon flow parameters (see, for example [2]);
- a method based on variational principles (see, for example [3]).

These and other adaptive redistribution methods are outlined in the review articles of Thompson [4] and Eiseman [5].

In the other basic approach, the adaptive refinement approach, mesh points are added by sudividing cells that are flagged for division. These methods include:

- ones in which entire zones are divided, yielding logically rectangular subgrids (see for example [6,7]);
- and ones in which single cells may be divided at random, yielding unstructured meshes (see for example [8,9,10]).

These methods are outlined in the review article of Berger [6]. The approach taken in this paper is that of adaptive refinement, in which any cell or group of cells may be refined, leading to an unstructured mesh.

Since, in general, the location of the high-gradient regions is not known in advance, a method of sensing them must be developed. This gives rise to the need to formulate a refinement parameter. If, for instance, shock waves are to be detected, pressure gradient could be be used as a parameter. That is, the pressure gradient could be calculated in each cell, and cells with high gradients of pressure would be refined. If, however, slip lines are to be detected, pressure gradient would not be a good parameter, since pressure is constant across a slip-line. Thus the primary goal in designing a refinement parameter is that it should be general enough to detect any region that requires refinement.

*Assistant Professor, AIAA Member
†Research Assistant, AIAA Member

Once a refinement parameter has been chosen, this gives a way of determining to what degree the solution would be improved by refining a given cell. In an adaptive redistribution method, grid points are redistributed based on this parameter in an analog manner. In an adaptive refinement method, the relation is digital; a cell is either flagged for refinement or left alone. For the adaptive refinement scheme, then, a threshold must be set, so that cells with a value of the refinement parameter greater than the threshold value are refined. Determination of this threshold also must be as general as possible. Setting a fixed value of the threshold or a fixed percentage of cells to be refined is not general enough.

The only remaining problem is to design a scheme that works on an unstructured mesh that has been adaptively refined. These three components; the solution scheme, the refinement criterion and the refinement threshold are described below for the solution of the conical Euler equations about a delta wing in supersonic flow. This problem makes a particularly good test problem for adaptive-refinement schemes, because of the presence of shocks, vortex cores and vortex sheets in the flows [11,12,13].

Governing Equations

The governing equations applied here are the conical Euler equations. They are derived from the three-dimensional Euler equations via an assumption of conical self-similarity. The three-dimensional Euler equations may be written in vector form as

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho E\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ \rho uh_0\end{bmatrix}+\frac{\partial}{\partial y}\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vw\\ \rho vh_0\end{bmatrix}+\frac{\partial}{\partial z}\begin{bmatrix}\rho w\\ \rho uw\\ \rho vw\\ \rho w^2+p\\ \rho wh_0\end{bmatrix}=0\,,$$

where h_0 is the stagnation enthalpy,

$$h_0 = E+\frac{p}{\rho}.$$

The ideal gas law, which may be written as

$$p=(\gamma-1)\,\rho\left[E-\frac{u^2+v^2+w^2}{2}\right],$$

closes the set of equations. Introducing the conical variables

$$r = \sqrt{x^2+y^2+z^2}$$

$$\eta = \frac{y}{x}$$

$$\zeta = \frac{z}{x}$$

and assuming conical self-similarity (that the solution is independent of r), the Euler equations become

$$\frac{r}{\kappa}\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho E\end{bmatrix}+\frac{\partial}{\partial \eta}\begin{bmatrix}\rho\bar{v}\\ \rho u\bar{v}-\eta p\\ \rho v\bar{v}+p\\ \rho w\bar{v}\\ \rho h_0\bar{v}\end{bmatrix}+\frac{\partial}{\partial \zeta}\begin{bmatrix}\rho\bar{w}\\ \rho u\bar{w}-\zeta p\\ \rho v\bar{w}\\ \rho w\bar{w}+p\\ \rho h_0\bar{w}\end{bmatrix}+2\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ \rho uh_0\end{bmatrix}=0\,,$$

where

$$\bar{u} = u+\eta v+\zeta w$$

$$\bar{v} = v-\eta u$$

$$\bar{w} = w-\zeta u$$

and

$$\kappa=\sqrt{1+\eta^2+\zeta^2}\,.$$

These conical Euler equations may be expressed in terms of the state vector $\mathbf{U}$, the Cartesian flux vector $\mathbf{F}$ and the conical flux vectors $\hat{\mathbf{G}}$ and $\hat{\mathbf{H}}$ as

$$\frac{r}{\kappa}\frac{\partial \mathbf{U}}{\partial t}+\frac{\partial \hat{\mathbf{G}}}{\partial \eta}+\frac{\partial \hat{\mathbf{H}}}{\partial \zeta}+2\mathbf{F}=0\,.$$

The unsteady terms have been included so that an iterative procedure may be employed to reach a conically self-similar steady-state. The equations are solved on the unit sphere by setting $r=1$. They could be solved for any value of r and would give the same solution.

Solution Procedure

The solution scheme used is a finite-volume, multi-stage scheme in which the state variables are stored at the nodes (i.e. a cell-vertex scheme). Similar schemes have been used by other researchers [14,15]. The scheme used here is formulated for an unstructured mesh, and allows for different cells in the mesh to be at different levels of refinement. The grid generation, spatial discretization, added artificial viscosity, temporal discretization and boundary and interface procedures are described below, along with the data structure that underlies the algorithm.

Grid Generation

For the cases presented here, the grid generation is carried out by a Joukowsky transformation. The (η,ζ) plane is mapped to a complex χ plane, in which the wing becomes a circle, by the transformation

$$\eta + i\zeta = \chi + \tan^2 \frac{\left(\frac{\pi}{2} - \Lambda\right)}{2\chi}$$

where Λ is the leading-edge sweep of the wing. In the χ plane, i = constant lines are equiangularly spaced rays emanating from the origin and j = constant lines are concentric rings. Grid points are generated along rays in the χ plane by an exponential stretching. This procedure yields near-conformal grids with good resolution near the wing. A grid generated by this procedure forms the base grid, i.e., the grid before adaptive local refinement has taken place. The refinement is handled by sub-dividing a cell into four sub-cells. This is done by adding a node on each face of the cell to be divided, and one in the center of the cell. Bilinear interpolations are used to find the coordinates of the new points. A typical grid is shown in Figure 3.

Spatial Discretization of Equations

The finite-volume discretization of the partial differential equations is formulated by integrating the conical Euler equations over a cell. This gives

$$\iint_\Omega \frac{r}{\kappa}\frac{\partial \mathbf{U}}{\partial t}\, d^2x + \iint_\Omega \left[\frac{\partial \hat{\mathbf{G}}}{\partial \eta} + \frac{\partial \hat{\mathbf{H}}}{\partial \zeta}\right] d^2x + \iint_\Omega 2\mathbf{F}\, d^2x = 0.$$

Using Gauss' theorem and the mean value theorem, this may be rewritten as

$$A\overline{\frac{r}{\kappa}\frac{\partial \mathbf{U}}{\partial t}} + \oint_{\partial\Omega} \left[\hat{\mathbf{G}}d\zeta - \hat{\mathbf{H}}d\eta\right] + 2A\overline{\mathbf{F}} = 0$$

where an overbar denotes a cell-average and A is the cell area.

The line integral of the fluxes is carried out by a trapezoidal integration about the cell, i.e.

$$\oint_{\partial\Omega} \left[\hat{\mathbf{G}}d\zeta - \hat{\mathbf{H}}d\eta\right] =$$

$$\sum_{faces} \left[\frac{1}{2}\left(\hat{\mathbf{G}}_1 + \hat{\mathbf{G}}_2\right)(\zeta_2 - \zeta_1) - \frac{1}{2}\left(\hat{\mathbf{H}}_1 + \hat{\mathbf{H}}_2\right)(\eta_2 - \eta_1)\right],$$

where the subscripts denote the two nodes that define the face, ordered so that the integral is carried out in a counter-clockwise sense.

The cell-average of the source term is calculated by averaging the source term at the four nodes defining the cell. This gives the residual at the center of the cell, $\overline{\partial \mathbf{U}/\partial t}$.

Added Artificial Viscosity

The added artificial viscosity is a blend of a nonlinear second-difference and a linear fourth-difference. It is constructed from a weighted Laplacian, $\overline{L}$, and an unweighted Laplacian squared, L^2, respectively. The weighting function $\mathcal{W}$ is a normalized Laplacian of the pressure,

$$\mathcal{W} = \left|\frac{L(p)/p}{||L(p)/p||_\infty}\right|,$$

where $||\ ||_\infty$ denotes the L_∞ norm, so that $0 < \mathcal{W} < 1$. This weighting is chosen for the efficient capturing of shocks and vortices. It causes the second-difference term to be first-order in high-gradient regions and small elsewhere; the fourth-difference term is third-order everywhere. Thus the artificial viscosity is given by

$$\begin{aligned} D(\hat{\mathbf{U}}) &= \epsilon_2 D_2(\hat{\mathbf{U}}) - \epsilon_4 D_4(\hat{\mathbf{U}}) \\ &= \epsilon_2 \overline{L}(\mathcal{W}, \hat{\mathbf{U}}) - \epsilon_4 L^2(\hat{\mathbf{U}}) \end{aligned}$$

where $\hat{\mathbf{U}}$ is a modified state vector, with the energy term ρE replaced by ρh_0, so that the discrete equations permit a solution with constant total enthalpy.

Temporal Discretization of Equations

The spatial discretization resulted in a semi-discrete equation for $\overline{\partial \mathbf{U}/\partial t}$, the residual for a cell. The temporal discretization is made up of two process: one to distribute $\overline{\partial \mathbf{U}/\partial t}$ to the nodes, and one to integrate the changes at the nodes. For the first process, a simple $(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ distribution is used. For the second, a multi-stage scheme is used. The multi-stage scheme is given by, for iteration n,

$$\begin{aligned} \mathbf{U}^{(0)} &= \mathbf{U}^n \\ \mathbf{U}^{(1)} &= \mathbf{U}^{(0)} - \alpha_1 \left[\Delta t \overline{\frac{\kappa}{rA}} \mathbf{C}^{(0)} - \lambda \mathbf{D}^{(0)}\right] \\ &\vdots \\ \mathbf{U}^{(k)} &= \mathbf{U}^{(0)} - \alpha_1 \left[\Delta t \overline{\frac{\kappa}{rA}} \mathbf{C}^{(k-1)} - \lambda \mathbf{D}^{(k-1)}\right] \\ \mathbf{U}^{n+1} &= \mathbf{U}^{(k)} \end{aligned}$$

where $\mathbf{C}$ is the convective operator

$$\mathbf{C}^{(k)} = \oint_{\partial\Omega} \left[\hat{\mathbf{G}}\left(\mathbf{U}^{(k)}\right) d\zeta - \hat{\mathbf{H}}\left(\mathbf{U}^{(k)}\right) d\eta\right] + 2A\overline{\mathbf{F}}\left(\mathbf{U}^{(k)}\right)$$

and $\mathbf{D}$ is the damping operator

$$\mathbf{D}^{(k)} = \mathbf{D}_2\left(\hat{\mathbf{U}}^{(k)}\right) - \mathbf{D}_4\left(\hat{\mathbf{U}}^{(k)}\right).$$

The time-step factor, $\Delta t\overline{\kappa/rA}$, is given by

$$\Delta t \overline{\frac{\kappa}{rA}} = \lambda \frac{\overline{\kappa}}{r} \max_{faces} \left[\frac{1}{u_i n_i + c\sqrt{n_i n_i}}\right].$$

where λ is the CFL number and n_i is the i^{th} component of the non-normalized face normal. In the cases presented here, a four-stage scheme was chosen, with coefficients

$$\alpha_1 = \frac{1}{4} \qquad \alpha_2 = \frac{1}{3}$$

$$\alpha_3 = \frac{1}{2} \qquad \alpha_4 = 1$$

Boundary Conditions

There are boundary conditions to be enforced at the physical boundaries;

1. No flux through the wing,
2. Free-stream conditions upstream of the bow-shock,
3. Kutta condition at the leading-edges;

and at the numerical boundaries;

1. no flow through the symmetry plane for zero-yaw cases,
2. conservation at the embedding interfaces.

The wing boundary condition is met by retaining only the pressure terms in the flux calculation on cell faces which abut the wing. No pressure extrapolation is necessary since the state variables are stored at the nodes.

The free-stream boundary condition is implemented by ensuring that the outer boundary of the domain is outside the bow shock and enforcing free-stream conditions there.

The Kutta condition is enforced implicitly by ensuring that the artificial viscosity is present near the leading-edges. This is done by arbitrarily setting the pressure switch $\mathcal{W}$ to one at several (four or five) nodes in the vicinity of the leading-edges. Numerical experience demonstrates that this is adequate to ensure smooth separation [11].

The symmetry plane is introduced so that the flow past a wing at zero yaw may be solved on a half-plane. The condition to be enforced at the symmetry line is that the through-flow velocity, v, is zero. This is implemented by setting v to zero initially and zeroing the distributed flux for the y-direction momentum equation at each iteration.

The condition of conservation at the embedding interfaces is met by careful treatment of the flux integration and the artificial viscosity calculation there. Proper treatment of the flux integration is ensured by treating the midpoint on the face at which a coarse cell abuts a fine cell as a dummy point. That is, the state vector and flux vector at that point are taken to be the average of the two endpoints of the face. In this way, the flux that the coarse cell sees through this face is exactly the same as the sum of the two half-faces, thus ensuring conservation. The artificial viscosity is constructed in terms of face-centered first differences, with any difference that is added to one node of the face subtracted from the other node. This ensures that the artificial viscosity is conservative, even at boundaries and interfaces.

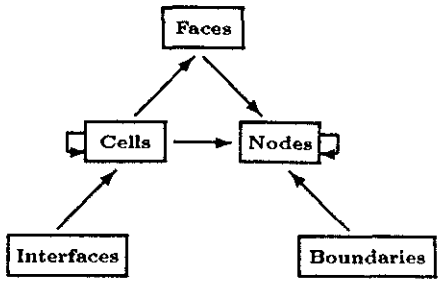


Figure 1: Data Structure

Data Structure

Local refinement of cells leads to grids that are inherently unstructured. That is, they cannot be "unwrapped" to give an (i, j) grid in computational space. This leads to the necessity of a different data structure than is normally used in CFD codes. The data structure used in this paper is shown in Figure 1. Each cell, each face and each node are numbered. The following arrays describe the connectivity of the grid:

- a pointer from each cell to its nodes;
- a pointer from each cell to its faces;
- a pointer from each cell to its neighbors;
- a pointer from each node to its neighbors;
- a pointers from each face to its nodes;
- a pointer to each interface cell;
- a pointer to each boundary node.

This choice of data structure is not unique. Many of the arrays are not "necessary," but help to speed up the computation. There is a trade-off between storage and computation in the design of a data structure; the more connectivity arrays available, the quicker the computation.

Refinement Parameter Choice

To develop an adaptive method of deciding which cells to refine, some parameter must chosen on which to base the decision. This parameter can be physical or numerical in its basis. For a physical refinement parameter, for instance, cells in which the pressure gradient

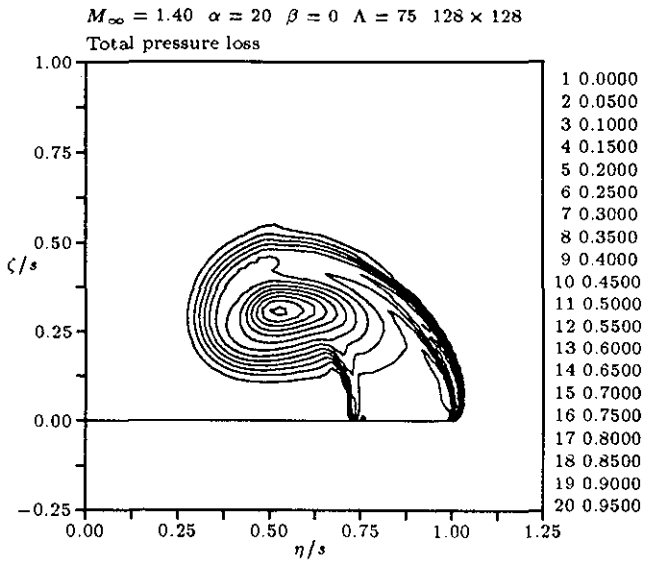


Figure 2: Total Pressure Loss for Shock-Vortex Case

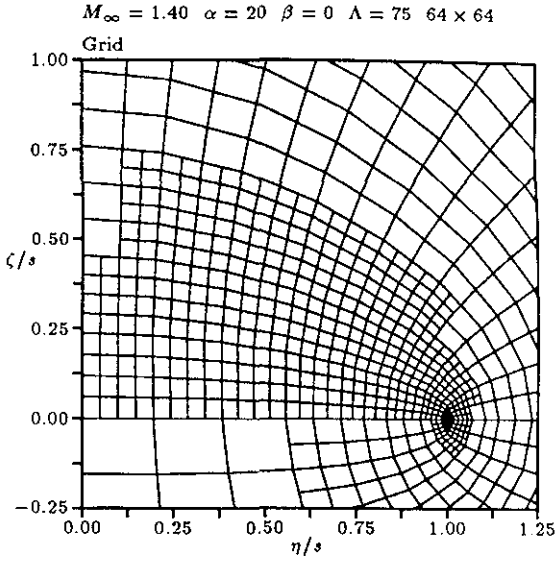


Figure 3: Grid before Refinement for Shock-Vortex Case

is high could be refined. This would lead to refinement at shocks and stagnation points, but not at constant-pressure features such as vortex sheets. For a numerical refinement parameter, some estimate of the local truncation error could be used. Local truncation error is difficult to ascertain, however.

Much of the work in adaptive schemes to date has concentrated on the detection of shocks. In the cases presented here, both shocks and vortices are present, and both must be detected. Refinement parameters that work well in the detection of shocks do not necessarily detect vortices. Contours of total pressure loss for a shock-vortex case are shown in Figure 2. The grid before embedding is shown in Figure 3. The results of using an undivided difference of pressure as a refinement parameter are shown in Figure 4. The feeding sheet is not detected at all, since it is a constant-pressure feature. Instead, the inboard portion of the vortex, where the pressure gradient is the highest, is flagged for refinement. Another refinement parameter popular for detecting shocks is an undivided difference of density. This gives similar results, as shown in Figure 5.

It is not difficult to design a refinement parameter that will detect the vortex. Since there is a total pressure loss that is localized to the vicinity of the vortex in computations of vortex flows [11,16], one could use p_0 or its gradient as the refinement and thus detect the vortex, as shown in Figures 6 and 7.

It is clear that, if it is known in advance what high-gradient features will occur in the flow, a physical criterion can be chosen that detects that particular feature. In general, however, flows will have different types of features, each requiring a different physical criterion for detection. One possible approach is to refine all cells

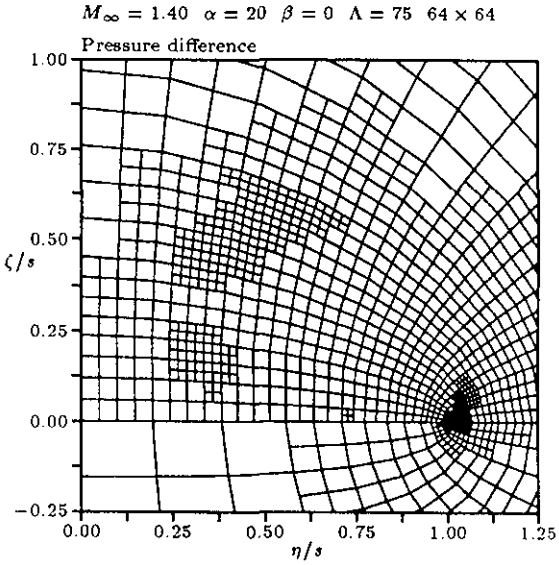


Figure 4: Grid Resulting from Pressure Difference-based Criterion

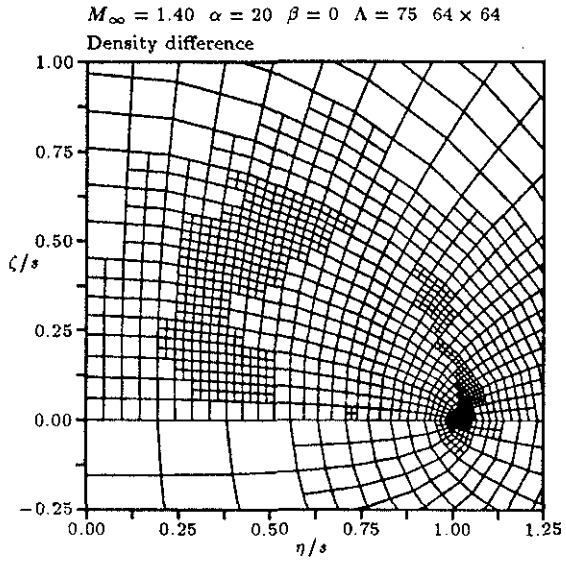


Figure 5: Grid Resulting from Density Difference-based Criterion

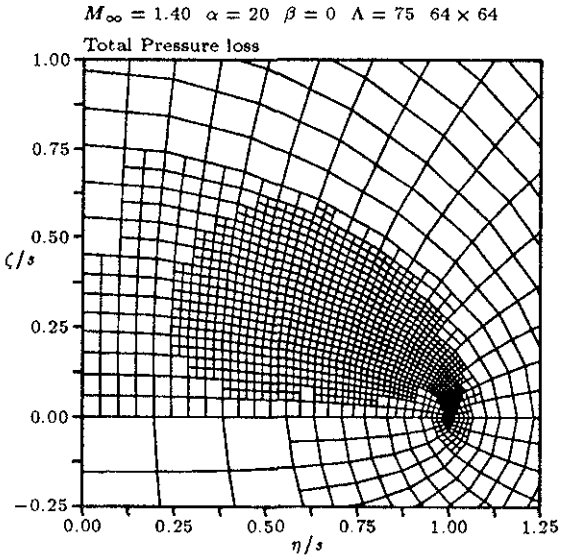


Figure 6: Grid Resulting from Total Pressure-Based Criterion

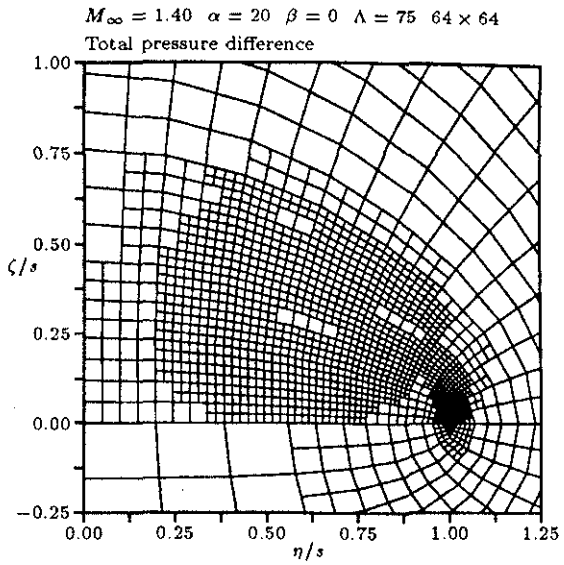


Figure 7: Grid Resulting from Total Pressure Difference-Based Criterion

that are flagged by any of several parameters. In other words, refine each cell in which the pressure gradient is high, *or* the density gradient is high, *or* the velocity gradient is high. This still requires *a priori* knowledge of the types of features that might occur, however.

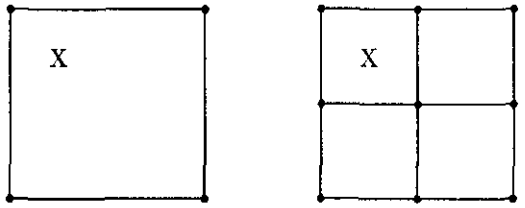
Numerical refinement parameters are attractive because they are in some sense more general that physical parameters. No *a priori* knowledge of how a high-gradient feature will manifest itself is necessary. Estimates of truncation error are relatively difficult to obtain, however. Estimates based on Taylor series will not be valid in non-smooth regions of the flow; estimates based on Richardson extrapolation rely on knowing the order of the scheme, and will not hold near boundaries or regions of grid stretching or skewing.

The refinement parameter chosen in this paper is a numerical one that is general and easy to compute. It is based on a test of whether the solution is mesh-converged in a given region. For a cell that has been refined, the solution at the center of one of the sub-cells after refinement is computed by bilinear interpolation among the nodes. This is then compared to the solution at the same point, before refinement, computed by bilinear interpolation among the corners of the coarse cell (See Figure 8). This is a measure of mesh-convergence; when the refinement has added no structure to the solution, the solution is locally mesh-converged.

Any one of a number of quantities can be measured for mesh-convergence; a logical choice is a normalized L_1 norm over the state-vector, i.e.

$$\epsilon_{refine} = \left|\frac{\rho_f - \rho_c}{\rho_f + \rho_f}\right| + \left|\frac{\rho q_f - \rho q_c}{\rho q_f + \rho q_c}\right| + \left|\frac{\rho E_f - \rho E_c}{\rho E_f + \rho E_c}\right|$$

where ϵ_{refine} is the refinement, a subscript c denotes a



Coarse Grid Fine Grid

Figure 8: Mesh-Convergence Refinement Criterion

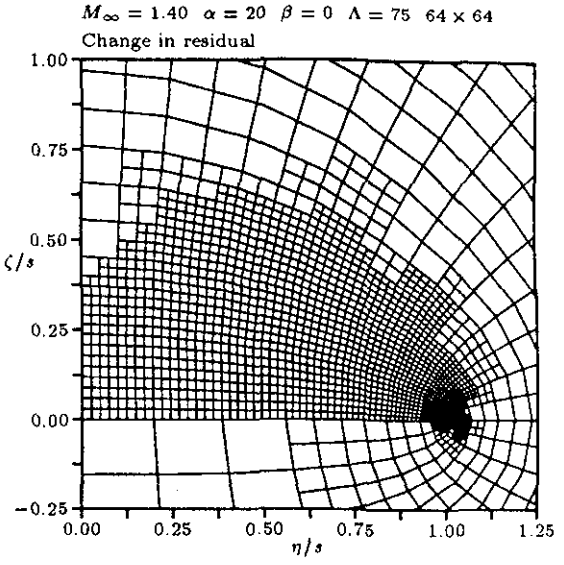


Figure 9: Grid Resulting from Mesh-Convergence-Based Criterion

value obtained from bilinear interpolation of the coarse mesh solution (i.e. before the previous refinement), and a subscript f denotes a value obtained from bilinear interpolation of the fine mesh solution (i.e. after the previous refinement). The grid resulting from this criterion is shown in Figure 9.

Refinement Threshold Choice

The choice of a refinement threshold determines how many cells will be refined. Too low a threshold will produce many cells that do not add to the quality of the solution, but slow down the computation. Too high a threshold will lead to a grid that does not provide refinement where needed.

Without *a priori* knowledge of the structure of the flow, it is impossible to determine the percentage of cells that might need refinement. If the grid is sufficiently coarse, *all* of the cells might need to be refined. If the solution is basically grid-converged, *none* of the cells might need to be refined. Clearly, for a truly general adaptive procedure, the refinement threshold must be determined on the fly.

Once a refinement parameter has been chosen, the threshold is determined in the following manner:

1. A histogram is constructed which shows the number of cells flagged for refinement if the threshold is set at a certain value;
2. A least-squares fit of a high-order polynomial to this curve is done to produce a differentiable version of the histogram;
3. The curvature of this polynomial is calculated;
4. The threshold is set at the lowest value of the refinement parameter that produces a local maximum in the curvature plot.

To illustrate this process, the histogram, smoothed histogram and the curvature of the histogram for a particular case are shown in Figures 10–12. The threshold, which is set by the leftmost extremum of the curvature, is marked on the original histogram plot. The abscissae are normalized by twice the mean of the refinement parameter; the ordinates are normalized by the total number of cells in the grids.

That a truly general method of choosing the threshold must be taken can be seen in Figures 13–16. Figures 13 and 14 are the histograms for the vortex-only case, at two different levels of adaptation. The shape of the histogram changes as the solution becomes more refined. Figures 15 and 16 are the histograms for the shock-vortex case. Each of the four curves leads to a different threshold, and a different number of cells flagged for refinement.

"Smoothing" the Grid

If the refinement parameter and threshold outlined above were applied to a typical run, the grid that would result could have isolated fine cells in coarse regions, and isolated coarse cells in fine regions. Although the finite volume scheme has been constructed to make interfaces as transparent as possible, convergence is reached more quickly if these isolated regions do not exist. Therefore, it is advisable to "smooth" the

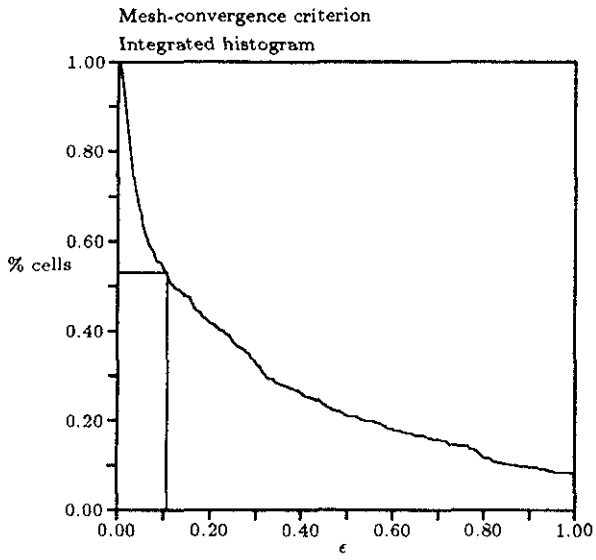


Figure 10: Histogram

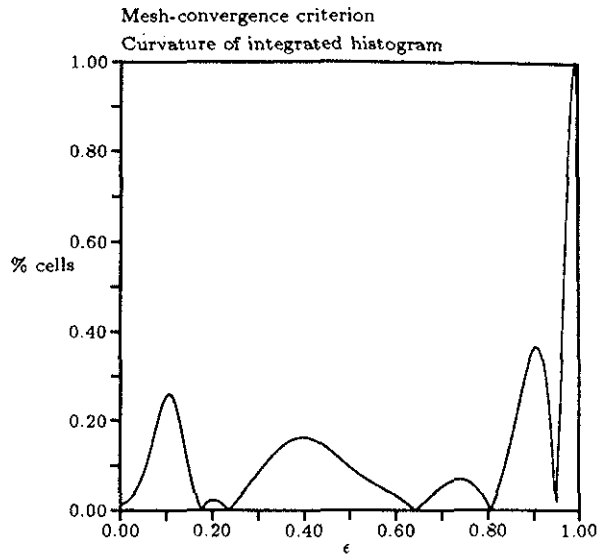


Figure 12: Curvature of Histogram Curve

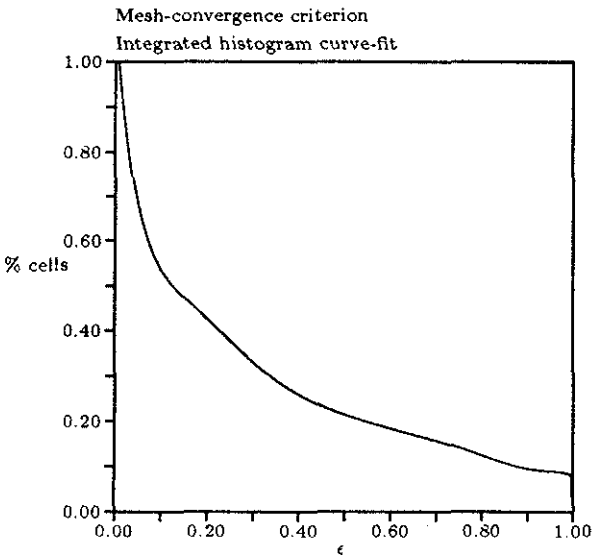


Figure 11: Smoothed Histogram (Least-Squares Fit)

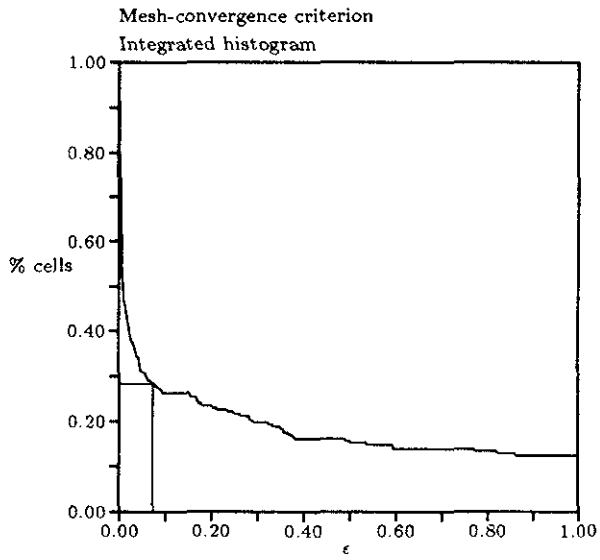


Figure 13: Histogram for Vortex Case (Initial Grid)

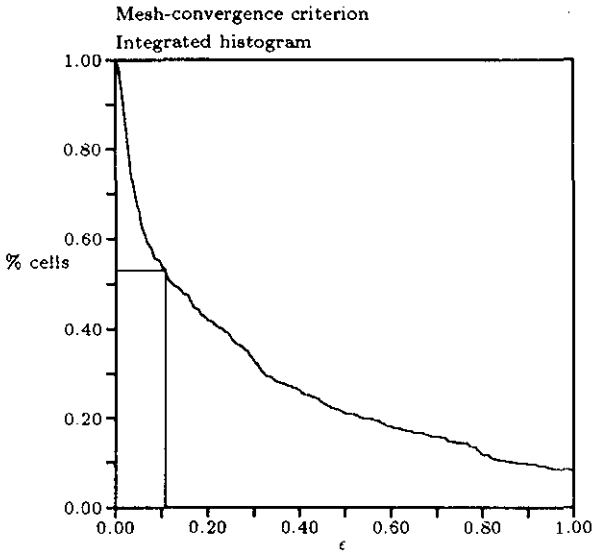


Figure 14: Histogram for Vortex Case (Final Grid)

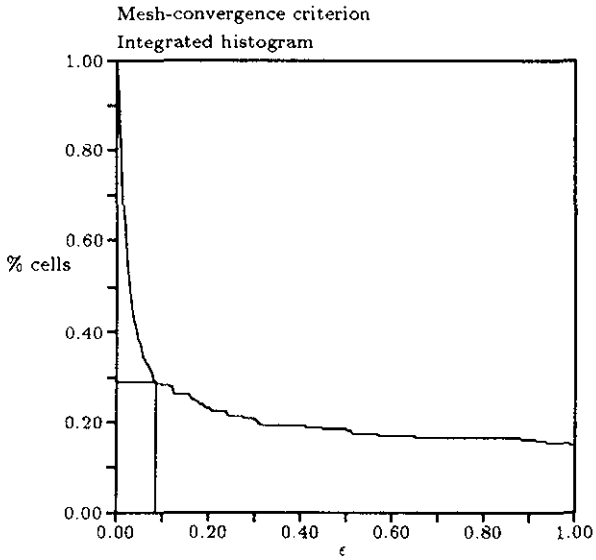


Figure 15: Histogram for Shock-Vortex Case (Initial Grid)

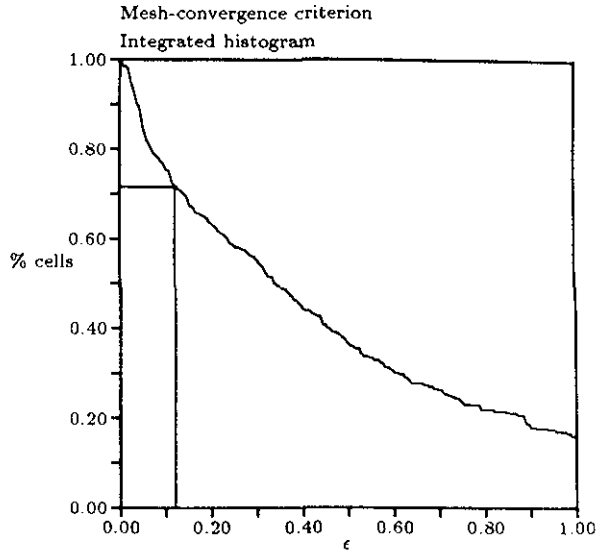


Figure 16: Histogram for Shock-Vortex Case (Final Grid)

grid after the refinement parameter and threshold have flagged cells to be divided.

The grid smoother is based on avoiding several cases that are (heuristically) "undesirable." These cases are:

1. a coarse cell between two fine cells (FCF);
2. a coarse cell between a fine cell and a boundary (FCB);
3. a fine cell between two coarse cells (CFC);
4. two interfaces without a buffer (e.g. F^2FC, as opposed to F^2FFC).

where C denotes a coarse cell, F denotes a fine cell and F^2 denotes a doubly fine cell (i.e. refined one level more than an 'F' cell).

If these steps were to be carried out one by one, the result from one step could create a new cell that is "undesirable" by the standards of an earlier step. For this reason, the grid smoother is made up of carrying out these four steps recursively, until no "undesirable" cells exist. At this point, the embedded region is expanded by two cells. This acts to make the interfaces less jagged, which further enhances convergence. The effects of these grid smoothing steps is shown in Figures 17–23.

Results

Results are presented for three cases:

1. $M_\infty = 1.1$, $\alpha = 10°$, $\beta = 0°$;
2. $M_\infty = 1.4$, $\alpha = 20°$, $\beta = 0°$;

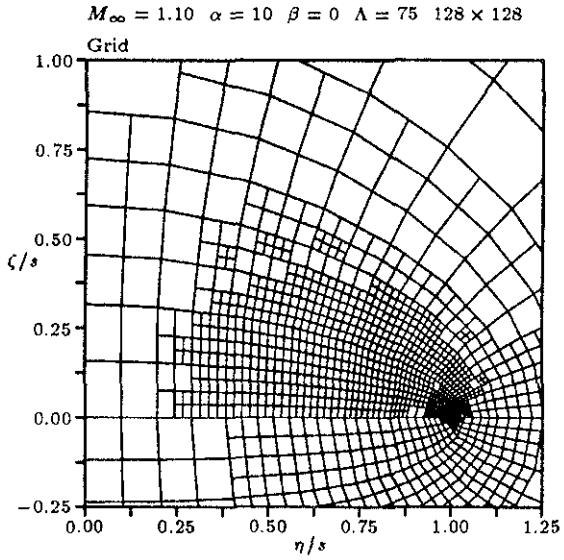


Figure 17: Grid without any smoothing

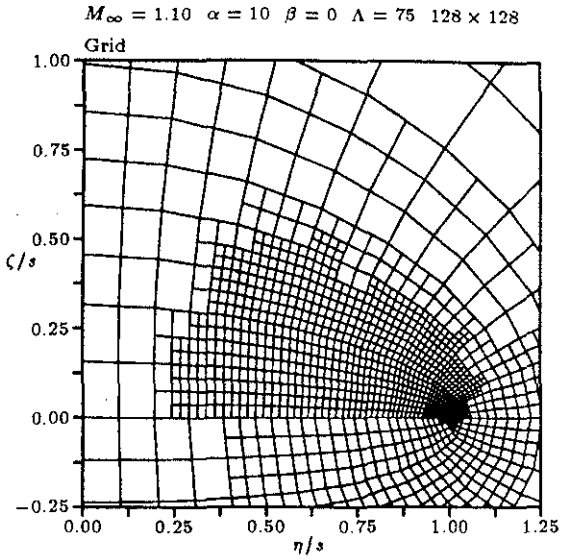


Figure 19: Grid with FCF and FCB smoothing

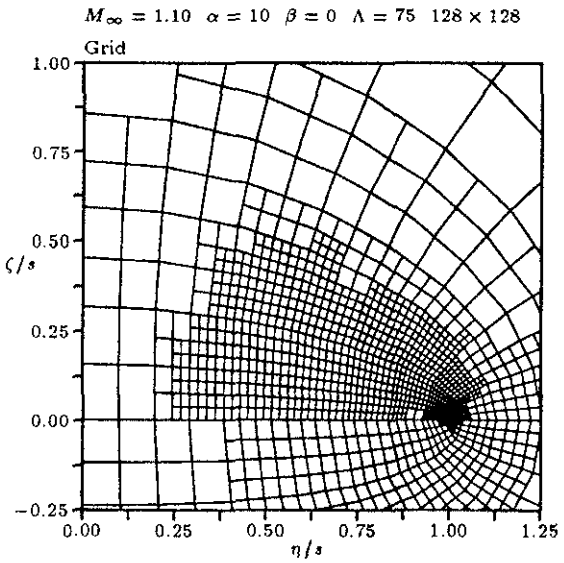


Figure 18: Grid with only FCF smoothing

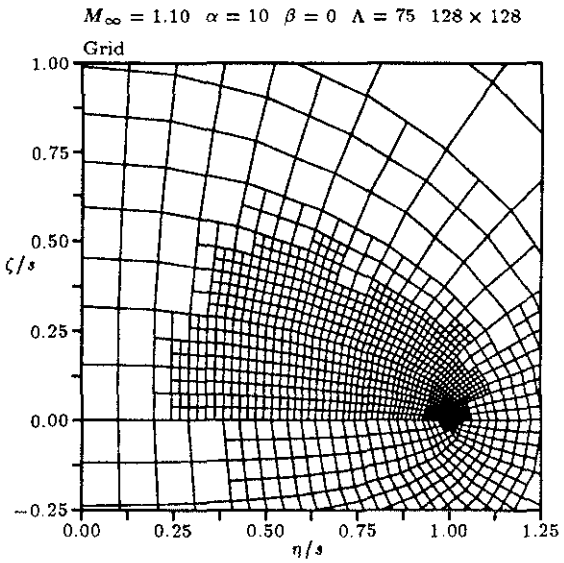


Figure 20: Grid with FCF, FCB and CFC smoothing

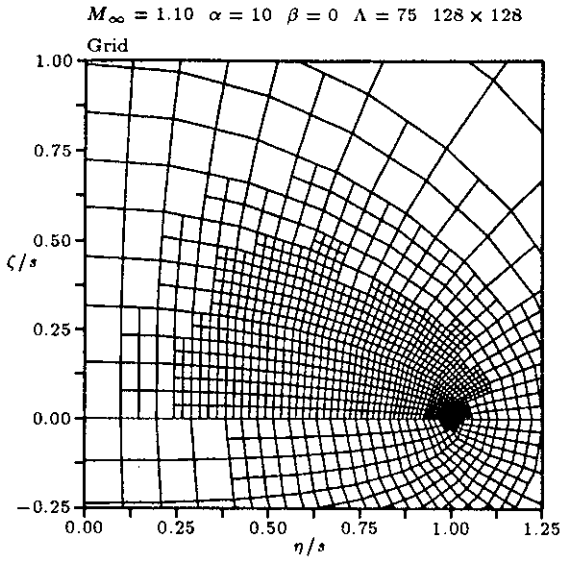


Figure 21: Grid with FCF, FCB, CFC and F^2FC smoothing

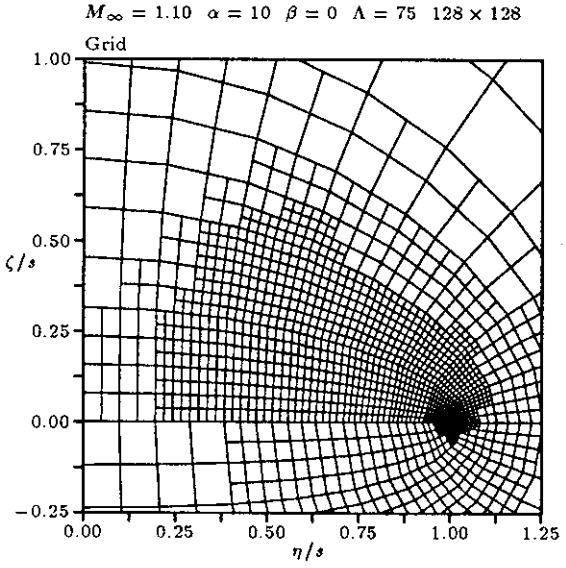


Figure 22: Fully smoothed grid, expanded one cell outward

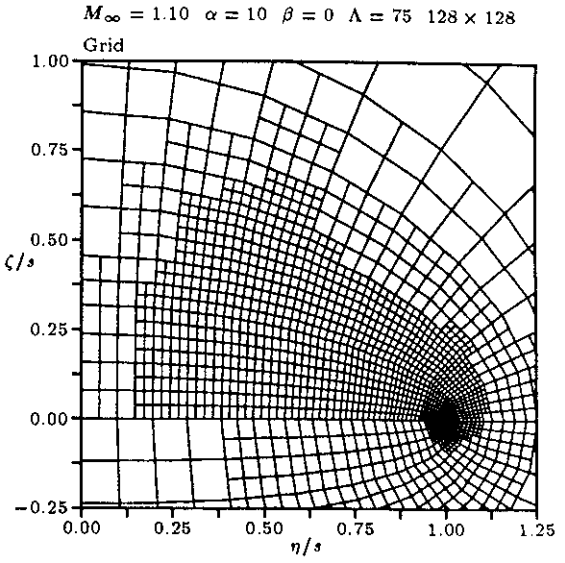


Figure 23: Fully smoothed grid, expanded two cells outward

3. $M_\infty = 1.4$, $\alpha = 20°$, $\beta = 10°$.

For each case, the final grid and total pressure loss contours are shown, along with the cross-flow Mach number contours on the initial and final grids.

The results for Case 1 are shown in Figures 24–27. Figure 24 shows the grid for this case; there are 3967 cells in this grid, a factor of fifteen less than the same resolution on a fixed grid. Cells have been refined near the leading edge of the wing, in the vicinity of the vortex, and at the cross-flow stagnation point at the leeward symmetry line. The improvement in resolution over the base grid is shown in the contours of cross-flow Mach number on the base grid (Figure 26) and the final grid (Figure 27). The transparency of the interfaces can be seen in Figures 27 and 25.

The results for Case 2 are shown in Figures 28–31. The grid (Figure 28) contains 2224 cells, a savings of a factor of seven over a fixed grid with the same resolution. The reasons for the lower savings in this case than in the last is that the vortex dominates a larger fraction of the computational space, and that only three levels of refinement were carried out in this case, as opposed to four in the last case. Further refinement of this case led to instability in the vortex; this is not a function of the adaptation, but of vortex-capturing in general. Even on a fixed grid, vortices can not be captured on too fine a grid without leading to instability [11]. The total pressure loss (Figure 29) shows the vortex and the shock. The improvement from the base grid to the final grid is shown in Figures 30 and 31. Again the interfaces are virtually transparent.

The results for the third case are shown in the remaining figures. This case did not use the refinement

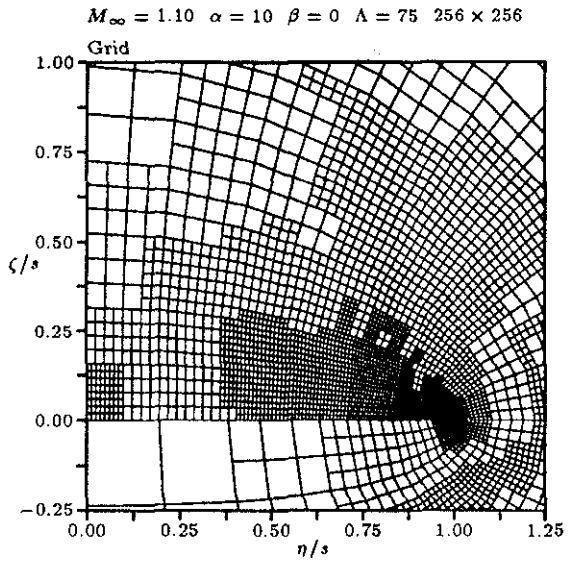


Figure 24: Case 1 — Final Grid

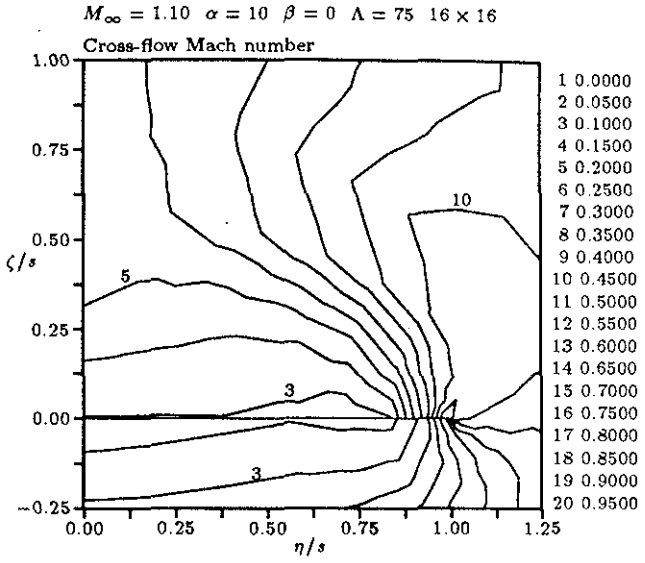


Figure 26: Case 1 — Cross-Flow Mach Number Contours on Initial Grid

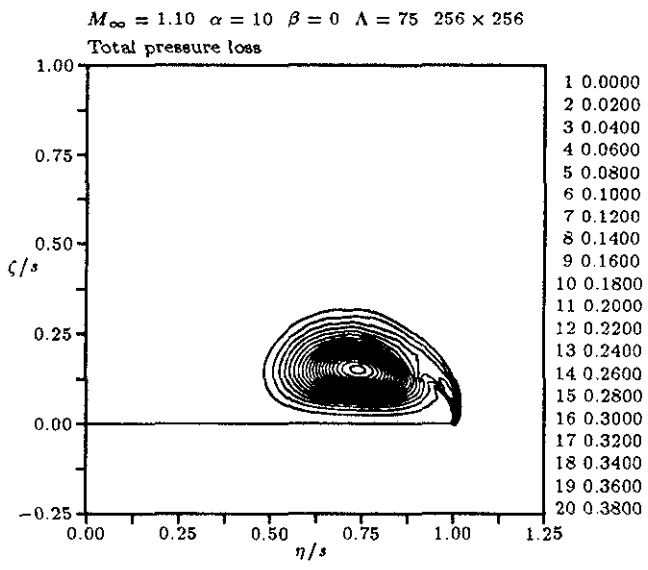


Figure 25: Case 1 — Total Pressure Loss Contours

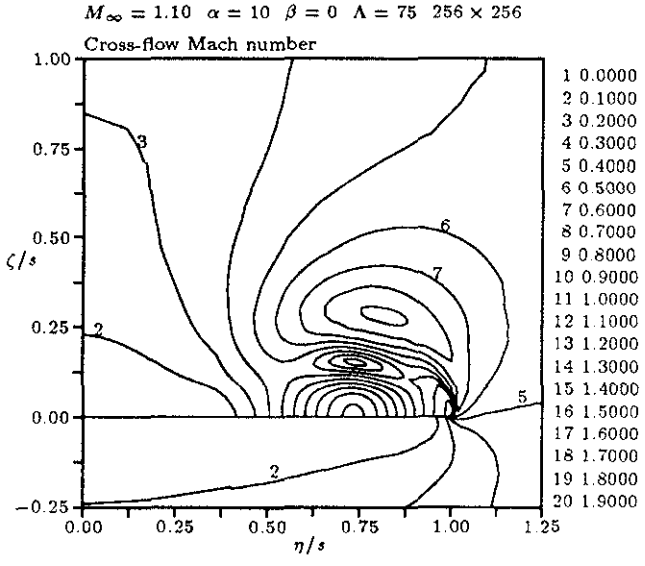


Figure 27: Case 1 — Cross-Flow Mach Number Contours on Final Grid

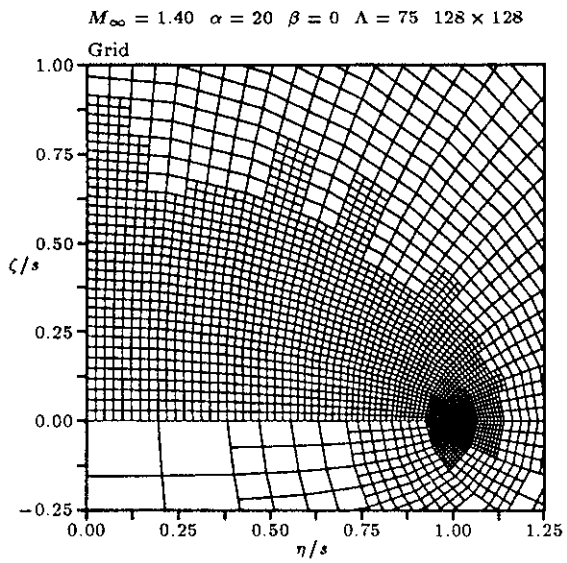


Figure 28: Case 2 — Final Grid

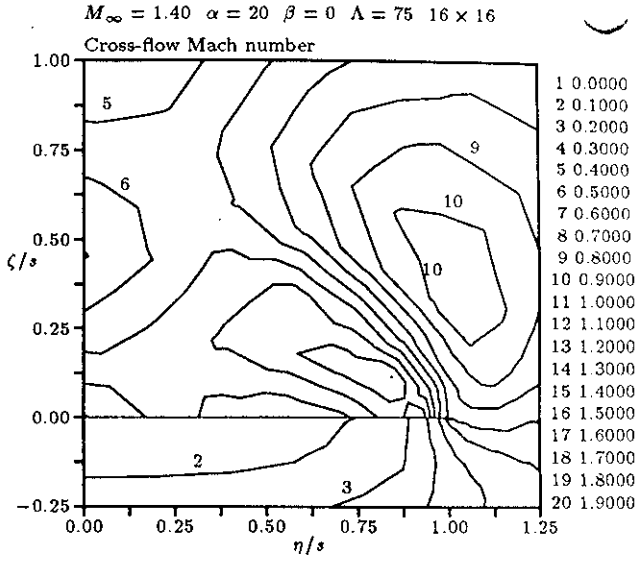


Figure 30: Case 2 — Cross-Flow Mach Number Contours on Initial Grid

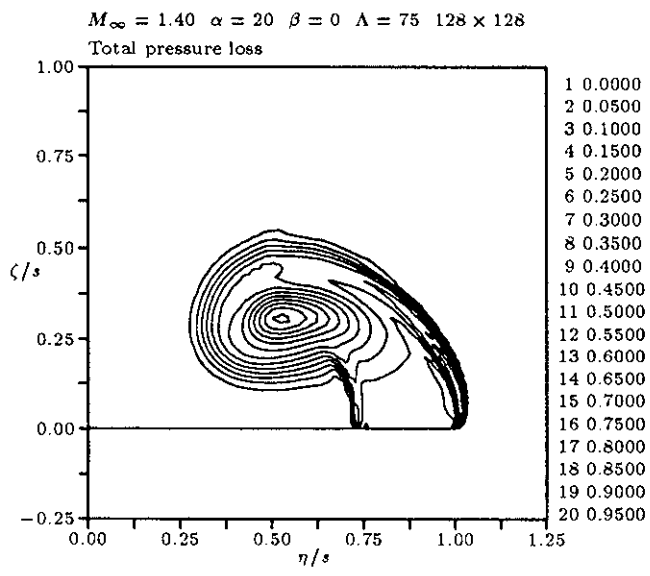


Figure 29: Case 2 — Total Pressure Loss Contours

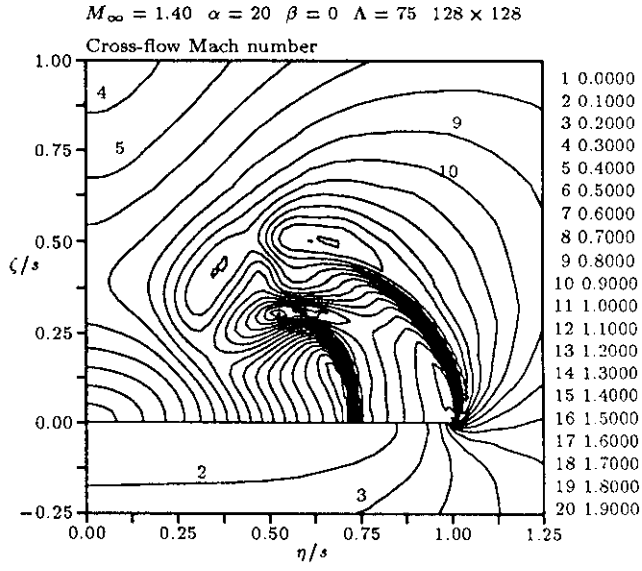


Figure 31: Case 2 — Cross-Flow Mach Number Contours on Final Grid

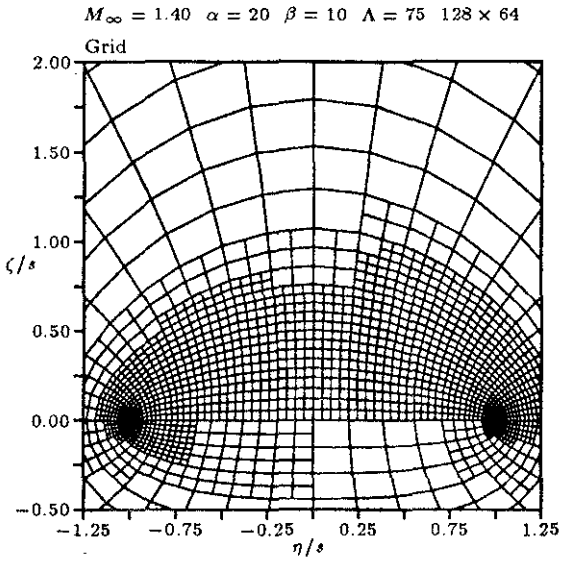


Figure 32: Case 3 — Final Grid

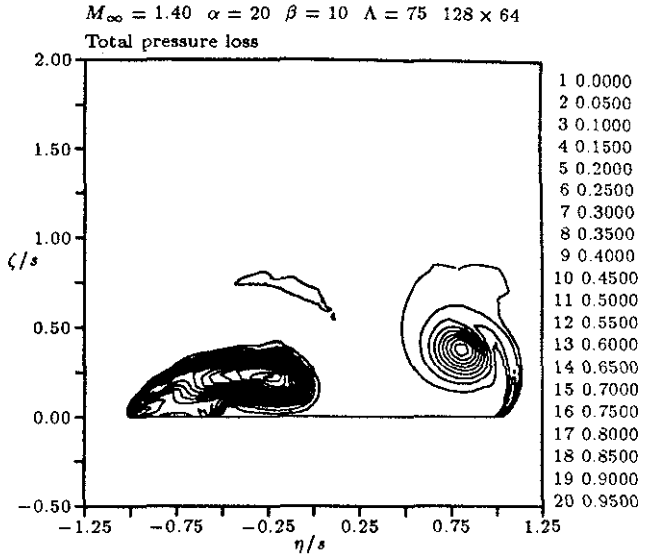


Figure 33: Case 3 — Total Pressure Loss Contours

parameter and threshold described above; total pressure loss was used as the refinement parameter, and a fixed threshold value was set. Because the wing is at yaw, an asymmetric flow pattern results, with the starboard vortex lifted off from the wing, and a shock-vortex pair on the on the port side. The grid for this case (Figure 32) has 1943 cells, a factor of eight less than a fixed grid would have. Again, the cross-flow Mach number contours on the initial and final grids (Figures 34 and 35) show the improvement due to refinement. In this case there is some total pressure loss at one of the grid interfaces; the fixed threshold led to too few cells being refined.

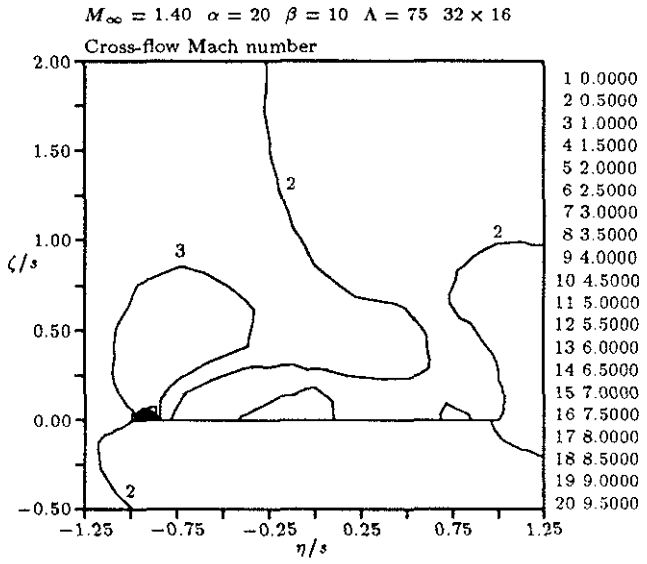


Figure 34: Case 3 — Cross-Flow Mach Number Contours on Initial Grid

Summary

An adaptive embedded mesh procedure for solving the conical Euler equations has been presented. It consists of a solution scheme that works on an unstructured mesh, a numerically based refinement parameter, and a method for determining a threshold for the refinement parameter. The refinement parameter is a measure of mesh-convergence. The advantage of this refinement parameter over one based on a physical parameter is its generality. The threshold determination is based on the curvature of the curve relating the number of cells flagged for refinement to the value of the threshold. It is also very general. The solution scheme was tested on cases with different numbers and types of features, and was shown to capture the features well.

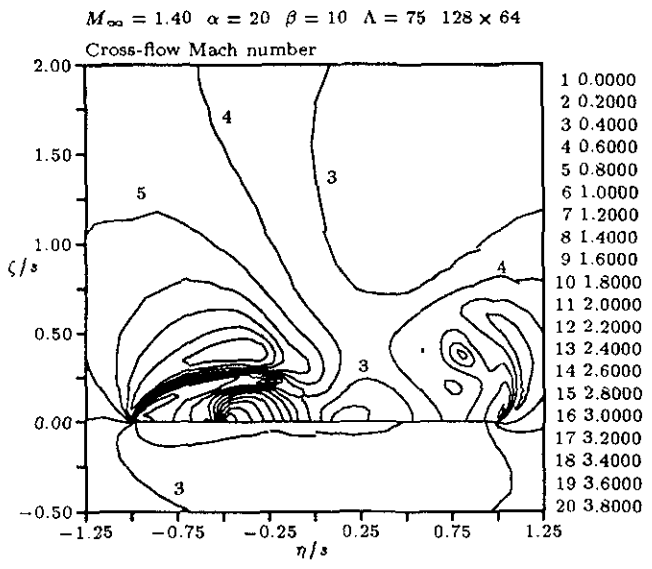


Figure 35: Case 3 — Cross-Flow Mach Number Contours on Final Grid

Acknowledgments

The authors would like to thank Dr. John F. Dannenhoffer III for discussions of his work in adaptive refinement. This work was partially funded by NASA Langley Research Center, Hampton VA, under grant NAG-1-869.

References

[1] H. A. Dwyer, M. D. Smooke, and R. J. Kee, "Adaptive gridding for finite difference solutions to heat and mass transfer problems," in *Numerical Grid Generation* (J. F. Thompson, ed.), Elsevier Science Publishing Company, Inc., 1982.

[2] K. Nakahashi and G. S. Deiwert, "A practical adaptive-grid method for complex fluid-flow problems," NASA TM 85989, 1984.

[3] J. Saltzman and J. Brackbill, "Applications and generalizations of variational mehtods for generating adaptive meshes," in *Numerical Grid Generation* (J. F. Thompson, ed.), Elsevier Science Publishing Company, Inc., 1982.

[4] J. F. Thompson, "A survey of dynamically-adaptive grids in the numerical solution of partial differential equations," AIAA Paper 84-1606, 1984.

[5] P. R. Eiseman, "Adaptive grid generation," *Computer Methods in Applied Mechanics and Engineering*, vol. 64, 1987.

[6] M. J. Berger, "Adaptive finite difference methods in fluid dynamics," Tech. Rep. DOE/ER/0377-277, Courant Mathematics and Computing Laboratory, 1987.

[7] K. G. Powell and E. M. Murman, "An embedded mesh procedure for leading-edge vortex flows," in *Proceedings of the Transonic Symposium*, 1988.

[8] J. G. Kallinderis and J. R. Baron, "Adaptation methods for a new Navier-Stokes algorithm," in *AIAA 8th Computational Fluid Dynamics Conference*, 1987.

[9] J. F. D. III, *Grid Adaptation for Complex Two-Dimensional Transonic Flows*. ScD thesis, Massachusetts Institute of Technology, 1987.

[10] R. Löhner, K. Morgan, and O. Zienkiewicz, "Adaptive grid refinement for the compressible Euler Equations," in *Accuracy Estimates and Adaptivity for Finite Elements*, Wiley, 1984.

[11] K. G. Powell, *Vortical Solutions of the Conical Euler Equations*. ScD thesis, Massachusetts Institute of Technology, 1987.

[12] J. L. Thomas and R. W. Newsome, "Navier-Stokes computations of lee-side flows over delta wings," AIAA Paper 86-1049, 1986.

[13] F. Marconi, "Flat plate delta wing separated flows with zero total pressure losses," AIAA Paper 87-0038, 1987.

[14] A. Jameson, "A vertex based multigrid algorithm for three-dimensional compressible flow calculations," 1986. Presented at the AME Symposium for Numerical Methods for Compressible Flow.

[15] M. J. Siclari and P. D. Guidice, "A hybrid finite volume approach to Euler solutions for supersonic flows," AIAA Paper 88-0225, 1988.

[16] K. G. Powell, E. M. Murman, E. S. Perez, and J. R. Baron, "Total pressure loss in vortical solutions of the conical Euler equations," *AIAA Journal*, vol. 25, 1987.