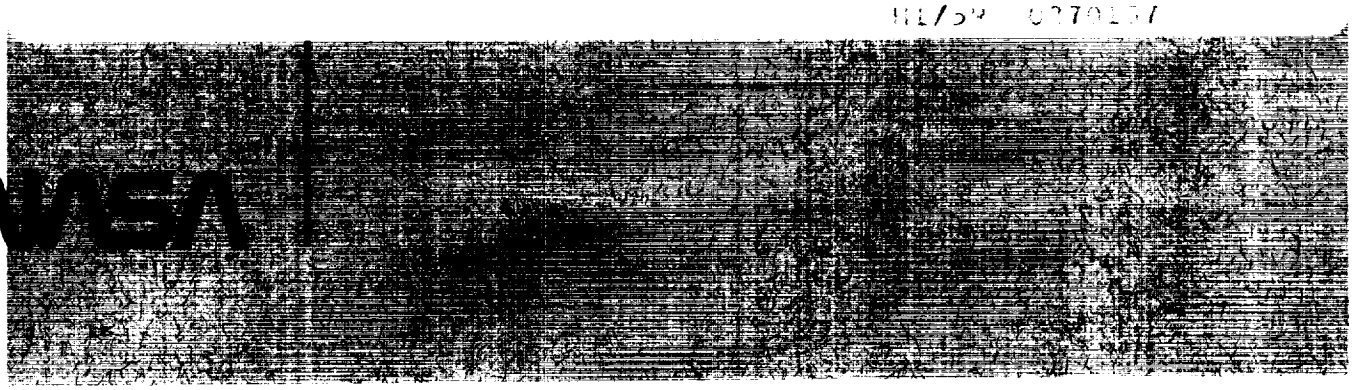
NASA Technical Paper 2998

June 1990

A Time-Accurate Adaptive Grid Method and the Numerical Simulation of a Shock-Vortex Interaction

Michael J. Bockelie
and Peter R. Eiseman



NASA Technical Paper 2998

1990

A Time-Accurate Adaptive Grid Method and the Numerical Simulation of a Shock-Vortex Interaction

Michael J. Bockelie
Langley Research Center
Hampton, Virginia

Peter R. Eiseman
Columbia University
New York, New York

National Aeronautics and Space Administration
Office of Management
Scientific and Technical Information Division

Introduction

Adaptive grid methods have rapidly emerged as important tools in numerical simulations because of their potential for improving the accuracy and efficiency of the solution. With adaptive (i.e., *dynamic*) grids, each grid point monitors the solution as it develops. The grid point distribution over the solution domain is correspondingly adjusted dynamically to concentrate grid points in regions of larger solution variation. In contrast, when using a traditional *static* grid, the investigator must use prior knowledge of the solution to design a grid which contains good grid resolution in all of the regions in which the solution has large variations.

Many solution adaptive strategies have been proposed in recent years (refs. 1 to 15). The adaptive grid solution method developed herein consists of three distinct parts: a grid movement scheme, a partial differential equation (PDE) solver, and a means of coupling the dynamic grid action to the PDE solver. A key element in this adaptive grid method is that the adaptive grid is generated separately from the determination of the solution of the physical problem, and therefore the grid movement scheme is only invoked by the temporal coupling routine as needed. With this approach, grid movement schemes can be developed that can be interfaced with a large number of PDE solvers.

The adaptive grid method used herein is a general-purpose technique that is suitable for multidimensional field problems. The method is simple to use and requires few user-specified parameters. The technique continues the adaptive philosophy used in references 1, 2, and 4. Although the focus is on methods for use on a structured grid with a well-ordered node connectivity pattern, many of the described techniques can be modified for use on an unstructured grid in which there can be arbitrarily connected grid points.

The adaptive method contains grid controls that correctly identify the solution behavior requiring additional grid resolution and that accurately redistribute the grid points into the targeted regions. The grid point movement is performed with localized intrinsic properties of the solution features being monitored for severe behavior. In the absence of behavior requiring grid point clustering, a uniform grid is obtained. The grid movement scheme includes a grid smoothing operation that ensures that the grid point locations and spacing change smoothly along the coordinate curves. The smoothing operation is interwoven with the grid point redistribution to help alleviate grid skewness.

The grid movement scheme contains several capabilities not available in other adaptive grid generation methods. Unlike many adaptive techniques which can only track one solution feature, the grid generation method used herein has the capability of accurately tracking multiple solution features. This capability is particularly useful in time-dependent problems in which the solution features monitored for adaptive purposes merge and then separate. Also presented is a method to eliminate "noisy" values in the adaptive data; the occurrence of noisy data can be quite severe when the adaptive data are formed from derivatives of the conserved variables in the solution vector. Using the described method allows the grid to be adapted to solution features that could not otherwise be used for grid attraction. Altogether the adaptive grid method used herein is a very robust scheme which can generate grids with good structure for a wide range of problems.

The PDE solver used in this study is an "off-the-shelf" Euler equation solver which employs a finite volume, shock capturing method to discretize the Euler equations on a *static* grid. The static grid feature of the Euler equation solver is emphasized because the movement of the grid points during the adaptive action introduces a time dependence (i.e., the grid becomes *dynamic*). From a strictly mathematical viewpoint, when computations are performed on a dynamic grid, additional grid velocity terms must be included in the governing equations of the physical problem to account for the moving frame of reference.

As described in reference 4, the manner in which the temporal coupling of the adaptive grid to the PDE solver is performed greatly affects the accuracy of the solution. In steady-state and time-asymptotic problems the grid velocity can typically be ignored, or computed with a simple backward difference in time, and adequate solutions obtained because the solution and the grid settle into a final configuration; such solutions are not time accurate because the solution is computed on a grid which lags the solution in time. The time accuracy of the grid and solution can be maintained by directly computing the grid velocity or determining the new grid as part of the solution at the forward time step; however, these methods can lead to folded, or singular, grids if the grid velocity is not computed accurately. In this report, a temporal coupling method is presented which maintains the time accuracy of the solution but which avoids the need to estimate the grid velocity terms and correspondingly modify the PDE solver to include a grid velocity capability.

A "prediction-correction" method is used for the temporal coupling. The prediction-correction

method treats the time integration as a series of initial-value problems over short time intervals in which the solution is first advanced to create a new grid and then recomputed on the new grid. In effect, the new grid provides a carpet of resolution in the regions over which the solution will evolve, and thereby ensures the evolving solution is always computed on a grid that has locally fine grid resolution in the regions where the solution exhibits severe behavior. The prediction-correction method used herein bears a philosophical similarity to a scheme developed by Blom, Sanz-Serna, and Verwer (ref. 7) but is more flexible in the type of equation solvers that can be used in the adaptive solution. Unlike that of reference 7, the approach described herein is not limited to using only equation solvers with a grid velocity capability.

Using the prediction-correction method for the temporal coupling provides several advantages. First, the method maintains the temporal accuracy of the solution; the grid does not lag the solution in time because the new grid is based on solution data forward in time. Second, the prediction-correction method does not require modification of the static grid Euler equation solver to include grid velocity terms because the predicted and corrected solutions are computed on static (although different) grids; however, if the PDE solver had a grid velocity capability, this scheme would provide a means to accurately compute the grid velocity. Last, the prediction-correction method provides a large degree of simplicity and flexibility, and thus can reduce the human effort required to develop time-accurate adaptive grid PDE solvers for a wide range of time-dependent field problems.

After the basic elements of the adaptive grid method are introduced, the technique for tracking multiple solution features, the method for eliminating noisy values in the adaptive data, the Euler equation solver, and the prediction-correction routine used for the temporal coupling are discussed. The capabilities of the adaptive grid method are demonstrated through the determination of the unsteady two-dimensional inviscid flow field created by a shock wave moving toward, and eventually over, a solid-core vortex. The shock-vortex problem provides a good test of the adaptive solution method because the solution and the grid do not settle into a steady state. To accurately capture all the complexities that develop in the shock front and the vortex as they approach each other, merge, and continue requires a grid with locally high resolution in which the location of the high-resolution region changes with time.

Adaptive Grid

In this section grid movement concepts and algorithms are discussed. Because many of the techniques used in multidimensional problems have their roots in one-dimensional problems, the formulation for a one-dimensional problem is presented first, after which the necessary extensions for two-dimensional problems are described. It is assumed that the reader understands the distinction between representing a grid in physical space and in computational space. Briefly, for a one-dimensional problem the grid in physical space is a coordinate curve that consists of n arbitrarily spaced grid points with locations described by the grid point position vectors $\mathbf{P}_i$ or the arc length along the curve s_i. Through application of a transformation to the points on the curve in physical space, a new representation of the curve can be constructed in a computational space in which the curve consists of n uniformly spaced points ξ_i ranging in value from 0 to $n-1$. For further details, see reference 12.

Monitor Surface

The adaptive grid method used herein utilizes a *monitor surface* to identify regions where finer grid resolution is needed. The monitor surface, or actually the monitor curve in the one-dimensional context, is a smooth piecewise linear surface positioned over the physical domain. The monitor surface $\psi(x_i)$ is typically defined by a simple scalar function (e.g., a linear combination) of one or more of the variables which drive the physics of the problem. Hence, regions in which the solution has a sharp gradient or a sharp transition between vastly different gradients are represented in the monitor surface as, respectively, sharp gradients or sharp bends. An example of a monitor surface for a one-dimensional problem is illustrated in figure 1.

The adaptive grid method operates on a surface grid lying on the monitor surface. The surface grid is obtained by "lifting" the grid points in the physical domain up to the monitor surface. The nodes in the surface grid are subsequently repositioned on the monitor surface to better resolve its geometric features. The repositioning is based on the arc length of the coordinate curves, which leads to gradient clustering, and the bends of the monitor surface, which are treated directly with curvature clustering. At the completion of the node redistribution process, the surface grid is projected back down to the physical domain to yield the new grid on which a new solution can be computed. The assumption implicit to the entire process is that using a grid which more accurately represents the physics of the governing

problem will reduce the overall error in the numerical solution computed on the grid.

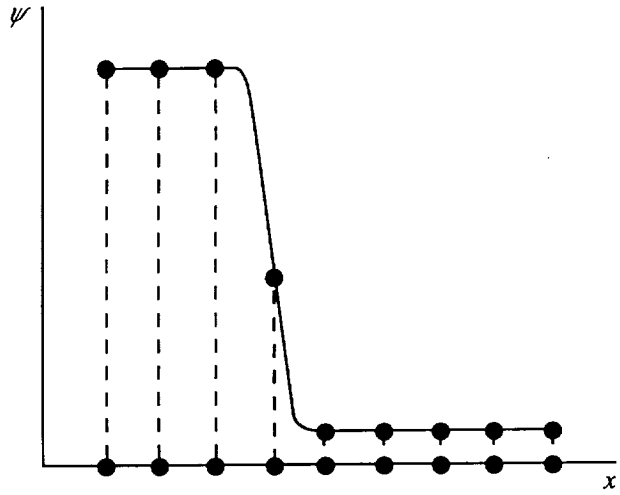


Figure 1. One-dimensional monitor surface with grid points uniformly distributed in physical domain.

Equidistribution Statement

The repositioning of the grid points along the curve on the monitor surface is achieved by equally distributing a positive weight function over the curve. That is, for each interval of the curve, the grid points are repositioned to satisfy the equidistribution statement

$$w_{i+1/2}\ \Delta s_i = A\ \Delta\xi_i = \text{Constant} \qquad (1)$$

where $w_{i+1/2} = \frac{1}{2}(w_i + w_{i+1})$ is the average of the weight function over the interval, $w_i = w(s_i)$ is the weight function evaluated at the ith node in physical space, $\Delta s_i = s_{i+1} - s_i$ is the interval arc length along the given curve on the monitor surface in physical space where the grid points are unevenly distributed, A is a constant to be determined, and $\Delta\xi_i = 1$ is the uniform spacing in computational space of the grid points that satisfy the equidistribution condition. As can be seen from equation (1), the node redistribution process is driven by the balancing of the weight function against the interval arc length. Notice that as the weight function increases or decreases, the associated interval length on the monitor surface must correspondingly shrink or expand.

Weight Function

For a one-dimensional problem, the form of the weight function is

$$w_i = 1 + C\,|\kappa_i| \qquad (2)$$

where C is a constant and κ_i is the curvature of the monitor surface at node i. If $C = 0$ in equation (2), the weight function reduces to $w_i = 1$, the equidistribution statement becomes $\Delta s_i = \text{Constant}$, and the grid points are redistributed on the monitor surface with a uniform spacing. Figure 2 shows that when the uniform surface grid is projected down onto the physical domain, it clusters grid points into the sharp-gradient regions and provides a uniform grid spacing in the regions away from the sharp gradients. Analytically, the constant arc-length increments imply that $(1 + \psi_x^2)^{1/2}\,dx$ is constant. For $\psi_x^2 \gg 1$, the term $(1+\psi_x^2)^{1/2}\,dx$ can be approximated by $|\psi_x|$ and hence $|\psi_x|\ dx$ is constant, the result being a spacing of points in the physical domain which is inversely proportional to the magnitude of the gradient of the monitor surface. For $\psi_x^2 \ll 1$, the uniform arc-length spacing implies dx is constant, the result being a uniform distribution of points in the physical domain.

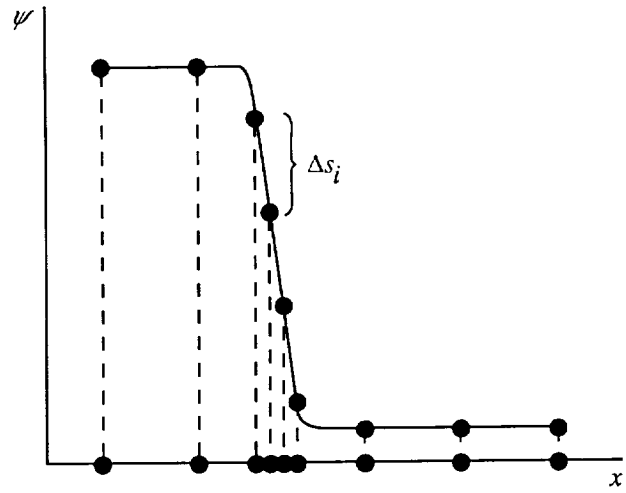


Figure 2. One-dimensional monitor surface with grid points uniformly distributed on monitor surface.

Although a uniform distribution of points on the monitor surface may provide adequate resolution in many problems, there are often certain features which require additional resolution. In particular, a uniform distribution does not provide adequate resolution in the regions containing a sharp "knee" or bend in the monitor surface. In figure 2, the lack of resolution at the foot and crest of the steep face in the monitor surface is evident. A bend in the monitor surface represents a transition between two vastly different gradients in the solution and hence should also be refined.

The presence of the sharp bends in the monitor surface can be identified from the curvature of the monitor surface κ. The curvature is simply the rate of change of the unit vector tangent to the curve (i.e., the monitor surface) with respect to the arc length: $\kappa\hat{\mathbf{n}} = d\hat{\mathbf{t}}/ds = \mathbf{P}''$, where $\hat{\mathbf{t}} = \mathbf{P}'/\|\mathbf{P}'\|$ is

the unit vector tangent to the curve, $\mathbf{P}'' = d\mathbf{P}'/ds$, $\mathbf{P}' = d\mathbf{P}/ds$, and $\mathbf{P}$ is the surface grid point position vector. Figure 2 shows that the tangent vector is approximately constant (i.e., $\kappa = 0$) everywhere on the monitor surface except in the two knee regions; in these regions $\kappa \neq 0$. Thus, the weight function is greater in magnitude than elsewhere, and this imbalance forces ds to be smaller by equidistribution and thereby results in grid points being attracted to the bend regions of the monitor surface. (See fig. 3.)

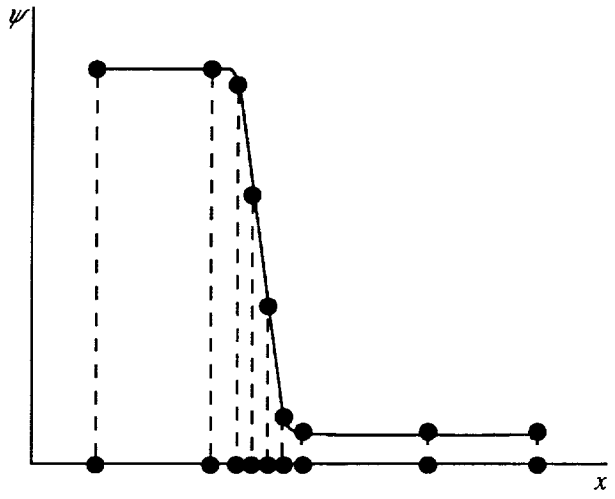


Figure 3. One-dimensional monitor surface with grid points distributed with moderate amount of curvature attraction.

The leading coefficient C in equation (2) controls the intensity of the curvature clustering. Hence, the value of C must be chosen carefully. With a technique originally developed by Dwyer (ref. 1), the coefficient C is computed based on the prescribed percentage of the total number of grid points f that are to be attributed to curvature clustering:

$$f = \frac{\int_{s_1}^{s_n} C|\kappa| \, dr}{\int_{s_1}^{s_n} w(r) \, dr} \tag{3}$$

The function for computing $C(f)$ used in this study includes a "switch function," as described by Eiseman (ref. 2), which dynamically turns C off and on based on the ratio of the arc length of the coordinate curve to its shortest possible length; if the ratio is near one, then curvature clustering is not needed (i.e., $C = 0$).

Grid Point Redistribution Algorithm

The equidistribution process starts with a given grid defined by the position vectors $\mathbf{P}_i$ and ends with a new grid defined by the position vectors $\mathbf{Q}_j$ for the grid points which satisfy the equidistribution condition. Assuming that the weight functions w_i have been computed and then using a series of integrations of the weight function over portions of the curve, we can derive a piecewise linear approximation which allows the new position vectors to be computed directly from the old position vectors (refs. 1, 2, and 4). Thus, the position vectors for the new node locations can be computed in a component-by-component fashion with the interpolation formula

$$\mathbf{Q}_j = \mathbf{P}_m + \alpha_j(\mathbf{P}_{m+1} - \mathbf{P}_m) \tag{4}$$

where

$$\alpha_j = \frac{\xi_j - \xi(s_m)}{\xi(s_{m+1}) - \xi(s_m)}$$

and the index $m = m(j)$ is the interval that brackets ξ_j, or $\xi(s_m) \leq \xi_j < \xi(s_{m+1})$. In the above, ξ_j are the uniformly spaced locations in computational space of the new grid points that satisfy equidistribution. The term $\xi(s_m)$ represents the arbitrarily spaced locations in computational space of the old grid points. The value of $\xi(s_i)$ is obtained by first integrating (with a trapezoidal rule) the equidistribution statement over the curve to eliminate A in equation (1) and then integrating only to point i and inserting boundary values for ξ to yield

$$\xi(s_i) = (n-1)\frac{F(s_i)}{F(s_n)} \tag{5}$$

where

$$F(s_i) = \int_{s_1}^{s_i} w(r) \, dr = \sum_{k=1}^{i-1} w_{k+1/2} \, \Delta s_k$$

To obtain the positions of the new grid in the physical domain the Cartesian coordinates are extracted from the new position vectors in equation (4).

With the weight function and node redistribution algorithm described above, an adaptive grid for a one-dimensional problem can be generated that automatically resolves the sharp gradient and transition regions of the solution and that provides a uniform grid cell spacing in the regions away from the severe solution behavior.

Two-Dimensional Problems

The previously described grid movement scheme for one-dimensional problems is extended to two-dimensional problems by operating on the surface grid on a curve-by-curve basis (ref. 4). The order in which the curves are adapted in the curve-by-curve scheme is based on directional sweeps; that is, with

the directions of the curvilinear coordinate curves of the surface grid denoted as ξ and η, the nodes are redistributed along each curve in the ξ direction, followed by the same process for the curves in the η direction. The mechanics of redistributing the points along the individual coordinate curves is the same as that for one-dimensional problems. It should be noted that, in general, the resulting grid is not invariant to the order in which the curves are adapted (i.e., first ξ and then η, or vice versa). Hence, for problems in which the grid must contain a very high degree of symmetry, to generate the adaptive grid with a curve-by-curve scheme might require special techniques, such as the use of symmetric operators for all computations, or the modification of the ordering of the curves in the directional sweeps.

The form of the weight function is the same as equation (2), except that the curvature is replaced by κ_{n_i}, the normal curvature of the monitor surface at node i, to yield

$$w_i = 1 + C|\kappa_{n_i}| \tag{6}$$

As discussed in detail by Eiseman (ref. 2), in two-dimensional problems to correctly identify the segments of a monitor surface that need resolution requires the use of weight functions based on the basic curvature properties of the coordinate curves. With the theory of Cartan Frames (ref. 16), it is possible to distinguish true changes in the monitor surface along the coordinate curves from wiggles and bends in the coordinate curves on the monitor surface.

The Cartan Frames provide a means of splitting the curvature of a curve lying on a two-dimensional surface into two components, the normal and geodesic curvatures. The normal curvature κ_n identifies bends in the monitor surface, which are where node clustering should occur. The normal curvature is computed by tracking the changes in the orientation of a plane tangent to the surface as the point of tangency is moved along a curve in the surface; that is, as an observer moves along the curve, changes in the tangent plane indicate changes in the monitor surface and thus the regions requiring additional resolution. The geodesic curvature κ_g signifies lateral bending of the curve within the monitor surface. The geodesic curvature is computed by tracking the changes in orientation of the plane that is orthogonal to the surface tangent plane and tangent to the curve. Typically, clustering based on the geodesic curvature is warranted only if the curve lies on or near a curved boundary that must maintain its physical shape (ref. 2); applications requiring the use of geodesic curvature have not been investigated in this study. In the regions where the normal curvature is approximately zero, the weight function in equation (6) yields an equal arc length spacing of points on the monitor surface and hence continues to provide a uniform spacing in the physical domain in the regions away from the severe solution behavior.

The precise mathematical expression for the curvature is (ref. 2)

$$\kappa\hat{\mathbf{n}} = \mathbf{P}'' = \kappa_g\hat{\mathbf{T}} + \kappa_n\hat{\mathbf{N}} \tag{7}$$

where κ is the curvature of the curve on the monitor surface and $\hat{\mathbf{n}}$, $\hat{\mathbf{N}}$, and $\hat{\mathbf{T}}$ are, respectively, the unit vector normal to the curve, the surface which defines the surface tangent plane, and the curve lying in the surface tangent plane. (See fig. 4.) The unit vectors $\hat{\mathbf{N}}$ and $\hat{\mathbf{T}}$ are defined by

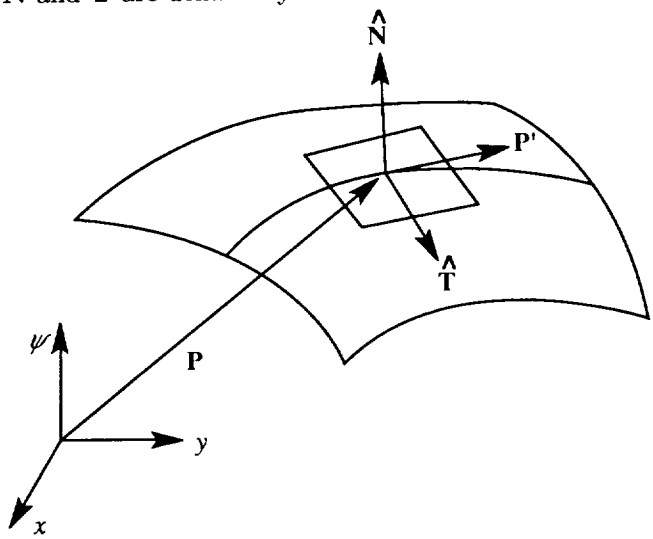


Figure 4. Orientation of vectors used to compute curvature properties of curve in two-dimensional monitor surface.

$$\hat{\mathbf{T}} = \mathbf{P}' \times \hat{\mathbf{N}} \qquad \text{and} \qquad \hat{\mathbf{N}} = \frac{\mathbf{B}}{\|\mathbf{B}\|} \tag{8}$$

where

$$\mathbf{B} = \frac{\partial \mathbf{P}}{\partial u} \times \frac{\partial \mathbf{P}}{\partial v}$$

and u and v are coordinate directions corresponding to the curvilinear coordinate curves on the monitor surface. The normal and geodesic curvatures can be computed from

$$\kappa_n = \hat{\mathbf{N}} \cdot \mathbf{P}'' \qquad \text{and} \qquad \kappa_g = \hat{\mathbf{T}} \cdot \mathbf{P}'' \tag{9}$$

because $\hat{\mathbf{N}}$ and $\hat{\mathbf{T}}$ are orthogonal by definition. In most applications, a one-dimensional smoothing operation is applied to κ_{n_i} before the weight function is assembled to eliminate abrupt jumps that can occur in the computed values. The leading coefficient of the curvature term in the weight function is computed in a manner analogous to that described for

a one-dimensional problem. Further details on the curvature computations are available in reference 8.

Active-Passive Phases

For problems in which the monitor surface contains very sharp gradients or complex geometries, the basic curve-by-curve scheme can result in a grid with unsmooth changes in grid positions and skewed coordinate curves. The irregularities in the grid could create large truncation errors in the numerical approximations of derivatives on the grid, and thus yield inaccurate solutions if a numerical simulation is performed on the grid.

In this study, smooth, nonskewed grids are obtained by splitting the node movement within each directional sweep into active and passive phases (ref. 4). The active phase redistributes the nodes on the monitor surface along each curve in the current direction through use of the previously described curve-by-curve adaptive scheme. In the passive phase, the grid is relaxed by application of a "low-pass" filter to the grid to remove any wiggles and abrupt changes in spacing created by the active phase.

The low-pass filter is a grid smoothing operation designed to remove high-frequency variations in the grid positions while approximately retaining the lower frequency variations in node spacing which correspond to the positions determined by the equidistribution statement (ref. 4). The low-pass filter is derived by applying a Laplacian operator to the current position vector and then writing a Gauss-Siedel relaxation formula to solve for the new grid point locations. For a two-dimensional grid, the relaxation formula is

$$\mathbf{Q}_{i,j} = \frac{1}{2}\frac{\mathbf{Q}_{i-1,j} + \mathbf{Q}_{i,j-1} + \mathbf{P}_{i+1,j} + \mathbf{P}_{i,j+1}}{4} + \frac{1}{2}\mathbf{P}_{i,j} \quad (10)$$

where $\mathbf{Q}$ and $\mathbf{P}$ are respectively the new and old position vector values at the grid point. Alternatively, equation (10) could have been obtained by defining the new position vector as the old value averaged with the average of its four neighbors. To maintain the shape of the physical domain, along the boundaries the relaxation formula is based on the finite-difference template for the interior points (eq. (10)), but a ghost point is used for the neighboring node which lies outside of the physical domain. The ghost point location is computed by extending the coordinate curve transverse to the boundary beyond the boundary, along a trajectory with slope equal to that of the transverse curve at the boundary, for a distance that is equal to the distance between the boundary and the first interior grid point on the transverse curve (ref. 8).

Vector Monitor Surface

One must be especially careful in defining the monitor surface for applications in which multiple solution features are to be used for grid adaptation. The simplest approach in such cases is to form the monitor surface as a scalar function defined as the linear combination of the desired features. However, with a scalar monitor surface, if the solution features merge the solution gradients can cancel each other and thereby destroy the grid resolution in the region of merger. The gradient cancellation can be seen in the arc-length computations. For a problem in which two solution features are being tracked, the arc length is

$$ds = (d\mathbf{x}\cdot d\mathbf{x} + d\psi^2)^{1/2} = [\,d\mathbf{x}\cdot d\mathbf{x} + (d\psi_1 + d\psi_2)^2\,]^{1/2} \quad (11)$$

where $\mathbf{x} = (x, y, z)$. Thus, if the solution features should merge, $d\psi_1$ and $d\psi_2$ can cancel each other. Illustrated in figure 5 are a scalar monitor surface and the resulting adaptive grid for the case described in equation (11). Here, the physical domain is two dimensional and the two solution features being tracked are represented by a hyperbolic tangent plane with a curved front

$$\psi_1 = \tanh\left\{10\left[(x-1) - \frac{y}{4}\left(1 - \frac{y}{4}\right)^3 \operatorname{sgn} y\right]\right\} \quad (12)$$

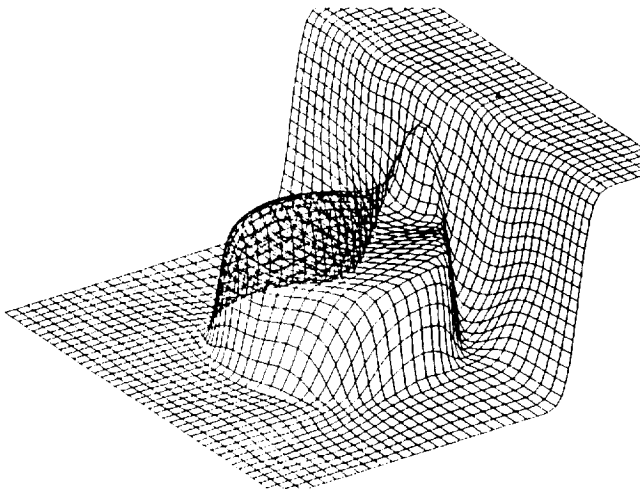
and a pillbox function

$$\psi_2 = \frac{1}{2}(1 + \tanh\{10[1 - (x^2 + y^2)]\}) \quad (13)$$

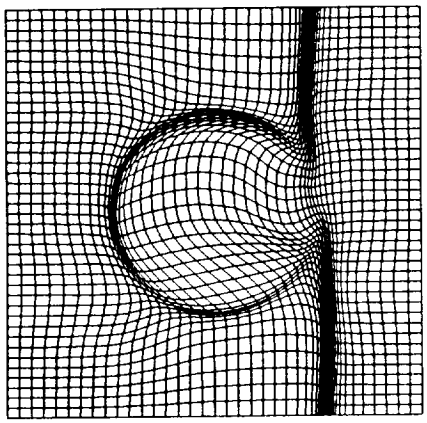
The lack of grid resolution can be seen along the horizontal center of the grid, where the two features merge.

The poor grid resolution that can occur with a scalar monitor surface can be overcome with a monitor surface defined as a vector function in which each solution feature to be tracked is a component of the vector. The vector monitor surface is defined as an N-dimensional vector $\boldsymbol{\Psi} = (\psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \ldots, \psi_N(\mathbf{x}))$, where ψ_k is one of the N features of the solution that is to be tracked or resolved. The terms ψ_k are scalar functions and are typically formed from a single variable or property of the solution. The vector monitor surface is positioned over the physical domain and the location of each point on the monitor surface is described by the position vector $\mathbf{P} = (\mathbf{x}, \psi_1(\mathbf{x}), \ldots, \psi_N(\mathbf{x}))$. Thus, to solve a problem for which N solution features are being tracked in an M-dimensional physical domain, the scalar monitor surface is an M-dimensional surface in an $(M+1)$-dimensional space, whereas the

vector monitor surface is an M-dimensional surface embedded in an $(M + N)$-dimensional space.



(a) Projection onto monitor surface.



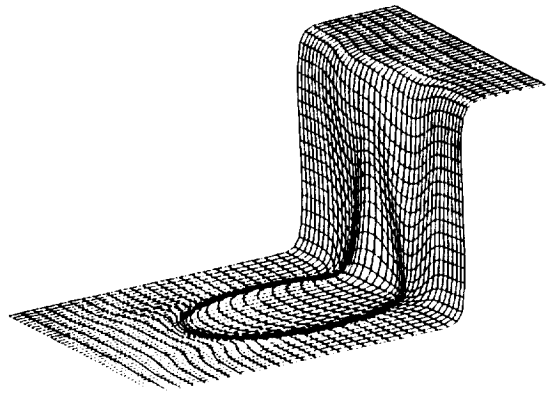
(b) Projection on physical domain.

Figure 5. Scalar monitor surface with surface grid projected onto monitor surface (without hidden lines removed) and physical domain.

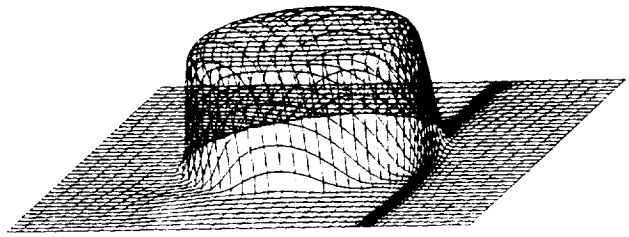
The advantage of using the vector monitor surface is that when the solution features merge, the gradients cannot cancel, and thus good grid resolution is maintained in the region of merger. Again, this can be observed from the arc length computations. For the example in equation (11), the arc length is computed as

$$ds = (d\mathbf{x} \cdot d\mathbf{x} + d\psi_1^2 + d\psi_2^2)^{1/2} \qquad (14)$$

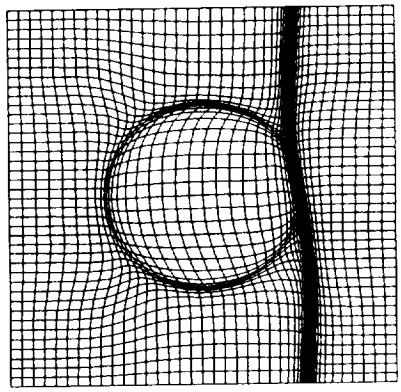
from which it can be seen that the contributions to the arc length of the individual solution features being tracked (i.e., $d\psi_i^2$) cannot cancel. Illustrated in figure 6 are the vector monitor surface projected onto the individual components and the resulting adaptive grid for the example problem in figure 5. Through comparison of the two figures, the improved grid resolution in the region of merger is evident.



(a) Projection onto component of monitor surface formed by hyperbolic tangent plane.



(b) Projection onto component monitor surface formed by pillbox function.



(c) Projection onto physical domain.

Figure 6. Vector monitor surface with surface grid projected onto component of monitor surface formed by hyperbolic tangent plane, component of monitor surface formed by pillbox function, and physical domain.

The process for adapting a grid to a vector monitor surface is essentially the same as that for a scalar monitor surface. The only differences are in the computations for the curvature clustering properties and having to use a more general form of the weight function. As described in reference 5, node attraction

based on curvature properties can be difficult to implement with a vector monitor surface because variations in the surface normal directions used in the curvature computations are not uniquely defined. We overcome this difficulty by projecting the surface grid onto a component of the vector monitor surface ψ_k to compute geometrical properties (e.g., curvatures) for the component (ref. 8). Thus, at each node there is a normal and a geodesic curvature value computed for each of the ψ_k components of the vector monitor surface. The values of the normal and the geodesic curvature for the kth component of the monitor surface, $\kappa_n^{(k)}$ and $\kappa_g^{(k)}$, respectively, are computed from equations (7) to (9), where the position vector to be used is $\widetilde{\mathbf{P}} = (\mathbf{x}, \psi_k(\mathbf{x}))$ and the associated arc length $\tilde{s}$ is computed along the curve projected onto component ψ_k, $\tilde{s} = \tilde{s}(\widetilde{\mathbf{P}})$. When computed in this fashion, the resulting geometrical properties are uniquely defined.

The weight function used in the equidistribution statement is

$$w_i = 1 + \sum_{k=1}^{N} C_k(f_k)\,|\kappa_{n_i}^{(k)}| \qquad (15)$$

where $C_k(f_k)$ is the curvature coefficient for ψ_k and $\kappa_{n_i}^{(k)}$ is the normal curvature at node i for the monitor surface projected onto ψ_k. As was the case for the scalar monitor surface, the κ_n values are passed through a one-dimensional smoothing operation before the weight function is computed. The dynamic curvature coefficient $C_k(f_k)$ for each component of Ψ is computed in a manner analogous to that for a scalar monitor surface, where f_k is the percentage of the grid points to be attributed to curvature attraction for the kth solution feature. Further details on the computations for the vector monitor surface and on the example problems discussed herein are contained in reference 8.

Smoothing and Clipping the Monitor Surface

In many applications it is desirable to modify the monitor surface before commencing the computations to redistribute the nodes. Otherwise, nonsmoothness induced in the monitor surface by numerical inaccuracies can lead to substantial errors in the computations for grid attraction properties and thus results in nodes clustering in the wrong regions and even possibly singular grids. As an example, in the shock-vortex problem the vortex component of the monitor surface can contain large, but unimportant, data variations which must be eliminated. (See fig. 7.)

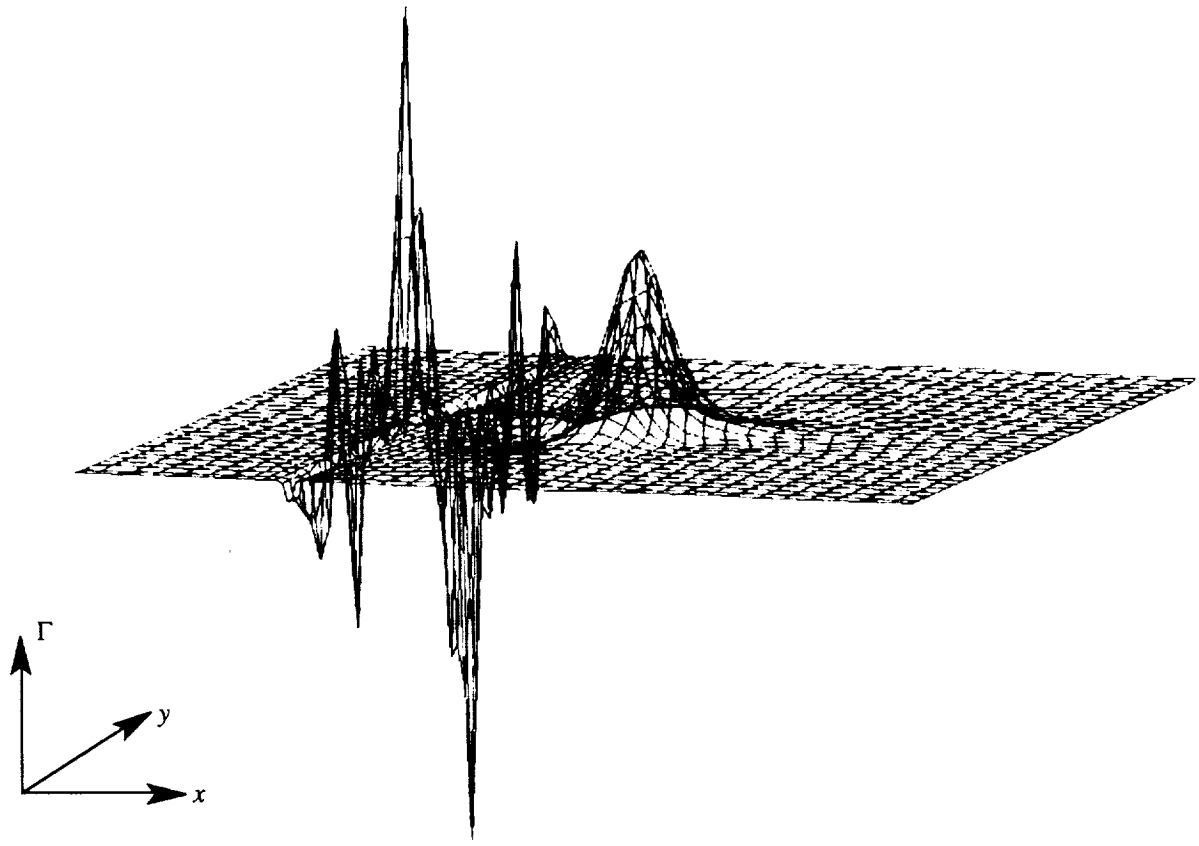


Figure 7. Vortex component of monitor surface before modification.

To eliminate unwanted data variations which have a large amplitude and high frequency, a data clipping scheme is applied to the monitor surface values before the grid movement is invoked (ref. 8). The data clipping scheme identifies the points having a monitor surface value that is not consistent with neighboring values and then sets, or clips, the monitor surface value of the targeted point to the average value of the surrounding points. The clipping is performed by checking each monitor surface value against an average value computed over a three-by-three patch of nodes centered on the node in question:

$$\bar{\psi}_{i,j} = \frac{1}{8}\left(\psi_{i-1,j-1} + \psi_{i-1,j} + \psi_{i-1,j+1} + \psi_{i,j-1} + \psi_{i,j+1} + \psi_{i+1,j-1} + \psi_{i+1,j} + \psi_{i+1,j+1}\right) \quad (16)$$

If the value of $\psi_{i,j}$ exceeds the average value $\bar{\psi}_{i,j}$ by a specified amount, then the value of $\psi_{i,j}$ is set to the average value:

$$\psi_{i,j} = \begin{cases} \psi_{i,j} & (\psi_{i,j} < \alpha\bar{\psi}_{i,j}) \\ \bar{\psi}_{i,j} & (\psi_{i,j} \geq \alpha\bar{\psi}_{i,j}) \end{cases} \quad (17)$$

where $\alpha = 1.15$, based on numerical tests. When the data clipping scheme is applied to a quantity which should be nonnegative, the $\psi_{i,j}$ values are "preprocessed" by setting all the negative $\psi_{i,j}$ values to zero. To eliminate any low-amplitude, high-frequency residual data variations (e.g., Gibbs phenomena) left over from the data clipping, the low-pass filter used to smooth the grid positions in the passive phase is applied to the monitor surface values after the data clipping routine is performed. In practice, the typical sequence of events is to apply the data clipping scheme to the monitor surface data two times and then to apply two passes of the smoothing operation. After the clipping and smoothing are completed, the monitor surface values are rescaled to the prescribed maximum value. In the case of a vector monitor surface, the components of $\mathbf{\Psi}(\mathbf{x})$ are processed individually. The benefit of using the data clipping scheme and the smoothing operation can be seen by comparing figures 7 and 8.

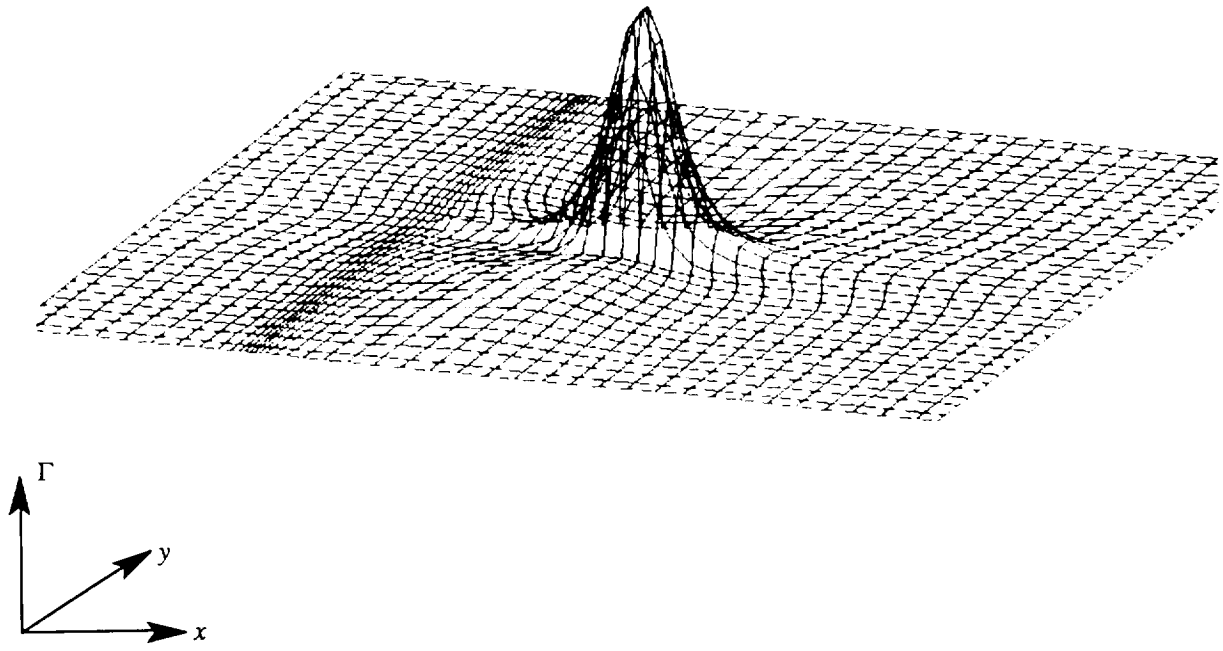


Figure 8. Vortex component of monitor surface after modification.

Summary of Grid Movement Scheme

The adaptive grid movement scheme is summarized below. Given a *smooth* monitor surface (i.e., after any monitor surface data clipping or smoothing has been performed), the procedure for obtaining the new grid in a two-dimensional-flow problem is as follows:

1. Perform the *active* phase for each curve in the ξ-direction.

   A. Compute the arc lengths and curvatures of the monitor surface at each node on the curve in question.

   B. Form the weight functions for each node.

C. Redistribute the points on the monitor surface along the current curve to satisfy the equidistribution statement.

2. Perform the *passive* phase for the ξ-direction by applying the low-pass filter to the grid.
3. Repeat steps 1 and 2 for each curve in the η-direction.

When a global iteration cycle on steps 1 to 3 is performed, the grid typically settles into a final configuration within a few iterations.

The Euler Equation Solver

The adaptive grid movement is coupled to an Euler equation solver (ref. 8). The solver uses a finite volume formulation to discretize the Euler equations in generalized coordinates on a static grid. The solution method uses a characteristic-based scheme that captures crisp monotone shock profiles. The flux terms are computed with Roe flux-difference splitting (refs. 17 and 18), and the spatial variation of the fluid is approximated with the Van Leer MUSCL scheme (ref. 19). The solution is integrated in a time-accurate manner with an explicit two-stage Runge-Kutta scheme (ref. 15).

Temporal Coupling

A simple grid prediction-correction technique is used to incorporate the adaptive grid movement into the static grid Euler equation solver (ref. 8). The prediction-correction algorithm can be divided into six basic steps. Given the grid and solution at some time level T, to advance the solution in time,

1. Choose a time interval τ.
2. Predict the solution over τ.
3. Form a monitor surface over τ.
4. Adapt the grid.
5. Transfer the solution data.
6. Solve over τ, then repeat steps 1 to 6 for the new time level $T + \tau$.

In the following description, the grid at the initial time T is denoted by $\mathbf{x}$, and the solution at time T on the given grid is denoted by $\mathbf{q}(\mathbf{x}, T)$.

Choosing a Time Interval τ

In this study, the time interval τ is determined by integrating forward γ time steps. Hence, the value of τ is $\tau = \sum_{k=1}^{\gamma} \Delta t_k$, where Δt_k is the time step determined by the Euler equation solver (i.e., $\Delta t_k = \min_{i,j} \Delta t_{i,j}$) during the forward time integration. The value of γ is predetermined and remains fixed for all time levels. Based on numerical experiments the value of γ is set at 10; this value of γ provides a good compromise between the grid resolution requirements over the time interval τ and reduction of the diffusion of the solution that occurs during the data transfers.

Predicting the Solution

The solution is predicted up to the forward time level $T + \tau$ by integration of the given solution forward on the given static grid in a time-accurate manner. At each time step of the prediction stage, a monitor surface ψ_i is computed and stored. In the case of a vector monitor surface, the components of the vector are computed and stored separately in a disk file. If needed, clipping and smoothing of the monitor surface are performed when the individual monitor surface is computed. In addition to the γ individual monitor surfaces computed during the forward time integration, a monitor surface is also formed from the initial condition to the prediction stage (i.e., $\mathbf{q}(\mathbf{x}, T)$) to provide a buffer region of grid resolution "behind" the evolving solution and thereby help provide a smooth transition in node spacing between successive grids.

Forming the Monitor Surface

After the prediction stage is completed, a composite monitor surface is computed by averaging the intermediate monitor surfaces computed during the forward integration on the given grid $\mathbf{x}$ from T to $T + \tau$. The composite monitor surface is computed as $\Psi = \frac{1}{\gamma+1} \sum_{i=1}^{\gamma+1} \psi_i$.

Adapting the Grid

With the composite monitor surface defined, the grid adaptation module is invoked to obtain a grid for the time period T to $T + \tau$. Before the grid movement is started, the individual components of the composite monitor surface are smoothed with equation (10), regardless of how the intermediate monitor surfaces were computed; this smoothing ensures that the composite monitor surface produced by the sum of the discrete monitor surfaces is itself smooth. The new grid is generated by performing only two cycles of the curve-by-curve scheme rather than by iterating until the grid has converged. Numerical tests have shown that the grid positions are essentially determined within two iterations. It is doubtful the improved grid resolution that would be obtained with a more precise placement of the grid points would offset the additional expense of computing such a node placement. In addition, the new grid is typically generated with a relatively low degree of normal curvature clustering because the composite monitor surface represents the grid resolution requirements for

several time steps. For applications containing propagating wave fronts, the normal curvature attraction results in a buffer region of resolution immediately behind and ahead of, respectively, the initial and final solution used to create the composite monitor surface.

Transferring the Solution Data

After the new grid is computed, the stored numerical solution from time level T on the previous grid, $\mathbf{q}(\mathbf{x}, T)$, is transferred to the new grid $\mathbf{x}^*$ to yield $\mathbf{q}(\mathbf{x}^*, T)$. The data transfer is performed with local bilinear interpolation. The bilinear interpolation does not create oscillations in the interpolated solution vector but is dissipative, does not maintain conservation of the solution, and uses data from ahead and behind the shock wave to perform the interpolation; it therefore somewhat defeats the purpose of using an upwind differencing scheme in the PDE solver. Hence, there could be a benefit to using a conservative data transfer scheme (refs. 20 to 22) or a nonoscillatory higher order interpolation method (refs. 23 and 24).

Solving Over τ

The last step of the prediction-correction algorithm is to integrate $\mathbf{q}(\mathbf{x}^*, T)$, the solution at time level T on the new static grid, forward to the time level $T + \tau$. To ensure that the solution does not outrun the grid resolution provided by the new grid, during the correction stage the solution is integrated only up to the time level $T + \tau'$, where $\tau' = 0.95\tau$. Thus, the final solution obtained from the prediction-correction procedure is $\mathbf{q}(\mathbf{x}^*, T + \tau')$. If additional temporal accuracy in the solution is desired, the prediction-correction procedure can be repeated several times before the solution is advanced to the next time level.

Repeating the Process

With the solution and the grid established at the time level $T + \tau'$, the entire prediction-correction procedure is repeated to advance the solution to another time level. By successive repetition of the prediction-correction procedure, the solution is marched forward through time to the desired final time level T_f.

The Shock-Vortex Problem

To test the adaptive grid solution method, the unsteady inviscid flow field for a shock wave passing through a vortex is calculated. The shock-vortex problem is of interest at the physical level because it provides an idealized model for studying certain acoustic problems and problems associated with turbulence amplification from shock waves (refs. 25 to 27). It also provides a model for studying blade-vortex interaction for helicopter blades operating at supercritical speeds (ref. 3). From a computational viewpoint the problem is of interest because the solution exhibits severe solution behavior and the position of the severe solution gradients changes with time. Thus, a locally high-resolution grid is required to capture accurately all the complexities that develop in the shock front and the vortex as they approach each other, merge, and continue.

The adaptive grid solution method developed herein has the important capability of readily extending beyond the given shock-vortex case to address patterns with further complexity. This same extensibility is not as directly accessible with the shock fitting methods employed in earlier numerical studies of the shock-vortex problem (refs. 25 and 26). The adaptive grid solution also has the potential to improve the efficiency compared with solutions computed on very fine stationary grids using shock-capturing methods (ref. 27). The adaptive grid method used herein can adapt the grid to both the shock wave and the vortex and thereby reduce the number of grid points required to capture the important physics of the problem.

Problem Definition

The shock-vortex problem modeled herein consists of an initially planar shock wave marching toward, and eventually over, a solid-core vortex (fig. 9).

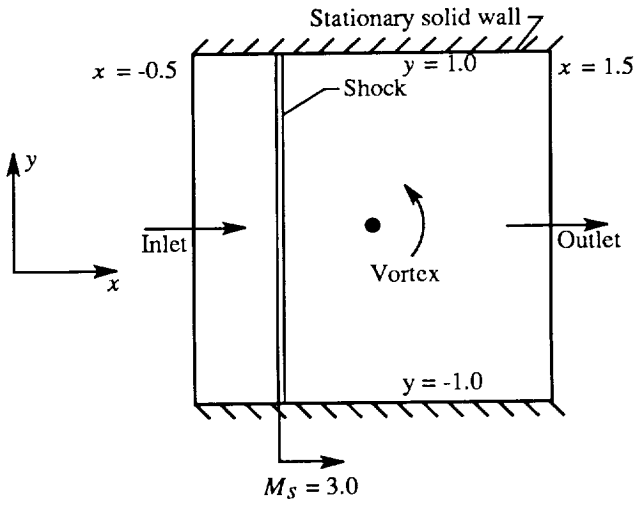


Figure 9. Definition of shock-vortex problem.

The shock wave and vortex are assumed to lie within a channel having solid walls. All lengths are nondimensionalized with respect to a reference length

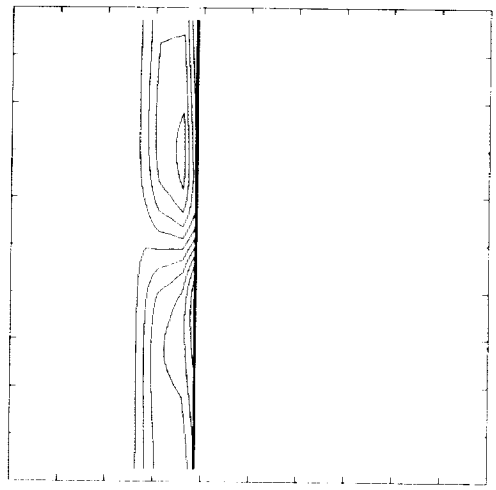
(e.g., the channel half-width) and the flow-field variables with respect to reference conditions at $x = \infty$. The governing equations for the model problem are the unsteady, two-dimensional, compressible Euler equations, which are well documented in the literature (ref. 28) and therefore are not presented herein.

The initial flow field consists of a planar shock located at $x = 0$ and a solid-core vortex rotating counterclockwise located at $x = 0.5$. To the left of the shock wave is the uniform supersonic flow field that would follow behind a shock wave propagating at a relative Mach number M_s of 3.0 if there were no upstream disturbance. To the right (i.e., ahead) of the shock the initial flow field is obtained by assuming a constant density field and calculating the velocity from the stream function $G(x,y) = -\frac{\lambda}{2\pi}\log\{[(x-x_o)^2+(y-y_o)^2+b^2]^{1/2}\}$, the pressure field from Bernoulli's equation, and the total energy from the equation of state. For this study $\lambda = 0.40$, $b = 0.1$, and $(x_o, y_o) = (0.5, 0)$. At the leading edge of the initial shock front (i.e., $x = 0$), the supersonic flow field behind the shock and the subsonic flow field ahead of the shock are smoothly merged in the space of a few grid cells in order to eliminate a numerically induced "bump" which occurs in the solution if the two flow fields are merged abruptly (i.e., within one grid cell). The additional bump in the solution occurs in the supersonic flow behind the shock front, has a magnitude which is a function of the grid cell size, and occurs when there is a vortex upstream of the shock and when there is no upstream disturbance. The flow field for the smooth shock front is obtained from the solution of a planar shock wave propagating into a region with no upstream disturbance. The modification of the initial flow field only affects the grid points near $x = 0$; away from this region the initial flow field is as described above. Further details are available in reference 8.

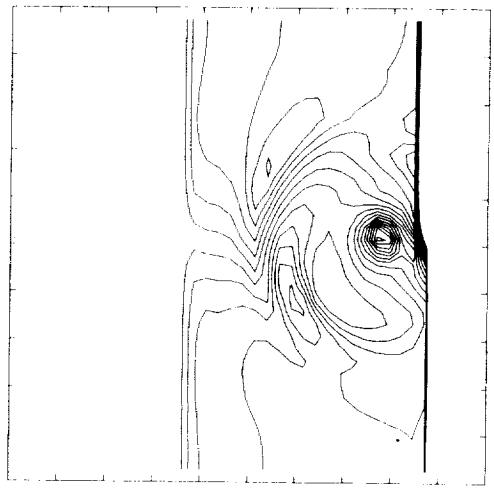
On the boundaries of the domain it is assumed that (1) at the inlet the flow field is that of the initial uniform supersonic flow, (2) at the outlet the flow field is that of the initial vortex, and (3) along the top and bottom of the domain the flow is tangent to the boundary (i.e., solid walls).

The Fine-Grid Solution

The results for a solution computed on a static grid with fine grid resolution are presented first to provide a basis of comparison for the adaptive grid solutions. Illustrated in figure 10 are contour plots for the pressure field of the solution computed on a static grid with 200 uniformly spaced cells in the axial direction and 50 cells in the transverse direction. The cells have a uniform size in computational space, where the transformation from physical to computational space is given by $Y = \tanh 2y/\tanh 2$ and $Y \in [-1, 1]$. The pressure field for the entire computational domain is plotted at two time levels that correspond to before the shock reaches the rim of the vortex core ($t = 0.10$) and to a long time after the interaction has occurred ($t = 0.42$). The pressure contour levels have been chosen to illustrate the small-scale flow-field characteristics in the region trailing the shock wave (i.e., 30 contours uniformly spaced in the interval $0.85 \le p/p_{\text{inlet}} \le 1.10$) (ref. 8); the change in pressure between the contours is less than 1 percent of the pressure jump across the shock wave.



(a) $t = 0.10$.



(b) $t = 0.42$.

Figure 10. Contour plots of pressure field for fine-static-grid solution.

Only the key features that can be easily discerned from the attached pressure plots are highlighted herein; for a thorough review of the physics involved for the dynamic interaction of the shock with the vortex, see Pao and Salas (ref. 25), or Meadows et al. (ref. 27). As shown in figure 10(a), before the shock wave reaches the vortex core, the shock front is still essentially planar and the pressure field behind the shock wave is slightly asymmetric but mostly constant. When the shock wave passes over the vortex core, the shock front becomes curved. A long time after the interaction has occurred (fig. 10(b)), the shock wave has moved almost to the outlet, the shock front is still curved, and the vortex, which can be identified by the concentric pressure contours behind the shock wave along the channel centerline, has been convected downstream. In figure 10(b) there is a visible separation between the shock wave and the vortex because the shock wave propagates at a faster speed than the vortex. With the basic solution established, the adaptive grid method can now be applied to reduce the number of grid points.

Adaptation to Only the Shock Wave

In the first adaptive grid solution an 80- by 32-cell grid is adapted to a monitor surface formed from the density field. Redistributing the grid points based on the density field clusters the grid points at the shock front because the density field is approximately constant within the vortex and far behind the shock front but undergoes a large increase across the shock front.

The monitor surface for this case is described by

$$\Psi_{i,j} = \frac{h(\rho_{i,j} - \rho_{\min})}{\rho_{\max} - \rho_{\min}} \tag{18}$$

where $\rho_{\max} = \rho_{\text{inlet}}$ is the maximum density, $\rho_{\min} = \rho_{\text{outlet}}$ is the minimum density, and h is the height of the monitor surface. By adjusting h, we can make the node cell spacing at the shock front as small as desired. Setting $h = 0.20$ results in a grid cell spacing at the shock wave (i.e., a numerical shock width) that is comparable to that of the fine-static-grid solution. Because there is little variation in the density field in the transverse direction, to maintain adequate grid resolution along the channel centerline, the grid point movement in the transverse direction occurs with respect to a background nodal distribution given by a hyperbolic tangent spacing ($Y = \tanh 2y/\tanh 2$). This node spacing is locally deviated from only in regions having a high normal curvature which receive additional node concentration. The percentage of grid points attributed to curvature clustering is set at $f = 0.075$. The initial grid (i.e., at $t = 0$) is obtained through adaptation of the grid to a monitor surface formed from the density field of the initial condition.

Illustrated in figure 11 are the grid and pressure-field contour plots for adapting the grid to only the shock wave. In the grid plots, the location of the shock wave can be determined by the densely packed strip of grid lines. Comparing the grid and pressure contour plots in figure 11 shows that the adaptive grid correctly tracks the shock wave. Comparing the pressure contour plots in figures 10 and 11 shows that there is good agreement between the adaptive and static grid solutions. However, in the adaptive grid case significantly fewer grid points have been used.

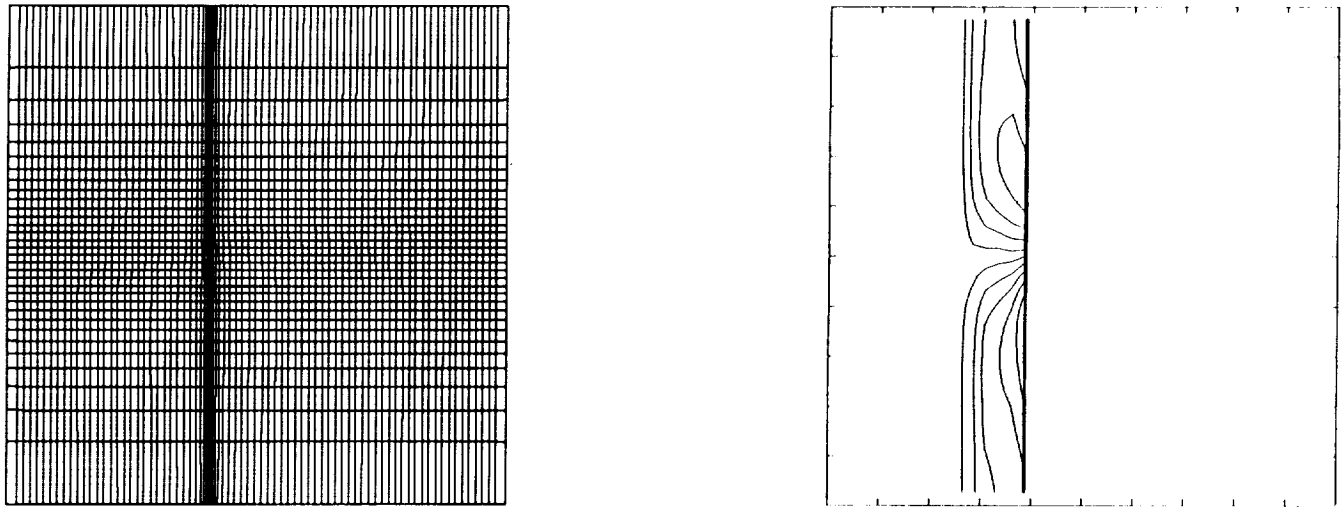
Adaptation to Both the Shock Wave and the Vortex

Illustrated in figure 12 are the results for a solution computed on an 80- by 32-cell grid adapted to a monitor surface containing data from both the shock wave and the vortex. Adapting the grid to the shock wave alone is much simpler than adapting it to the vortex because the shock wave is confined to a very narrow-banded region. Adapting the grid to the vortex requires clustering grid points into a broad area in the flow field in which the solution does not necessarily exhibit severe behavior. It should be noted that the flow-field variations created by the presence of the vortex in the region trailing the shock wave are almost two orders of magnitude smaller than the increase in the flow-field values across the shock wave.

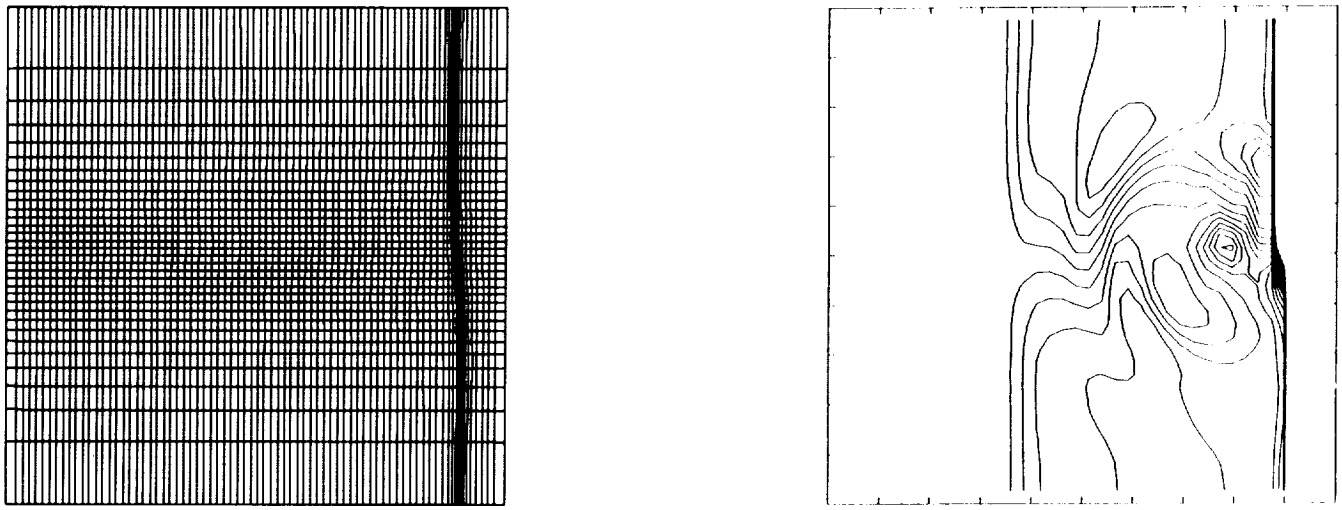
For this solution, the grid is obtained with a vector monitor surface defined from the density field and the circulation about each point. As described above, the density field identifies the shock front and concentrates points to it. The circulation is defined as $\Gamma = \oint \mathbf{V} \cdot d\mathbf{l}$, where $\mathbf{V}$ is the velocity vector and $d\mathbf{l}$ is the differential length along a closed path in the flow field. The circulation provides a means of tracking and clustering at the vortex core because Γ is approximately zero everywhere except within the vortex core. The circulation is computed at each node through use of a circuit around the node defined by the four adjacent grid points that surround it; on the boundaries of the domain the circulation is set to be $\Gamma = 0$.

The vector monitor surface is described by

$$\left.\begin{aligned} \Psi_{i,j} = (\psi^s_{i,j}, \psi^v_{i,j}) \qquad \psi^s_{i,j} &= h_s \frac{\rho_{i,j} - \rho_{\min}}{\rho_{\max} - \rho_{\min}} \\ \psi^v_{i,j} &= h_v \frac{\Gamma_{i,j} - \Gamma_{\min}}{\Gamma_{\max} - \Gamma_{\min}} \end{aligned}\right\} \tag{19}$$



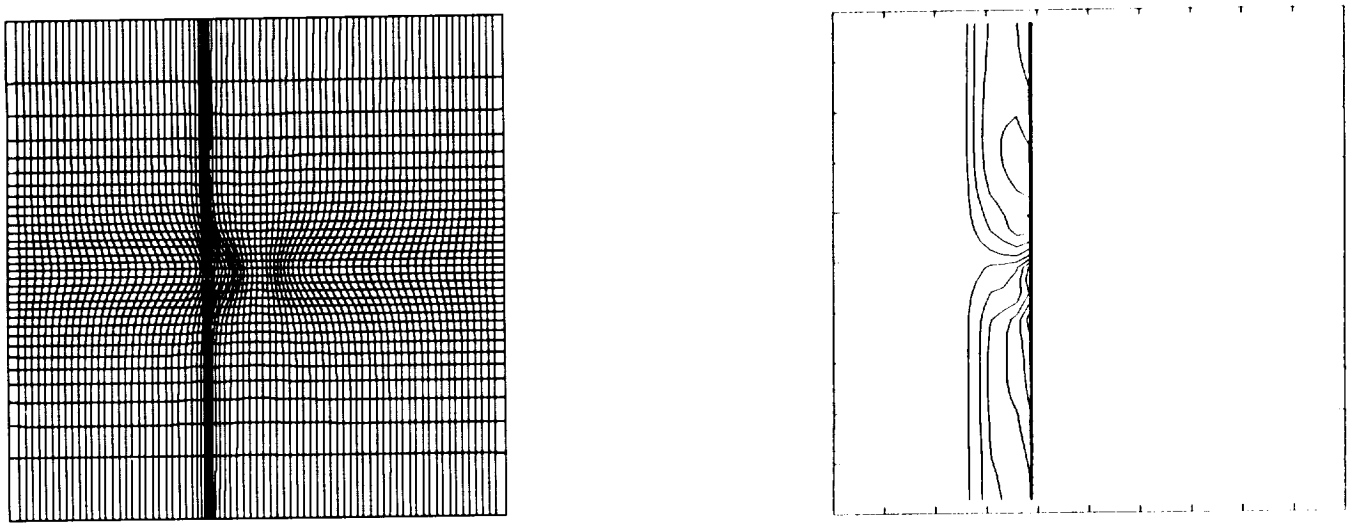
(a) $t = 0.10$.



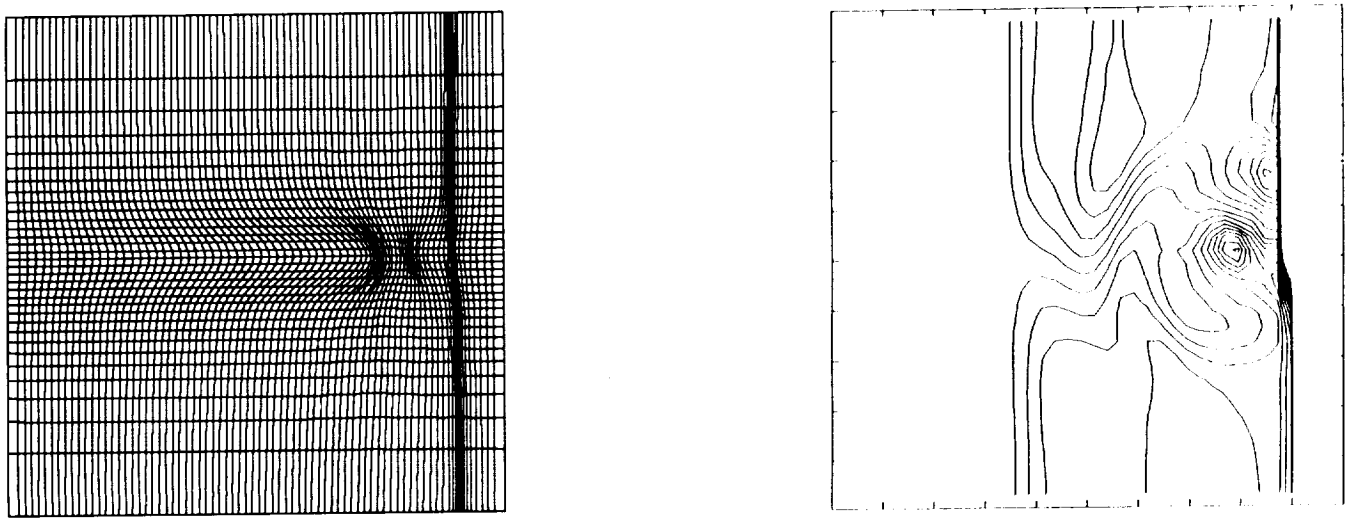
(b) $t = 0.42$.

Figure 11. Grid and pressure-field contour plots for adapting grid to only the shock wave.

where ρ and Γ are the density and circulation, $\rho_{\max}$ and $\rho_{\min}$ are as defined in equation (18), $\Gamma_{\min}$ and $\Gamma_{\max}$ are the minimum and maximum values of Γ that occur within the solution domain, and h_s and h_v are the heights of the shock wave and vortex components of the monitor surface. For the solution described here, $h_v = 0.35$, $h_s = 0.20$, and the normal curvature clustering parameters for the shock wave and vortex are $f_s = 0.075$ and $f_v = 0.0375$. Using these values results in approximately the same grid resolution on the shock wave as for the previous adaptive solution. Because the computed circulation values can contain a large amount of high-amplitude, but unimportant, variations in the regions along the shock front, the previously described data smoothing and clipping schemes are applied to the vortex component of the vector monitor surface. In addition, because the intent is to capture both the vortex core and its effect on the flow field trailing the shock wave, the vortex component of the vector monitor surface is smeared over a large number of grid points; the smearing is performed by passing the vortex component 25 times through the low-pass filter used to smooth the monitor surface values before any grid movement occurs.



(a) $t = 0.10$.



(b) $t = 0.42$.

Figure 12. Grid and pressure-field contour plots for adapting grid to both shock wave and vortex.

The key feature to notice in figure 12 is that the adaptive grid correctly mimics the shock wave and vortex portions of the flow field. That is, the grid is correctly capturing the time-dependent location and shape of the shock wave and the vortex; in the grid plots, the cluster of grid points near the channel centerline is due to the attraction to the vortex. Comparing figures 11 and 12 shows that adapting the grid to the vortex results in a slightly tighter packing of the pressure contours about the vortex, but near the channel centerline there is a slight degradation of the shock wave because the same number of grid points are used to resolve two solution features as are used to resolve just one.

Conclusions

A general adaptive strategy is described for use with structured grids. The basic grid movement scheme is capable of producing smooth grids that accurately place the grid points in regions of severe solution behavior. The multidimensional nature of the smoothing filter controls excessive skewness in the grid. New grid movement techniques have been

demonstrated that improve the tracking of multiple solution features and eliminate "noisy" values in the adaptive data. A simple and flexible prediction-correction scheme is demonstrated for coupling the adaptive grid to a static grid Euler equation solver. The adaptive solution method is applied to solve for the unsteady flow field of a shock-vortex interaction problem; the ability of the adaptive method to compute time-accurate solutions and to accurately adapt the grid to multiple solution features is thus demonstrated. The same adaptive solution method can be used to study a large number of compressible-flow problems. Furthermore, this method is also applicable to a wide range of problems.

It should be noted that the adaptive method described herein is still in the research stages and requires further improvement to be competitive with a fine-static-grid solution. Future work should concentrate on extending the current method to three-dimensional problems and improving the computational efficiency of the temporal coupling. The extension to three-dimensional problems will require deriving the intrinsic curvature properties for a curve contained in a three-dimensional surface embedded in a four-dimensional space. In the temporal coupling scheme the computational resources consumed by the prediction stage must be reduced. In particular, the storage on disk of the individual monitor surfaces should be replaced with computation of a running sum of the individual monitor surfaces in an internal array. In addition, for many applications an adequate estimate of the adaptive data needed to generate the new grid could be obtained with a coarse grid in the prediction stage. The coarse grid could be computed from the full grid by elimination of every other grid point in each direction, the result being the predicted solution is only integrated over only one-fourth as many grid points as for a two-dimensional problem. Furthermore, the coarse grid would allow use of a larger time step based on the Courant-Freidrichs-Lewy (CFL) constraint. Use of a larger time step would result in fewer time integrations to cover the time interval of the prediction stage and thereby provide a significant reduction of the computational time to perform the adaptive solution.

NASA Langley Research Center
Hampton, VA 23665-5225
March 27, 1990

References

1. Dwyer, H. A.: Grid Adaption for Problems in Fluid Dynamics. *AIAA J.*, vol. 22, no. 12, Dec. 1984, pp. 1705–1712.
2. Eiseman, Peter R.: Alternating Direction Adaptive Grid Generation. *AIAA J.*, vol. 23, no. 4, Apr. 1985, pp. 551–560.
3. Srinivasan, G. R.; McCroskey, W. J.; and Baeder, J. D.: Aerodynamics of Two-Dimensional Blade-Vortex Interaction. *AIAA J.*, vol. 24, no. 10, Oct. 1986, pp. 1569–1576.
4. Eiseman, Peter R.: Adaptive Grid Generation. Comput. Methods Appl. Mech. & Eng., vol. 64, nos. 1–3, Oct. 1987, pp. 321–376.
5. Eiseman, Peter R.; and Erlebacher, Gordon: *Grid Generation for the Solution of Partial Differential Equations.* NASA CR-178365, ICASE Rep. No. 87-57, 1987.
6. Abolhassani, Jamshid S.; Smith, Robert E.; and Tiwari, Surendra N.: Grid Adaption for Hypersonic Flow. *A Collection of Technical Papers—AIAA 8th Computational Fluid Dynamics Conference*, June 1987, pp. 726–733. (Available as AIAA-87-1169.)
7. Blom, J. G.; Sanz-Serna, J. M.; and Verwer, J. G.: On Simple Moving Grid Methods for One-Dimensional Evolutionary Partial Differential Equations. *J. Comput. Phys.*, vol. 74, no. 1, Jan. 1988, pp. 191–213.
8. Bockelie, Michael John: Adaptive Grid Movement Schemes and the Numerical Simulation of Shock Vortex Interaction. Ph.D. Thesis, Columbia Univ., 1988.
9. Eiseman, P. R.; and Bockelie, M. J.: Adaptive Grid Solution for Shock-Vortex Interaction. *11th International Conference on Numerical Methods in Fluid Dynamics, Volume 323 of Lecture Notes in Physics*, D. L. Dwoyer, M. Y. Hussaini, and R. G. Voigt, eds., Springer-Verlag, 1989, pp. 240–244.
10. Thompson, Joe F., ed.: *Numerical Grid Generation.* Elsevier Science Publ. Co., Inc., c.1982.
11. Thompson, Joe F.: Grid Generation Techniques in Computational Fluid Dynamics. *AIAA J.*, vol. 22, no. 11, Nov. 1984, pp. 1505–1523.
12. Thompson, Joe F.; Warsi, Z. U. A.; and Mastin, C. Wayne: *Numerical Grid Generation—Foundations and Applications.* North-Holland, c.1985.
13. Nakahashi, Kazuhiro; and Deiwert, George S.: Self-Adaptive-Grid Method With Application to Airfoil Flow. *AIAA J.*, vol. 25, no. 4, Apr. 1987, pp. 513–520.
14. Nakahashi, Kazuhiro; and Deiwert, George S.: Three-Dimensional Adaptive Grid Method. *AIAA J.*, vol. 24, no. 6, June 1986, pp. 948–954.
15. Nakahashi, Kazuhiro: An Automatic Grid Generator for the Unstructured Upwind Method. *AIAA 9th Computational Fluid Dynamics Conference*, June 1989, pp. 533–541. (Available as AIAA-89-1985-CP.)
16. O'Neill, Barrett: *Elementary Differential Geometry.* Academic Press, 1966.
17. Roe, P. L.: Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. *J. Comput. Phys.*, vol. 43, no. 2, Oct. 1981, pp. 357–372.
18. Roe, P. L.: Characteristic-Based Schemes for the Euler Equations. *Annual Review of Fluid Mechanics, Volume 18*, Milton van Dyke, J. V. Wehausen, and John L. Lumley, eds., Annual Reviews Inc., 1986, pp. 337–365.
19. Anderson, W. Kyle; Thomas, James L.; and Van Leer, Bram: Comparison of Finite Volume Flux Vector

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TP-2998		

4. Title and Subtitle	5. Report Date
A Time-Accurate Adaptive Grid Method and the Numerical Simulation of a Shock-Vortex Interaction	June 1990
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
Michael J. Bockelie and Peter R. Eiseman	L-16727
	10. Work Unit No.
9. Performing Organization Name and Address	505-90-21-02
NASA Langley Research Center Hampton, VA 23665-5225	11. Contract or Grant No.
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Technical Paper
National Aeronautics and Space Administration Washington, DC 20546-0001	14. Sponsoring Agency Code

15. Supplementary Notes

Michael J. Bockelie: NRC–NASA Research Associate, Langley Research Center, Hampton, Virginia.

Peter R. Eiseman: Columbia University, New York, New York.

Work partially supported NASA grant NAG1-427 and U.S. Air Force grant AFOSR-86-0307.

16. Abstract

A time-accurate, general-purpose adaptive grid method is developed that is suitable for multi-dimensional steady and unsteady numerical simulations. The grid point movement is performed in a manner that generates smooth grids which resolve the severe solution gradients and sharp transitions in the solution gradients. The temporal coupling of the adaptive grid and the partial differential equation solver is performed with a grid prediction-correction method that is simple to implement and ensures the time accuracy of the grid. Time-accurate solutions of the two-dimensional Euler equations for an unsteady shock-vortex interaction demonstrate the ability of the adaptive method to accurately adapt the grid to multiple solution features.

17. Key Words (Suggested by Authors(s))	18. Distribution Statement
Solution adaptive Grid generation Shock-vortex interaction	Unclassified—Unlimited Subject Category 59

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	18	A03

For sale by the National Technical Information Service, Springfield, Virginia 22161-2171

Splittings for the Euler Equations. *AIAA J.*, vol. 24, no. 9, Sept. 1986, pp. 1453–1460.

20. Dukowicz, John K.: Conservative Rezoning (Remapping) for General Quadrilateral Meshes. *J. Comput. Phys.*, vol. 54, no. 3, June 1984, pp. 411–424.
21. Dukowicz, John K.; and Kodis, John W.: Accurate Conservative Remapping (Rezoning) for Arbitrary Lagrangian-Eulerian Computations. *SIAM J. Sci. & Stat. Comput.*, vol. 8, no. 3, May 1987, pp. 305–321.
22. Ramshaw, John D.: Conservative Rezoning Algorithm for Generalized Two-Dimensional Meshes. *J. Comput. Phys.*, vol. 59, no. 2, June 1985, pp. 193–199.
23. Chakravarthy, Sukumar R.; Harten, Ami; and Osher, Stanley: Essentially Non-Oscillatory Shock-Capturing Schemes of Arbitrarily-High Accuracy. AIAA-86-0339, Jan. 1986.
24. Harten, Ami; and Osher, Stanley: Uniformly High-Order Accurate Nonoscillatory Schemes. I. *SIAM J. Numer. Anal.*, vol. 24, no. 2, Apr. 1987, pp. 279–309.
25. Pao, S. P.; and Salas, M. D.: A Numerical Study of Two-Dimensional Shock Vortex Interaction. AIAA-81-1205, June 1981.
26. Hussaini, M. Y.; Kopriva, D. A.; Salas, M. D.; and Zang, T. A.: *Spectral Methods for the Euler Equations: Chebyshev Methods and Shock-Fitting.* NASA CR-172295, 1984.
27. Meadows, Kristine R.; Kumar, Ajay; and Hussaini, M. Y.: A Computational Study on the Interaction Between a Vortex and a Shock Wave. AIAA-89-1043, Apr. 1989.
28. Anderson, Dale A.; Tannehill, John C.; and Pletcher, Richard H.: *Computational Fluid Mechanics and Heat Transfer.* Hemisphere Publ. Corp., c.1984.

National Aeronautics and
Space Administration
Code NTT-4

Washington, D.C.
20546-0001