NASA Technical Memorandum 106774
AIAA–94–0319

An Approach for Dynamic Grids

John W. Slater and Meng-Sing Liou
Lewis Research Center
Cleveland, Ohio

Richard G. Hindman
Iowa State University of Science and Technology
Ames, Iowa

Prepared for the
32nd Aerospace Sciences Meeting and Exhibit
sponsored by the American Institute of Aeronautics and Astronautics
Reno, Nevada, January 10–13, 1994

National Aeronautics and
Space Administration

AN APPROACH FOR DYNAMIC GRIDS

J. W. Slater[*] and M.S. Liou[†]
NASA Lewis Research Center
Cleveland, Ohio

and

R.G. Hindman[‡]
Iowa State University
Ames, Iowa

Abstract

An approach is presented for the generation of two-dimensional, structured, dynamic grids. The grid motion may be due to the motion of the boundaries of the computational domain or to the adaptation of the grid to the transient, physical solution. A time-dependent grid is computed through the time integration of the grid speeds which are computed from a system of grid speed equations. The grid speed equations are derived from the time-differentiation of the grid equations so as to ensure that the dynamic grid maintains the desired qualities of the static grid. The grid equations are the Euler-Lagrange equations derived from a variational statement for the grid. The dynamic grid method is demonstrated for a model problem involving boundary motion, an inviscid flow in a converging-diverging nozzle during startup, and a viscous flow over a flat plate with an impinging shock wave. It is shown that the approach is more accurate for transient flows than an approach in which the grid speeds are computed using a finite difference with respect to time of the grid. However, the approach requires significantly more computational effort.

Introduction

As computational methods become more efficient and computational resources become more powerful, the methods of computational fluid dynamics are being applied to more complex problems. Among these are problems involving the motion of the boundaries of the computational domain and problems involving transient flow conditions with solution adaptive grids. These generally require a dynamic grid. Much work has been done on methods for the generation of static grids with proper qualities. A dynamic grid should maintain these same qualities; however, this topic has not received much attention.

We present an approach for the generation of dynamic grids for which the qualities of grid smoothness, orthogonality, and cell volume adaptation are maintained. The dynamic grids are generated through the time-integration of the grid speeds computed from a system of grid speed equations. The grid speed equations are derived from the time-differentiation of the grid equations. This approach was first presented by Hindman et al. [1] for an application involving the motion of a shock-fitted boundary. Holcomb and Hindman [2] extended the approach for the adaptation of the grid to the physical solution. The previous cited work involved grid equations based on the Poisson equations and did not account for grid orthogonality. The present approach uses a system of grid equations based on the variational approach of reference [3].

An alternative approach for computing the grid speeds is to use a time-difference of the grid. This approach is examined. It will be shown that solving the grid speed equations reduces the grid-motion-induced errors for the inviscid, unsteady flow in a nozzle.

The objectives of this paper are to present the system of grid and grid speed equations for two-dimensional space, present an explicit dynamically adaptive grid method, and demonstrate the behavior of the method through applications. These applications include a simple problem involving a moving boundary, an inviscid flow through a converging-diverging nozzle during startup, and a viscous flow over a flat plate with an impinging oblique shock. The last two flows involve flow conditions which are initially transient; however, after time, the flow conditions and the flow reach a steady-state.

The Grid Equations

The grid equations follow the variational approach of Brackbill and Saltzman [3] which derive the grid equations from the Euler-Lagrange equations,

$$\left(\frac{\partial}{\partial \xi}\frac{\partial}{\partial x_\xi}+\frac{\partial}{\partial \eta}\frac{\partial}{\partial x_\eta}-\frac{\partial}{\partial x}\right)L=0 \qquad (1)$$

and

$$\left(\frac{\partial}{\partial \xi}\frac{\partial}{\partial y_\xi}+\frac{\partial}{\partial \eta}\frac{\partial}{\partial y_\eta}-\frac{\partial}{\partial y}\right)L=0. \qquad (2)$$

The Lagrangian L is defined so as to include measures of smoothness, orthogonality, and cell volume adaptation and is of the form

$$L \;=\; L_S \;+\; L_O \;+\; L_A. \qquad (3)$$

The expressions for L_O and L_A presented here are a modification of those presented in reference [3] to nondimensionalize the Lagrangians. The Lagrangians are of the form

$$L_S = \lambda_S \frac{\vec{r}_\xi \cdot \vec{r}_\xi + \vec{r}_\eta \cdot \vec{r}_\eta}{J}, \qquad (4)$$

$$L_O = \lambda_O \left(\frac{\vec{r}_\xi \cdot \vec{r}_\eta}{J} \right)^2, \qquad (5)$$

and

$$L_A = \lambda_A \; (\; W \; J \;)^2 \; / \; K^2. \qquad (6)$$

The λ_S, λ_O, and λ_A are specified constants which weigh the importance of each Lagrangian. The cell volume adaptation Lagrangian L_A is essentially a statement of the equidistribution of the product WJ over the grid. The W is a weighting function usually directly related to the gradients in the flow solution. The J is the Jacobian and is a measure of the cell volume. The K is the average value of WJ over the grid.

The system of grid equations are derived by inserting the expressions for the Lagrangians into the Euler-Lagrange equations and is expressed in the form

$$G(\, \vec{r} \,) \; = \; A \, \vec{r}_{\xi\xi} + B \, \vec{r}_{\xi\eta} + C \, \vec{r}_{\eta\eta} + D \, \vec{r}_\xi + E \, \vec{r}_\eta = 0 \quad (7)$$

where $\vec{r}^{\,T} = [\; x, \; y \;]$. The coefficient matrices are sums of the form

$$A = A_S + A_O + A_A.$$

The B, C, D, and E matrices are of the same form.

The expressions for the smoothness matrices are

$$A_S = \frac{\lambda_S \; \alpha}{J^3} [S], \qquad (8)$$

$$B_S = \frac{- \, 2 \; \lambda_S \; \beta}{J^3} [S], \qquad (9)$$

$$C_S = \frac{\lambda_S \; \gamma}{J^3} [S], \qquad (10)$$

and D_S and E_S are the null matrix [0]. Also,

$$S = \begin{bmatrix} \delta & - \, \epsilon \\ - \, \epsilon & \rho \end{bmatrix}. \qquad (11)$$

The remaining variables are defined as

$$J \; = \; x_\xi \; y_\eta \; - \; x_\eta \; y_\xi, \qquad (12)$$

$$\alpha \; = \; x_\eta^2 + y_\eta^2, \qquad (13)$$

$$\beta \; = \; x_\xi \; x_\eta \; + \; y_\xi \; y_\eta, \qquad (14)$$

$$\gamma \; = \; x_\xi^2 + y_\xi^2, \qquad (15)$$

$$\delta \; = \; y_\xi^2 + y_\eta^2, \qquad (16)$$

$$\epsilon \; = \; x_\xi \; y_\xi \; + \; x_\eta \; y_\eta, \qquad (17)$$

and

$$\rho = x_\xi^2 + x_\eta^2. \qquad (18)$$

The expressions for the orthogonality matrices are

$$A_O = \frac{\lambda_O \alpha}{J^4} \begin{bmatrix} 2\beta y_\eta y_\xi + \alpha y_\xi^2 & \psi \\ \psi & 2\beta x_\eta x_\xi + \alpha x_\xi^2 \end{bmatrix}, \quad (19)$$

$$B_O = - \, \frac{\lambda_O (2\beta^2 + \alpha\gamma)}{J^4} \begin{bmatrix} 2 y_\xi y_\eta & - \, \phi \\ - \, \phi & 2 x_\xi x_\eta \end{bmatrix}, \qquad (20)$$

$$C_O = \frac{\lambda_O \gamma}{J^4} \begin{bmatrix} \gamma y_\eta^2 + 2\beta y_\xi y_\eta & \kappa \\ \kappa & \gamma x_\eta^2 + 2\beta x_\xi x_\eta \end{bmatrix}, \quad (21)$$

and D_O and E_O are the null matrix [0]. The ϕ, ψ, and κ are defined as

$$\phi \; = \; x_\xi \; y_\eta \; + \; x_\eta \; y_\xi, \qquad (22)$$

$$\psi \; = \; - \; (\; \gamma \; x_\eta \; y_\eta + 2 \; \alpha \; x_\xi \; y_\xi \;), \qquad (23)$$

and

$$\kappa \; = \; - \; (\; \alpha \; x_\xi \; y_\xi + 2 \; \gamma \; x_\eta \; y_\eta \;). \qquad (24)$$

The expressions for the cell volume adaptation matrices are

$$A_A = \frac{\lambda_A W^2}{K^2} \begin{bmatrix} y_\eta^2 & - \, x_\eta y_\eta \\ - \, x_\eta y_\eta & x_\eta^2 \end{bmatrix}, \qquad (25)$$

$$B_A = \frac{\lambda_A W^2}{K^2} \begin{bmatrix} - \, 2 y_\xi y_\eta & \phi \\ \phi & - \, 2 x_\xi x_\eta \end{bmatrix}, \qquad (26)$$

$$C_A = \frac{\lambda_A W^2}{K^2} \begin{bmatrix} y_\xi^2 & - \, x_\xi y_\xi \\ - \, x_\xi y_\xi & x_\xi^2 \end{bmatrix}, \qquad (27)$$

$$D_A = \frac{\lambda_A J W W_\eta}{K^2} \begin{bmatrix} 0 & - \, 1 \\ 1 & 0 \end{bmatrix}, \qquad (28)$$

and

$$E_A = \frac{\lambda_A J W W_\xi}{K^2} \begin{bmatrix} 0 & 1 \\ - \, 1 & 0 \end{bmatrix}. \qquad (29)$$

The Grid Speed Equations

The grid speed equations are obtained from the time-differentiation of the grid equations. Applying the chain rule of differentiation and collecting terms results in the grid speed equations of the form

$$\begin{aligned} G_\tau(x, y) \; &= \; A^* \; \vec{z}_{\xi\xi} \; + \; B^* \; \vec{z}_{\xi\eta} \; + \; C^* \; \vec{z}_{\eta\eta} \\ &+ \; D^* \; \vec{z}_\xi \; + \; E^* \; \vec{z}_\eta \; + \; \vec{T}^* \; = \; 0 \quad (30) \end{aligned}$$

where the grid speed vector is defined as

$$\vec{z}^{\,T} = [\, x_\tau ,\ y_\tau \,] .$$

This involves performing the time differentiations A_τ, B_τ, C_τ, D_τ, and E_τ and factoring out $\vec{z}$. The procedure is straight-forward; however, the algebra is quite involved. The details are presented in reference [4]. One observation is that the time-differentiation of the matrices only result in first-order time derivatives of the grid and so

$$[A^*] = [A], \quad [B^*] = [B], \quad \text{and} \quad [C^*] = [C].$$

The Weighting Function

The weighting function W is defined in the form

$$W = \lambda_0 + \lambda_1 \, |\nabla f(U)| + \lambda_2 \, \exp(\, \lambda_3 \ |\nabla f(U)| \,) . \quad (31)$$

The $\nabla f(U)$ is the gradient with respect to the computational coordinates ξ and η of a function f, such as density, of the physical solution. The function f may be discontinuous, thus, an elliptic smoothing is used to make f a continuous function. The λ_0, λ_1, λ_2, and λ_3 are specified constants which influence the spatial character of W.

The grid equations require the derivatives W_ξ and W_η. The grid speed equations also require the derivatives W_τ, $W_{\xi\tau}$, and $W_{\eta\tau}$. These are computed by applying the chain rule and using second-order spatial and first-order temporal finite-differences.

Solving the Grid and Grid Speed Equations

The grid equations are non-linear partial differential equations while the grid speed equations are linear partial differential equations. This is attractive since linear equations are easier to solve than non-linear equations.

The smoothness terms in the grid equations form a system of elliptic equations. The orthogonality and cell volume adaptation terms form a system of equations which is not elliptic and can produce non-unique solutions. However, the combined system of equations remains fairly elliptic and iterative methods for solving elliptic equations can be applied with success.

The grid and grid speed equations are discretized using second-order, central differences and solved using the Gauss-Seidel point relaxation method.

Neumann boundary conditions are applied to impose orthogonality of the grid lines at the boundaries. The orthogonality condition is

$$\vec{r}_\xi \cdot \vec{r}_\eta = 0. \quad (32)$$

The boundary is parameterized according to the curve-length along the boundary. A local linearization is applied and an expression can be developed for the change in the curve-length coordinate s needed for orthogonality for the iteration. For the $j = 1$ boundary, this expression is

$$(\Delta s)^m_{i,1} = \left[\frac{(\vec{r}_s)_{i,1} \cdot [\, 4\vec{r}_{i,2} - 3\vec{r}_{i,1} - \vec{r}_{i,3} \,]}{3 \ (\vec{r}_s \cdot \vec{r}_s)_{i,1}} \right]^m , \quad (33)$$

where m is the iteration index. Thus

$$s^{m+1} = s^m + \ (\Delta s)^m \quad (34)$$

and the new boundary point grid point is

$$\vec{r}_B^{\,m+1} = \vec{r}_{\text{fit}} \,(\, s^{m+1} \,) \quad (35)$$

where the subscript B denotes a boundary point and $\vec{r}_{\text{fit}}$ is the curve fit of the boundary.

The grid speeds at the boundaries are of the mixed type to impose a prescribed boundary motion and to impose the orthogonality condition at the boundary. The boundary condition for imposing orthogonality is derived from the time differentiation of the grid orthogonality condition

$$\frac{\partial}{\partial \tau} (\, \vec{r}_\xi \cdot \vec{r}_\eta \,) \ = \ 0. \quad (36)$$

Using the parameterization of the boundary and a local linearization, an expression for the the time derivative of the curve-length coordinate can be obtained. For the $j = 1$ boundary, the expression is

$$s_\tau = \frac{\vec{z}_s \cdot (4\vec{r}_{i,2} - 3\vec{r}_{i,1} - \vec{r}_{i,3}) + \vec{r}_s \cdot (4\vec{z}_{i,2} - \vec{z}_{i,3})}{3\,(\vec{r}_s \cdot \vec{r}_s)} . \quad (37)$$

Thus for each iteration, a new s_τ is computed and

$$s^{m+1} = s^m + \ \Delta\tau \ s_\tau \quad (38)$$

and from the curve fit of the boundary,

$$\vec{r}_B^{\,m+1} = \vec{r}_{\text{fit}} \,(\, s^{m+1} \,) . \quad (39)$$

The mixed boundary condition is imposed as

$$\vec{z}_B(t) \ = \ \vec{z}_{B(motion)}(t) + (\, \vec{r}_B^{\,m+1} - \vec{r}_B^{\,m} \,) \ / \, \Delta\tau . \quad (40)$$

Solving the grid speed equations for the grid speeds and then integrating to obtain the grid is expected to generate a grid which satisfies the grid equations. However, small deviations may develop. One way to ensure that the grid continues to satisfy the grid equations is to rewrite the grid speed equations in the form of a first-order homogeneous dynamic system [2] as

$$G_\tau \ + \ \lambda_C \, G \ = \ 0, \quad (41)$$

where λ_C is a prescribed damping constant which damps deviations from the grid equations.

Coupling of the Flow and Grid Equations

The flow equations and the grid speeds are integrated in time using an explicit, two-stage Lax-Wendroff method [5] of the form

$$\Phi^*_{i,j} = \Phi^n_{i,j} + \Delta\tau\ \Psi^n_{i,j}, \qquad (42)$$

$$\Phi^{**}_{i,j} = \Phi^*_{i,j} + \Delta\tau\ \Psi^*_{i,j}, \qquad (43)$$

and

$$\Phi^{n+1}_{i,j} = \frac{1}{2}\ \{\ \Phi^n_{i,j} + \Phi^{**}_{i,j}\ \}. \qquad (44)$$

For the grid, $\Phi = \vec{r}$ and $\Psi = \vec{z}$. For the flow equations, $\Phi = \hat{U}$ and $\Psi = \hat{R}$. The $\hat{U}$ and $\hat{R}$ denote the algebraic vectors of the generalized conservative variables and fluxes defined from a cell-vertex, finite-volume approximation of the unsteady Navier-Stokes equations. Thus,

$$\hat{U} = U\ V, \qquad (45)$$

where V is the area of the planar finite-volume cell and U is the algebraic vector of conservative variables. The $\hat{R}$ is the flux residual defined as

$$\hat{R}_{i,j} = \hat{F}_{i+1/2,j} - \hat{F}_{i-1/2,j} + \hat{F}_{i,j+1/2} - \hat{F}_{i,j-1/2}, \quad (46)$$

where the flux vector for a cell face is defined as

$$\hat{F} = \mathbf{H} \cdot \hat{n} dS. \qquad (47)$$

The $\mathbf{H}$ is the flux dyadic, which for a time-varying control volume is

$$\mathbf{H} = \mathbf{F} - \vec{z}\ U. \qquad (48)$$

The $\mathbf{F}$ is the Cartesian flux dyadic for the two-dimensional, unsteady Navier-Stokes equations. The flow equations are complete with Sutherland's formula, the definition of the Prandtl number, and the assumptions of a perfect gas (air) and laminar viscous flow.

The time step is computed using the CFL condition. The inviscid fluxes are computed using the Roe flux-difference splitting with a TVD limiter as presented in reference [5]. The viscous fluxes are computed using differences and averages computed at the cell faces. Characteristic boundary conditions are used to compute the solution points on the boundaries.

The V needed to decode $\hat{U}$ is computed from the grid conservation law as discussed in references [6] and [7]. The grid conservation law relates the change in volume of the cell to the motion of the cell faces and is derived from the flow integration equations with the assumption of a uniform solution for the conservative variables. The grid conservation law follows the form of equations (42) to (44) with $\Phi = V$ and $\Psi = \hat{Z}$where $\hat{Z}$ is the vector sum of speeds of the cell faces.

The flow and grid equations are coupled through the grid speed terms in the flow equations and the W function in the grid speed equations. The two-stage approach advances the solution and grid in time and space in ideally second-order accuracy. The steps in the explicit, dynamically adaptive grid method using the grid speed equations include:

1) Establish an initial grid $\vec{r}^n$ satisfying the grid equations and a consistent initial solution U^n.
2) Solve the grid speed equations for $\vec{z}^n$.
3) Compute or specify the time step, $\Delta\tau$.
4) Compute $\hat{U}^*$ for the first stage,

$$\hat{U}^*_{i,j} = \hat{U}^n_{i,j} - \Delta\tau\ \hat{R}^n_{i,j}. \qquad (49)$$

5) Solve the grid conservation law,

$$V^*_{i,j} = V^n_{i,j} - \Delta\tau\ \hat{Z}^n_{i,j}. \qquad (50)$$

6) Compute the first-stage flow solution,

$$U^*_{i,j} = \hat{U}^*_{i,j}\ /\ V^*_{i,j}. \qquad (51)$$

7) Compute the flow boundary conditions.
8) Integrate the grid speeds to obtain the grid,

$$\vec{r}^{\,*}_{i,j} = \vec{r}^{\,n}_{i,j} + \Delta\tau\ \vec{z}^{\,n}_{i,j}. \qquad (52)$$

9) Recompute the cell volumes and cell-face areas.
10) Solve the grid speed equations for $\vec{z}^{\,*}$.
11) Compute $\hat{U}^{**}$ for the second stage,

$$\hat{U}^{**}_{i,j} = \hat{U}^*_{i,j} - \Delta\tau\ \hat{R}^*_{i,j}. \qquad (53)$$

12) Compute $\hat{U}^{n+1}_{i,j}$.

$$\hat{U}^{n+1}_{i,j} = \frac{1}{2}\left[\ \hat{U}^n_{i,j} + \hat{U}^{**}_{i,j}\ \right]. \qquad (54)$$

13) Solve the grid conservation law,

$$V^{**}_{i,j} = V^*_{i,j} - \Delta\tau\ \hat{Z}^*_{i,j}. \qquad (55)$$

14) Compute the volume at the $n+1$ time level,

$$V^{n+1}_{i,j} = \frac{1}{2}\ (\ V^n_{i,j} + V^{**}_{i,j}\ ). \qquad (56)$$

15) Compute the flow solution,

$$U^{n+1}_{i,j} = \hat{U}^{n+1}_{i,j}\ /\ V^{n+1}_{i,j}, \qquad (57)$$

16) Compute the flow boundary conditions.
17) Integrate the grid speeds to obtain the grid,

$$\vec{r}^{\,n+1}_{i,j} = \vec{r}^{\,n}_{i,j} + \frac{1}{2}\ \Delta\tau\ (\ \vec{z}^{\,n}_{i,j} + \vec{z}^{\,*}_{i,j}\ ). \qquad (58)$$

18) Recompute the cell volumes and cell-face areas.
19) Solve the grid speed equations for $\vec{z}^{\,n+1}$.
20) Go to step 3 and repeat until the final time.

Computing the grid speeds using a time-difference of the grid requires solving the grid equations at each stage prior to the differencing.

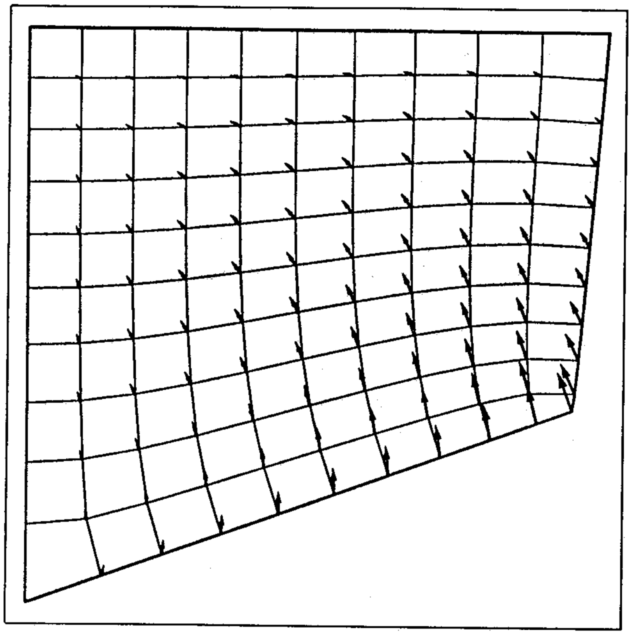

Figure 1: The grid and grid speed vectors for the rotation of the lower boundary of a unit square

Results

A Simple Dynamic Boundary

The dynamically adaptive grid method was applied to a uniform grid on a unit square domain in which the lower boundary was prescribed to rotate at a rate of 20 degrees/second about the lower left corner. This problem tested the ability of the method to compute the grid speeds and integrate them in time to create a grid which continuously satisfied the grid equations, and so, maintain the desired grid qualities.

The time integrations were performed for 50 time steps with a time step of $\Delta\tau = 0.02$ seconds to a final time of 1 second. The bottom boundary rotated 20 degrees about the lower left corner. The grid parameters were $(\lambda_S, \lambda_O, \lambda_A) = (1.0,1.0,1.0)$ and $\lambda_C = 30.0$.

For an iteration tolerance of $1.0x10^{-8}$, the computation required 17308 iterations of the grid speed equations. The average error was 0.0396% and the maximum error was 0.8176%. The errors are the differences in grid point locations between the final grid obtained from the time integration and the grid which satisfied the grid equations at the final time. The errors are expressed as percentages of the length of the uniform grid spacing. Figure 1 shows the grid and grid speed vectors at the final time. The final grid shows that the grid maintained the desired grid qualities, specifically, the orthogonality of the grid lines to the lower boundary.

The effectiveness of the grid control law damping factor, λ_C, is shown in figure 2. The average error was reduced for increased values of λ_C. The fact that the curve bottomed out indicates that one should be able to pick a value of λ_C and obtain a good reduction in error without having to search for an optimum value. However, while the errors decreased, the equations became stiffer, and more iterations of the grid speed equations were required. The effectiveness of λ_C remained consistent with other test problems, and so, these results guided the selection of λ_C for the flow problems.

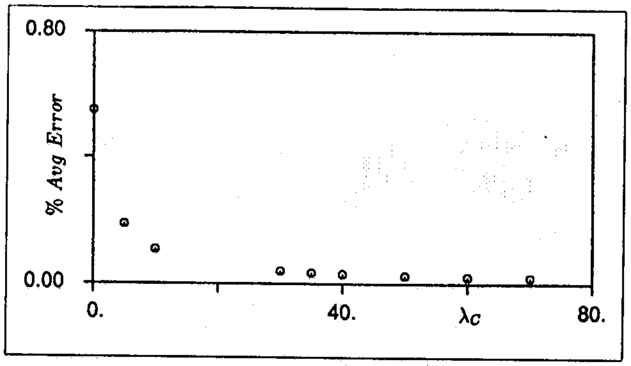

Figure 2: The effectiveness of the λ_C damping factor

The pointwise iteration of the grid speed equations demonstrated excellent convergence characteristics. When the iteration tolerance was increased to $1.0x10^{-4}$, only 304 iterations were required and the average error and maximum error only increased to 0.1065% and 2.8635%, respectively. When the number of iterations per call was limited to 5 iterations, the total number of iterations was 332 and the average and maximum errors were only 0.1023% and 2.0723%, respectively. This provided some guidance for setting the number of iterations and tolerances appropriate for the efficient solution of the grid speed equations.

Inviscid Startup Flow in a Nozzle

The dynamically adaptive grid method was applied to the computation of the inviscid startup flow in a converging-diverging (CD) nozzle. Figure 3(a) shows the shape of the nozzle. The flow was initially stationary at a total pressure of $p_t = 104074.6\ N/m^2$ and a total temperature of $T_t = 252.9\ K$. The exit pressure was then decreased from the total pressure value to a value of $p_{exit} = 84000\ N/m^2$ over a time interval of 10% of a reference time. The reference time was defined by the ratio L/V_{inflow} where L is the length of the nozzle (L = 10 meters) and V_{inflow} is the inflow velocity (V_{inflow} = 75.92 meters/sec). A two-point, cubic spline variation with zero end-slope conditions was used for the time variation of the exit pressure. After the transition time interval, the exit pressure was held fixed. The flow was computed until a final time of 0.3 seconds. All computations were performed on the NASA Lewis Cray XMP.

The grids consisted of 95 grid points in the streamwise direction and 15 grid points in the traverse di-

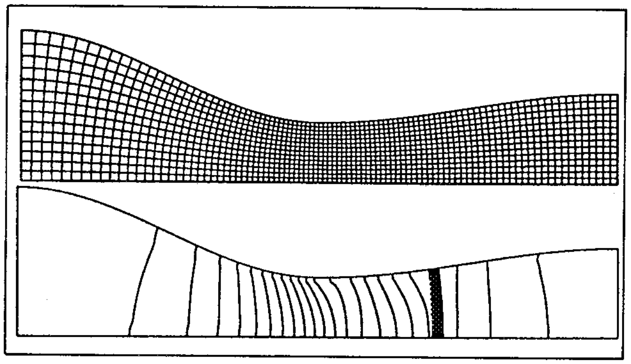

Figure 3: The grid and density contours for the static grid computation for the CD nozzle

rection. The static grid, shown in figure 3(a), was computed using the grid equations with parameters $(\lambda_S, \lambda_O, \lambda_A) = (1.0, 1.0, 0.0)$. The dynamic grids were computed with grid parameters $(\lambda_S, \lambda_O, \lambda_A) = (1.0, 1.0, 0.5)$. The function f used in computing the weighting function W was the density. The grid adaptation parameters were $\lambda_0 = 1.0$ and $\lambda_1 = 500$. Thirty smoothing passes were used to smooth the density solution prior to computing the weighting coefficient W and its derivatives.

The flow solution on the static grid was computed using a CFL number of $\nu = 0.7$ and required 3016 time steps and 111 CPU seconds to converge to a residual of $1.308x10^{-3}$ after a maximum of $6.284x10^{-1}$. Figure 3 shows the grid and density contours at the final time.

The flow solution on the dynamically adaptive grid using time-differenced grid speeds was computed using a CFL number of $\nu = 0.6$ and required 4423 time steps and 563 CPU seconds to converge to a residual of $1.198x10^{-3}$. The grid equations were iterated with a tolerance of $1.0x10^{-4}$ with a maximum of 3 iterations allowed for each stage of the Lax-Wendroff method. One would expect that integrating the grid speeds prior to solving the grid equations would provide a good first guess of the new grid. Experience has shown that for this problem, performing the integrations caused the boundary of the grid to deviate significantly from its true geometry and introduced perturbations into the flowfield.

An instability occurred when applying the inviscid wall boundary conditions for a dynamic grid. The grid points on the boundary are constrained to move along the boundary. Since the boundary conditions act only on information normal to the boundary, the grid speeds do not influence the boundary conditions. However, the numerical implementation of the boundary conditions may result in some influence from the grid point motion. For the dynamic grid computations, the grid speeds were neglected in the numerical boundary condition computations.

The flow solution on the dynamically adaptive grid using the grid speeds computed from the grid speed equations was computed using a CFL number of $\nu = 0.6$ and required 4520 time steps and 944 CPU seconds to converge to a residual of $9.633x10^{-4}$. The grid control law damping factor was set to a value of $\lambda_C = 50.0$. The grid speed equations were iterated with a tolerance of $1.0x10^{-4}$ with a maximum of 3 iterations allowed for each stage of the Lax-Wendroff method. At the start of the computation, the grid speed law required all 3 iterations. As the steady-state flow solution was reached, the residual fell below the tolerance and less iterations were required until only 1 iteration was required per stage. Figure 4 shows the grids and density contours at various times. As can be seen, the resolution of the shock is clearly improved over the computation on the static grid.

The variation of the static pressure in the streamwise direction for all of the computed solutions compared very well with quasi-one-dimensional, inviscid theory. The dynamically adaptive grid solutions show increased resolution of the shock. The TVD limiter eliminated any oscillations near the shock.

The convergence histories were similar for each computation. The convergence history for the dynamic grid computation which used the grid speed equations showed some occasional sharp spikes which may indicate movement of a grid point across the shock.

Using a dynamically adaptive grid required significantly more computational effort than a static grid. The dynamic grid computation using time-differenced grid speeds required 3.46 times more CPU time per step than the static grid computation, while the dynamic grid computation using the grid speed equations required 3.07 times more CPU time. Using the grid speed equations was more efficient for this problem because linear equations are solved rather than non-linear grid equations. Using a time difference required 3% more memory than the static grid computation, while using the grid speed equations required 20% more memory.

The existence of grid-motion-induced errors was examined by comparing the time variation of the static pressure at the location $x = 8.0$ meters in the nozzle for the static grid computation with the dynamic grid computations. As shown in figure 5, there is less error when the grid speed equations are used than when a time difference is used to compute the grid speeds.

Shock/Boundary-Layer Interaction

The dynamically adaptive grid method was applied to the computation of a viscous flow in which an oblique shock impinges on a laminar boundary layer along a flat plate at $x = 1.0$ meters and causes a small flow separation region. This problem was investigated

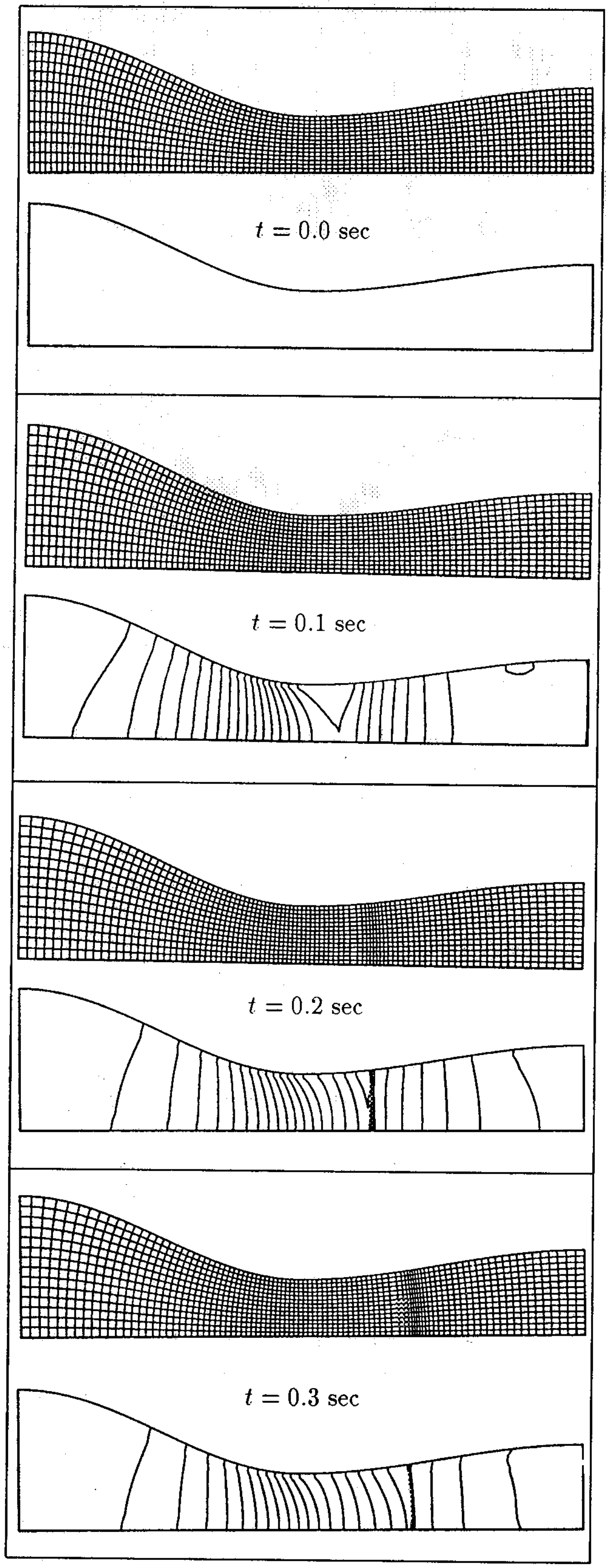

Figure 4: The grids and density contours at various times for the dynamically adaptive grid

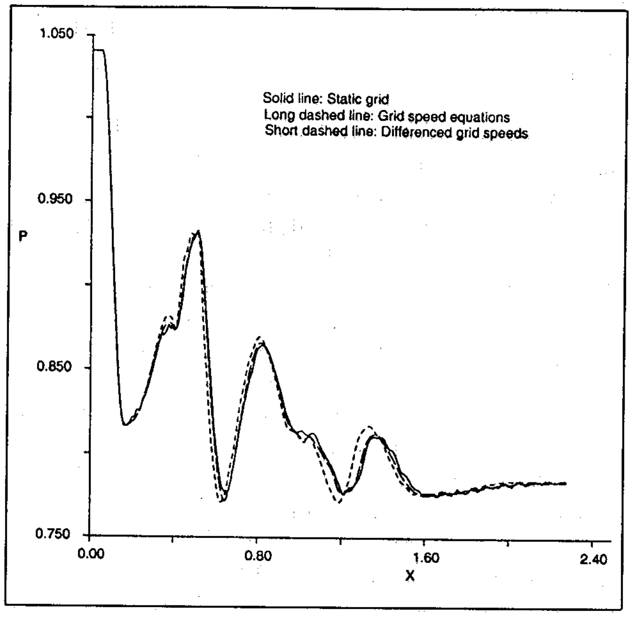

Figure 5: The time history of the pressure at the location $x = 8.0$ for the startup of the CD nozzle

experimentally by Hakkinen et al. [8] and is used extensively to validate CFD methods. The freestream conditions prior to the shock are a Mach number of 2.0, a Reynolds number of $2.96x10^5$ based on a length of 1.0 meters, and a shock angle of 32.585 degrees.

The flow domain extends from $x = -0.2$ to $x = 2.0$ meters and from $y = 0$ to $y = 1$ meters. The grids for the flow computations followed the suggestions of Liou [9] based on triple-deck theory. The grids consisted of 75 streamwise and 65 transverse grid points. The initial solutions included the incident and reflected oblique shocks whose properties were computed from shock theory. The flow solutions were computed without TVD limiters. The velocity at the plate was decreased during the flow computation from the inviscid flow velocity to zero velocity over a time interval of $3.95x10^{-3}$ seconds.

The static grid was generated using an interactive grid generator with a normal grid spacing at the plate of $1.6x10^{-4}$ meters and clustering to increase resolution along the inflow boundary and plate. The grid is shown in figure 6. The flow solution on the static grid was computed using a CFL number of $\nu = 0.8$ and required 90000 time steps and 11519 CPU seconds to converge to a residual of $9.8x10^{-5}$ after reaching a maximum of 5.0. Figure 6 shows the pressure contours for the steady-state solution.

The dynamic grids were computed with grid parameters $(\lambda_S, \lambda_O, \lambda_A) = (1.0, 1.0, 1.0)$. The function f used in computing the weighting function W was the Mach number. The grid adaptation parameters were $\lambda_0 = 1.0$ and $\lambda_1 = 500$. Thirty smoothing passes were used to smooth the Mach number prior to computing

W and its derivatives. The initial grids were obtained by solving the grid equations for the given adaptation parameters with the flow solution being defined by inviscid, oblique shock theory. Since no boundary layer present, there was no clustering normal to the plate; the clustering developed as the boundary layer developed. Figure 7(a) shows the initial grid.

The flow solution on the dynamically adaptive grid using the time-differenced grid speeds was computed using a CFL number of $\nu = 0.7$ and required 6658 time steps and 2118 CPU seconds to converge to a residual of $1.0x10^{-4}$. The CPU time per step averaged 2.49 times the time required for the static grid.

The flow solution on the dynamically adaptive grid using the grid speed equations used a CFL number of $\nu = 0.7$ and required 8392 time steps and 2703 CPU seconds to converge to a residual of $1.0x10^{-4}$. The grid control law damping factor was set to a value of $\lambda_C = 50.0$. The iterations of the grid speed equations were limited to 10 iterations per stage and the grid residual tolerance was set to $5.0x10^{-4}$. At the start of the computation, all 10 iterations were required. As the steady-state flow solution was reached, the residual fell below the tolerance and less iterations were required until only 1 iteration was required per stage. The CPU time per step averaged 2.52 times the time required for the static grid. Figure 7(b) shows the adapted grid at the final time and figure 7(c) shows the pressure contours.

The adapted grid seems to have improved the resolution of the shocks. Figure 8 shows the comparison of the pressure coefficients along the plate. Figure 9 shows the comparison of the skin friction coefficient along the plate. The static grid results compare well to the experimental data and are consistent with numerical results of other researchers. The dynamic grid predicted a smaller separation region. This is due to the improper resolution of the boundary layer. The spacing of the dynamic grid next to the plate was approximately $1.6x10^{-3}$ meters. At $x = 1.0$ meters, this corresponds to $|u/u_\infty| = 0.038$, compared to $|u/u_\infty| = 0.0025$ for the static grid. The u_∞ is the velocity just outside the boundary layer.

Obtaining a smaller grid spacing normal to the plate for the dynamically adaptive grid method relies on emphasizing adaptation over smoothness. This involves increasing the values of λ_A, λ_1, λ_2, and λ_3. Attempts to reduce the spacing have yet been unsuccessful and is under further study.

Conclusions

An explicit, dynamically adaptive grid method has been presented and demonstrated for two-dimensional problems. The dynamic grids maintained the desired grid qualities while allowing grid motion due to motion of the boundaries and to adaptation of the grid to solution gradients. The results suggest that using the grid speed equations to compute the grid speeds reduces the grid-motion-induced errors more than using a backwards time-difference of the grid. The grid adaptation improved the shock resolution; however, the resolution of boundary layers needs to be improved. Using a dynamically adaptive grid required significantly more computational effort than using a static grid. The problems presented attempted to demonstrate the approach, but static grids are probably adequate for these problems. Future efforts will focus on applying the method to truly unsteady flow problems and problems involving boundary motion. The method is applicable to three-dimensional problems and development has begun.

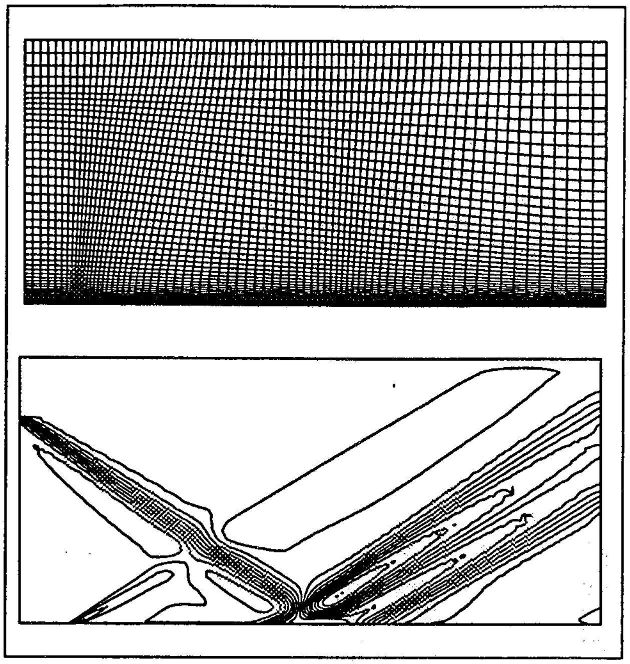

Figure 6: The grid and pressure contours for the shock/boundary-layer interaction computation on the static grid

Acknowledgments

This work was supported through the NASA Graduate Student Researchers Program through the NASA Lewis Research Center, contract number NGT-50441, and was part of the doctoral dissertation of John Slater through Iowa State University. John Slater is currently a NRC Research Associate at the NASA Lewis Research Center.

References

[1] Hindman, R.G., P. Kutler, and D.A. Anderson. "Two-Dimensional Unsteady Euler-Equation Solver for Arbitrarily Shaped Flow Regions." AIAA Journal 19 (1981): 424-431.

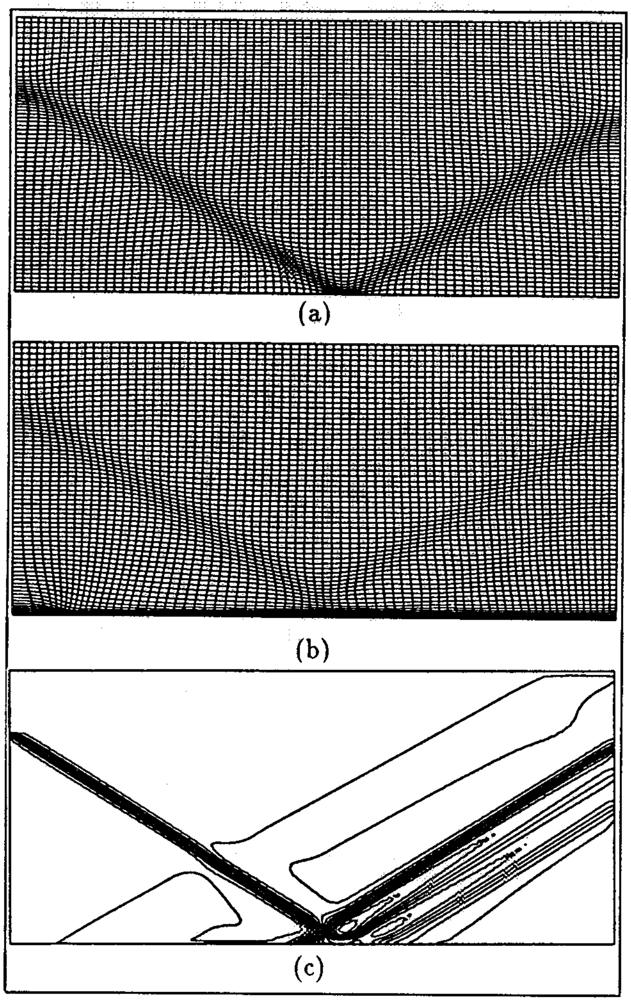

Figure 7: The grids and shock/boundary-layer interaction computation on the dynamic grid: (a) initial grid; (b) final grid; (c) pressure contours

[2] Holcomb, J.E. and R.G. Hindman. "Development of a Dynamically Adaptive Grid Method for Multidimensional Problems." AIAA Paper 84-1668, June 25-27, 1984.

[3] Brackbill, J.U. and J.S. Saltzman. "Adaptive Zoning for Singular Problems in Two Dimensions." Journal of Computational Fluid Dynamics 46 (1982): 342-368.

[4] Slater, J.W., "A Dynamically Adaptive Mesh Method for Internal Flows," Ph.D. Dissertation, Iowa State University, Ames, Iowa 1992.

[5] Liou, M.-S. and A.T. Hsu. "A Time-Accurate Finite Volume High Resolution Scheme for Three Dimensional Navier-Stokes Equations." AIAA Paper 89-1994, June 13-15, 1989.

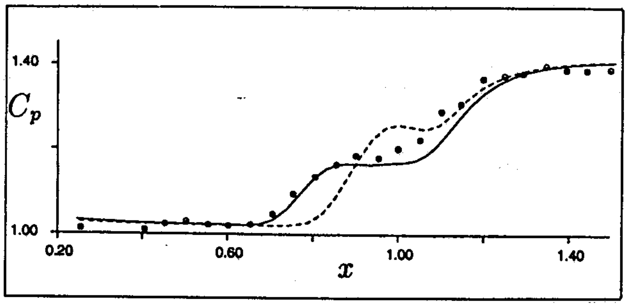

Figure 8: The pressure coefficients along the plate for the shock/boundary-layer interaction problem: (circles) experimental data; (solid line) static grid; (dashed line) dynamically adapted grid

[6] Thomas, P.D. and C.K. Lombard. "Geometric Conservation Law and Its Application to Flow Computations on Moving Grids." AIAA Journal 17 (1979): 1030-1037.

[7] Hindman, R.G. "Generalized Coordinate Forms of Governing Fluid Equations and Associated Geometrically Induced Errors." AIAA Journal 20 (1982): 1359-1367.

[8] Hakkinen, R.J., I Greber, L. Trilling, and S.S. Abarbanel, "The Interaction of an Oblique Shock Wave with a Laminar Boundary Layer," NASA Memo 2-18-59W, 1959.

[9] Liou, M-S. "A Newton/Upwind Method and Numerical Study of Shock Wave/Boundary Layer Interactions." International Journal for Numerical Methods in Fluids 9 (1989): 747-761.

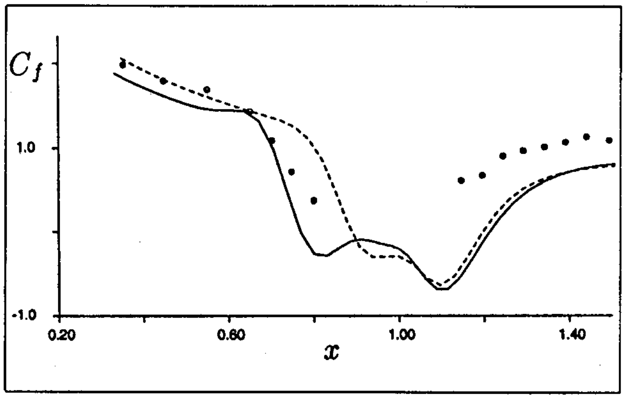

Figure 9: The skin friction coefficients along the plate for the shock/boundary-layer interaction problem: (circles) experimental data; (solid line) static grid; (dashed line) dynamically adapted grid

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE November 1994	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE

An Approach for Dynamic Grids

5. FUNDING NUMBERS

WU–505–62–52

6. AUTHOR(S)

John W. Slater, Meng-Sing Liou, and Richard G. Hindman

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–9124–1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM–106774
AIAA–94–0319

11. SUPPLEMENTARY NOTES

Prepared for the 32nd Aerospace Sciences Meeting and Exhibit sponsored by the American Institute of Aeronautics and Astronautics, Reno, Nevada, January 10–13, 1994. John W. Slater and Ming-Sing Liou, NASA Lewis Research Center and Richard G. Hindman, Iowa State University of Science and Technology, Ames, Iowa 50010–1858. Responsible person, John W. Slater, organization code 2610, (216) 433–8513.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 02

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

An approach is presented for the generation of two-dimensional, structured, dynamic grids. The grid motion may be due to the motion of the boundaries of the computational domain or to the adaptation of the grid to the transient, physical solution. A time-dependent grid is computed through the time integration of the grid speeds which are computed from a system of grid speed equations. The grid speed equations are derived from the time-differentiation of the grid equations so as to ensure that the dynamic grid maintains the desired qualities of the static grid. The grid equations are the Euler-Lagrange equations derived from a variational statement for the grid. The dynamic grid method is demonstrated for a model problem involving boundary motion, an inviscid flow in a converging-diverging nozzle during startup, and a viscous flow over a flat plate with an impinging shock wave. It is shown that the approach is more accurate for transient flows than an approach in which the grid speeds are computed using a finite difference with respect to time of the grid. However, the approach requires significantly more computational effort.

14. SUBJECT TERMS

Computational grids; Computational fluid dynamics; Navier-Stokes equation

15. NUMBER OF PAGES

12

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT