JOURNAL OF PROPULSION AND POWER
Vol. 10, No. 4, July–Aug. 1994

Solution-Adaptive Structured-Unstructured Grid Method for Unsteady Turbomachinery Analysis, Part I: Methodology

Sanjay R. Mathur*
Iowa State University, Ames, Iowa 50011
Nateri K. Madavan†
MCAT Institute, NASA Ames Research Center, Moffett Field, California 94035
and
R. Ganesh Rajagopalan‡
Iowa State University, Ames, Iowa 50011

A solution-adaptive method for the time-accurate analysis of two-dimensional flows in multistage turbomachinery is presented. The method employs a hybrid structured-unstructured zonal grid topology in conjunction with appropriate modeling equations and solution techniques in each zone, thus combining the advantages of both structured and unstructured grid methods. The viscous flow region in the immediate vicinity of the airfoils is resolved on structured O-type grids, while the rest of the domain is discretized using an unstructured mesh of triangular cells. In the inner regions, the Navier-Stokes equations are solved using an implicit, third-order accurate, upwind-biased scheme. The use of both central difference and upwind schemes is explored for the solution of the Euler equations in the outer regions. An efficient and robust grid adaptation strategy, including both grid refinement and coarsening capabilities, is developed for the unstructured grid regions. Methodologies for the accurate, conservative transfer of information at the interface between the structured and unstructured domains, as well as that between two unstructured grids in relative motion, are also developed. For generality, three-dimensional effects of stream-tube contraction are modeled. The numerical methodology is presented in detail in the present article (Part I). Results obtained using this method and comparisons of these results with experimental data and earlier structured-grid based methods are presented in a companion article (Part II).

Introduction

THE numerical simulation of turbomachinery fluid dynamics is a challenging problem that involves unsteady flow phenomena in complex geometries. The problem has received the attention of computational fluid dynamics researchers ever since the emergence of the discipline, and several techniques have been applied towards its solution. References 1–10 represent a brief, but by no means exhaustive, list of relevant work in unsteady turbomachinery flow prediction. More general and comprehensive reviews of computational methods for turbomachinery flow analysis are also available.[11]

The relative motion between the stationary and rotating rows of a turbomachine makes domain discretization using a single grid rather impractical. Most methods and associated computer codes in use today employ structured grids in combination with some type of zonal strategy for computing such flows.[4,10,12,13] The zonal strategy provides a convenient way to treat complex flow domains. The grids in the individual regions have only to conform to the local geometry and can thus be generated with relative ease. Zonal methods also offer flexibility in selecting the optimal governing equations and solution algorithms in each domain, based on the local flow characteristics. However, the division of the flow domain into multiple zones has some drawbacks. Artificial zone boundaries are introduced where appropriate boundary conditions are required to ensure accurate information exchange between the zones. The stability of the overall algorithm requires proper overlapping and patching of grids, and calls for an appreciable degree of user judgment and experience during the initial grid generation stage. In addition, the large overlap between the inner and outer grids that is sometimes necessitated is computationally inefficient, and detracts from the ability to refine grids elsewhere.

Unstructured grid-based methods have gained popularity in recent years as an alternative approach for analyzing flows in complex geometries. These methods offer two main advantages over traditional structured grid-based approaches. First, by not requiring the grids to be logically rectangular, they offer considerable flexibility in discretizing complex domains without resorting to large-scale zoning. General-purpose grid generation techniques can be used and the process can be automated easily. Second, unstructured grids offer control of grid density and stretching in a more flexible manner and permit optimal use of available grid points. This capability becomes even more powerful when used in a dynamic, adaptive setting to resolve flow features evolving in time and space. Unstructured-grid based methods are gaining popularity because of such advantages, but there are several issues that still need to be resolved. These methods usually employ explicit time-stepping and have been used quite successfully for solutions of the Euler equations.[14] However, the small grid spacings that are required to resolve boundary-layer features result in rather severe time-step restrictions when explicit schemes are used for viscous calculations. Progress has been reported recently[15,16] on the development of Navier-Stokes and implicit methods for unstructured grids, but it is somewhat unclear whether these techniques will be competitive with structured-grid based methods. Other concerns in-

Presented as Paper 93-0387 at the AIAA 31st Aerospace Sciences Meeting, Reno, NV, Jan. 11–14, 1993; received April 23, 1993; revision received Aug. 23, 1993; accepted for publication Aug. 23, 1993.

*Graduate Student, Department of Aerospace Engineering and Engineering Mechanics; currently Engineer, Fluent Inc., 10 Cavendish Court, Lebanon, NH 03766. Member AIAA.
†Senior Research Scientist, M/S T047-1. Senior Member AIAA.
‡Associate Professor, Department of Aerospace Engineering and Engineering Mechanics. Member AIAA.

clude the effect on solution accuracy of the highly stretched grids typically used in viscous regions, and the development of higher-order accurate schemes on unstructured grids. Implementation of some simple turbulence models and surface boundary condition procedures also becomes complicated for unstructured grids where the grid lines are not orthogonal to the body surfaces.

The relative ease with which solution adaptive capabilities can be built into the framework of unstructured grids in comparison to structured grids makes the former particularly attractive for unsteady turbomachinery applications. In such applications, accurate treatment of the wakes is important since the unsteady flowfield can depend to a large extent on the interaction of these wakes with downstream airfoils. Numerical diffusion inherent in most computational procedures can cause the wakes to be distorted. One remedy for reducing this effect is to adaptively refine the grid in the vicinity of the wakes; however, this is somewhat difficult to achieve in a structured grid setting. In zonal methods, the wakes are subjected to additional numerical diffusion and distortion as they convect through a series of grids with differing topologies. These problems become even more critical in multistage configurations where the interactions are potentially more complex and the wakes traverse through several zones.

The present method combines the advantages of both unstructured and structured grid methodologies in a hybrid procedure for the analysis of unsteady turbomachinery flows. The method is implemented by generalizing the zonal framework of Ref. 10 to include both structured and unstructured grid domains. The region in the immediate vicinity of the airfoils (inner region) is discretized using structured grids, while the rest of the domain (outer region) is discretized using an unstructured triangular grid. In the viscous inner regions, the Navier-Stokes equations are solved using an implicit, third-order accurate, upwind-biased scheme. In the inviscid outer region, Euler equations are solved using either a central difference scheme or an explicit upwind scheme that incorporates a linear reconstruction procedure. The solution in the outer unstructured region is advanced in time explicitly using the same time-step values as for the structured regions which are time-advanced in an implicit manner. An efficient and robust grid adaptation strategy with both grid refinement and coarsening capabilities is also used for the unstructured grid. A similar hybrid-grid approach for turbomachinery computations has been reported recently in Ref. 13. The present method, in addition to its solution-adaptive capability, is capable of treating multistage turbomachinery configurations.

The following sections describe the domain decomposition, grid generation, inner and outer grid solution procedures, grid adaptation strategy, and the various boundary conditions used. Computed results and comparisons of these results with experimental data and earlier structured-grid based methods are presented in a companion article (Part II).

Domain Decomposition and Grid Generation

The domain decomposition strategy adopted is illustrated in Fig. 1 for a single-stage axial turbine configuration. The figure shows a row of stator airfoils adjacent to a row of rotor airfoils. The inner zones occupy the immediate vicinity of the airfoils. There is one outer zone for each airfoil row in which all the inner zones corresponding to that airfoil row are embedded. Structured grids are used in the inner zones, while unstructured grids are used in the outer zones. Multistage configurations with several adjacent rows of airfoils are accommodated within the same framework.

In the inner zones O-type grids are used. These are generated easily since the location of outer boundary of the grid is not a major constraint. This outer boundary is located at a specified distance normal to the airfoil surface. The remainder of the geometrically complex domain is discretized using an unstructured triangular mesh. The outer boundaries of the inner grids form the inner boundaries of the outer

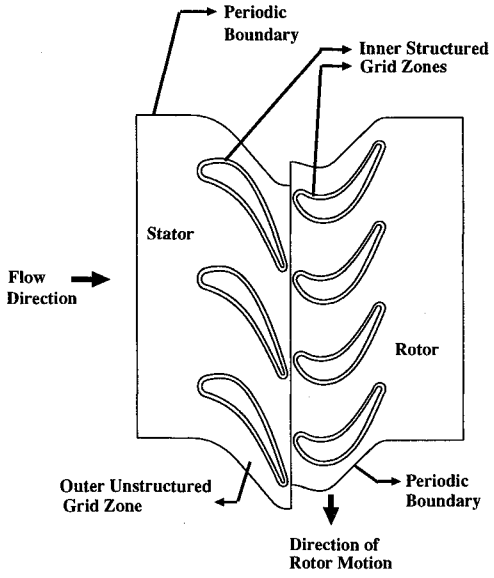

Fig. 1 Typical zonal decomposition.

domain. Unlike the approach of Ref. 10, this outer region for each airfoil row is not discretized into identical grids, but is retained as one multiply-connected domain with several "holes." The distribution of grid points on the inner hole boundaries is already known from the inner grids; on all other boundaries it is prescribed in accordance with the expected flow features and the desired clustering. The domain is then triangulated using an advancing-front technique that produces a smooth grid with point clustering being controlled by the specified boundary points. The technique allows for control of the magnitude as well as directionality of the stretching. Inlet and exit boundary placement is accomplished by extending the domain for the first and last airfoil rows by a few chord lengths upstream and downstream, respectively. The boundary between adjacent airfoil rows is located midway between the two rows.

Inner Grid Solution Procedure

The Navier-Stokes equations, including the stream-tube contraction terms,[17] form the set of governing equations in the inner regions. The thin-layer approximation is introduced and the resulting equations in generalized coordinates are given as

$$\frac{\partial \tilde{Q}}{\partial \tau} + \frac{\partial \tilde{E}}{\partial \xi} + \frac{\partial \tilde{F}}{\partial \eta} = Re^{-1}\frac{\partial \tilde{F}_v}{\partial \eta} + S \tag{1}$$

where

$$\tilde{Q} = Q/J$$

$$\tilde{E} = \left(\frac{\partial \xi}{\partial t}Q + \frac{\partial \xi}{\partial x}E + \frac{\partial \xi}{\partial y}F\right)\Big/J$$

$$\tilde{F} = \left(\frac{\partial \eta}{\partial t}Q + \frac{\partial \eta}{\partial x}E + \frac{\partial \eta}{\partial y}F\right)\Big/J$$

$$\tilde{F}_v = \left(\frac{\partial \eta}{\partial x}E_v + \frac{\partial \eta}{\partial y}F_v\right)\Big/J$$

$$J = \frac{\partial \xi}{\partial x}\frac{\partial \eta}{\partial y} - \frac{\partial \eta}{\partial x}\frac{\partial \xi}{\partial y}$$

where $[(\partial\xi/\partial t), (\partial\xi/\partial x), (\partial\xi/\partial y)]$ and $[(\partial\eta/\partial t), (\partial\eta/\partial x), (\partial\eta/\partial y)]$ are the metrics, and J the Jacobian of the transformation from Cartesian to body fitting coordinates, such that the ξ direction wraps around the airfoil and the η direction is normal to the

surface. Re and Pr denote the nondimensional parameters Reynolds number and Prandtl number, respectively. $\boldsymbol{Q}$ is the vector of the conserved flow variables, and $\boldsymbol{E}$ and $\boldsymbol{F}$ are the inviscid fluxes. In terms of the primitive variables they can be written as

$$\boldsymbol{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \quad \boldsymbol{E} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (e + p)u \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (e + p)v \end{bmatrix} \tag{2}$$

where $\boldsymbol{E}_v$ and $\boldsymbol{F}_v$ are the viscous fluxes

$$\boldsymbol{E}_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \beta_x \end{bmatrix}, \quad \boldsymbol{F}_v = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \beta_y \end{bmatrix}$$

with

$$\tau_{xx} = 2\mu \frac{\partial u}{\partial x} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) - \left[\frac{2}{3}\mu \frac{u}{h}\frac{dh}{dx} \right]$$

$$\tau_{xy} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right)$$

$$\tau_{yy} = 2\mu \frac{\partial v}{\partial y} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) - \left[\frac{2}{3}\mu \frac{u}{h}\frac{dh}{dx} \right]$$

$$\beta_x = u\tau_{xx} + v\tau_{xy} + \gamma\mu Pr^{-1} \frac{\partial e_{\text{int}}}{\partial x}$$

$$\beta_y = u\tau_{xy} + v\tau_{yy} + \gamma\mu Pr^{-1} \frac{\partial e_{\text{int}}}{\partial y}$$

$$e_{\text{int}} = \frac{e}{\rho} - \frac{1}{2}(u^2 + v^2)$$

The inclusion of stream-tube contraction terms modifies the viscous stress terms as shown in square brackets above, and also introduces the source term $\boldsymbol{S}$ given by

$$\boldsymbol{S} = \frac{1}{h}\frac{dh}{dx}(Re^{-1}\boldsymbol{E}_v - \boldsymbol{E}) + \frac{1}{h}\frac{dh}{dx}\begin{bmatrix} 0 \\ p - \tilde{\tau} \\ 0 \\ 0 \end{bmatrix} \tag{3}$$

where

$$\tilde{\tau} = 2\mu \frac{u}{h}\frac{dh}{dx} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{u}{h}\frac{dh}{dx} + \frac{\partial v}{\partial y} \right)$$

where h is the normalized area of the stream tube and is assumed to be a known function of x. Note that for constant h, the above equations reduce to the standard two-dimensional thin-layer Navier-Stokes equations.

The governing equations are discretized to give the following factored, iterative, implicit algorithm:

$$\left[r\boldsymbol{I} + \frac{1}{r\Delta\xi}(\nabla_\xi \boldsymbol{A}^+_{i,j} + \Delta_\xi \boldsymbol{A}^-_{i,j}) \right] \times \left[r\boldsymbol{I} + \frac{1}{r\Delta\eta}(\nabla_\eta \boldsymbol{B}^+_{i,j} + \Delta_\eta \boldsymbol{B}^-_{i,j} - Re^{-1}\delta_\eta \boldsymbol{M}) \right]^p \times (\tilde{\boldsymbol{Q}}^{p+1}_{i,j} - \tilde{\boldsymbol{Q}}^p_{i,j}) = -\boldsymbol{R}^p_{i,j} \tag{4}$$

where $r = (\frac{3}{2}\Delta\tau)^{1/2}$ and

$$\boldsymbol{A}^\pm = \left(\frac{\partial \tilde{\boldsymbol{E}}}{\partial \tilde{\boldsymbol{Q}}} \right)^\pm$$

$$\boldsymbol{B}^\pm = \left(\frac{\partial \tilde{\boldsymbol{F}}}{\partial \tilde{\boldsymbol{Q}}} \right)^\pm \tag{5}$$

$$\boldsymbol{M} = \left(\frac{\partial \tilde{\boldsymbol{F}}_v}{\partial \tilde{\boldsymbol{Q}}} \right)$$

and Δ, ∇, and δ are the forward, backward, and central difference operators, respectively. The residual R^p is given by

$$\boldsymbol{R}^p = \frac{3\tilde{\boldsymbol{Q}}^p - 4\tilde{\boldsymbol{Q}}^n + \tilde{\boldsymbol{Q}}^{n-1}}{2\Delta\tau} + \frac{\partial \tilde{\boldsymbol{E}}^p}{\partial\xi} + \frac{\partial \tilde{\boldsymbol{F}}^p}{\partial\eta} - Re^{-1}\frac{\partial \tilde{\boldsymbol{F}}^p_v}{\partial\eta} - \boldsymbol{S}^p$$

The temporal integration scheme employs a Newton-type iterative process. In Eq. (4), $\tilde{\boldsymbol{Q}}^p$ is an approximation to the next time-step value, $\tilde{\boldsymbol{Q}}^{n+1}$. At the beginning of the iteration process, $p = 0$, $\tilde{\boldsymbol{Q}}^p = \tilde{\boldsymbol{Q}}^n$, and at convergence, $\tilde{\boldsymbol{Q}}^p = \tilde{\boldsymbol{Q}}^{n+1}$. In this manner, the linearization and factorization errors can be driven to zero at each time step. Typically, three to four such iterations are sufficient to reduce the residuals by two orders of magnitude.

The inviscid flux terms of the residual are discretized using a third-order accurate, upwind-biased scheme. It will suffice to consider the term $(\partial\tilde{\boldsymbol{E}}/\partial\xi)$

$$\frac{\partial \tilde{\boldsymbol{E}}}{\partial\xi} = (\hat{\boldsymbol{E}}_{i+1/2} - \hat{\boldsymbol{E}}_{i-1/2}) + \frac{1}{6}(-\Delta\tilde{\boldsymbol{E}}^-_{i+1,j+2} - \Delta\tilde{\boldsymbol{E}}^-_{i,j+1} + 2\Delta\tilde{\boldsymbol{E}}^-_{i-1,j} + 2\Delta\tilde{\boldsymbol{E}}^+_{i,j+1} - \Delta\tilde{\boldsymbol{E}}^+_{i-1,j} - \Delta\tilde{\boldsymbol{E}}^+_{i-2,j-1})$$

where

$$\hat{\boldsymbol{E}}_{i+1/2} \equiv \tilde{\boldsymbol{E}}_{i+1} - \Delta\tilde{\boldsymbol{E}}^+_{i,j+1} \tag{6}$$

The fluxes $\Delta\tilde{\boldsymbol{E}}^\pm$ can be calculated in many different ways, and in the present study they are evaluated based on Roe's scheme[18]

$$\Delta\tilde{\boldsymbol{E}}^\pm_{i,j+1} = \boldsymbol{A}^\pm(\tilde{\boldsymbol{Q}}_{i+1/2}, \xi_t, \xi_x, \xi_y)(\tilde{\boldsymbol{Q}}_{i+1} - \tilde{\boldsymbol{Q}}_i)$$

where the intermediate state $\tilde{\boldsymbol{Q}}_{i+1/2}$ at which the Jacobian $\boldsymbol{A}$ is evaluated is obtained by density weighted averaging as follows:

$$u_{i+1/2} = \frac{u_i\sqrt{\rho_i} + u_{i+1}\sqrt{\rho_{i+1}}}{\sqrt{\rho_i} + \sqrt{\rho_{i+1}}}$$

$$v_{i+1/2} = \frac{v_i\sqrt{\rho_i} + v_{i+1}\sqrt{\rho_{i+1}}}{\sqrt{\rho_i} + \sqrt{\rho_{i+1}}}$$

$$h_{i+1/2} = \frac{h_i\sqrt{\rho_i} + h_{i+1}\sqrt{\rho_{i+1}}}{\sqrt{\rho_i} + \sqrt{\rho_{i+1}}}$$

and the metric terms are averaged as

$$(\xi_t/J)_{i+1/2} = \tfrac{1}{2}[(x_\eta y_\tau - y_\eta x_\tau)_{i+1} + (x_\eta y_\tau - y_\eta x_\tau)_i]$$

$$(\xi_x/J)_{i+1/2} = \tfrac{1}{2}[(y_\eta)_{i+1} + (y_\eta)_i]$$

$$(\xi_y/J)_{i+1/2} = -\tfrac{1}{2}[(x_\eta)_{i+1} + (x_\eta)_i]$$

The viscous flux, $(\partial\tilde{\boldsymbol{F}}_v/\partial\eta)$ is evaluated using central differences as

$$\left(\frac{\partial \tilde{\boldsymbol{F}}_v}{\partial\eta} \right)_j = \tilde{\boldsymbol{F}}_v[\tilde{\boldsymbol{Q}}_{j+1/2}, (\tilde{\boldsymbol{Q}}_\eta)_{j+1/2}] - \tilde{\boldsymbol{F}}_v[\tilde{\boldsymbol{Q}}_{j-1/2}, (\tilde{\boldsymbol{Q}}_\eta)_{j-1/2}]$$

where

$$\tilde{Q}_{j+1/2} = \tfrac{1}{2}(\tilde{Q}_{j+1} + \tilde{Q}_j)$$

$$(\tilde{Q}_\eta)_{j+1/2} = \tilde{Q}_{j+1} - \tilde{Q}_j$$

Outer Grid Solution Procedure

The Euler equations chosen as the governing equations in the outer region are given as

$$\frac{\partial \tilde{Q}}{\partial \tau} + \frac{\partial \tilde{E}}{\partial \xi} + \frac{\partial \tilde{F}}{\partial \eta} = S_i$$

where S_i represents the inviscid terms of Eq. (3), and (ξ, η) are the Cartesian coordinates fixed to the body. The equations are spatially discretized using a finite volume technique. Integrating the Euler equations over a control volume Ω yields

$$\frac{\partial}{\partial \tau}\int\int_\Omega \tilde{Q}\, d\xi\, d\eta + \int_{\partial\Omega} (\tilde{E}\, d\eta - \tilde{F}\, d\xi) = \int\int_\Omega S_i\, d\xi\, d\eta \quad (7)$$

Various spatial discretization schemes are distinguished by the choice of the control volume and the manner in which the boundary integral is evaluated. The spatial discretization results in a set of coupled, ordinary differential equations that are advanced in time using a second-order accurate, four-step Runge-Kutta scheme. The same time-step values as in the inner zones are used. Two different spatial discretization schemes were used in the current work as discussed below.

Central Difference Scheme

The first method is a "vertex-based" scheme in which the polygon surrounding each grid point is chosen as the control volume Ω (see Fig. 2). The boundary integral in Eq. (7) is approximated by adding together the contribution of each side of the control volume. The flux along each edge can be evaluated by either averaging the endpoint Q first, or by averaging the endpoint fluxes. The latter approach is used here. For example, the contribution of the edge i connecting the points A and B in Fig. 2 to the integral at point C is given by

$$\tfrac{1}{2}(\tilde{E}_A + \tilde{E}_b)\Delta\eta - \tfrac{1}{2}(\tilde{F}_A + \tilde{F}_B)\Delta\xi$$

where $\Delta\xi$ and $\Delta\eta$ are the increments in the ξ and η directions, respectively, along the edge i. The edge i also contributes the same flux term, with a negative sign, to the integral at point D.

This procedure is analogous to a central difference scheme on a structured grid, and as with any central differenced scheme, artificial dissipation is required to stabilize the solution. A combination of fourth and second differences of the conserved

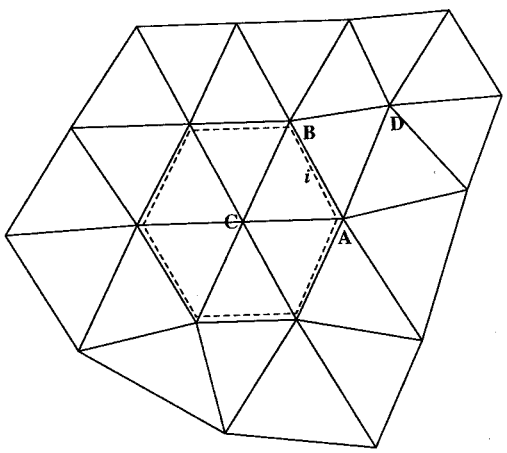

Fig. 2 Control volume for the central difference scheme.

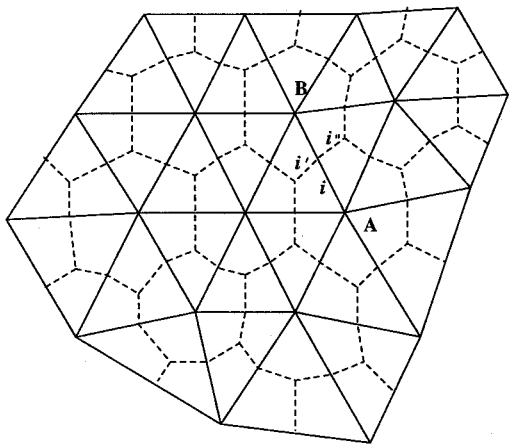

Fig. 3 Control volume for the upwind scheme.

variables is added to the numerical fluxes, following the procedure of Mavriplis.[14]

Upwind Scheme

Upwind schemes can be constructed on unstructured grids by considering a control volume made up from a dual grid. Dual grids can be constructed geometrically by connecting the centroids or circumcenters of the triangles, but not all of them are good candidates for use as control volumes. In the present study, the control volume around a point is made up of line segments connecting the centroids of the triangles to the midpoints of the edges emanating from the point, as shown in Fig. 3. Therefore, each edge of the original grid has two corresponding edges on the dual grid, the contribution of which to the flux integral is evaluated using the Roe flux scheme. For example, the edge i, directed from point A to B in Fig. 3, has two corresponding dual edges, i' and i''. The contribution of a dual edge to the integral at point A is given by $\hat{E}\Delta s$, where $\hat{E}$ is defined by Eq. (6) and represents the flux normal to the edge, and Δs is the length of the edge. $\tilde{Q}_A$ and $\tilde{Q}_B$ are the left and right states used in evaluation of the Jacobian A [Eq. (5)], and the metric terms are

$$\left(\frac{\xi_t}{J}\right) = -\frac{1}{J}(x_\tau \xi_x + y_\tau \xi_y)$$

$$\left(\frac{\xi_x}{J}\right) = \frac{\Delta\eta}{\Delta s}$$

$$\left(\frac{\xi_y}{J}\right) = \frac{-\Delta\xi}{\Delta s}$$

where $\Delta\xi$ and $\Delta\eta$ are ξ and η increments along the edge.

The scheme outlined above is spatially first-order accurate. To improve accuracy, a linear reconstruction procedure, proposed in Refs. 19 and 20, is used whereby instead of assuming $\tilde{Q}_A$ and $\tilde{Q}_B$ as the left and right states for use in the Roe flux evaluation, $\tilde{Q}^+$ and $\tilde{Q}^-$ are obtained as

$$\tilde{Q}^- = \tilde{Q}_A + (\nabla\tilde{Q})_A \cdot r_1$$

$$\tilde{Q}^+ = \tilde{Q}_B + (\nabla\tilde{Q})_B \cdot r_2$$

where r_1 and r_2 are vectors directed from points A and B, respectively, to the midpoint of the dual edge. The components of the gradient of $\tilde{Q}$ are evaluated using the Gauss divergence theorem

$$\tilde{Q}_\xi = \frac{1}{A}\int_{\partial\Omega_c} \tilde{Q}\, d\eta$$

$$\tilde{Q}_\eta = \frac{-1}{A}\int_{\partial\Omega_c} \tilde{Q}\, d\xi$$

where Ω_c is the control volume used for the central difference scheme, as shown in Fig. 2. The reconstruction process can create local extrema, and it is usually necessary to limit the gradients $\tilde{Q}_\xi$ and $\tilde{Q}_\eta$ in order to ensure monotonicity of the solution.

The upwind scheme is naturally dissipative and requires no additional artificial dissipation for stability. Also, like its structured grid counterparts, it results in lesser smearing of flow discontinuities. The central difference scheme, on the other hand, has the advantage of being very easy to implement and requires lesser number of operations at each time step. Another reason for preferring the upwind scheme in the present study is the fact that it is identical to the flux evaluation procedure used in the inner, structured grids. Thus, flux conservation between the different zones can be ensured, at least to the order of accuracy used in the outer grids.

Grid Adaptation Strategy

To be viable for unsteady flow computations, solution-adaptive grid techniques must satisfy several additional requirements beyond those required for steady computations. First, the adaptation algorithm must be very efficient, since it will be used repeatedly. This is not a major concern in steady computations where adaptation is done only a few times during the entire solution procedure. Second, the adaptation strategy and attendant error indicators should be automatic and robust. Since the grid is adapted every few time steps, user intervention is not feasible whenever the grid changes. Third, the adaptation method must incorporate a coarsening capability that allows point deletion to prevent an unbounded increase in the total number of points. Finally, the adaptation technique should incorporate some means of detecting unacceptable cells and restoring grid quality, since repeated refinement and coarsening tend to degrade the quality of the grid.

Grid adaptation algorithms currently in use can be classified into three general categories: 1) point relocation, 2) remeshing, and 3) h refinement. In point relocation methods the total number of grid points is kept constant, but their location is varied to achieve the desired clustering in specific regions. These methods are very easy to implement and are useful when grid adaptation is used to relocate interior points following any changes in the boundaries,[21] or in cases where only a mild clustering is sufficient.[22] This approach does have drawbacks in situations where excessive gradients develop because additional points are not introduced. Grid quality also deteriorates easily if there is excessive point motion. In remeshing methods, the entire domain is (or portions of it are) fully remeshed at each adaptation. The drawbacks of this approach are that complete or even partial remeshing can be very expensive, and that the general spatial interpolation of the solution from the original to the adapted grid can lead to numerical errors for transient or periodic problems. In the h-refinement approach, points are added and/or deleted from the grid according to specified criteria, usually by subdivision of some element (edge, face, or volume) of the original grid. Interpolations on the new grids can be made conservative and do not require global searches. The degree of refinement can be controlled precisely and the procedure is quite robust. A drawback of this technique is that the transition from refined- to coarse-grid regions is not very smooth, and the change in cell areas can be rather large.

The grid adaptation method used in the present study is based primarily on h refinement. However, some additional steps are incorporated to improve the quality of the adaptive grids. These steps are described below.

Grid Refinement

Refinement is effected by subdividing those edges of current grids for which the indicator function (computed as an average of the indicator function values at the two endpoints) exceeds a specified tolerance. This approach is chosen over other alternatives (such as point addition at the centroid of an existing triangle) because it can be accomplished using only the edge data structure. Also, since the new points are added midway between two existing points, the interpolation of the flow variables is straightforward.

Grid Coarsening

Point removal from a grid is more complicated than point addition. Coarsening cannot be based simply on removing all the points where the error indicator is below a specified tolerance, since this is likely to result in removal of all points from smooth regions. The procedure used here is as follows. A deletion flag is defined at each grid point and set to 0, except at certain points which must be protected from deletion (such as some boundary points which define the ends of the domain etc.) where the flag is set to 1. All the points are then successively visited, and if both the error indicator is below tolerance and the flag is 0, that particular point is marked for deletion by setting its flag to -1. Each time a point is marked for deletion, all its neighbors are forcibly protected by setting their flags to 1. This ensures that the grid refinement is gradual, but does not necessarily delete the points with the lowest indicator values. Somewhat better results are obtained if the points are visited in the order of their error indicator values rather than according to their indices in the point lists. An alternative strategy, which can be used if the original grid is deemed to be sufficiently coarse in the smooth regions of the flow, is to delete only the points that were introduced as a result of refinement. This approach is faster, but does not make optimum use of adaptation capabilities since the original set of grid points is always preserved.

Retriangulation

An important consideration during grid adaptation is the triangulation of the new set of grid points. One possible means of accomplishing this would be to generate the new list of grid points by adding and deleting points from the old grid as required, and then to fully retriangulate the new set of points. However, this would be very inefficient since triangulation methods, such as the Delaunay triangulation, typically require $\mathcal{O}(n \log n)$ operations, where n is the number of points. It is highly desirable that the triangulation be obtained simultaneously with refinement or coarsening. For the grid refinement strategy chosen here, this is very easy to accomplish, as shown in Fig. 4. The figure on the left depicts the original grid, the dotted lines in the central figure are the two new edges created joining the new point with the neighbors of the original edge. This default triangulation is always possible on an arbitrary grid following a refinement and the updates to the point, edge and cell data structures of the grid are determined easily, using only the edge data structure. The figure on the right in Fig. 4 showing the final triangulation will be discussed later under edge swapping.

Retriangulation following coarsening is more complicated than retriangulation after refinement since there is no automatic way of reconnecting the remaining points. The number of neighbors varies and there is no unique triangulation of the polygon left after the point is removed. The current approach uses a general algorithm for triangulation of a polygon which seeks to create a valid triangle out of three successive points of the polygon such that the length of the new side

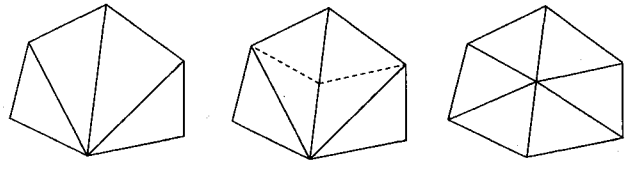

Fig. 4 Illustration of grid refinement followed by edge swapping.

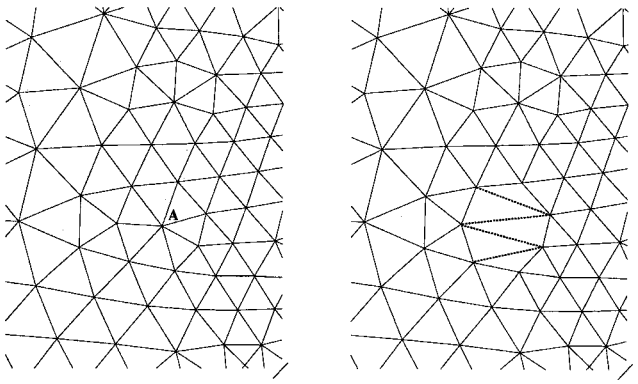

Fig. 5 Illustration of grid coarsening.

generated is minimized. The polygon remaining after the new triangle has been formed is then treated similarly, and this process is continued till the entire polygon has been retriangulated, as illustrated in Fig. 5, which depicts a section of the mesh before and after deletion of the point A.

Edge Swapping

Although the refinement and coarsening procedures described above produce valid grids in a very efficient manner, the grids are not necessarily globally optimal and can have elements with small angles or areas. A simple localized correction procedure can be used during the refinement and coarsening process to ensure better grid quality. The idea is based on a property of Delaunay triangulation which guarantees that in any quadrilateral, of the two possible diagonals, the one chosen is such that the minimum angle is maximized. This principle provides a very convenient mechanism for converting any existing triangulation of a set of points to the Delaunay triangulation by visiting each edge and swapping it in favor of the alternative if it fails to meet the criteria. The process is repeated until all edges are made to conform in this manner. This approach is particularly attractive in the present context, since it only needs to be applied locally. A very simple and elegant implementation of edge swapping is possible if programmed in a language that supports recursion, since every time an edge is swapped, the procedure can simply invoke itself for all the edges that are affected.

An illustration of the improvement that can be obtained by edge swapping is shown for the situation following edge refinement in Fig. 4. It is easy to show that the four edges meeting at the newly created point will be part of the final Delaunay triangulation, and hence, only the edges of the quadrilateral need to be checked for possible swapping. The final triangulation after two of these edges have been swapped is sketched in the figure on the right. It is quite clear that the edge swapping prevents triangles with small angles. It is also interesting to observe from the figure that the new triangulation has edges with smaller lengths, although only one of the original edges was refined. A final point to note regarding edge swapping is that the decision to swap need not be based always on the Delaunay criterion. If some other grid optimality principle is desired, such as the minimizing of the maximum angle, edge swapping can still be used to obtain a grid satisfying that criterion, starting from any given triangulation.

Grid Smoothing

When the edge swapping procedure is used simultaneously with the refinement and coarsening procedures, the resulting grid can be optimal in the sense of satisfying the Delaunay criterion everywhere, with the possible exception of boundary regions. However, this still does not ensure that the variation in adjacent cell areas is smooth. This situation is most easily visualized in case of coarsening of a regular triangular grid. If the points are deleted such that all neighbors of each deleted point are protected from deletion, the Delaunay triangulation of the new set of points will not be as regular as the original grid. However, a few sweeps of grid smoothing, where each interior point is successively relocated to the centroid of its surrounding polygon, can be used to recover a smooth coarsened grid.

The main disadvantage of grid smoothing is that the solution must be interpolated at the new locations of the points. However, since every point moves only within the surrounding polygon, global searches are not required in order to locate the cell in which the new location lies, and thus this interpolation is considerably faster than what would be required following a complete remeshing. In addition, this smoothing does not have to be invoked after every adaptation, but only when the grid quality becomes very poor. Also, only those points which are farther away from the centroid of their surrounding polygon than a specified tolerance are actually moved to the new location.

Boundary Conditions

The success of any zonal scheme depends on the proper implementation of interface boundary conditions. The boundaries that occur in the present method can be classified into two categories—the "natural" boundaries that are inherent to the problem, and the "artificial" interface boundaries introduced by the decomposition of the domain.

Airfoil Surface Boundaries

At all points on airfoil surfaces, the conditions of no-slip, adiabatic wall, and zero normal derivative of pressure are imposed. Note that in the case of the rotor, "no-slip" implies zero relative (not absolute) velocity. Reference 4 provides details of the implicit implementation of these boundary conditions.

Inlet and Outflow Boundaries

Subsonic inflow and outflow boundary conditions are imposed at the left boundary of the first airfoil row and the right boundary of the last airfoil row, respectively. At the inlet, three quantities (a Riemann invariant, the total pressure, and the inlet flow angle) are prescribed while the fourth quantity (also a Riemann invariant) is extrapolated from the interior of the solution domain. At the outflow boundary, one Riemann invariant is extrapolated from the interior, along with the v velocity and entropy, while the static pressure is prescribed.

Except for one detail, the implementation of these boundary conditions is identical to that in case of structured grids. Extrapolation of flow quantities from the interior is usually achieved by taking those values from the next inner line parallel to the boundary. However, in the case of unstructured grids there is, in general, no such set of points. To avoid expensive interpolation, the inlet boundary is extended by

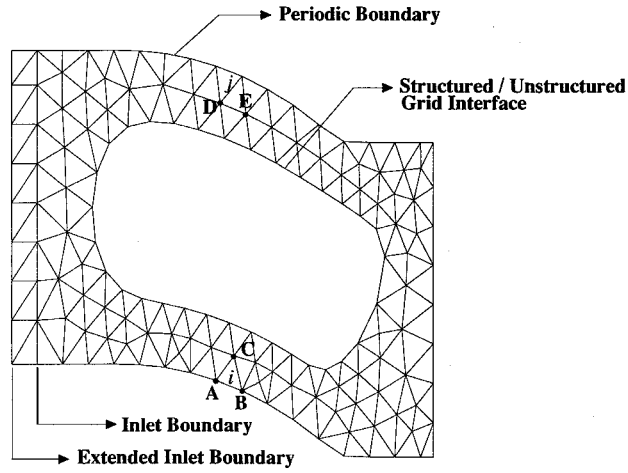

Fig. 6 Unstructured outer grid with extended inlet boundary.

adding a set of points that mimics the situation found in structured grids, as shown in Fig. 6.

The specification of static pressure at the exit leads to a reflective boundary condition. Following Ref. 23, both one- and two-dimensional nonreflective boundary conditions were implemented in the present method. These boundary conditions are based on the work of Refs. 24 and 25. The results of these implementations are not discussed here; the reader is referred to Ref. 26 for details.

Periodic Boundaries

For ease in imposition of periodicity over one rotor or stator pitch (or composite pitch when multiple airfoils are considered in each row), a one-to-one correspondence is maintained between the point distributions on the upper and lower boundaries of the unstructured grids. The flux terms for the points on the upper boundary are calculated based on the half control-volume surrounding them and then added to the corresponding points on the lower boundary. In unstructured grids this can be accomplished easily by modifying properly the edge-based data structure used to express the connectivity of the grids. For example, the data structure for the edge i in Fig. 6 consists of A and B as its endpoints, and the points C and D as the left and right neighboring points. Likewise, the edge j is considered to be directed from D to B, and has A and E as the two neighbors. Once the data structure for all edges that have an end-point or a neighbor on the periodic boundary has been similarly modified, the solution procedure described above implicitly accounts for the periodicity, and no special procedures are necessary.

Inner-Outer Grid Interfaces

The inner boundary of the outer unstructured grid and the outer boundary of the inner structured grid share the same set of points, thus obviating any need for expensive interpolation. The flow variables at these points are not updated during the inner grid solution procedure. Updating them during the outer grid solution process requires the definition of a complete control volume around these points, and this can be accomplished by extending the unstructured grid inside the structured zone. Two possible approaches are depicted in Figs. 7 and 8 where the augmented unstructured grid is shown by the dotted lines. In the first approach, which is used in the central difference algorithm, triangular cells are formed with additional points that are introduced midway between the segments on the next inner ξ line of the inner grid. This arrangement is preferred over the triangulation of the inner grid quadrilaterals because it results in a smoother grid. In the second approach, depicted in Fig. 8, the outer triangular mesh is augmented with a set of quadrilateral cells. The points near the interface now have common control-volume boundaries, and therefore flux conservation across the grid interface can be ensured.

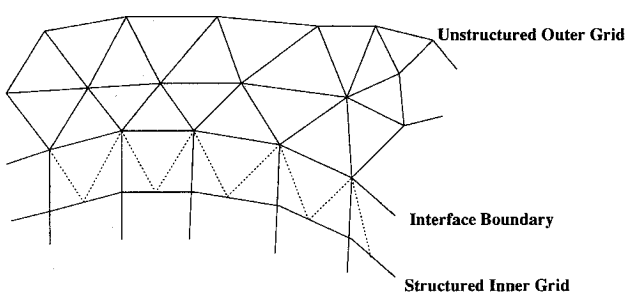

Fig. 7 Inner-outer grid interface for central difference scheme.

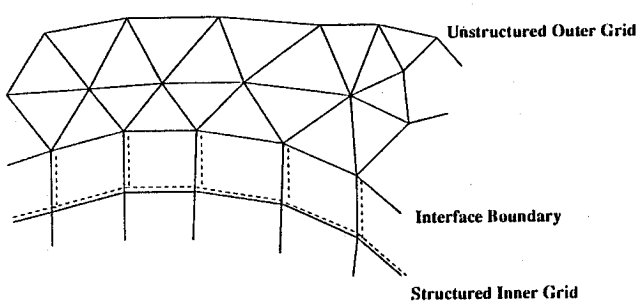

Fig. 8 Inner-outer grid interface for upwind scheme.

The above procedure raises an issue regarding time accuracy of the solution at the zonal interfaces. In any zonal algorithm the solution in the different zones is updated (to the next time level) in a sequential manner. If an iterative scheme is employed at each time step, then the grids are updated sequentially during each iteration, with the information across zonal interfaces "lagged," i.e., held constant at values from the previous iteration. Any errors associated with this lagging of information can be driven to zero by performing a sufficient number of iterations to converge the solution in each zone at each time step. In the present method, this strategy is not feasible at the interface between the outer and inner grids, since different time stepping schemes are employed in the two regions and there is no correspondence between the intermediate solutions of the Runge-Kutta stages of the outer grids and the Newton iteration steps of the inner grids. Therefore, the solution at the next time level is obtained first for all the outer zones by performing the four explicit Runge-Kutta integration steps, keeping the solution at the additional points that lie inside the inner region constant at the values from the previous time step. One can compensate easily for this lagging of information by introducing additional layers of unstructured grid cells that extend deeper into the inner grid. Thus, in the first stage of the four-stage Runge-Kutta integration, only the innermost layer of points have to be held constant while the rest can be updated; in the second stage, two layers of points are held constant while the rest are updated, and so on for the third and fourth stages. The solution is then updated in the inner zones, followed by a post-update correction where all outer grid points that lie inside the inner grids are interpolated from the current inner grid values. An analysis of the effective influence stencil over the four-stage scheme employed in the present work indicates that to ensure full second-order accurate update of the points on the interface, the augmented unstructured grid should extend eight cells deep into the inner grids. Numerical tests carried out using this approach did not result in any appreciable improvement in the temporal accuracy of the solution (in terms of surface pressure amplitudes, and other similar quantities) for the problems considered. Hence, the simpler inner-outer interface treatment described above was used to obtain all the solutions reported here.

Rotor-Stator Grid Interfaces

This interface occurs between the outer unstructured grids of two adjacent rows of airfoils which slip past each other. Flux conservation across this interface is ensured in the following manner. When the grid on the left is being updated, the points on its right boundary cannot be updated to the next time-step since only half a control volume exists around them, as shown in Fig. 9 for point A. This partial contribution to the flux terms for the boundary points is computed and stored. The flux contribution at the points is considered as a piece-wise constant distribution of fluxes along the interface boundary. Therefore, the flux calculated for point A is distributed over half of edges $A-C$ and $A-B$. During the subsequent update of the grid on the right, this contribution is transferred to the edges on the left boundary of that grid in a conservative manner. The edge distribution is then accumulated at the boundary points on the next grid. The contribution of the half-control volume to the right is added to the left boundary points of the right grid as usual, and the flow quantities at the next time step are thus obtained at these points.

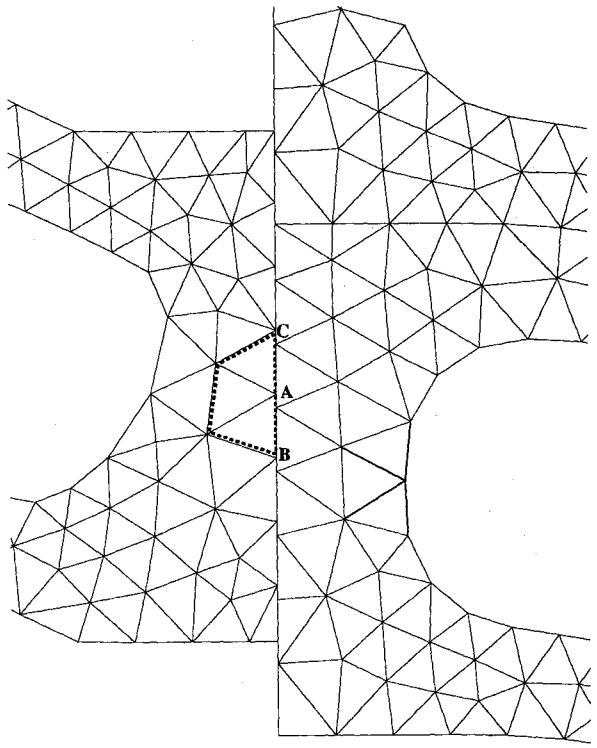

Fig. 9 Stator-rotor grid interface.

Once all the grids have been updated in this fashion, the dependent variables on the right boundaries of all but the last airfoil row are interpolated from the new values on the left boundaries of the next grid, thereby completing the update to the next time level at all the points on the unstructured outer grids.

Grid Adaptation for Moving Boundaries

Apart from enhancing the grid to improve the solution quality, the adaptation capability of unstructured grids can also be exploited for other purposes. In the present method, one such use of grid adaptation is possible at the interface between two unstructured grids in relative motion. If the rotor and stator grids in a turbomachinery stage computation are not modified, then, as the rotor moves past the stator, the interface changes at each time step. Interpolation searches are thus necessary at each time step in order to transfer information across the interface. However, if the stator grid is adapted such that each boundary point at the edge always coincides with some point on the rotor grid boundary, then the interpolation becomes trivial. This is achieved in the following manner. At the beginning of each time step, the rotor grid is moved to its new location based on the rotor speed. The right boundary points of the stator grid are then moved along the boundary, in either direction, to the nearest point on the rotor boundary. This procedure is illustrated in Fig. 10 which depicts the region in the vicinity of the interface at four instances of a rotor-passing cycle. (A cycle is defined as the time taken by the rotor grid on the right to move through a distance equal to one pitch.) Close inspection of the figure reveals that the outer unstructured grid around the two stator airfoils, marked S1 and S2, remains unchanged, except for the points on its right boundary. The figure also shows the unstructured grid around the rotor airfoils, R1 and R2, translating downwards, but without any relative motion between the grid points. This procedure is quite efficient since only the points on the boundary are moved, and furthermore, this motion is confined along the boundary line. The grid connectivity remains unchanged for both grids. The distortion introduced by moving the boundary points is quite small since the maximum distance a point can traverse is half of the maximum grid spacing at the boundary. The only requirement for this procedure is that the number of points on the two boundaries should be the same, but this is not a concern for unstructured grids since this requirement does not impose any restrictions on the grid densities in the interior of either grids. It is obvious from Fig. 10 that with this type of grid adaptation, the entire outer unstructured region appears essentially to be discretized with a single grid at all instants. Thus, this type of grid adaptation is also very useful for implementing higher order methods since no reduction in accuracy at the interfaces is incurred.

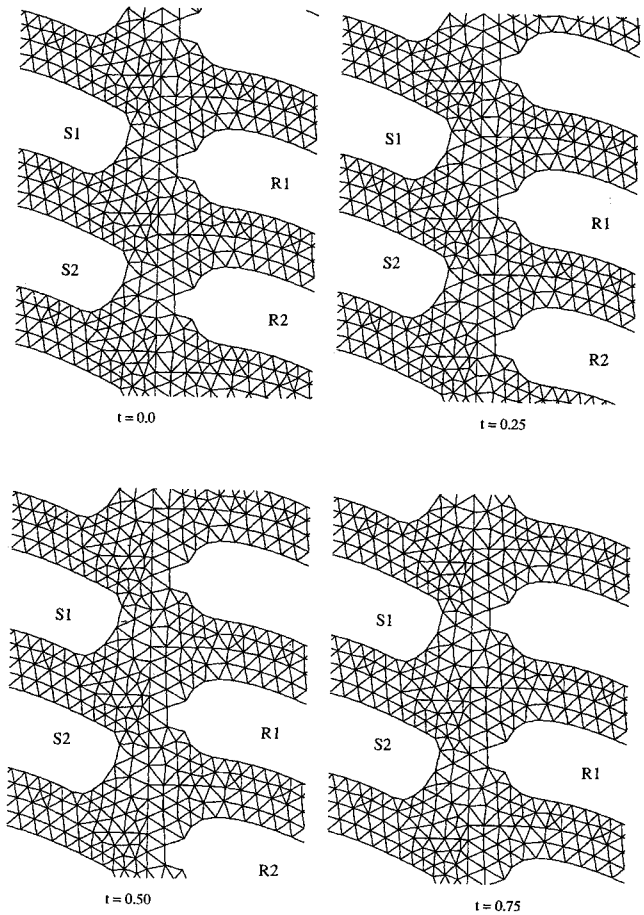

Fig. 10 Grid adaptation for moving boundaries.

Overview of the Algorithm

The main steps of the overall hybrid algorithm are summarized as follows. The solution process begins with an initial guess for the conserved variables (usually the inlet values) on all grids. The following steps are then carried out at each time step for the outer unstructured grid of each airfoil row in sequence.

1) The flow variables at the augmented grid points on the inner-outer grid interfaces are calculated from the current values on the inner grid.

2) The flux terms are calculated for all points. Except for the last row, the partial fluxes at the right boundary are stored for use in the next grid and, for all but the first row, the stored flux terms from the previous row are interpolated onto the points at the left boundary.

3) The flow variables are then updated to the next iteration level.

These steps are repeated for all the rows and for each of the four stages of the Runge-Kutta scheme at the end of which all the unstructured grid points have been updated to the next time step. The inner structured grids are then updated in sequence as follows:

1) The flow variables at the outer boundaries are copied from the unstructured grid. (Although the inner and outer grids share the same set of points at their interface, for ease of implementation, the points are maintained in data structures associated with both the grids.)

2) The solution for the inner grids is advanced to the next time step using several iterations of the iterative implicit algorithm at all points except the outer boundary where boundary conditions are prescribed.

A post-update correction step is carried out at the end of the time step. This involves imposition of airfoil surface boundary conditions for the inner grids, inflow and outflow boundary conditions at the left and right boundaries of the first and last unstructured grids, respectively, and the interpolation of flow variables at the right boundaries of all grids (except the grid for the last airfoil row). The grid adaptation procedure is invoked once every few (typically, 50) time steps.

Summary

A solution-adaptive hybrid-grid method has been developed for the two-dimensional analysis of unsteady flows in turbomachinery. The present approach uses a hybrid structured-unstructured zonal grid topology along with modeling equations and solution techniques that are most appropriate in the individual domains, thus combining the advantages of both structured and unstructured grid methods. An efficient and robust grid adaptation strategy, including both grid refinement and coarsening capabilities, is incorporated. The grid adaptation is also exploited to simplify the transfer of information across the moving interface between adjacent stator and rotor grids. Although results have been presented here only for a single-stage turbine, the present method is also capable of treating multistage turbomachinery configurations. For generality, three-dimensional effects of streamtube contraction are also included in the analysis. Details of the numerical method are presented in this article. Results obtained using this method and comparisons of these results with experimental data and earlier structured-grid based methods can be found in the companion article (Part II).

Acknowledgments

Funding for S. R. Mathur was provided by NASA Ames Research Center through University Grant NCA2-541. Funding for N. K. Madavan was provided partially by NASA Ames Research Center through Cooperative Agreement NCC2-755, and by the Office of Naval Research. The authors would like to acknowledge the technical inputs provided by M. M. Rai of NASA Langley Research Center, A. A. Rangwalla, and K. L. Gundy-Burlet of NASA Ames Research Center, and T. H. Okiishi of Iowa State University.

References

[1]Erdos, J. I., Alzner, E., and McNally, W., "Solution of Periodic Transonic Flow Through a Fan Stage," *AIAA Journal*, Vol. 15, 1977, pp. 1559–1568.

[2]Koya, M., and Kotake, S., "Numerical Analysis of Fully Three-Dimensional Periodic Flows Through a Turbine Stage," *Journal of Engineering for Gas Turbines and Power*, Vol. 107, Oct. 1985, pp. 945–952.

[3]Fourmaux, A., "Unsteady Flow Calculation in Cascades," American Society of Mechanical Engineers 86-GT-178, 1986.

[4]Rai, M. M., "Navier-Stokes Simulations of Rotor-Stator Interaction Using Patched and Overlaid Grids," *Journal of Propulsion and Power*, Vol. 3, No. 5, 1987, pp. 387–396.

[5]Lewis, J. P., Delaney, R. A., and Hall, E. J., "Numerical Prediction of Turbine Vane-Blade Interaction," AIAA Paper 87-2149, June 1987.

[6]Chen, Y. S., "3-D Stator-Rotor Interaction of the SSME," AIAA Paper 88-3095, July 1988.

[7]Giles, M. B., "Stator/Rotor Interaction in a Transonic Turbine," *Journal of Propulsion and Power*, Vol. 6, No. 5, 1990, pp. 621–627.

[8]Jorgenson, P. C. E., and Chima, R. V., "An Unconditionally Stable Runge-Kutta Method for Unsteady Flows," AIAA Paper 89-0205, Jan. 1989.

[9]Rao, K. V., and Delaney, R. A., "Investigation of Unsteady Flow Through a Transonic Turbine Stage, Part I—Analysis," AIAA Paper 90-2408, June 1990.

[10]Madavan, N. K., and Rai, M. M., *Computational Analysis of Rotor-Stator Interaction in Turbomachinery Using Zonal Techniques*, edited by P. A. Henne, Vol. 125, Progress in Astronautics and Aeronautics, AIAA, Washington, DC, 1990, pp. 481–529.

[11]Lakshminarayana, B., "An Assessment of Computational Fluid Dynamic Techniques in the Analysis and Design of Turbomachinery—The 1990 Freeman Scholar Lecture," *Journal of Fluids Engineering*, Vol. 113, Sept. 1991, pp. 315–352.

[12]Gundy-Burlet, K. L., Rai, M. M., and Dring, R. P., "Two-Dimensional Computations of Multi-Stage Compressor Flows Using a Zonal Approach," AIAA Paper 89-2452, June 1989.

[13]Giles, M. B., and Haimes, R., "Validation of a Numerical Method for Unsteady Flow Calculations," *Journal of Turbomachinery*, Vol. 115, Jan. 1993, pp. 110–117.

[14]Mavriplis, D., "Accurate Multigrid Solution of the Euler Equations on Unstructured Adaptive Meshes," *AIAA Journal*, Vol. 28, 1990, pp. 213–221.

[15]Mavriplis, D., Jameson, A., and Martinelli, L., "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes," AIAA Paper 89-0120, Jan. 1989.

[16]Venkatakrishnan, V., and Mavriplis, D., "Implicit Solvers for Unstructured Meshes," *Proceedings of the 10th AIAA CFD Conference*, July 1991, pp. 115–124 (AIAA Paper 91-1537).

[17]Chima, R. V., "Explicit Multigrid Algorithm for Quasi-Three-Dimensional Viscous Flows in Turbomachinery," *Journal of Propulsion and Power*, Vol. 3, No. 5, 1987, pp. 397–405.

[18]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, Oct. 1981, pp. 357–372.

[19]Barth, T. J., and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA Paper 89-0366, Jan. 1989.

[20]Barth, T. J., "Aspects of Unstructured Grids and Finite-Volume Solvers for the Euler and Navier-Stokes Equations," Special Course on Unstructured Grid Methods for Advection Dominated Flows, AGARD Rept. 787, 1992.

[21]Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," AIAA Paper 89-0115, Jan. 1989.

[22]Kennon, S. R., "Supersonic Inlet Calculations Using an Upwind Finite-Volume Method on Adaptive Unstructured Grids," AIAA Paper 89-0113, Jan. 1989.

[23]Rangwalla, A. A., and Rai, M. M., "A Kinematical/Numerical Analysis of Rotor-Stator Interaction Noise," AIAA Paper 90-0281, Jan. 1990.

[24]Bayliss, A., and Turkel, E., "Far Field Boundary Conditions for Compressible Flows," *Journal of Computational Physics*, Vol. 48, No. 2, 1982, pp. 182–189.

[25]Giles, M. B., "Non-Reflecting Boundary Conditions for the Euler Equations," MIT Computational Fluid Dynamics Lab., TR CFDL-TR-88-1, Cambridge, MA, 1988.

[26]Mathur, S. R., "A Solution Adaptive Structured-Unstructured Grid Procedure for Unsteady Flows," Ph.D. Dissertation, Iowa State Univ., Ames, IA, 1993.