Proceedings of
ASME TURBO EXPO 2001
June 4-7, 2001, New Orleans, Louisiana, USA

2001-GT-0357

CONSERVATIVE INTERFACING FOR TURBOMACHINERY APPLICATIONS

Philip M. Cali*
CFD Research Branch
Air Force Research Laboratory
Wright-Patterson Air Force Base, Ohio 45433

Vincent Couaillier‡
CFD and Aéroacoustics Dept.
Office National D'Études et de Recherches Aérospatiales
92320 Châtillon, France

Antony Jameson§
Dept. of Aeronautics and Astronautics
Stanford University
Stanford, California 94305

ABSTRACT

This paper presents the development of a conservative interface technique designed for turbomachinery applications involving multiple rows of blades. Instead of interpolating solution information between blade rows, unique patch boundaries are defined at rotor/stator interfaces on which numerical fluxes are evaluated directly. The fluxes are then distributed to both sides of the interface in a fully conservative fashion. Accuracy is maintained by representing the primitive variables in each interface cell using a system of quadratic polynomials. The coefficients of each polynomial are calculated using neighboring flow information from both sides of the interface. In order to make the reconstruction process more tenable, the current implementation is restricted to pairs of blade rows which maintain identical axisymmetric radial distributions of points at their juncture. Preliminary inviscid flow results show significant improvements in solution quality relative to calculations conducted using conventional nonconservative interpolation techniques.

INTRODUCTION

Prohibitive computational requirements have until recently limited unsteady turbomachinery calculations to small engine sections containing modest numbers of circumferentially reduced stages. Shared and distributed memory massively-parallel platforms, however, such as those being investigated by the Department of Energy's Accelerated Strategic Computing Initiative (ASCI), have the potential to render large-scale unsteady turbomachinery calculations possible in the foreseeable future [18].

Given the prospect of simulations consisting of large numbers of stages, an important concern becomes the manner in which flow information is passed between adjacent rows of blades. Conventional methods are commonly based on interpolating flow information between rotor and stator meshes. Some recent examples can be found in the following references [4][5][14][18]. While convenient to implement, such interpolation methods do not maintain conservation across blade-row interfaces. While a lack of conservation is disconcerting even for turbomachinery calculations involving a limited number of stages, conservation errors would quickly overwhelm a solution containing, say, 10 to 20 stages.

The conservative interface technique presented here was developed to provide an alternative to traditional nonconservative methodologies for passing flow information between consecutive blade rows in relative motion. It is based on the approach described in reference [3] and has been developed and tested using ONERA's multi-block flow solver, *Canari*.

OVERVIEW OF *CANARI*

The *Canari* multi-block flowsolver [13][17], which was developed in the CFD Research Branch of ONERA, served as the framework for the current development. *Canari's* fundamental algorithm is based on the compressible, mass-averaged, Navier-Stokes equations and includes several turbulence models. Since the scope of the current work is limited to inviscid flow, only a description of *Canari's* Euler algorithm is provided here.

*Aerospace Engineer
‡ Senior Research Scientist
§T.V. Jones Professor of Engineering

Using the freestream total density (ρ_o), total speed of sound (c_o) and characteristic length scale (L) to nondimensionalize the flow variables, the governing equations may be cast in the following conservative form,

$$\frac{\partial U}{\partial t}+\frac{\partial E}{\partial x}+\frac{\partial F}{\partial y}+\frac{\partial G}{\partial z}=0. \qquad (1)$$

Here t is time and (x,y,z) are the Cartesian spatial coordinates. The state vector (U) and flux terms (E,F,G) are given by,

$$U=\begin{bmatrix}\rho & \rho u & \rho v & \rho w & \rho e_o\end{bmatrix}^T$$

$$E=\begin{Bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ \rho uh_o\end{Bmatrix}\quad F=\begin{Bmatrix}\rho v\\ \rho vu\\ \rho v^2+p\\ \rho vw\\ \rho vh_o\end{Bmatrix}\quad G=\begin{Bmatrix}\rho w\\ \rho wu\\ \rho wv\\ \rho w^2+p\\ \rho wh_o\end{Bmatrix}$$

$$\text{where } e_o=\frac{p}{(\gamma-1)\rho}+\tfrac{1}{2}(u^2+v^2+w^2)$$

$$\text{and } h_o=e_o+\frac{p}{\rho}.$$

In the above expressions, (u,v,w) are the Cartesian velocity components, ρ is the density and p is the pressure.

Canari is based on the Jameson-Schmidt-Turkel (JST) [11] multi-step explicit finite-volume algorithm. Common forms of both linear and nonlinear artificial-dissipation terms are employed to augment stability using well-established values of the damping coefficients [13][17]. *Canari's* algorithm has proven reliable for a variety of internal and external fluid-flow problems [2][6][7].

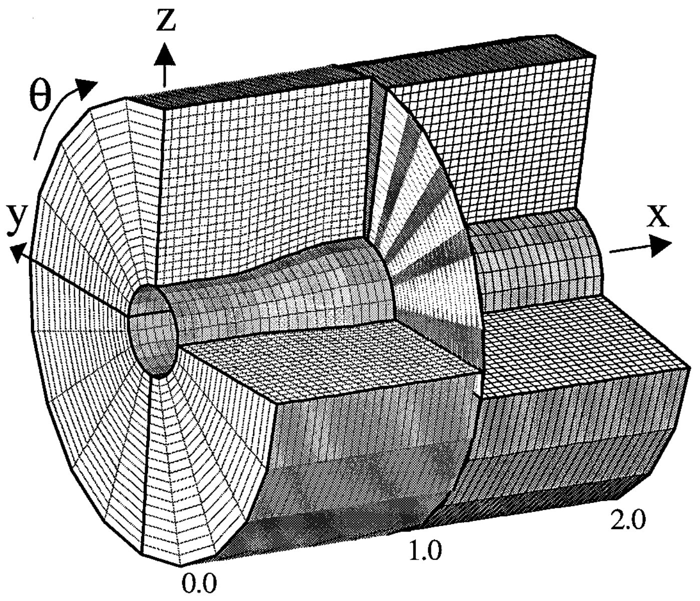

Figure 1 Eight-mesh axisymmetric duct configuration.

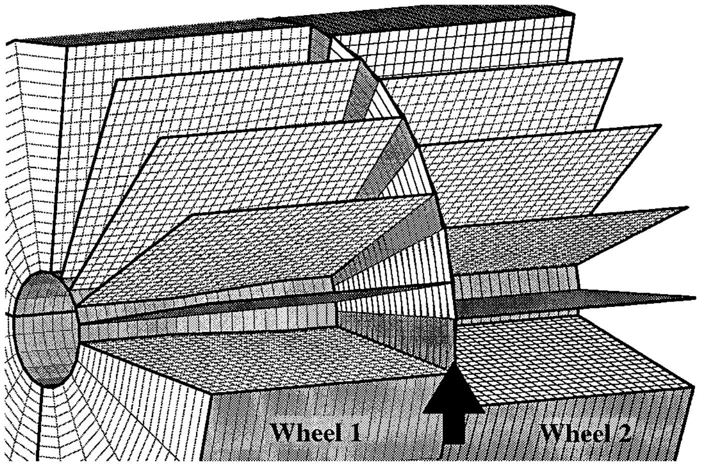

Figure 2 Non-coincident interface between wheels.

DEVELOPMENTAL TEST CASE

A simple test case was chosen to conduct the development. The grid configuration is shown in Figures 1 and 2. The test case consisted of the inviscid M_∞ = 2.0 flow through an axisymmetric duct with an 11.31° ramp in the lower wall. In the figures, the geometry is modeled using two separate 'wheels' each consisting of four pie-slice-shaped domains of $\pi/2$ each. For simplicity, we will refer to the wheels as 'blade-rows' and to the four grid-blocks that make up each wheel as 'blades' (although no actual blades exist). Each wheel is 1.0 units long and has an outer wall radius of 1.0 units. The inner radii at the inlet and outlet are 0.20 and 0.28 units, respectively, and the ramp between these two values extends from x = 0.4 to x = 0.8. Each of the eight meshes making up the configuration maintains uniform spacing and possesses axial, radial and circumferencial grid dimensions of (26×21×6). In the figures, the downstream wheel has been rotated by $\pi/25$ to create the non-coincident interface shown. Note that the lack of coincidence occurs in the θ direction only, as radii on opposite sides of the inter-wheel boundary are equal.

CONSERVATIVE INTERFACE TREATMENT

Overview

The conservative interface algorithm is simple in concept. Instead of interpolating the solution from the interior region of one grid to overlapping ghost cells created at the boundary of another (and vice versa), we endeavor to evaluate numerical fluxes directly along blade-row interfaces. The interface flux contributions are then distributed to the cells on opposite sides in a fully conservative fashion.

The process begins at the start of each global iteration when the angular position of the rotating wheel is incremented by $\Delta\theta$. Because the interface treatment was designed to accommodate blade rows of non-reducible counts, the interface cells from each individual blade grid are first concatenated to form a set of wheel arrays which extend through a full 2π. This mapping serves as a simple means for cross-referencing grid and flow information between consecutive blade rows as rotation occurs. These arrays are updated after each incremental rotation such that their members always remain ordered between 0 and 2π. This is conducted using a system of pointers which avoids the need to rearrange each wheel array after every iteration.

The above mapping is next used to create a unique patch boundary at the juncture between two adjacent blade rows. For the present work, this task is greatly simplified because the interfaces between consecutive blade rows are constrained to be planar and oriented in the x-direction (see Figure 2). Referring to Figure 3, the construction of the patch boundary reduces to superposing the upstream (solid lines) and downstream (dotted lines) blade-row boundary meshes. The superposition is carried out by merge-sorting the angular positions of the radial grid lines from both sides of the interface. Using this merged list, we can readily calculate the 'small-face' areas created by the superposition process. Some typical small faces are shaded in Figure 3 for clarity.

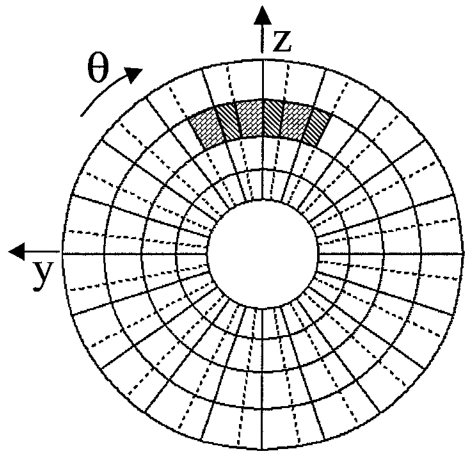

Figure 3 Superposed interface grids.

The unique patch boundary (as defined above) is then used at the beginning of each Runge-Kutta stage to reconstruct the fluxes along the interface. To this end, we first represent the primitive flow variables (ρ, u, v, w, p) in each interface cell using a quadratic polynomial. Interface cells are simply defined as the cells along the boundary (one row deep). The coefficients for each variable's polynomial are calculated using information from the surrounding cells on both sides of the interface. Stencils are taken to include all of the cell's face and edge neighbors. Two typical stencils are shown in Figure 4a. In the figure, each stencil contains eight points. Note that to limit the reconstruction to two dimensions and thus render it tenable for practical applications, we have constrained the radii of corresponding circumferential gridlines on opposite sides of the interface to be equal and constant with θ. Note that this restriction only affects interface cells.

The cell-wise polynomials are next used to reconstruct the flow at the centroid of each small face (see Figure 4b). The polynomial contributions from each side result in two reconstructed states at each small face. A unique flux value is then obtained by averaging the flux vectors, multiplying by the small-face area and subtracting an appropriate numerical smoothing term, *{D}*. Two different forms of artificial dissipation were investigated. Both will be explained in detail later in the paper.

The resulting small-face fluxes are next distributed to the interface cells on both sides by summing the appropriate small-face contributions. The distributed fluxes are finally included in the normal flux summation of the respective interior schemes.

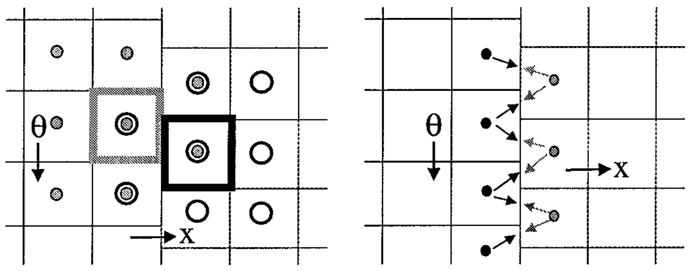

Figure 4 a. Typical stencils. b. Reconstruction along interface.

Areas, Normals and Volumes

To facilitate the process of maintaining conservation across mesh interfaces, grid metrics were calculated based on a cylindrical coordinate system. The metrics were then converted into Cartesian coordinates so that they were consistent with *Canari's* algorithm. This process is accomplished to ensure that the sum of the cell-face areas corresponding to a given annulus will be equal on both sides of the interface and also equal to the sum of the small-face areas corresponding to the same annulus (see Figure 3). The directional areas in cylindrical coordinates for each quadrilateral cell face are given by:

$$S_x = \frac{1}{2}\oint_{face} r^2 d\theta = -\oint_{face} r\theta dr \tag{2}$$

$$= \frac{1}{6}\begin{bmatrix} (r_4\theta_4 - r_2\theta_2)(r_3 - r_1) - (r_3\theta_3 - r_1\theta_1)(r_4 - r_2) \\ +(\theta_4 - \theta_2)(r_3^2 - r_1^2) - (\theta_3 - \theta_1)(r_4^2 - r_2^2) \end{bmatrix}$$

$$S_r = \oint_{face} r\theta dx = -\oint_{face} xr d\theta \tag{3}$$

$$= \frac{1}{2}\bar{r}_{face}\left[(\theta_3 - \theta_1)(x_4 - x_2) - (\theta_4 - \theta_2)(x_3 - x_1)\right]$$

$$S_\theta = \oint_{face} r dx = -\oint_{face} x dr \tag{4}$$

$$= \frac{1}{2}\left[(x_3 - x_1)(r_4 - r_2) - (x_4 - x_2)(r_3 - r_1)\right]$$

The corresponding cell volumes are,

$$Vol = \frac{1}{3}\oiint_{faces}(xrdrd\theta + \tfrac{1}{2}r^2 d\theta dx + r\theta drdx) \tag{5}$$

$$\approx \frac{1}{3}\sum_{faces} \bar{x}rdrd\theta + \tfrac{1}{2}\bar{r}d\theta dx + \bar{r}\bar{\theta}drdx$$

$$\approx \frac{1}{3}\sum_{faces}\left[\overline{x}_{face}S_x + \tfrac{1}{2}\overline{r}_{face}S_r + \overline{r}_{face}\overline{\theta}_{face}S_\theta\right].$$

We then transform the above areas via,

$$\begin{bmatrix} S_x \\ S_y \\ S_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} S_x \\ S_r \\ S_\theta \end{bmatrix}, \qquad (6)$$

to obtain the face normals in Cartesian coordinates,

$$\hat{n}_{face} = (n_x, n_y, n_z) = (S_x, S_y, S_z)/|S|. \qquad (7)$$

Cell-wise Polynomial Reconstruction

The cell-wise reconstruction is based on a quadratic polynomial of the form,

$$f_r(\Delta\theta,\Delta x) = f_o + a_1\Delta\theta + a_2\Delta x + a_3\Delta\theta^2 + a_4\Delta x^2 + a_5\Delta x\Delta\theta. \qquad (8)$$

Here f_r represents the reconstruction of any primitive variable f. f_o is the primitive's point-wise value at the cell's centroid from which $(\Delta\theta,\Delta x)$ are measured. The constants a_m are calculated using a singular-value-decomposition (SVD) formulation of the least-squares problem with information from neighboring cells. A typical stencil includes all face and edge neighbors on both sides of the interface (see Figure 4a).

Two slightly different formulations were implemented. In the first formulation, referred to here as the 'least-squares' (LS) formulation, the problem reduces to the following. Given N surrounding cells, we wish to choose the M coefficients of equation (8) so that the difference between the reconstructed solution and the discrete solution at the surrounding cell centers is as small as possible. Writing equation (8) for each neighboring cell results in the following system of equations,

$$\{\Delta f_n\} = [\mathbf{X}_{nm}]\{a_m\} \qquad (9)$$

where $\Delta f_n = (f_n - f_o)$, $n = 1,2...N$ and $m = 1,2...M$ (note that $M = 5$ for a quadratic polynomial). Thus, we want to find $\{a_m\}$ such that

$$\chi^2 = \left|[\mathbf{X}_{nm}]\{a_m\} - \{\Delta f_n\}\right|^2 \qquad (10)$$

is minimized. If $N \geq M$, we can decompose $[\mathbf{X}_{nm}]$ into the product of two orthogonal and one diagonal matrices as follows,

$$[\mathbf{X}_{nm}] = [\mathbf{U}_{nm}][diag(\mathbf{W}_m)][\mathbf{V}_{mm}]^T. \qquad (11)$$

Using this decomposition, the solution to (10) is given by

$$\{a_m\} = [\mathbf{V}_{mm}][diag(1/\mathbf{W}_m)][\mathbf{U}_{nm}]^T\{\Delta f_n\} \qquad (12)$$

$$\text{or} \quad a_m = \sum_{l=1}^{M}\frac{\mathbf{V}_{ml}}{\mathbf{W}_l}\sum_{n=1}^{N}\mathbf{U}_{nl}\Delta f_n \qquad (13)$$

The routine used to construct the matrices (**U**,**V**,**W**) is presented in reference [15].

The second formulation, which was developed by Delanaye, et al. [9], is a slight variation on the above theme. It is based on calculating the first-order gradients using a 'Green-Gauss' (GG) loop as shown in Figure 5. An error term based on a SVD calculation of the second-derivative terms is then subtracted from the loop gradients to produce second-order accuracy.

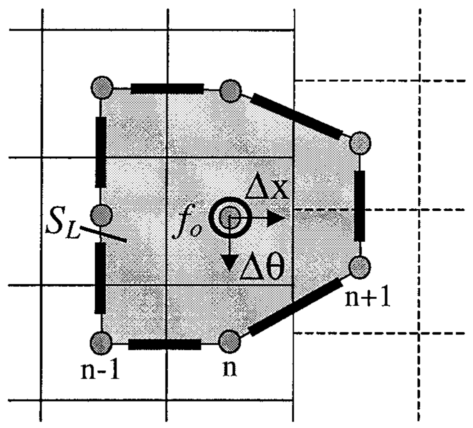

Figure 5 Typical 'Green-Gauss' loop.

The first-order gradients are given by the following,

$$\hat{\nabla}f = (\tfrac{1}{r}\hat{f}_\vartheta, \hat{f}_x) = \frac{1}{S_L}\oint_{Loop}(f - f_o)\vec{n}\,ds, \qquad (14)$$

where S_L is the loop area. In discrete form, equation (14) may be written,

$$\begin{Bmatrix} \hat{f}_\theta \\ \hat{f}_x \end{Bmatrix} = \frac{r_o}{2S_L}\sum_n (f_n - f_o)\begin{Bmatrix} x_{n+1} - x_{n-1} \\ \theta_{n-1} - \theta_{n+1} \end{Bmatrix} = [B]\begin{Bmatrix} \vdots \\ \Delta f_n \\ \vdots \end{Bmatrix}$$

$$\text{where} \quad [B] \equiv \frac{r_o}{2S_L}\begin{bmatrix} \cdots & (x_{n+1} - x_{n-1}) & \cdots \\ \cdots & (\theta_{n-1} - \theta_{n+1}) & \cdots \end{bmatrix}$$

and r_o is the radius at the cell center. We next expand $[B]\{\Delta f_n\}$ in a Taylor series about the point (θ_o, x_o),

$$[B]\{\Delta f_n\} = \overbrace{[B]\{\Delta\theta_n \;\; \Delta x_n\}}^{[I]}\begin{Bmatrix} f_\theta \\ f_x \end{Bmatrix} + \underbrace{[B]\{\frac{\Delta\theta_n^2}{2} \;\; \frac{\Delta x_n^2}{2} \;\; \Delta\theta_n\Delta x_n\}}_{[C]}\begin{Bmatrix} f_{\theta\theta} \\ f_{xx} \\ f_{\theta x} \end{Bmatrix} + \vartheta(\Delta h^3)$$

Truncating and replacing $(f_{\theta\theta}, f_{xx}, f_{\theta x})$ by (a_3, a_4, a_5), we arrive at a second-order gradient approximation,

$$\begin{Bmatrix} a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \hat{\hat{f}}_\theta \\ \hat{\hat{f}}_x \end{Bmatrix} = [B]\{\Delta f_n\} - [C]\begin{Bmatrix} a_3 \\ a_4 \\ a_5 \end{Bmatrix}. \qquad (15)$$

Equation (15) depends on obtaining at least first-order accurate approximations to the second derivative terms (a_3, a_4, a_5). To this end, we substitute (15) back into (9) which gives us the following overdetermined system for the three unknowns (a_3, a_4, a_5),

$$\overbrace{([I] - \{\Delta\theta_n \;\; \Delta x_n\}[B])\{\Delta f_n\}}^{\{\Delta g_n\}} = \underbrace{(-\{\Delta\theta_n \;\; \Delta x_n\}[C] + \{\frac{\Delta\theta_n^2}{2} \;\; \frac{\Delta x_n^2}{2} \;\; \Delta\theta_n\Delta x_n\})}_{[\mathrm{Y}_{nl}]}\begin{Bmatrix} a_3 \\ a_4 \\ a_5 \end{Bmatrix}. \qquad (16)$$

Because equation (16) has the same form as equation (9) (note l = 1,2,3), we use the aforementioned SVD solution procedure to solve for (a_3, a_4, a_5). We then back substitute into equation (15) for (a_1, a_2).

Although slightly more involved than the previously described least-squares procedure, the Green-Gauss method has the advantage that it decouples the calculation of the first and second-order gradient terms. This is useful when limiting the reconstruction in regions of high gradients. One can apply a full second-order quadratic reconstruction in smooth flow regions while switching to a robust first-order linear reconstruction near shocks. In the present work, Barth's limiter [1] was employed using the implementation found in reference [9].

Artificial Dissipation

We experimented with two different types of artificial dissipation at the interface. The first was based on Roe's [16] standard Jacobian linearization. Note that since the interface was aligned in the x-direction, we are only concerned here with the x-Jacobian matrix, $[A_x] = [\partial E / \partial U]$. The so-called matrix-smoothing term takes the following form,

$$\{D\}_{MATRIX} = \frac{1}{2}\left[|A_x|\right]\{\Delta U\} \qquad (17)$$

$$\text{where } \left[|A_x|\right] = [T]\left[diag\{|\lambda|\}\right][T]^{-1}$$

$$\text{and } \{\lambda\} = \{Q, Q, Q, Q + cS, Q - cS\}.$$

In the above, $Q \equiv \vec{V} \cdot \vec{S}$ and c is the speed of sound. Decomposed forms of the transformation matrices $[T]$ and $[T]^{-1}$ can be found, for example, in reference [12].

The second type of artificial dissipation was based on a simple scalar term of the form,

$$\{D\}_{SCALAR} = \frac{1}{2}\left[diag\{\lambda_{\max}\}\right]\{\Delta U\} \qquad (18)$$

$$\text{where } \lambda_{\max} = |Q| + cS.$$

Time-Integration

All calculations performed in this study were conducted using a four-step Runge-Kutta time-integration scheme with CFL = 3.0. As stated earlier, the current implementation assumes that consecutive blade-row grids have been constructed such that they abut perfectly at a planar interface between them. Because there is no disruption of the grid structure whatsoever, traditional acceleration techniques such as implicit residual smoothing and multigrid remain fully operational.

Test Results

Steady (Non-Rotating) Case

The ultimate goal in developing a technique to pass flow information through non-matching zonal grid boundaries is that the application of the method to a non-coincident interface reproduce the results that the baseline numerical scheme would yield when applied to a single, continuous mesh. To evaluate the present method in this regard, a steady (non-rotating) inviscid solution (M_∞ = 2.0) was first conducted using a single smoothly varying grid to represent the entire axisymmetric channel (as described previously). The single grid had dimensions of (51×21×21) and maintained a mesh spacing that was consistent with the eight-grid configuration shown in Figures 1 and 2. This case served as a reference in comparing subsequent solutions.

Steady (non-rotating) calculations were next carried out using the previously described eight-grid configuration illustrated in Figures 1 and 2. The non-coincident interface shown in Figure 2 was created by rotating the downstream wheel by $\pi/25$ and then fixing it. Calculations of the eight-grid configuration were conducted using both a standard linear-interpolation approach and the current conservative technique. Four separate conservative calculations were conducted to examine different reconstruction techniques (Green-Gauss vs. least-squares) and different numerical smoothing methods (matrix vs. scalar). In the figures below, these cases are referred to as 'GG + Matrix', 'LS + Matrix', 'GG + Scalar' and 'LS + Scalar'.

All four calculations conducted using the conservative interface treatment showed improvements over the linearly-interpolated results in both convergence and accuracy. Figure 6 shows a comparison of the iteration histories for the steady cases. The four conservative solutions were virtually identical. All four converged to machine zero in 35% fewer iterations than did the linear-interpolation case. The playing field was somewhat leveled, however, when we examined CPU cost. Figure 7 shows a plot of convergence vs. CPU seconds on a DEC Alpha machine. As expected, the 'LS + Scalar' conservative case was the cheapest of the four. The unoptimized code added approximately 30% extra CPU time per iteration in comparison to the traditional linear-interpolation scheme. The most expensive of the four was the Green-Gauss (GG) reconstruction with matrix smoothing. It added approximately 35% extra CPU time per iteration. Although the additional CPU expense is significant when using a scalar machine, operating in a massively-parallel environment would considerably lighten the burden. Indeed, individual processors could be dedicated specifically to the interface treatment making the extra CPU cost inconsequential.

Figure 8 presents line plots of several quantities along a constant θ plane. These include mass flow (ρuA), pressure (p), total pressure (p_o) and entropy (s). Entropy is defined here as,

$$s = \log(p/\rho^\gamma) - \log(p/\rho^\gamma)_{inlet} = \log(p/\rho^\gamma) - \log(1/\gamma).$$

With the exception of massflow which is a cross-sectional property, the line plots are taken at the average channel radius, $(r_{inner} + r_{outer})/2$. This coincides with the location at which a shock crosses the interface. Note that the interface itself is located at $x_{interface}$ = 1.0.

Using the single-grid calculation as the reference, we see that the results obtained using the conservative interface treatment were markedly improved relative to the linearly-interpolated results. Examining first the massflow (ρuA), we see that in all cases the conservative approach eliminated the discrepancy produced by the nonconservative linear-interpolation scheme. As a result, the pressure peaks of the conservative solutions were captured more precisely. In addition, the conservative total pressure and entropy plots matched the single-grid results considerably better than did the linearly-interpolated solution. Although the mass production attributed to the nonconservative approach was a mere 0.02% for this simple test case, the associated increase in entropy was a resounding 28%. Such a discrepancy highlights the importance of an accurate interface boundary treatment.

Among the steady conservative cases, there was no discernable difference in accuracy between the least-squares and the Green-Gauss reconstruction methods. The matrix and scalar smoothing calculations, however, did produce somewhat different results. This is most readily seen from the entropy (s) plot in Figure 8. Relative to the single grid solution, the calculation performed using matrix smoothing underpredicted the entropy jump through the shock and the scalar smoothing calculation overpredicted it. Of the two, the scalar approach actually matched *Canari*'s interior scheme slightly better. This is probably due to the fact that the scalar term given in equation (18) is very similar in nature to the second-order artificial-dissipation term of *Canari*'s internal algorithm. Indeed, the internal scheme's second-order smoothing term would be the dominant artificial-dissipation term near a shock. Further work is needed to more smoothly blend the interior and interface schemes' numerical-dissipation terms.

'Pseudo-Unsteady' (Rotating) Case

A 'pseudo-unsteady' test was next carried out to ascertain the functionality of the interface procedure for rotating meshes. 'Pseudo-unsteady' is defined here to mean that although the downstream mesh was indeed rotating geometrically, rotational accelerations and angular velocities were not taken into account. In other words, the physical locations of the flow-quantity array addresses were indeed rotating, although the values of the flow quantities themselves did not reflect that rotation. Because the flow was inviscid and axisymmetric, the steady-state solution should be recovered. Performing the calculation in this way (instead of in 'full-unsteady' mode), provided us with a convenient method with which to evaluate the performance of the rotational scheme using the steady-state solution as a reference.

The Green-Gauss reconstruction with scalar smoothing was chosen to perform this test. The downstream wheel was rotated by a fixed amount of $\pi/100$ after each timestep. Referring to the convergence history shown in Figure 9, we see that the solution converged less than three-orders of magnitude before entering a limit cycle. In investigating the reason for this, we discovered that *Canari's* treatment of the circumferential boundary condition between grids of the same wheel (unrelated to the current interface procedure) caused a cyclic asymmetry to persist throughout the calculation. The observed limit-cycle was the result of these asymmetries interacting with one another at the interface as the downstream wheel was rotated. Nonetheless, the calculation proved to be stable and produced a flowfield that was practically identical to the previous steady results. Note that the pseudo-unsteady case is labeled 'Rotating' in Figure 8.

CONCLUSION

Given the prospect of performing large-scale turbomachinery calculations consisting of dozens of stages, the nonconservative nature of conventional blade-row interface techniques render them inadequate. As an alternative, this paper has presented a conservative methodology that is based on creating unique patch boundaries between adjacent rows of blades. Numerical fluxes are evaluated directly along these interfaces, eliminating the need to interpolate flow information to a system of overlapping ghost cells. Interface fluxes are then distributed in a fully conservative fashion to the cells on both sides of the patch boundary. Preliminary inviscid test results have demonstrated the method's potential for providing improved accuracy and convergence characteristics relative to traditional nonconservative methods.

FUTURE WORK

The conservative interface treatment documented in this paper has recently been implemented into the *TFLOW* code at Stanford University and the results presented here reproduced. In the future, the methodology will be incorporated into ONERA's object oriented flow solver, *ELSA*. Future developments will focus on the extension of the algorithm to viscous and turbulent flows and its application to more complicated turbomachinery configurations.

ACKNOWLEDGMENTS

This work was conducted at ONERA in Châtillon, France as part of an engineering exchange program with the U.S. Air Force Research Laboratory (AFRL) at Wright-Patterson Air Force Base, Ohio. The first author thanks Dr. Vincent Couaillier for his invaluable assistance and guidance throughout his exchange tour. The first author also thanks Prof. Antony Jameson, Dr. Jixian Yao and Mr. Sriram Shankaran for many helpful conversations and discussions during his brief sabbatical with them at Stanford University. Their insights and suggestions were instrumental in bringing this work to fruition.

REFERENCES

[1] Barth, T. and Jesperen, D., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA-89-0366.
[2] Cahen, J., Couaillier, V., Delery, J. and Pot, T., "Validation of a Navier-Stokes Code Using a k-ε Turbulence Model Applied to a Three-Dimensional Transonic Channel," *AIAA Journal*, Vol. 23, No. 4 (pp. 671-679), April 1995.
[3] Cali, P. and Couaillier, V., "Conservative Interfacing for Overset Grids," AIAA-2000-1008.

[4] Chen, J. and Barter, J., "Comparison of Time-Accurate Calculations for the Unsteady Interaction in Turbomachinery Stages," AIAA-98-3292.
[5] Cizmas, P. and Dorney, D., "Parallel Computation of Turbine Blade Clocking," AIAA-98-3598.
[6] Collercandy, R., "Multigrid Strategy for Euler and Navier-Stokes Computations for Flows around Complex Aerospace Configurations," Proceedings of the ECCOMAS 1998 Conference, Athens, Greece.
[7] Couaillier, V. and Collercandy, R., "Numerical Methods and Results for Turbulent Flows around Transport Aircraft," Proceedings of the ODAS Aerospace Symposium, June 1999, Paris, France.
[8] Couaillier, V., "Numerical Simulation of Separated Turbulent Flows Based on the Solutions of the RANS / Low Reynolds Number Two-Equation Model," AIAA-99-0154.
[9] Delanaye, M., Geuzaine, P. and Essers, J., "Development and Application of Quadratic Reconstruction Schemes for Compressible Flows on Unstructured Adaptive Grids," AIAA-97-2120.
[10] Geuzaine, P., Delanaye, M. and Liu Y., "Application of a High-Order Reconstruction Scheme to Turbulent Flow Calculations Using Hybrid-Cartesian Adaptive Unstructured Grids," AIAA-98-0546.
[11] Jameson A., Schmidt W. and Turkel E., "Numerical Simulation of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," AIAA-81-1259.
[12] Jameson, A., "Analysis and Design of Numerical Schemes for Gas Dynamics 2: Artificial Diffusion and Discrete Shock Structure," *International Journal of Computational Fluid Dynamics*, Vol. 5 (pp. 1-38), 1995.
[13] Liamis, N. and Couaillier, V., "Unsteady Euler and Navier-Stokes Flow Simulations with an Implicit Runge-Kutta Method," Proceedings of the ECCOMAS 1994 Conference, Stuttgart, Germany.
[14] Nozaki, O., Kikuchi, K., Nishizawa, T., Matsuo, Y., Hirai, K. and Kodama, H., "Three-Dimensional Viscous Analysis of Rotor-Stator Interaction in a Transonic Compressor," AIAA-99-0239.
[15] Press, W., Teukolsky, S., Vetterling, W. and Flannery, B., Numerical Recipes in Fortran, Cambridge University Press, 1992.
[16] Roe, P., "Approximate Riemann Solvers, Parameter Vectors and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2 (pp. 357-393), 1981.
[17] Vuillot, A., Couaillier, V. and Liamis N., "3D Turbomachinery Euler and Navier-Stokes Calculations with a Multi-Domain Cell-Centered Approach," AIAA-93-2576.
[18] Yao, J., Jameson, A., Alonso, J., and Liu, F., "Development and Validation of a Massively Parallel Flow Solver for Turbomachinery Flows," AIAA-2000-0882.

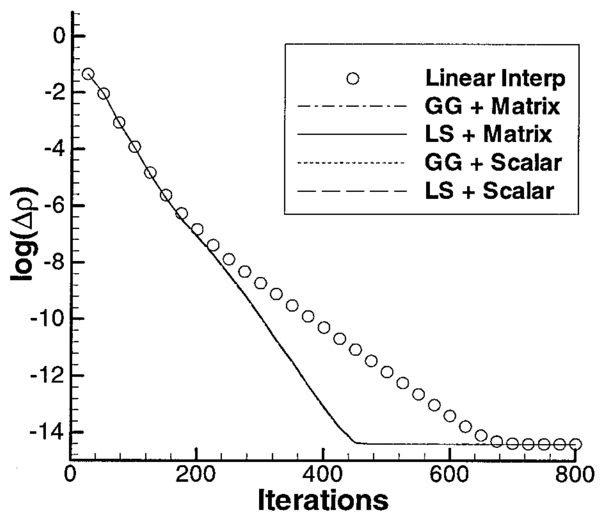

Figure 6 Residual history for stationary solutions.

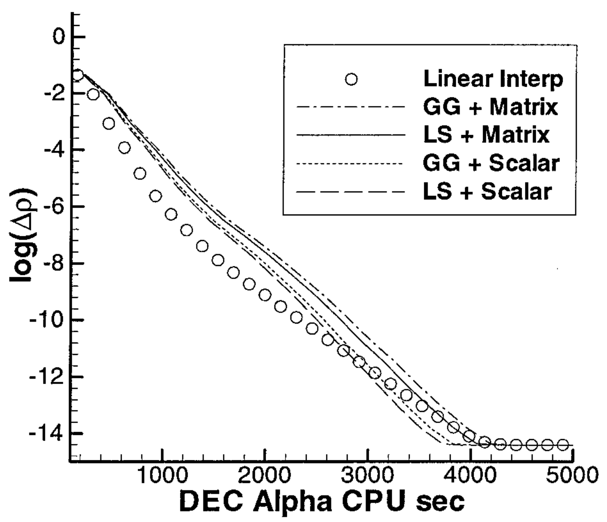

Figure 7 DEC Alpha CPU cost (not optimized).

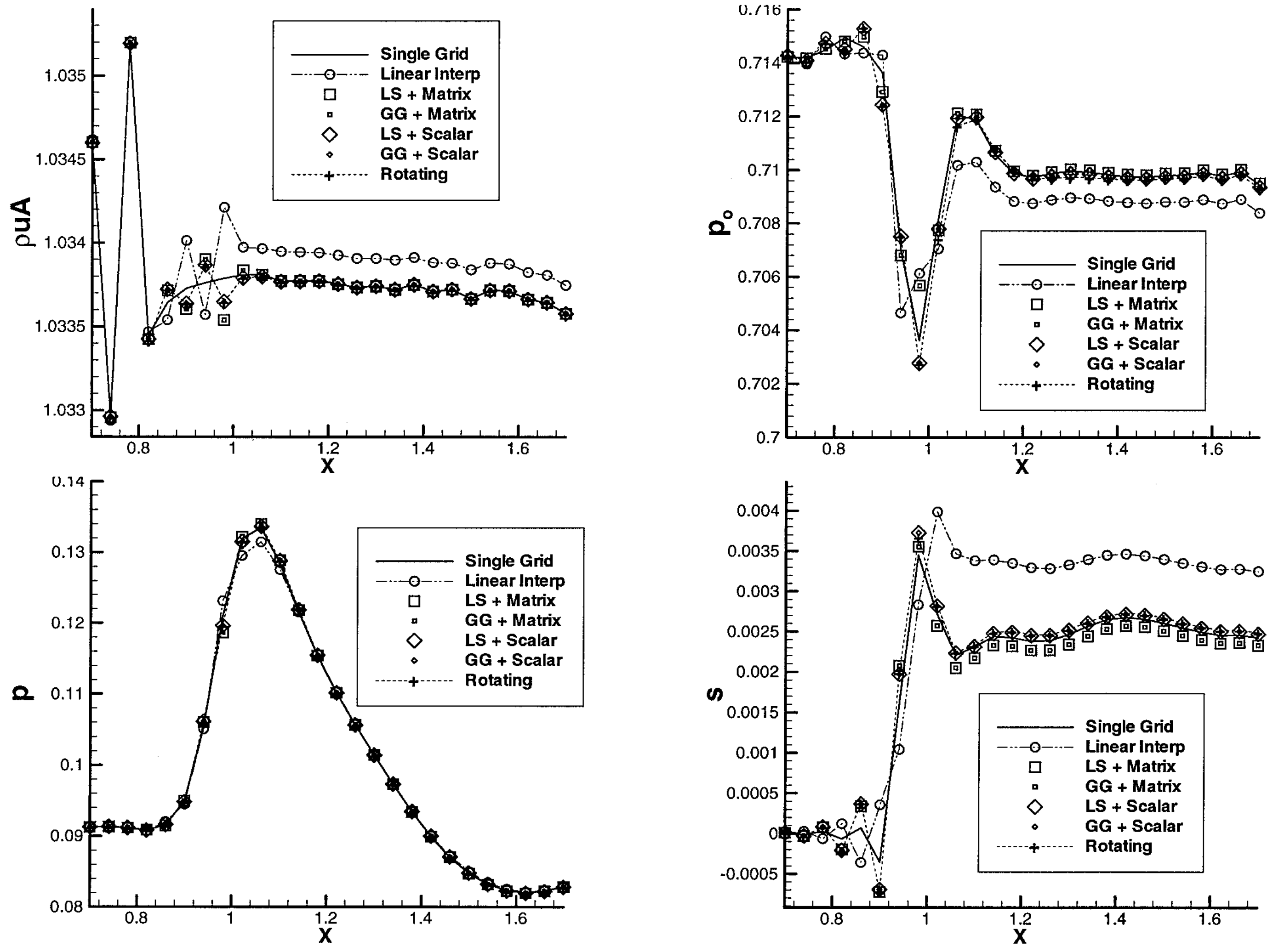

Figure 8 Line-plot comparisons at $(r_{inner}+r_{outer})/2$.

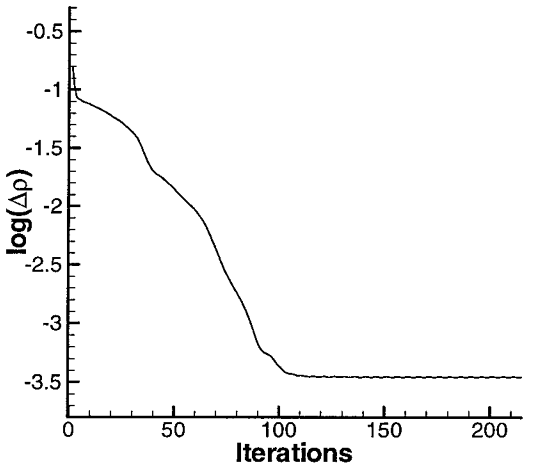

Figure 9 'Pseudo-unsteady' iteration history.