AIAA-00-0882

Development and Validation of a Massively Parallel Flow Solver for Turbomachinery Flows

J. Yao, A. Jameson, J. J. Alonso
Stanford University
Stanford, CA 94305

F. Liu
University of California, Irvine
Irvine, CA 92697

38th Aerospace Sciences Meeting and Exhibit
January 10-13, 2000/Reno, NV

Development and Validation of a Massively Parallel Flow Solver for Turbomachinery Flows

Jixian Yao* Antony Jameson† Juan J. Alonso‡
Department of Aeronautics and Astronautics
Stanford University

Feng Liu§
Department of Mechanical and Aerospace Engineering
University of California, Irvine

This paper presents the development and validation of the unsteady, three-dimensional, multiblock, parallel turbomachinery flow solver, TFLO. The Unsteady Reynolds Averaged Navier-Stokes (Unsteady RANS) equations are solved using a cell-centered discretization on arbitrary multiblock meshes. The solution procedure is based on efficient explicit Runge-Kutta methods with several convergence acceleration techniques such as multigrid, residual averaging, and local time-stepping. The algebraic Baldwin-Lomax, the one-equation Spalart-Allmaras, and the two-equation Wilcox k-ω turbulence models are implemented. The solver is parallelized using domain decomposition, an SPMD (Single Program Multiple Data) strategy, and the Message Passing Interface (MPI) Standard. A mixing model and a sliding mesh interface approach have been implemented to exchange flow information between blade rows in both steady and unsteady rotor/stator interaction flows. The dual-time stepping technique is applied to advance unsteady computations in time. This paper focuses heavily on the initial validation of the flow solver, TFLO, with emphasis on steady-state calculation of multiple blade-row flows. For validation and verification purposes, results from TFLO are compared with both existing experimental data and computational results from other software used in industry. The large set of cases tested increases our confidence in the ability of TFLO to accurately predict flows inside typical turbomachinery geometries, and sets the stage for the large-scale computation of unsteady, multiple blade-row flows.

1. Introduction

Computational Fluid Dynamic (CFD) simulations play an essential role in the design of modern gas turbine jet engines, providing engineering predictions of aerodynamic performance, heat transfer, and flow behavior. Steady-state flow predictions are commonplace for problems ranging in size from the design of an individual compressor or turbine blade, to large or whole sections of a complete component, such as a combustor or a low-pressure turbine. A few examples of steady, multi-stage turbomachinery flow prediction capability include those developed by Adamczyk et. al. [1], Dawes [2], Denton and Singh [3], Ni et. al. [4, 5] and Rhie et. al. [6, 7]. Adamczyk et. al. [8, 9], Hall [10], and others have extended the steady-flow prediction schemes to model various unsteady-flow effects in multi-stage turbomachinery. Multi-component steady-flow simulations have not yet become routine in the design process because of the exceedingly large resource requirements to perform these analyses and the disparate flow physics prevalent in each component.

The use of unsteady flow simulations in the design process remains limited to small sections of the engine, such as a single stage of a compressor or turbine. The main reasons for this lack of unsteady numerical results are the large computational requirements necessary to calculate the flow solution and the long integration times necessary to achieve meaningful time averages. As the size of an unsteady flow turbomachinery simulation increases beyond two or three blade rows, the overall required computer memory size and solution time rapidly

*Research Associate, Member AIAA
†T.V.Jones Professor of Engineering, AIAA Fellow
‡Assistant Professor, Member AIAA
§Associate Professor, Member AIAA

grow to a point where the simulation is no longer feasible with most available computer systems. This rapid growth is due to the different number of airfoils in each blade row and the requirement to maintain a constant and equal circumferential section along the entire axial length of the domain for the sake of circumferential periodicity. Examples of current unsteady, multi-stage turbomachinery flow prediction procedures include those of Arnone et. al. [11], Dorney et. al. [12], Giles [13, 14], Gundy-Burlet [15], Jorgenson and Chima [16], Lewis et. al. [17], Rai et. al. [18], and Rao et. al. [19]. Other components in the engine have similar requirements that make it just as difficult to utilize unsteady-flow simulations as a design tool.

Shared and distributed memory parallel computers have helped to greatly extend the feasible size and reduce the solution time of large-scale gas-turbine design and analysis problems. Many new advances in computer hardware, communication networks, and simulation software are required, however, to bring large-scale simulations into practical, everyday use. The design and analysis of gas turbine jet engines is not the only engineering or scientific arena that has encountered these bottlenecks. As a result, the Department of Energy (DoE) has launched the Accelerated Strategic Computing Initiative (ASCI) to promote the development of large massively-parallel computer systems and the simulation software that can take advantage of them.

As part of this initiative, the current effort has been focused on developing a new massively parallel computational fluid dynamic solution procedure. Although the simulation software has been written to allow for the computation of steady and unsteady flows in general configurations, the target application of the current effort has been gas turbine turbomachinery flows. The long-term goal is to develop the capability to simulate the steady or unsteady flow through an entire compressor or turbine component. Before reaching this goal, the fundamental ability of our program to predict some of the most basic turbomachinery flows needs to be addressed thoroughly. The purpose of this paper is to document the numerical, data structure, and parallel computing techniques used in the baseline procedure as well as the results of a series of validation test cases used to verify the prediction accuracy for different flow regimes.

2. Overview of TFLO

The Unsteady Reynolds Averaged Navier-Stokes (Unsteady RANS) equations are solved using a cell-centered discretization on arbitrary multiblock meshes. The solver is parallelized using domain decomposition, an SPMD (Single Program Multiple Data) strategy, and the Message Passing Interface (MPI) Standard.

The solution procedure is based on efficient explicit modified Runge-Kutta methods with several convergence acceleration techniques such as multigrid, residual averaging, and local time-stepping. These techniques, multigrid in particular, provide excellent numerical convergence and fast solution turnaround. Two numerical dissipation schemes have been implemented: the JST (Jameson-Schmidt-Turkel) switched scheme [20], and the more refined CUSP (Convective Upwind Split Pressure) dissipation model [21, 22], which provides sharper resolution of shock waves and contact discontinuities at a moderate increase in computational cost.

The multiblock strategy facilitates the treatment of arbitrarily complex geometries using a series of structured blocks with point-to-point matching at their interfaces. This point-to-point matching ensures that global conservation of the flow variables is preserved. The structure of the mesh is specified via a connectivity file which allows for arbitrary orientations of the blocks. Two layers of halo cells are used for inter-block information transfer and an efficient communication scheme is implemented for the halo cell data structures. The load of each processor is balanced on the basis of a combination of the amount of computation and communication that each processor performs. Communication of halo cell values is conducted at every stage of the Runge-Kutta integration and in every level of the multigrid cycle in order to guarantee fast convergence rates.

The resulting pre-processor and flow solver combination have been ported to a variety of today's most advanced parallel computers. TFLO is routinely run on the CRAY T3E at the Pittsburgh Supercomputing Center, on SGI Origin 2000 systems at both Stanford University and the Los Alamos National Laboratory, and on the latest IBM-SP systems at the Lawrence Livermore National Laboratory.

The solver incorporates a variety of boundary conditions and turbulence models which allow for a wide range of applications in internal flows. Currently supported boundary conditions include both viscous and inviscid solid walls, inflow, outflow, far-field, symmetry, circumferential periodic, open gap, flow through, and specified mass flux conditions. Additional boundary conditions will be incorporated as needed.

Several turbulence models have been implemented for the computation of the Reynolds stress. Current options include the Baldwin-Lomax algebraic model, the one-equation Spalart-Allmaras model, and the two-equation Wilcox k-ω model. These options are

being extended to include a family of k-ϵ models, and the Stanford-developed v^2-f model. The k-ω model is mostly used in this work and is strongly coupled with the Navier-Stokes solution procedure. A point-implicit treatment is applied to the source terms of the turbulence equations to achieve faster convergence.

For steady-state flow calculations in a multiple blade-row environment, the conservative flow variables at the mixing plane between adjacent blade rows are circumferentially averaged and exchanged. For unsteady calculations of rotor/stator interaction, the conservative flow variables at the sliding interface are suitably interpolated both upstream and downstream thus preserving the spatial variations in the flow field. The dual-time stepping technique [23, 24, 25, 26] is used for time-accurate simulations that account for the relative motion of rotors and stators as well as other sources of flow unsteadiness.

The main goal in the development of TFLO is to ensure that the solver can be scaled to large numbers of processors (in the thousands range) so that problems of far larger size than ever attempted before can be computed efficiently. These problems will involve more data, more complexity, and a succession of more powerful computing platforms. The issue is not simply bigger and faster, but rather a fundamental shift in the way problems are solved. For this purpose, the solver has to be robust, efficient, and must be able to predict the proposed problems accurately. However, our intention has also been to develop a code that can be used for small routine calculations that are repeatedly encountered during the process of component design.

Intensive validation has been an intrinsic part of TFLO's development. Some fundamental test cases, such as boundary layers on a flat plate, the Bachalo-Johnson transonic flow over a bump, the Hobson II cascade, and the vortex shedding over a circular cylinder have been computed to validate the characteristics of the basic solver and turbulence models in TFLO. Several turbomachinery test cases have been carefully selected to validate the predictions of multistage compressor and turbine flows. These test cases include the VPI turbine cascade, the 1.5-stage Aachen turbine, and the 3.5-stage Pennsylvania State University Research Compressor (PSRC).

2.1 TFLO Origins

TFLO has been developed from scratch using the experience in both turbomachinery computations and parallel computing gained during the development of two earlier simulation codes:

- FLO107-MB [27] is a general multiblock parallel Navier-Stokes solver that uses the Baldwin-Lomax turbulence model for the solution of external flows. It implements a multiblock strategy and has a scalable, well-tuned parallel algorithm for block-to-block communication.
- Turbo90 [28, 29, 30] is a single block, single processor Navier-Stokes solver for turbomachinery flows that uses the k-ω turbulence model.

2.2 Components and Main Features

The major components which comprise the software are a grid generator, a simplified through-flow solver for the generation of initial flow conditions, a grid blocker and duplicator, a multiblock pre-processor, the flow solver, TFLO, and a host of visualization and post-processing utilities. The main features of the complete system can be summarized as follows:

- Multiblock structured mesh with double halos at block boundaries.
- Parallel implementation using a Single Program Multiple Data (SPMD) strategy and the Message Passing Interface (MPI) Standard for communication.
- Pre-processor for domain decomposition and load balancing.
- Dual-time stepping for unsteady flow simulation.
- Moving mesh attached to rotors with sliding mesh interface between rotor and stator meshes.
- Modified Runge-Kutta time-stepping for steady-state simulation.
- Convergence acceleration via multigrid and implicit residual smoothing.
- JST and CUSP algorithms for artificial dissipation.
- Turbulence modeling: Baldwin-Lomax, Spalart-Allmaras, and Wilcox k-ω models.

3. Numerical Methods

3.1 Governing Equations

The governing equations solved by TFLO are formulated in a general moving coordinate system. Let $\mathcal{V}(t)$ be a moving control volume with bounding surface $\mathcal{S}(t)$. The functional dependence of $\mathcal{V}$ and $\mathcal{S}$ on t implies that the control volume and its boundary can be time dependent. Let ρ, $\vec{V}$, E be the density, absolute velocity and absolute total energy per

unit mass, respectively. Using a coordinate system which rotates with angular velocity $\vec{\omega}$ and neglecting body forces, the integral form of the unsteady Navier-Stokes equations can be shown to be

$$\frac{d}{dt}\int_{\mathcal{V}(t)} \mathbf{W} d\mathcal{V} + \oint_{\mathcal{S}(t)} \mathbf{F}\cdot\vec{n} dS = \oint_{\mathcal{V}(t)} \mathbf{S} d\mathcal{V}, \quad (1)$$

where

$$\mathbf{W} = \begin{bmatrix} \rho \\ \rho\vec{V} \\ \rho E \end{bmatrix}, \quad (2)$$

$$\mathbf{F} = \begin{bmatrix} \rho(\vec{V}-\vec{V}_b) \\ \rho\vec{V}(\vec{V}-\vec{V}_b) - \bar{\bar{\sigma}} \\ \rho E((\vec{V}-\vec{V}_b) - \vec{V}\cdot\bar{\bar{\sigma}} + \vec{q} \end{bmatrix}, \quad (3)$$

$$\mathbf{S} = \begin{bmatrix} 0 \\ \rho\vec{\omega}\times\vec{V} \\ 0 \end{bmatrix}. \quad (4)$$

According to Stokes' postulate, the total stress tensor can be written as

$$\sigma_{ij} = -p\delta_{ij} + \tau_{ij}, \quad (5)$$

where τ_{ij} is the shear stress tensor

$$\tau_{ij} = 2\mu S_{ij} - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}, \quad (6)$$

and S_{ij} is the strain rate tensor defined by

$$S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \quad (7)$$

The coefficient of viscosity of the fluid is represented by μ, and p is the thermodynamic pressure that can be related to the conservative variables by an equation of state. Under a calorically and thermally perfect gas assumption, we have

$$p = (\gamma-1)\rho(E - \frac{1}{2}\vec{V}\cdot\vec{V}); \quad p = \rho RT, \quad (8)$$

where R is the universal gas constant, and γ is the specific heat ratio.

Using Fourier's law the heat flux vector is given by

$$\vec{q} = -\kappa\nabla T, \quad (9)$$

where κ is the Fourier coefficient of heat conductivity.

3.2 Discretization

A cell-centered finite volume scheme is used to discretize the governing equations. Upwind biasing is achieved with the addition of numerical dissipation. The current version of TFLO uses either a switched scalar dissipation scheme (Jameson-Schmidt-Turkel, JST) or the more sophisticated Convective Upstream Split Pressure (CUSP) scheme, coupled with an Essentially Local Extremum Diminishing (ELED) formulation. Details of these techniques and an extensive validation of the schemes for both inviscid and viscous flows, can be found in [21, 22, 31].

3.3 Dual-Time Stepping

When unsteady flows are considered, the Navier-Stokes equations must be marched forward in time in a time-accurate fashion. Two basic alternatives exist: one can advance the system forward in time using either an explicit or an implicit method.

Explicit methods require the selection of the time-accurate time-step based on numerical stability requirements. Except for very high frequency flow phenomena, the time-step required for reasonable accuracy of the calculation is often much larger than the limit imposed by numerical stability considerations.

Implicit methods are, in general, more costly per time step, but set less restrictive limitations on the allowable time-step. Therefore, they allow the selection of the time-step based on the characteristic frequency of the physical phenomena to be resolved. Implicit methods also pose additional difficulties to the efficient parallelization of the scheme.

The dual-time stepping technique combines the advantages of both implicit methods and the fast solution techniques that have been developed for steady-state solutions (multigrid, implicit residual smoothing etc).

The governing equations (1) can be discretized implicitly as follows:

$$D_t(\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)}) + \mathcal{R}(\mathbf{W}^{(n+1)}) = 0, \quad (10)$$

where D_t is a k-th order accurate backward difference operator of the form proposed by Jameson [23].

$$D_t = \frac{1}{\Delta t}\sum_{q=1}^{k}\frac{1}{q}\left[\Delta^-\right]^q, \quad (11)$$

where

$$\Delta^-\mathbf{W}^{(n+1)} = \mathbf{W}^{(n+1)} - \mathbf{W}^{(n)}.$$

The superscripts $(n+1)$ denote the time level $(n+1)\Delta t$ at which the flow variables are considered, and $\mathcal{R}(\mathbf{w})$ is the symbol for the combined convective and dissipative residuals (which can include viscous residuals as well). Following Melson et al. [32], the

physical time derivative operator can be written as follows

$$
\begin{aligned}
& D_t(\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)}) \\
&= \frac{1}{\Delta t}\sum_{m=0}^{M} a_m \mathbf{W}^{(n+1-m)}\mathcal{V}^{(n+1-m)} \\
&= \frac{1}{\Delta t}[a_0 \mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)} \\
&+ E(\mathbf{W}^{(n)}\mathcal{V}^{(n)}, \\
&\quad \mathbf{W}^{(n-1)}\mathcal{V}^{(n-1)}, \cdots, \\
&\quad \mathbf{W}^{(n+1-M)}\mathcal{V}^{(n+1-M)})].
\end{aligned}
$$

where the operator E represents the part of the time derivative operator that is due to the values of the flow variables and cell volumes at previous time-steps, and is therefore a fixed source term in the solution at each time-step.

The second-order time discretization is then given by:

$$
\begin{aligned}
D_t(\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)}) &= \frac{3}{2\Delta t}\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)} \\
&- \frac{2}{\Delta t}\mathbf{W}^{(n)}\mathcal{V}^{(n)} \\
&+ \frac{1}{2\Delta t}\mathbf{W}^{(n-1)}\mathcal{V}^{(n-1)},
\end{aligned}
$$

with

$$
a_0 = \frac{3}{2}, \quad E = -2\mathbf{W}^{(n)}\mathcal{V}^{(n)} + \frac{1}{2}\mathbf{W}^{(n-1)}\mathcal{V}^{(n-1)},
$$

and the third-order discretization is given by:

$$
\begin{aligned}
D_t(\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)}) &= \frac{11}{6\Delta t}\mathbf{W}^{(n+1)}\mathcal{V}^{(n+1)} \\
&- \frac{3}{\Delta t}\mathbf{W}^{(n)}\mathcal{V}^{(n)} \\
&+ \frac{3}{2\Delta t}\mathbf{W}^{(n-1)}\mathcal{V}^{(n-1)} \\
&- \frac{1}{3\Delta t}\mathbf{W}^{(n-2)}\mathcal{V}^{(n-2)}.
\end{aligned}
$$

with

$$
\begin{aligned}
a_0 &= \frac{11}{6}, \\
E &= -3\mathbf{W}^{(n)}\mathcal{V}^{(n)} \\
&+ \frac{3}{2}\mathbf{W}^{(n-1)}\mathcal{V}^{(n-1)} \\
&- \frac{1}{3}\mathbf{W}^{(n-2)}\mathcal{V}^{(n-2)}.
\end{aligned}
$$

Let $\mathbf{W}^l$ denote the l-th iterative approximation to the flow solution at time step $(n+1)$, such that

$$
\mathbf{W}^{(n+1)} = \lim_{l\to\infty} \mathbf{W}^l.
$$

Equations (10) can now be rewritten as follows:

$$
\frac{a_0}{\Delta t}\mathbf{W}\mathcal{V}^{(n+1)} + \frac{E}{\Delta t} + \mathcal{R}(\mathbf{W}) = 0. \qquad (12)
$$

These non-linear coupled ordinary differential equations can be re-cast into a modified steady-state calculation for each implicit time-step as follows:

$$
\frac{\partial \mathbf{W}}{\partial t^*} + \mathcal{R}^*(\mathbf{W}) = 0, \qquad (13)
$$

where the modified residual, $\mathcal{R}^*(\mathbf{W})$ contains the usual steady-state residual with the addition of two source terms that arise from the discretization of the time derivative operator

$$
\mathcal{R}^*(\mathbf{W}) = \frac{a_0}{\Delta t}\mathbf{W} + \frac{1}{\mathcal{V}^{(n+1)}}\left[\frac{E}{\Delta t} + \mathcal{R}(\mathbf{W})\right], \qquad (14)
$$

and t^* denotes the fictitious time used to reach a pseudo steady state which advances the solution forward in time from $t = n\Delta t$ to $t = (n+1)\Delta t$. The solution $\mathbf{W}$ can be marched in fictitious time through successive approximations $\mathbf{W}^l$ with inner time step Δt^* until a steady state in pseudo-time is reached. Once this is accomplished the solution vector $\mathbf{W}$ which satisfies equation (13) is actually the new solution at time-step $(n+1)$, $\mathbf{W}^{(n+1)}$ that we were looking for. Repeating this procedure at every time step, the time accurate behavior of the flow can be predicted as a sequence of pseudo-time steady-state solutions.

Notice that when advancing Equation (13) in pseudo-time, if $\frac{\Delta t^*}{\Delta t}$ is a small number, the product of the modified residual with the fictitious time step $\Delta t^*\mathcal{R}^*(\mathbf{W})$ differs from the steady-state residual times its time step by a small perturbation only, and, therefore, the solution methods used for steady-state problems ought to be almost as efficient for this problem. It must be pointed out that it is precisely these problems in which the physical time-step Δt is much larger than the fictitious numerical time-step Δt^* where the performance of an implicit discretization is at its best.

The second order accurate discretization is used for the unsteady simulation presented in this paper, as it is considered to be a good compromise between accuracy, memory requirements, stability, and robustness of the scheme. For some types of problems, the third order discretization may be a better choice. The stability and robustness of the dual-time stepping scheme has also been verified by other researchers [26, 33, 24, 32, 34, 35, 36].

The physical time-step used to advance the time-accurate solution in all the cells in the domain must be the same independently of the size of each of these cells. In the inner iteration, however, we are free to choose the pseudo time-step within the restrictions

imposed by numerical stability of the scheme. It pays off to use locally varying pseudo time-steps such that each cell in the domain is running close to its stability limit. For each cell, the time-step for the inner iteration is chosen as

$$\Delta t^* = \frac{\mathrm{CFL}}{\lambda_\xi + \lambda_\eta + \lambda_\zeta}, \tag{15}$$

where λ_ξ, λ_η, and λ_ζ are the spectral radii of the flux Jacobians of the governing equations in each of the three coordinate directions.

It must be mentioned that, although the overall formulation for the physical time integration is implicit in nature, advancement in time is driven by an explicit inner iteration. It is this characteristic of the dual-time stepping algorithm that allows for the use of well-tested explicit convergence acceleration techniques, such as multigrid and residual averaging, in order to obtain faster convergence of the pseudo-time iterations. The computational advantages of the dual-time stepping scheme are due, in large part, to the use of the multigrid technique; without it, a large number of iterations would be required to converge the flow at each physical time-step. Moreover, a second advantage of the explicitness of the inner iteration is that it allows for the possibility of a straightforward, efficient parallelization. Because of these two reasons, the dual-time method is perfectly suited for the computation of unsteady problems.

3.4 Time-Stepping Scheme

An m-stage generalized Runge-Kutta scheme can be formulated as follows:

$$\begin{aligned} \mathbf{W}^{(0)} &= \mathbf{W}^l \\ \mathbf{W}^{(k)} &= \mathbf{W}^{(0)} - \alpha_k \Delta t^* \mathcal{R}^*(\mathbf{W}^{k-1}) \\ \mathbf{W}^{l+1} &= \mathbf{W}^{(M)}. \end{aligned}$$

The residual, $\mathcal{R}^*$, includes contributions from the convective and dissipative residuals, as well as from the time derivative discretization. In order to optimize the smoothing properties of the scheme, the convective and dissipative parts of the original residual at the k-th stage $\mathcal{R}(\mathbf{W}^k)$ are treated separately in the following fashion:

$$\begin{aligned} \mathcal{R}(\mathbf{W}^k) &= \mathcal{Q}(\mathbf{W}^k) + \tilde{\mathcal{D}}(\mathbf{W}^k) \\ \tilde{\mathcal{D}}(\mathbf{W}^k) &= \beta_k \mathcal{D}(\mathbf{W}^k) + (1-\beta_k)\mathcal{D}(\mathbf{W}^{k-1}) \end{aligned}$$

where $\mathcal{Q}$ and $\mathcal{D}$ are the convective and diffusive flux balances, respectively.

The coefficients α_k are chosen to maximize the stability region along the imaginary axis, and the coefficients β_k are chosen to increase the stability interval along the negative real axis. A 5-stage scheme is used in the current code with coefficients

$$\begin{aligned} \{\alpha_k\} &= \{\tfrac{1}{4}, \tfrac{1}{6}, \tfrac{3}{8}, \tfrac{1}{2}, 1\}, \\ \{\beta_k\} &= \{1.0, 0.0, 0.56, 0.0, 0.44\}. \end{aligned}$$

The term, $\frac{a_0}{\Delta t}\mathbf{W}\mathcal{V}^{(n+1)}$, which has been lumped into the $\mathcal{R}^*$ operator and that originates from the discretization of the time derivative term (see Equation 12) can be treated implicitly within the Runge-Kutta integration since it is only a diagonal term, and a simple division is required. The Runge-Kutta integration is then reformulated as follows:

$$\begin{aligned} \mathbf{W}^{(0)} &= \mathbf{W}^l \\ \mathbf{W}^{(k)} &= \frac{1}{(1+\alpha_k\overline{\lambda})}(\mathbf{W}^{(0)} + \alpha_k\overline{\lambda}\mathbf{W}^{k-1} \\ &- \alpha_k \Delta t^* \mathcal{L}[\mathcal{R}^*(\mathbf{W}^{k-1})]) \\ \mathbf{W}^{l+1} &= \mathbf{W}^{(M)}, \end{aligned} \tag{16}$$

where $\overline{\lambda} = \frac{a_0 \Delta t^*}{\Delta t}$, and the operator $\mathcal{L}$ performs implicit residual averaging on the complete residual $\mathcal{R}^*(\mathbf{W})$ which vanishes as the solution converges to a steady state in pseudo-time.

3.5 The Wilcox k-ω Turbulence Model

The reformulated k-ω equations used in the code are given by

$$\underbrace{\frac{\partial}{\partial t}\begin{bmatrix}\rho k \\ \rho\omega\end{bmatrix}}_{\text{rate of change}} + \underbrace{\frac{\partial}{\partial x_j}\begin{bmatrix}\rho k \\ \rho\omega\end{bmatrix}(u-u_b)_j}_{\text{convection}} =$$

$$\underbrace{\begin{bmatrix}\mu_t P_d \\ \alpha^*\alpha\rho P_d\end{bmatrix} - \begin{bmatrix}\frac{2}{3}\rho k\Delta \\ \alpha\frac{2}{3}\rho\omega\Delta\end{bmatrix}}_{\text{production}} -$$

$$\underbrace{\begin{bmatrix}\beta^*\rho\omega k \\ \beta\rho\omega\omega\end{bmatrix}}_{\text{dissipation}} + \underbrace{\frac{\partial}{\partial x_j}\begin{bmatrix}(\mu+\sigma^*\mu_t)\frac{\partial k}{\partial x_j} \\ (\mu+\sigma\mu_t)\frac{\partial\omega}{\partial x_j}\end{bmatrix}}_{\text{diffusion}},$$

where

$$P_d = \frac{1}{2}\left[(\epsilon_{11}^2+\epsilon_{22}^2+\epsilon_{33}^2) + 2(\epsilon_{12}^2+\epsilon_{13}^2+\epsilon_{23}^2)\right] \geq 0,$$

$$\epsilon_{ij} = \left[2S_{ij} - \frac{2}{3}\Delta\delta_{ij}\right],$$

$$\Delta = \frac{\partial u_k}{\partial x_k} = \frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3},$$

and

$$\mu_t = \left(\alpha^*\frac{\rho k}{\omega}\right).$$

The basic algorithm for solving the k-ω equations follows that proposed by Liu and Zheng [28] except that a non-staggered finite volume discretization is used here. The k-ω equations are solved at

every stage of Runge-Kutta scheme, and are closely coupled with the Navier-Stokes equations. The k-ω equations are also solved on all multigrid levels to accelerate convergence. Communication of k and ω is performed simultaneously with the conservative flow variables at all block interfaces. The implicit residual smoothing technique is also applied to the k-ω equations.

The k and ω variables are defined at cell centers in a similar fashion to the main flow quantities. The velocity derivatives, which are used to calculate production and dilation terms at cell centers, are obtained at the vertices of the cell using the values in all of the eight cell centers that surround a given node. The convection terms are discretized with a second order MUSCL-type upwinding scheme (first order in coarser grids of the multigrid sequence).

The time integration for the k-ω equations requires special attention. The semi-discrete k-ω equations can be written as

$$\frac{\partial}{\partial t}(\rho k) + R_k(\rho k, \rho\omega) = 0$$

$$\frac{\partial}{\partial t}(\rho\omega) + R_\omega(\rho k, \rho\omega) = 0,$$

where R_k and R_ω are the residuals for the k and ω equations, respectively:

$$R_k(\rho k, \rho\omega) = \frac{1}{Vol}(C_k - D_k) - S_k$$

$$R_\omega(\rho k, \rho\omega) = \frac{1}{Vol}(C_\omega - D_\omega) - S_\omega.$$

C_k and C_ω are the discrete forms of the convective terms in the k and ω equations, and D_k and D_ω are the corresponding diffusive terms; S_k and S_ω are the discrete forms of the source terms which can be written as

$$S_k = \mu_t P_d - \frac{2}{3}(\nabla\cdot u)(\rho k) - \beta^*\frac{\omega}{k}\frac{(\rho k)^2}{\rho}$$

$$S_\omega = \alpha\alpha^*\rho P_d - \alpha\frac{2}{3}(\nabla\cdot u)(\rho\omega) - \frac{\beta}{\rho}(\rho\omega)^2.$$

The $\mu_t P_d$ and $\alpha\alpha^*\rho P_d$ terms are the major production terms for k and ω and are always positive. The $\frac{2}{3}(\nabla\cdot u)(\rho k)$ and $\alpha\frac{2}{3}(\nabla\cdot u)(\rho\omega)$ terms also contribute to the production of k and ω, but, however, may be either positive or negative. When the flow is undergoing an expansion $\nabla\cdot u > 0$, they dissipate k or ω. Conversely, when the flow is undergoing a compression, they produce k or ω. The $\frac{\beta^*}{\rho}(\rho\omega)(\rho k)$ and $\frac{\beta}{\rho}(\rho\omega)^2$ terms are dissipative terms which are always negative and thus annihilate k and ω. The larger these terms, the faster k and ω decay. In this case, the turbulence equations unfortunately become less stable because of larger negative eigenvalues. The update of the k and ω equations within each stage of Runge-Kutta scheme is modified to treat part of the source terms implicitly and therefore obtain increased numerical stability:

$$\left[\frac{1}{\Delta t}I - \frac{\partial S}{\partial W}\right]\Delta W = R(W) + S(W)$$

$$\frac{\partial S}{\partial W} = \begin{bmatrix} -2\beta^*\omega - \Delta^+ & 0 \\ 0 & -2\beta\omega - \alpha\Delta^+ \end{bmatrix},$$

where

$$\Delta^+ = \max(0, \frac{2}{3}\nabla\cdot u).$$

Even if the numerical scheme can guarantee the positivity of k and ω, the computation may lead to low levels of ω which are not physical, resulting in excessively large values of eddy viscosity. To prevent this, we take

$$(\rho\omega)_{min} = \alpha\alpha^*\rho\sqrt{P_d}.$$

The turbulence production term P_d is calculated using the mean rates of strain only on the finest grid, and is then injected to the coarser meshes in the multigrid sequence. Certain limiters for k and ω are applied when the solution is updated. This ensures the proper transfer of the residuals of the turbulence equations during the multigrid process.

The free-stream boundary condition for k and ω used currently can be described as follows:

$$k_\infty = \frac{1}{2}(q_\infty T_u)^2, \quad \omega_\infty = \alpha_\omega 10\frac{q_\infty}{L},$$

where $T_u = \frac{\sqrt{2k}}{q_\infty}$ is the freestream turbulent intensity and L is the turbulence length scale. T_u and α_ω are also prescribed.

For solid walls, the value of k at the first halo cell is set to the negative value of the first real cell inside the domain to ensure that $k = 0$ at the wall. The value of ω at the first cell away from the wall is set to $\omega_2 = \frac{6\mu_w}{\beta\rho y^2}$, which is valid for $y+ < 2.5$. In the first halo cell outside of the domain, ω is set to $\omega_1 = \frac{5}{2}\frac{6\mu_w}{\beta\rho y^2}$, where y is the distance from the first cell center to the wall.

On coarser grids, k and ω values at the first real and halo cells are transfered directly from the fine mesh.

4. Boundary Conditions

The boundary conditions currently implemented in TFLO are:

- linear extrapolation of flow variables
- solid inviscid boundary
- XZ symmetry plane
- XY symmetry plane
- YZ symmetry plane
- far field
- solid viscous adiabatic wall
- inflow
- outflow (fixed back pressure and non-reflection)
- periodic boundary
- inter-blade row interface

Since each processor in the calculation (as we will see later) will typically be responsible for more than one block, and since similar boundary conditions can be imposed on more than one face of each of these blocks, an unstructured approach to the implementation of the boundary conditions was followed. In the pre-processor, lists for each boundary condition type are constructed and initialized. These lists are later used in TFLO such that each processor can impose all boundary conditions of the same type, independently of the number of blocks, in a single loop.

Periodic boundary conditions require the transfer of both the values of the main flow and turbulence variables, and the rotation angle through which the velocity vectors need to be transformed. Special data structures are created in the pre-processor to deal with the communication in an efficient fashion.

5. Multiblock Domain Decomposition

In order to apply the finite volume technique to the solution of flows around complex configurations, we have chosen to implement a multiblock strategy. In a multiblock environment, a series of structured blocks of varying sizes is constructed such that these blocks fill the complete space and conform to the surface of the geometry of interest. This segmentation of the complete domain into smaller blocks avoids the topological problems in constructing a grid around complex configurations and multiply connected regions. The general strategy in the solution procedure of the multiblock flow solver is to construct a halo of cells that surrounds each block and contains information from cells in the neighboring blocks. This halo of cells, when updated at appropriate times during the numerical solution procedure, allows the flow calculation inside each block to proceed independently of the others.

This approach requires the identification of halo cells adjacent to block boundaries and the construction of lists of halo cells and their internal counterparts in the global mesh. In TFLO, we have chosen to carry out these setup procedures as part of a pre-processing module. During the pre-processing step, a two-level halo of cells is added around each block. The requirement of this double halo results from the need to calculate all the necessary fluxes for the internal cells of each block without reference to additional cell locations outside the block in question. In particular, the second differences used for the third-order artificial dissipation terms require the values of the flow variables in the two neighboring cells on all sides of any given cell.

The conservation laws (Equation 1) are applied to all cells in each block. The time integration scheme follows that used in the single block solver [20]. The solution proceeds by performing the cell flux balance, updating the flow variables, and smoothing the residuals at each stage of the time-stepping scheme and at each level of the multigrid cycle. The main difference in the multiblock integration strategy is the need to loop over all blocks during each stage of the process. The addition of the double-halo of cells around each block permits standard single-block subroutines to be used, without modification, for the computation of the flow field within each individual block. This includes the single-block subroutines for convective and dissipative flux discretization, viscous discretization, multistage time-stepping, and multigrid convergence acceleration.

The only difference between single block and multiblock strategies is in the implementation of the residual averaging technique. In the single-block solution strategy, tridiagonal systems of equations are set up and solved using flow information from the entire grid. Thus, each residual is replaced by a weighted average of itself and the residuals of its neighbors in the entire grid. In the multiblock strategy, the support for residual smoothing is reduced to the extent of each block. This eliminates the need to solve scalar tridiagonal systems spanning the blocks, which would incur a penalty in communication costs. Depending on the topology of the overall mesh, the setup of tridiagonal systems that follow coordinate lines may lose the physical interpretation it had in the single block implementation. This change has no effect on the final converged solution, and in all applications of the solver has not led to any significant

reduction in the rate of convergence.

Inter-block communication is performed at every stage of the Runge-Kutta scheme and at every level of the multigrid sequence so that the convergence characteristics of the multiblock parallel scheme are similar to those of a typical single-block solver.

A connectivity file is used to specify the topology of all the blocks in the domain. The orientation of the indices (I,J,K) for each block can be arbitrarily specified. Blocks and surfaces are grouped for boundary condition treatment, post-processing, and visualization.

6. Parallelization and Load Balancing

The previous section describes the partition of the domain into a series of connected blocks that can handle arbitrarily complex geometries. Once the complete flow domain has been partitioned, the computational workload has to be divided evenly among the processors participating in the calculation. The multiblock decomposition provides a natural approach to parallelization by assigning complete blocks to different processors in the parallel computer. This coarse grained parallelism is easily handled using the double-halo construct mentioned above and the MPI message passing library. Within each processor, fine-grain parallelism can be obtained using compiler-assisted techniques, such as OpenMP or multithreading, which are available on most platforms. We believe that this paradigm of parallel implementation used in TFLO maps extremely well to current and future generations of high-performance parallel computers.

Because entire blocks are assigned to any of the N processors participating in the computation, if our mesh has a number of blocks equal to M, we are restricted to using a maximum number of processors $N = M$. This limitation has not typically presented any problems, since our pre-processing software has the ability to decompose the original mesh into very large numbers of blocks depending on the size of the complete mesh. In practice, each and every one of the N processors participating in the calculation is assigned several blocks so that each processor virtually runs a copy of a multiblock flow solver.

The load balancing problem is that of *assigning* a set of M arbitrarily connected blocks of varying sizes to N processors of a parallel machine in a way that maximizes the overall computational performance. Load balancing is performed statically during the pre-processing step of TFLO. We have experimented with several load balancing algorithms and have settled on the one that distributes the blocks among participating processors in such a way that the differences between the amount of computation *and* communication that the processors must perform is a minimum. This algorithm has a tendency to place approximately the same number of cells in each processor (this is not always possible because of varying block sizes), but, in addition, attempts to place blocks that share a physical interface (and therefore must communicate with each other) on the same processor, thus decreasing the cost of communication. It is the combination of these two criteria, together with a careful implementation of the communication algorithm, that allows TFLO to obtain high parallel efficiency and to scale to large numbers of processors. A detailed explanation of all the techniques used to improve the parallel performance of the code has been presented earlier and can be found in [27].

7. Inter-Blade Row Interface

For multiple blade-row calculations we have chosen to use meshes that are attached to the individual blades (either rotors or stators) in the calculation. Due to the relative motion of rotors and stators, by design TFLO was required to handle areas of the overall mesh which share a common surface, but that slide over each other as the rotors turn. These surfaces are termed sliding mesh interfaces, and a special boundary condition module was developed in TFLO to handle this situation in a completely general multiblock, multiprocessor environment. The sliding mesh interface implementation is responsible for the exchange of all necessary information (flow variables, turbulence variables, etc.) across these surfaces in a way that does not decrease the parallel efficiency of the overall computation. Furthermore, in TFLO, the sliding mesh interface module is used for both multiple blade-row steady-state calculations and truly unsteady rotor/stator interaction situations. For the former, a mixing plane boundary condition was developed that requires the azimuthal average of the flow and turbulence variables at each radial station. For the latter, careful interpolation must be performed to transfer the details of the flow features in the upstream and downstream directions.

In a complex, general, multiblock-parallel environment, there is the added difficulty that the sliding mesh interface will usually be composed of faces from many different blocks with potentially arbitrary orientations and which will most likely reside in different processors across the parallel computer. Because of this situation, information has to be collected in the right order by one processor, and then, the relevant pieces of the information need to be distributed to the right blocks/processors.

Relative to each sliding interface we find upstream

and downstream sides. Figure 1 shows a simple sketch of the data structure used in TFLO. The idea of a *data pool* is defined in order to abstract the concept of the sliding mesh interface in the presence of multiple blocks and processors. A *data pool* is simply a two-dimensional arrangement of data that has an identity provided by the fact that the totality of the data is ordered and lies on a sliding mesh interface. Data pools can be both physical and fictitious (if they correspond to halo cells).

In the example below, the data pools #1 and #3 are used to exchange information from the upstream to the downstream sides of the sliding interface, while data pools #2 and #4 are used in the reverse direction. Data pools #1 and #2 are called source pools since they provide information about the physical flow variables that need to be transferred, while data pools #3 and #4 are called destination pools since they receive the information into the halo cells upstream and downstream of the sliding mesh interface.

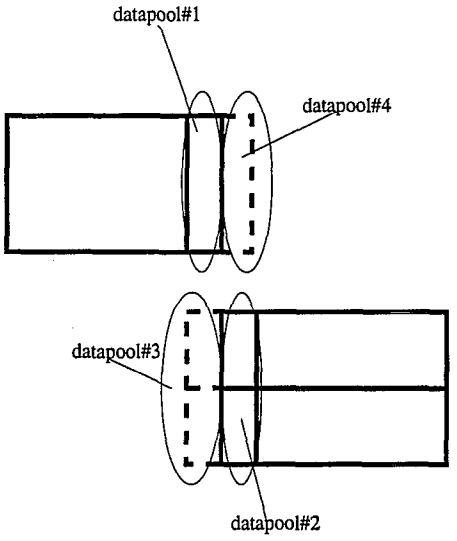


Figure 1: Data pools of a sliding interface

In order to simplify the data exchange operation, we currently require that, on both sides of the sliding mesh interface, the mesh lines in the circumferential direction be perfectly circular, and that the various radii of these circular mesh lines (in the blade spanwise direction) match upstream and downstream of the interface. This constraint, which is natural for turbomachinery flows, reduces the interpolation procedure to a one-dimensional problem. More advanced two-dimensional interpolation schemes are currently being pursued.

8. Validation

This section presents a range of fundamental test cases that have been chosen to validate the accuracy of the flow solver, TFLO.

8.1 Transonic Flow over a Bump

In order to verify the shock capturing properties of the numerical scheme, calculations were performed for the transonic flow over a circular arc bump on the lower wall of a channel. The bump has a 20.32 cm chord and a thickness of 1.905 cm. The flow field is inviscid with an exit Mach number of 0.875. Figure 2 shows the static pressure distribution on the lower channel wall for the fully converged solution. The results for both the JST and CUSP schemes are very similar. Note, however, that the CUSP scheme shows superior shock resolution with a single interior point as predicted by the theory [21, 22]. By reducing the amount of numerical dissipation introduced, the CUSP scheme achieves higher accuracy. The results were computed on a mesh of size $161 \times 73 \times 9$.

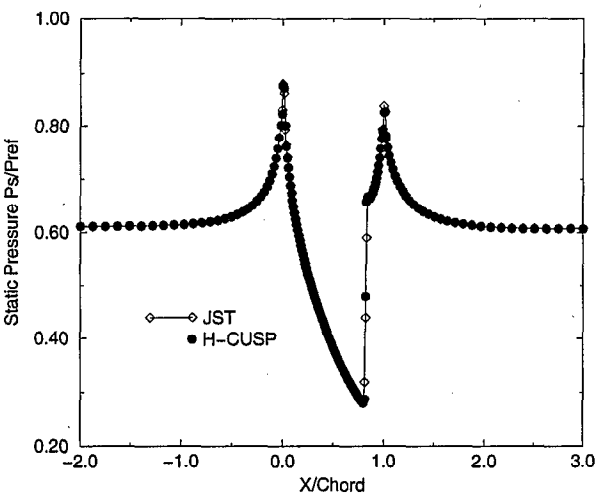


Figure 2: Static pressure profile on the lower wall

8.2 Hobson II Cascade

As a second test of the accuracy of the basic numerical scheme in TFLO, the inviscid flow through the Hobson II cascade was computed. This is a transonic shock-free cascade which is very sensitive to the accuracy of the numerical scheme. It's widely used as a test case for shock capturing schemes because numerical errors introduced by the solution procedure will cause a shock wave to appear. Figure 3 shows the pressure contours and the isentropic Mach number on the cascade surfaces. The pressure field and Mach number profiles are very close to being symmetric, showing a very small amount of dissipative loss.

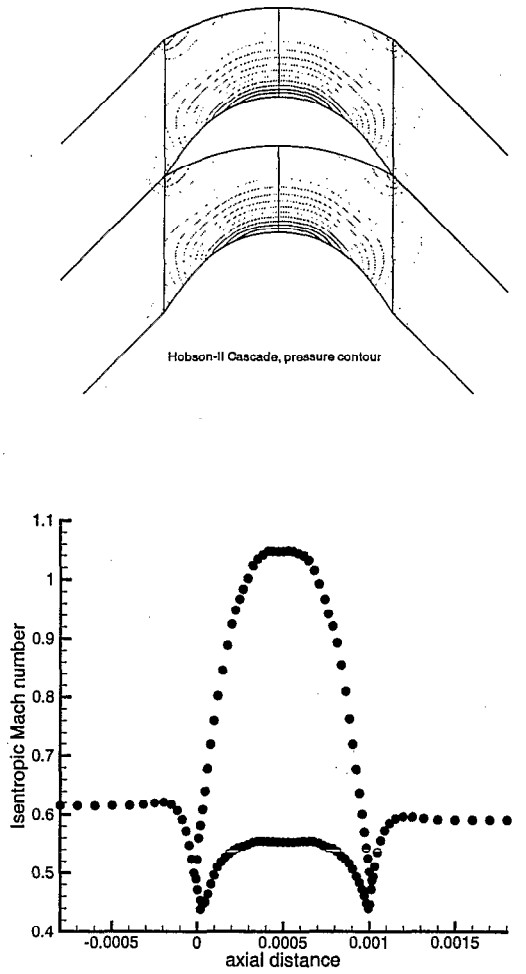


Figure 3: Pressure contours and isentropic Mach number distribution for Hobson II cascade

8.3 Viscous Flow over a Flat Plate

8.3.1 Laminar Boundary Layer

A laminar flow calculation was performed to verify the accuracy of the viscous discretization in the Navier-Stokes solver of TFLO. The geometry for this test case is a single passage of a flat plate cascade where the downstream end of the domain coincides with the trailing edge of the flat plates. The two flat plates in the geometry have unit chord and pitch, and a span of 0.1. A periodicity boundary condition was imposed on the portions of the mesh ahead of the leading edges of each of the flat plates, while inviscid flow tangency boundary conditions were imposed on the front and back faces (hub and case) of the domain. On the plate surfaces, a viscous, no-slip, adiabatic boundary condition was specified. The size of the mesh used in this calculation was $129 \times 9 \times 65$ in the axial, radial, and tangential directions respectively. Notice that the radial (spanwise) direction should have no flow variation and, therefore, only a small number of cells was required. The inlet Mach number was set to 0.3, and the Reynolds number based on the length of the plate was 35,000.

Figure 4 shows the calculated skin friction, boundary layer thicknesses, velocity, and viscous stress profiles. The results agree very well with the Blasius solution and show the expected similarity at various streamwise stations along the plate. Similarity and accuracy is also verified for the velocity component normal to the flat plate which is much harder to capture accurately due to its small magnitude.

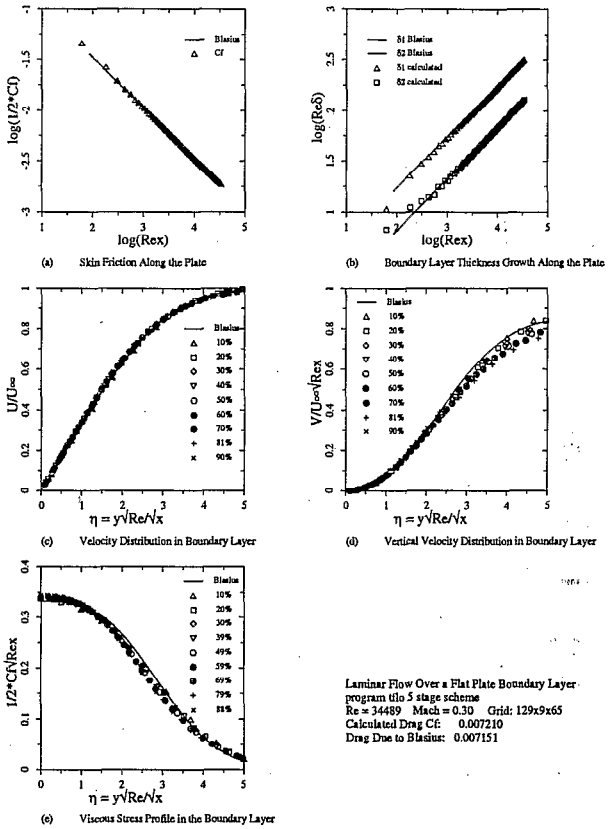


Figure 4: Laminar boundary layer on a flat plate

8.3.2 Turbulent Boundary Layer

The accuracy of the turbulence models in TFLO was initially tested using the case of a turbulent boundary layer over a flat plate in a zero pressure gradient. In this section, results are compared with those from NASA's CFL3D code using exactly the same grid. Unlike the cascade-type grid used for the laminar flow calculation, the grid in this case had a rectangular geometry with the flat plate occupying the lower surface of the mesh only. The mesh has 97 points in the direction normal to the plate and 65 points in the streamwise direction. Again, only 9 nodes were required in the spanwise direction due to

reasons mentioned in the previous section.

The portion of the lower surface ahead of the leading edge was treated with a symmetry plane boundary condition, while the entire upper surface was treated as a far-field boundary. As in the laminar case, the front and back surfaces were treated as inviscid walls. The inlet Mach number was set to 0.2, and the Reynolds number based on the length of the plate was approximately 6,000,000.

The CFL3D result using the Wilcox $k-\omega$ turbulence model is shown in Figure 5. The velocity profile shown corresponds to a streamwise station located at about 91% of the length of the plate. The TFLO result for the same turbulence model and using the exact same grid is shown in Figure 6 at the same streamwise station. The viscous sub-layer, log-law layer and the defect layer are predicted accurately. The results of the two codes are in close agreement and capture the log-law layer as expected.

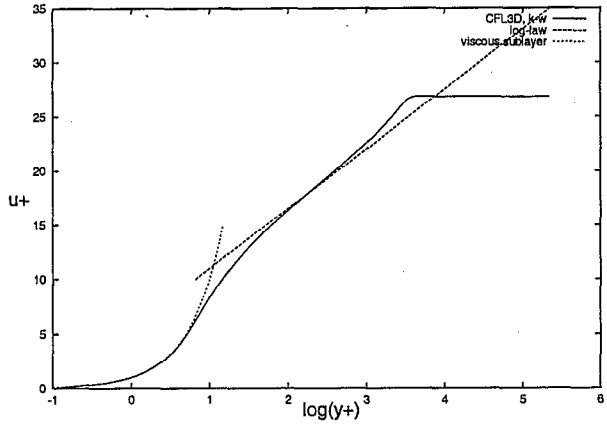


Figure 5: Flat plate boundary layer velocity profile, result from CFL3D, Re=6,000,000, Wilcox k-ω

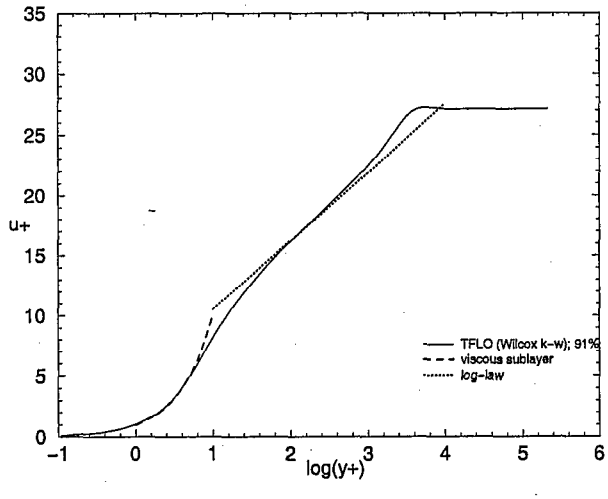


Figure 6: Flat plate boundary layer velocity profile, result from TFLO, Re=6,000,000, Wilcox k-ω

8.4 Bachalo and Johnson Transonic Flow

For further validation of the turbulence models, calculations were carried out for the Bachalo and Johnson test case [37]. This transonic test case consists of the flow over an annular bump on a circular cylinder aligned with the flow direction. Experimental data is available. The longitudinal section of the bump is a simple circular arc. The axisymmetric configuration is chosen to avoid the presence of sidewall boundary layers that can contaminate the inherent two-dimensionality of this flow. The resolved geometry has a baseline diameter of 0.152 m and it extends 61 cm upstream of the bump's leading edge. The chord of the circular bump is 0.203 m and it has a maximum thickness of 0.019 m. A $161 \times 73 \times 9$ mesh was used.

The freestream Mach number for this particular test case was set at 0.875 with a resulting Reynolds number of 13.1×10^6/m. At this freestream Mach number, a transonic shock wave appears over the rear portion of the bump. This shock wave is strong enough to produce a relatively large region of separated flow. The separation and re-attachment points can be found at distances of approximately 0.7 and 1.1 chords from the leading edge of the bump, respectively.

The surface pressure distributions obtained with TFLO and Allison's ADPAC code [38] are compared to each other and to experimental data in Figure 7. Identical meshes and flow conditions were used for all computational results.

TFLO's $k-\omega$ model predicts the separated region and the strength of the shock wave quite accurately, although the position of the shock is found somewhat downstream of the experiment.

8.5 Unsteady Flow behind a Circular Cylinder

In order to test the accuracy of the dual-time stepping scheme, we calculated the vortex shedding flow behind a circular cylinder at a Mach number of 0.2. For this case, a Karman vortex street is observed in the Reynolds number range from 40 to 5000 [39]. In this range, the Strouhal frequency of the vortex shedding, $St = \frac{nD}{U_\infty}$, where n is the real frequency, D is the diameter of the cylinder, and U_∞ is the free stream velocity, is primarily a function of Reynolds number.

For the TFLO calculation presented here, the Reynolds number was fixed at $Re_D = 150$. The mesh size of the cross-cylinder surface was 256×128. The TFLO calculation was executed for 10 vortex shedding periods with 40 real time steps within each period. The calculated Strouhal frequency is

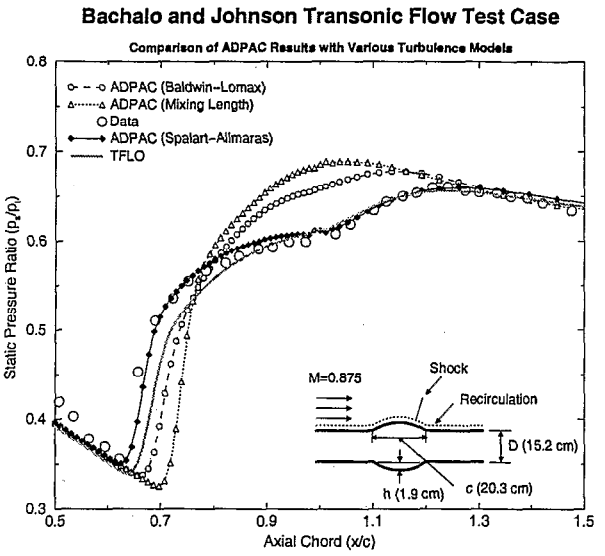


Figure 7: Static pressure comparison on bump surface

$St = 0.185$, which agrees very well with the experimental value of 0.182 from the incompressible flow curve in the laminar shedding regime.

The periodic response in the coefficients of lift and drag are shown in Figure 8. A series of entropy contours and streamlines at four instances of time within a shedding cycle is shown in Figure 9. The first four vortices can be accurately captured before the grid coarsens beyond the point where an accurate solution can be obtained.

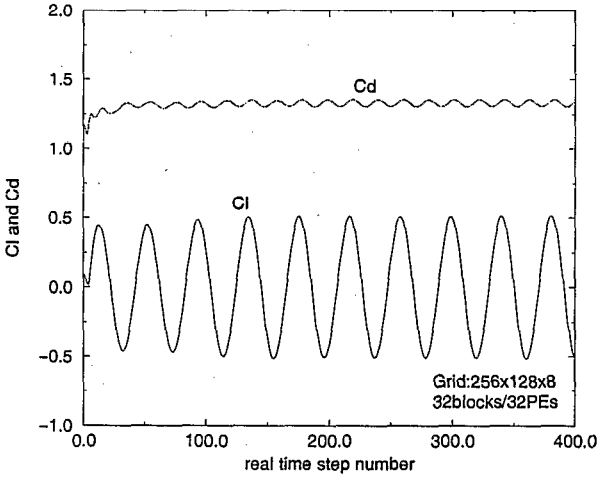


Figure 8: Time history of the coefficients of lift and drag

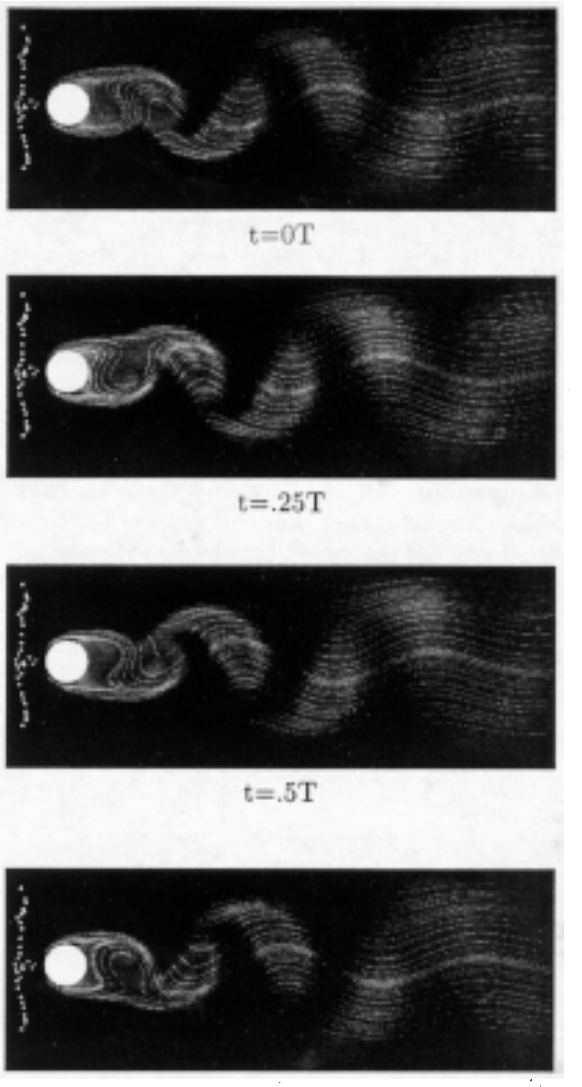


Figure 9: Entropy contours and instantaneous streamlines at four different times in the vortex shedding process

8.6 VPI Cascade

The results of subsections 8.1 – 8.5 confirm the accuracy of the flow solution scheme for a variety of basic flows. Subsections 8.6 – 8.8 present results for actual turbomachinery test cases.

The first of these cases is the VPI cascade, which is a linear transonic turbine nozzle. This viscous flow was calculated using both the Wilcox k-ω and the Baldwin-Lomax models. Two options were used for the exit boundary condition: uniform fixed back pressure and a non-reflecting boundary condition. Results for the surface pressure distribution produced by TFLO and APNASA-V5 [40] (NASA Glenn software run by GE personnel), are compared in Figure 10 together with the experimental data. All numerical calculations were carried out using exactly the same grid, whose dimensions were $145 \times 17 \times 81$.

Figure 10 shows satisfactory pressure comparisons between both codes and the experimental data. In addition, it shows the improvement in the prediction of the suction surface isentropic Mach number when the non-reflecting exit boundary conditions are used. The reason for this improvement can be found in Figure 11. This Figure shows the pressure variation in the circumferential direction at the exit plane and at a mid-span location. The two curves refer to the results with and without the non-reflecting boundary condition. The curve for the non-reflecting exit boundary condition shows a large jump in the pressure distribution at the point where the shock wave crosses the exit boundary. This pressure jump has been considerably smoothed when we fix the back pressure at the exit plane, thus distorting the shock wave structure and the blade isentropic Mach number.

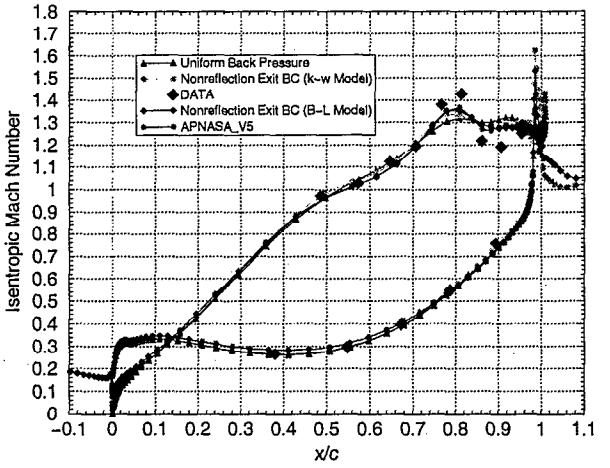


Figure 10: Isentropic Mach number on blade surfaces

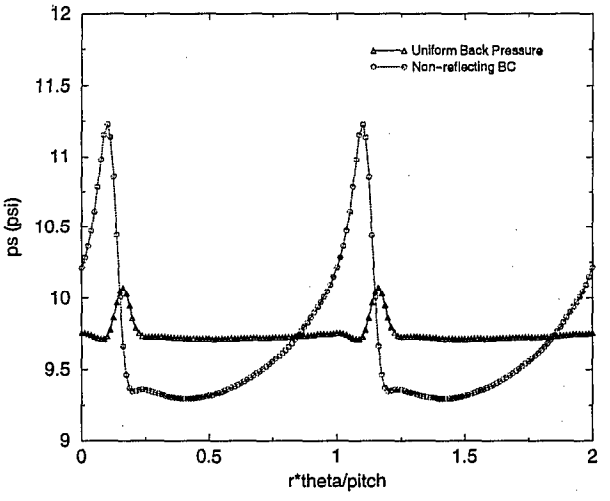


Figure 11: Static pressure in tangential direction at exit plane

8.7 PennState 1.5-Stage Research Compressor

The next test case is the PennState Research Compressor (PSRC). The PSRC facility is an axial flow compressor consisting of an inlet guide vane row and three stages of rotor and cantilever-mounted stator blading with a rotating hub. See Figure 12 for a schematic of the physical setup.

The results presented here are for a 1.5 stage subset of the complete configuration which consists of Rotor2-Stator2-Rotor3. The blade counts for these three blade rows are 72, 73, and 74. The tip gap of the rotor blade rows and the hub gap of the stator blade row were modeled in this calculation using TFLO's open-gap boundary condition. The inflow boundary conditions for this test case were obtained from a full rig simulation performed with ADPAC. The experimental data, computational grid, inlet conditions, and ADPAC results were provided by the Allison Engine Company.

Penn State Research Compressor Schematic

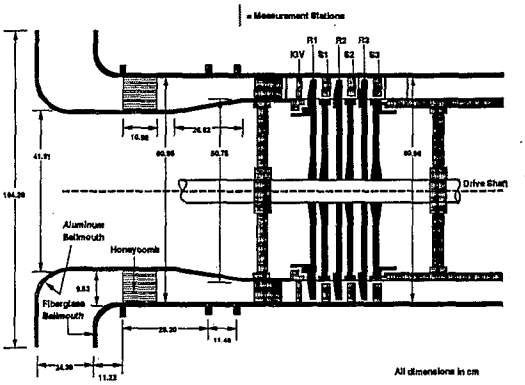


Figure 12: Schematic of PennState Research Compressor

In order to obtain a steady-state flow solution for this multiple blade row case, a mixing plane boundary condition was used between adjacent blade rows. Once again, the same grid is used for all computational results regardless of the code that was run. The two stators used meshes with $125\times65\times81$ nodes, while the single rotor used a mesh with $105\times65\times81$ nodes. All calculations were performed using 72 processors of a CRAY-T3E supercomputer. Figure 13 shows a comparison between the results from TFLO, ADPAC, and the existing experimental data. The flow conditions for this test case are those described in [41, 42]. The reader should note that all measurements were taken in the full rig environment. Figure 13 shows comparisons of the circumferentially averaged total pressure, and the axial and tangential velocity components along the blade spanwise direction at a distance of 5.6% of the chord behind

the trailing edge of Stator-2. TFLO predicts similar results to ADPAC and both sets of results agree well with the test data. The axial velocity profile predicted by TFLO is very close to the experimental data. However, the results from TFLO show a slight overshoot in the total pressure distribution near the hub.

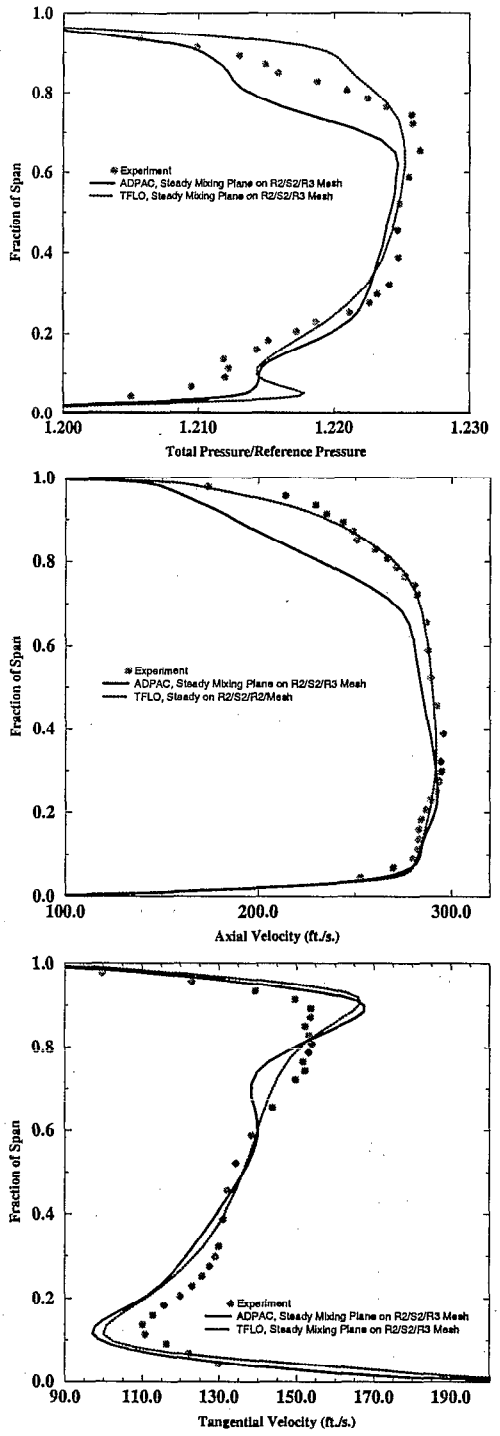


Figure 13: Circumferentially averaged parameters along spanwise direction at 5.6% chord after the trailing edge of stator-2.

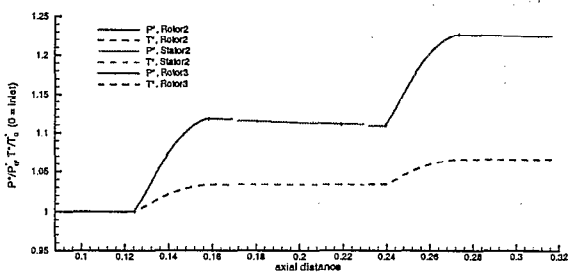


Figure 14: Predicted distribution of total pressure and total temperature in axial direction

The TFLO prediction of total pressure and total temperature in the axial direction is shown in Fig. 14. It shows very small but detectable losses of the total pressure caused by the mixing model. The total temperature remains constant cross the mixing plane.

8.8 Aachen 1.5-Stage Turbine

The final test case is the 1.5-Stage Aachen Turbine. The geometry and experimental data package were provided by ERCOFTAC (European Research Community On Flow, Turbulence And Combustion). This facility is an axial flow turbine consisting of three blade rows: the first vane, the blade, and the second vane. The geometry of the second vane is exactly the same as that of the first vane. A schematic of the configuration is presented in Figure 15. The blade counts for this case are 36, 41, and 36 respectively. The experimental results at the various axial locations were taken at different times, with slight variations in the inlet flow conditions. For this reason, it is very hard to obtain good comparisons at all axial stations with the results of a single simulation.

The mesh sizes used in this calculation can be summarized as follows: IGV: $137 \times 65 \times 81$, blade: $113 \times 65 \times 81$ for the blade passage and $89 \times 17 \times 17$ for the tip gap, stator: $153 \times 65 \times 81$. All calculations were run on 58 processors of an SGI Origin 2000 computer.

Comparisons are presented between the results of TFLO, United Technologies' solver 3DFLOW [4, 5], NASA/GE's solver APNASA-V5, and the experimental data for the low mass-flow rate condition. The calculated mass-flow rate was 7.1 Kg/s while the measured mass-flow rate was in the range 6.6 – 6.9 Kg/s. The computational grids were generated by personnel of the United Technologies Research Center. These grids have a separate block for the tip gap region of the rotor. The results from TFLO and 3DFLOW were calculated using the same grids.

The grid used by APNASA-V5 was similar in size, although the tip gap region was not resolved; the open gap boundary condition was applied instead.

TFLO's total pressure and total temperature axial distributions are shown in Figure 16. The mixing model in TFLO introduces a very small loss in the total pressure. The losses of total temperature across the mixing plane are not detectable. From additional comparisons performed, but not shown here, we were able to determine that TFLO and APNASA-V5 predict very similar distributions for these two quantities as well.

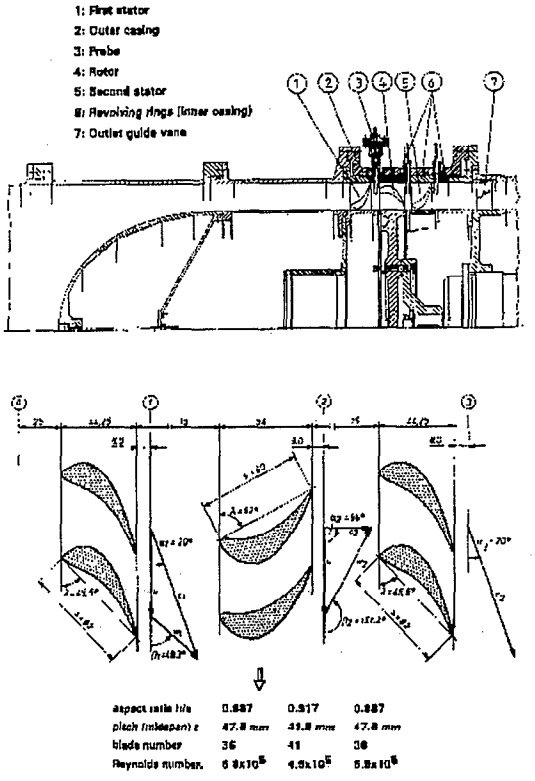


Figure 15: Schematic of the Aachen 1.5-stage turbine.

8.8.1 *The First Vane*

Table 1 compares the parameters at the inlet and exit (8.8 mm after the trailing edge) of the first vane. TFLO predicts a very slight mass-flow rate loss (0.03%). Results from 3DFLOW and experiment are also summarized in the Table.

Figure 17 shows the circumferentially averaged total pressure and total temperature at the first station (inlet) which simply indicate that the inlet boundary conditions are essentially the same.

The circumferentially averaged total pressure, total temperature, and absolute flow angle after the trailing edge of the first vane are compared in Figure 18.

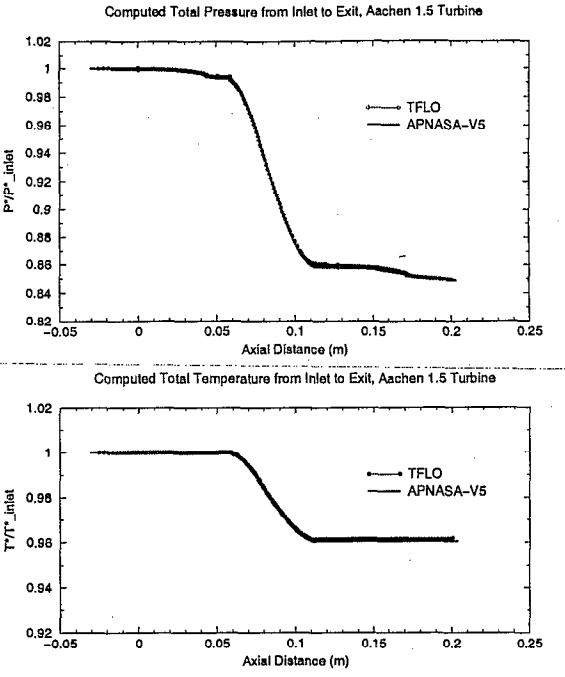
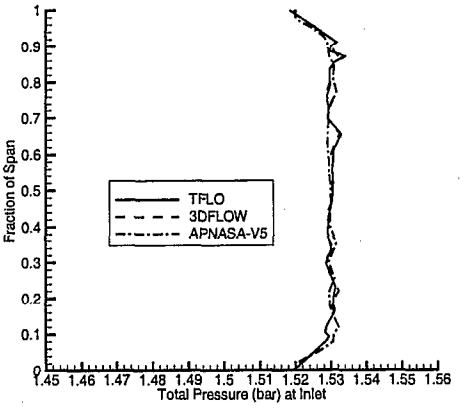


Figure 16: Predicted distribution of total pressure and total temperature in the axial direction



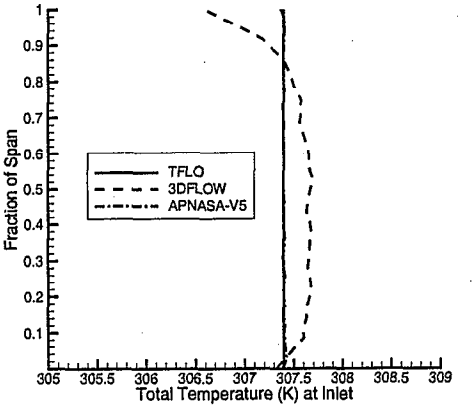
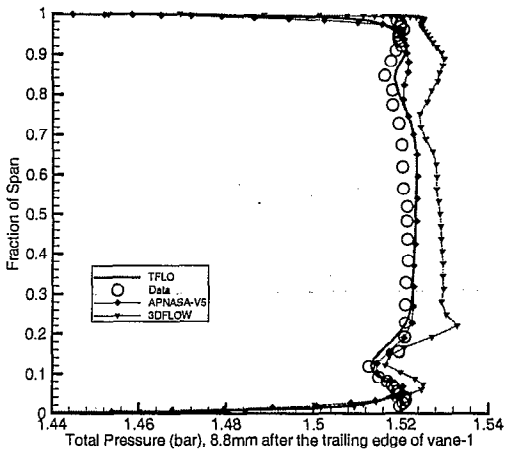
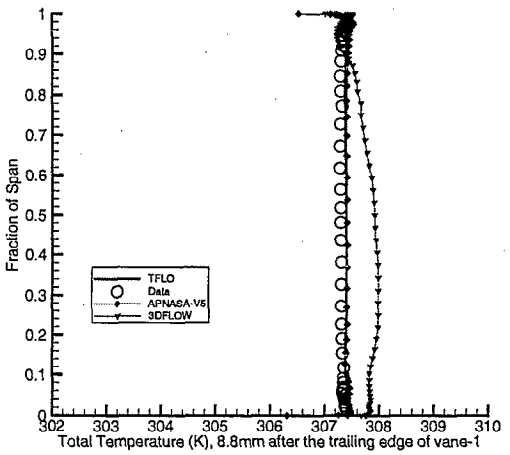


Figure 17: Predicted circumferentially averaged total pressure and total temperature at the inlet of the first vane





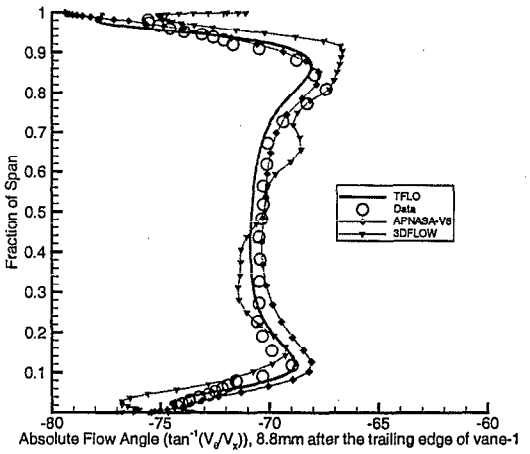
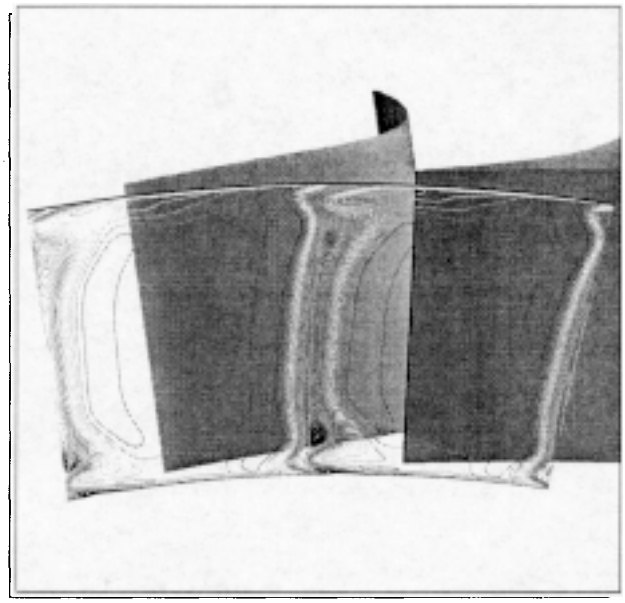
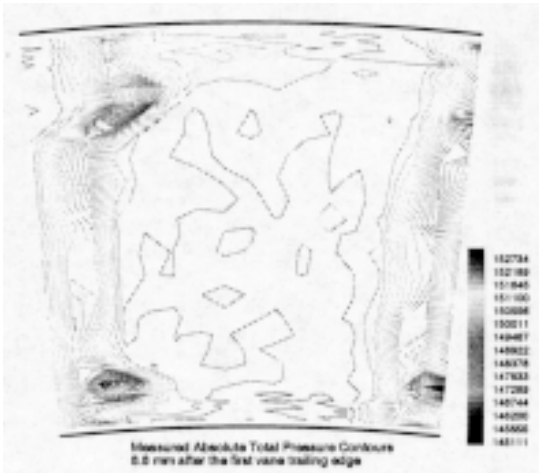


Figure 18: Predicted circumferentially averaged total pressure, total temperature, and flow angle at a station 8.8mm downstream of the first vane trailing edge

Figure 19 shows a comparison between the results of TFLO and the experimental data for the total pressure at a measurement plane located behind the first vane. The strong secondary flow near the end walls and in the wake behind the trailing edge cause major losses in total pressure. TFLO appears to capture the main features of the total pressure map.



(a) TFLO: Predicted contours of total pressure 8.8 mm behind the trailing edge of the first vane



(b) Measured contours of total pressure 8.8 mm behind the trailing edge of the first vane

Figure 19: Comparison of total pressure contours at the measurement plane behind the first vane

8.8.2 The Blade

Table 2 presents a comparison of various flow parameters at the inlet and exit (8.8 mm after the trailing edge) of the blade. TFLO predicts a slight mass-flow rate gain of 0.2% due to the mixing model used at

the interface plane. TFLO also preserves the flow angle quite well across the mixing plane. The results for 3DFLOW are also included in the Table as a reference of what can be done with typical flows solvers used in industry.

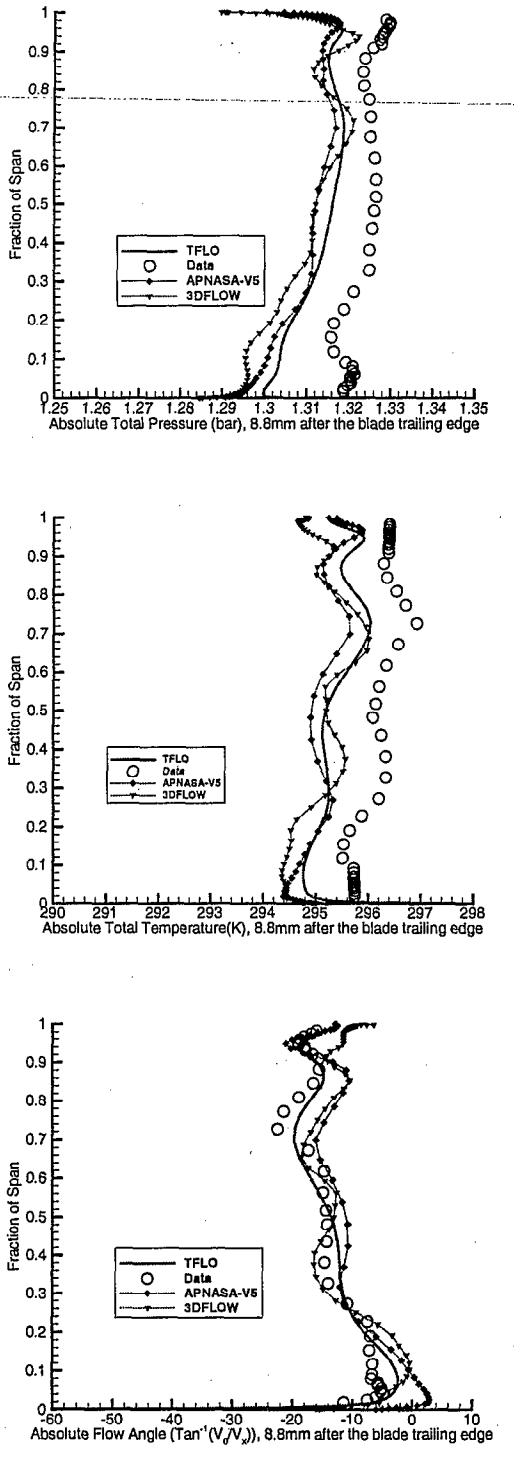
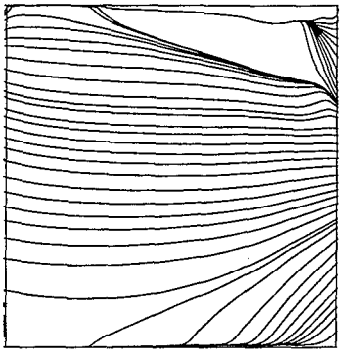


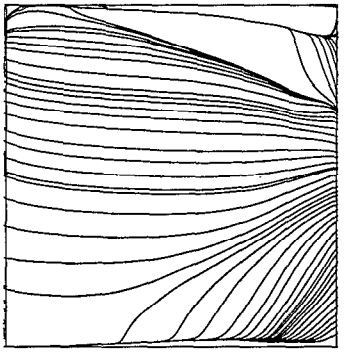
Figure 20: Predicted circumferentially averaged total pressure, total temperature, and flow angle at a station 8.8mm downstream of the blade trailing edge

The circumferentially averaged total pressure, total temperature, and absolute flow angle are compared in Figure 20 as was done for the first vane. Predictions of the different solvers seem to agree well with each other and the trends in the data but differ somewhat in the measured level, especially for the total pressure and total temperature.

The limiting streamlines on the suction surface of the blade predicted by TFLO and 3DFLOW are shown in Figure 21. These flow patterns are the signature of the hub secondary flow and blade tip vortex. The dividing streamline patterns in the area of the blade tip look quite similar for both solvers.



(a) TFLO: Predicted limiting streamlines on suction surface of the blade



(b) 3DFLOW: Predicted limiting streamlines on suction surface of the blade

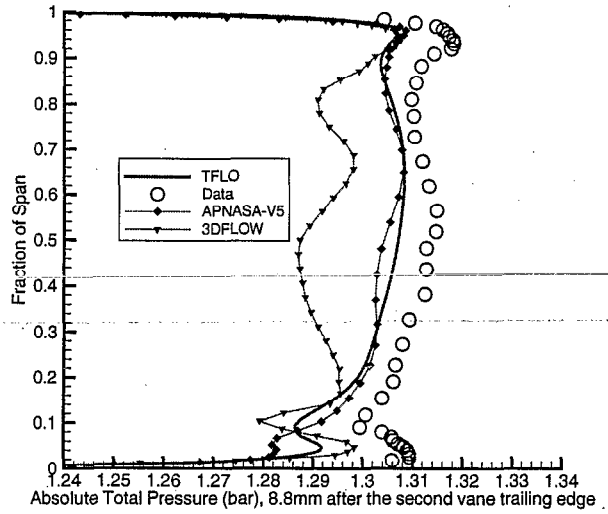
Figure 21: Predicted limiting streamlines on suction surface of the blade

8.8.3 The Second Vane

Finally, Table 3 compares flow parameters at the inlet and exit (8.8 mm after the trailing edge) of the second vane. TFLO predicts a slight mass-flow rate loss (0.2%) due to the details of the mixing model. Results from 3DFLOW and the experiment are included for comparison purposes.

The circumferentially averaged total pressure, total temperature, absolute flow angle, and absolute Mach number are compared in Figures 22 and 23. Similar conclusions drawn for the case of the first vane and blade can be arrived at here.

TFLO produces similar results to APNASA-V5 and 3DFLOW though the differences tend to be slightly larger for the second vane. TFLO shows close agreement with the experimental data especially in the spanwise direction.



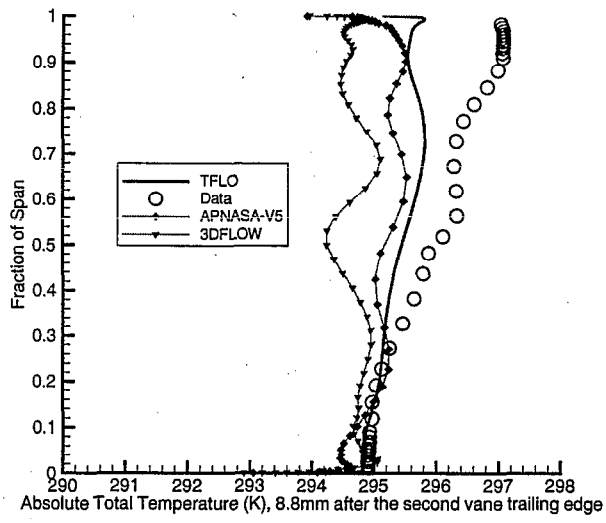
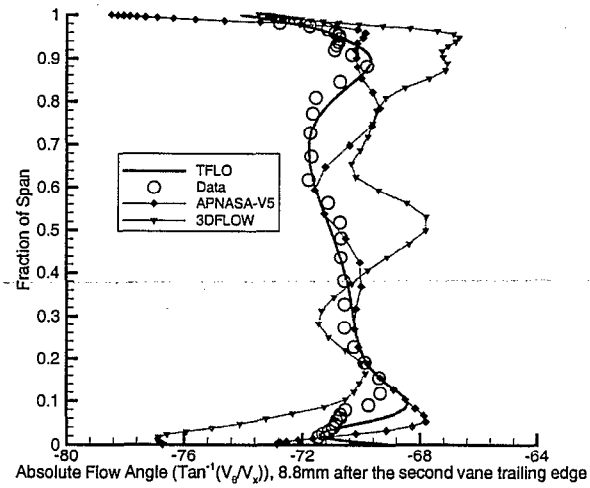


Figure 22: Predicted circumferentially averaged total pressure and total temperature at a station 8.8mm downstream of the trailing edge of the second vane



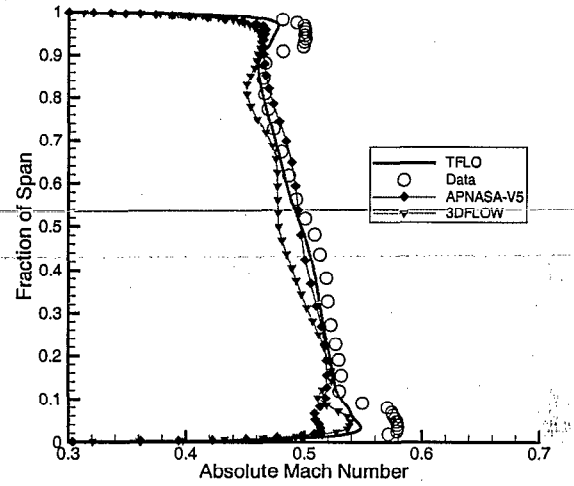


Figure 23: Predicted circumferentially averaged absolute flow angle and absolute Mach number at a station 8.8mm downstream of the trailing edge of the second vane

9. Concluding Remarks

The initial development and validation of the general turbomachinery flow solver TFLO has been completed. The result is a versatile program that can handle a range of general, multiple blade-row geometries in both steady and unsteady flow and that can be run on either simple workstations or large-scale parallel computers depending on the size of the test case in question. Detailed validation of the software has been presented and continues to this date. The results show that TFLO is able to obtain results that are very similar to those from other state-of-the-art simulation codes used in both government laboratories and industry. All of the basic elements of TFLO (steady-state and unsteady

CODE	Mr_1	Ma_1	α_1	β_1	Pt_{r_1} $(\times 10^5 Pa)$	Pt_{a_1} $(\times 10^5 Pa)$	Pt_r/Pt_{r_1}	Tt_{r_1} (K)	Tt_{a_1} (K)	Tt_{r_1}/Tt_{r_1}	$Flow_1$ (Kg/s)
TFLO	0.1251	0.1251	90	90	1.53	1.53	1	307.4	307.4	1	7.11
3DFLOW	0.1262	0.1262	90	90	1.53	1.53	1	307.5	307.5	1	7.17
Data	0.1048	0.1048	92.95	92.95	1.58	1.58	1	309.1	309.1	1	6.61

CODE	Mr_2	Ma_2	α_2	β_2	Pt_{r_2} $(\times 10^5 Pa)$	Pt_{a_2} $(\times 10^5 Pa)$	Pt_r2/Pt_{r_1}	Tt_{r_2} (K)	Tt_{a_2} (K)	Tt_{r_2}/Tt_{r_1}	$Flow_2$ (Kg/s)
TFLO	0.4022	0.4022	160.01	160.01	1.52	1.52	0.99237	307.3	307.3	0.99983	7.11
3DFLOW	0.4116	0.4116	160.21	160.21	1.53	1.53	0.99787	307.8	307.8	1.00078	7.20
Data	0.4298	0.4298	160.3	160.3	1.52	1.52	0.95879	307.3	307.3	0.99434	6.70

Table 1: Inlet(1) and exit(2) circumferentially averaged flow conditions for the first vane.

CODE	Mr_1	Ma_1	α_1	β_1	Pt_{r_1} $(\times 10^5 Pa)$	Pt_{a_1} $(\times 10^5 Pa)$	Pt_r/Pt_{r_1}	Tt_{r_1} (K)	Tt_{a_1} (K)	Tt_{r_1}/Tt_{r_1}	$Flow_1$ (Kg/s)
TFLO	0.163	0.4017	20.04	57.67	1.38	1.52	1	299.4	307.3	1	7.11
3DFLOW	0.170	0.4115	19.8	55.13	1.39	1.52	1	299.5	307.8	1	7.20
Data	0.1867	0.4298	19.7	53.07	1.37	1.52	1	298.5	307.3	1	6.70

CODE	Mr_2	Ma_2	α_2	β_2	Pt_{r_2} $(\times 10^5 Pa)$	Pt_{a_2} $(\times 10^5 Pa)$	Pt_r2/Pt_{r_1}	Tt_{r_2} (K)	Tt_{a_2} (K)	Tt_{r_2}/Tt_{r_1}	$Flow_2$ (Kg/s)
TFLO	0.2956	0.1481	103.61	29.15	1.37	1.31	0.99307	299.5	295.6	1.00042	7.13
3DFLOW	0.2977	0.1511	103.24	29.61	1.37	1.31	0.99050	299.1	295.2	0.99876	7.26
Data	0.2934	0.1544	104.08	30.5	1.38	1.32	1.00762	300.0	296.2	1.00482	6.84

Table 2: Blade inlet(1) and exit(2) circumferentially averaged flow conditions.

CODE	Mr_1	Ma_1	α_1	β_1	Pt_{r_1} $(\times 10^5 Pa)$	Pt_{a_1} $(\times 10^5 Pa)$	Pt_r/Pt_{r_1}	Tt_{r_1} (K)	Tt_{a_1} (K)	Tt_{r_1}/Tt_{r_1}	$Flow_1$ (Kg/s)
TFLO	0.1488	0.1488	76.02	76.02	1.31	1.31	1	295.7	295.7	1	7.14
3DFLOW	0.1511	0.1511	76.74	76.74	1.31	1.31	1	295.2	295.2	1	7.26
Data	0.1544	0.1544	75.92	75.92	1.32	1.32	1	296.2	296.2	1	6.84

CODE	Mr_2	Ma_2	α_2	β_2	Pt_{r_2} $(\times 10^5 Pa)$	Pt_{a_2} $(\times 10^5 Pa)$	Pt_r2/Pt_{r_1}	Tt_{r_2} (K)	Tt_{a_2} (K)	Tt_{r_2}/Tt_{r_1}	$Flow_2$ (Kg/s)
TFLO	0.4928	0.4928	160.32	160.32	1.30	1.30	0.98938	295.7	295.7	0.99989	7.16
3DFLOW	0.4824	0.4824	159.87	159.87	1.30	1.30	0.99547	295.7	295.7	1.00153	7.16
Data	0.5048	0.5048	160.82	160.82	1.31	1.31	0.98997	295.7	295.7	0.99912	6.71

Table 3: Inlet(1) and exit(2) circumferentially averaged flow conditions at the second vane.

discretization, domain decomposition, parallel implementation, sliding mesh interface boundary conditions, basic algorithms for viscous and artificial dissipation, pre- and post-processing) have been described, tested, and demonstrated.

Emphasis has been placed on the ability of TFLO to scale to large numbers of processors so that the more interesting problems of complete compressor/turbine unsteady calculations can be tackled. It is in this area where we hope that TFLO can make a contribution to the field, since, using computational resources from the ASCI project, we will be in a position to tackle the most complex calculations attempted to this date. We are currently running some large scale unsteady computations of the Aachen turbine case whose results will be presented at a later time.

As mentioned above, additional validation test cases for both steady and unsteady rotor/stator interaction flows in multi-stage turbomachinery are in progress. The validation strategy is composed of two main parts:

- α-test within the Stanford Turbomachinery Simulation Group for new code and algorithm developments and modifications.
- β-test and joint validation with industry:

  1. Demonstration on industrial computer platforms

  2. Continued comparison with results from other codes being used at the various engine companies

  3. Continued comparison with available experimental data for representative test cases of progressively increasing complexity

10. Acknowledgements

The authors would like to thank the U.S. Department of Energy (DoE) for its generous support under the ASCI Program. We are also deeply indebted to Dr. Roger L. Davis from the United Technologies Research Center, who, during his stay at Stanford University has made invaluable contributions to the completion and editing of this work. The authors would also like to thank Dr. Lyle D. Dailey from GE Aircraft Engines Company, and Drs. Edward J. Hall and Kurt Weber from the Rolls-Royce Allison Engine Company for their assistance and cooperation. We like to thank the Pittsburgh Super-Computing Center and Lawrence Livermore National Laboratory for the use of their parallel platforms. ERCOFTAC (European Research Community On Flow, Turbulence And Combustion) is gratefully acknowledged for providing the experimental data package for the Aachen case. Finally, we thank Dr. Creigh Y. McNeil for providing the CFL3D result of the flat plate case.

REFERENCES

[1] J. J. Adamczyk, M. L. Celestina, T. A. Beach, and M. Barnett. Simulation of 3-dimensional viscous flow within a multistage turbine. *Journal of Turbomachinery*, 112:370–376, 19890.

[2] W. N. Dawes. Simulation of unsteady blade row interaction with CFD: Applications. *VKI-LS-1998-02*, Von Karman Institute for Fluid Mechanics, 1998.

[3] J. D. Denton and U. K. Singh. Time marching methods for turbomachinery flow calculations. *VKI-LS-1979-07*, Von Karman Institute for Fluid Mechanics, 1979.

[4] R. H. Ni and J. C. Bogoian. Predictions of 3-D multi-stage turbine flow fields using a multiple-grid euler solver. *AIAA paper 90-2357*, 1990.

[5] R. H. Ni and O. P. Sharma. Using 3-D Euler flow simulations to assess effects of periodic unsteady flow through turbines. *AIAA paper 90-2357*, 1990.

[6] C. M. Rhie, A. J. Gleixner, D. A. Spear, C. J. Fischberg, and R. M. Zacharias. Development and application of a multistage navier-stokes solver: Part I - multistage modeling using body forces and deterministic stresses. *ASME Journal of Turbomachinery*, 120(2):205–214, 1997.

[7] C. R. Lejambre, R. M. Zacharias, B. P. Biederman, A. J. Gleixner, and C. J. Yetka. Development and application of a multistage navier-stokes flow solver: Part II - application to a high-pressure compressor design. *ASME Journal of Turbomachinery*, 120:215, 1998.

[8] J. J. Adamczyk. Model equation for simulating flows in multistage turbomachines. *ASME paper 85-GT-226*, 1985.

[9] J. J. Adamczyk. A model for closing the inviscid form of the average passage equation system. *Journal of Turbomachinery*, 108:180, 1986.

[10] E. J. Hall. Aerodynamic modeling of multistage compressor flowfields. *ASME paper 97-GT-344*, 1997.

[11] A. Arnone and R. Pacciani. Rotor-stator interaction analysis using the Navier-Stokes equations and a multigrid method. *ASME paper 95-GT-117*, 1995.

[12] D. J. Dorney, R. L. Davis, D. E. Edwards, and N. K. Madavan. Unsteady analysis of hot streak migration in a turbine stage. *AIAA paper 90-2354*, 1990.

[13] M. B. Giles. Stator/rotor interaction in a transonic turbine. *AIAA paper 88-3093*, 1988.

[14] M. B. Giles. UNSFLO: A numerical method for unsteady inviscid flow in turbomachinery. MIT Gas Turbine Laboratory 195, 1988.

[15] K. L. Gundy-Burlet. Computations of unsteady multistage compressor flows in a workstation environment. *ASME paper 91-GT-336*, 1991.

[16] P. C. E. Jorgeson and R. Chima. An explicit Runge-Kutta method for unsteady rotor/stator interaction. *AIAA paper 88-0049*, 1988.

[17] J. P. Lewis, R. A. Delaney, and E. J. Hall. Numerical prediction of turbine vane-blade aerodynamic interaction. *ASME Journal of Turbomachinery*, 111:387–393, 1989.

[18] M. M. Rai. Navier-Stokes simulations of rotor/stator interactions using patched and overlaid grids. *AIAA Journal of Propulsion and Power*, 3(5):387–396, 1987.

[19] K. Rao and R. Delaney. Investigation of unsteady flow through transonic turbine stage, part I: Analysis. *AIAA paper 90-2408*, 1990.

[20] A. Jameson, W. Schmidt, and E. Turkel. Numerical solutions of the Euler equations by finite volume methods with Runge-Kutta time stepping schemes. *AIAA paper 81-1259*, January 1981.

[21] A. Jameson. Analysis and design of numerical schemes for gas dynamics 1, artificial diffusion, upwind biasing, limiters and their effect on multigrid convergence. *Int. J. of Comp. Fluid Dyn.*, 4:171–218, 1995.

[22] A. Jameson. Analysis and design of numerical schemes for gas dynamics 2, artificial diffusion and discrete shock structure. *Int. J. of Comp. Fluid Dyn.*, 5:1–38, 1995.

[23] A. Jameson. Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA paper 91-1596*, AIAA 10th Computational Fluid Dynamics Conference, Honolulu, HI, June 1991.

[24] J. J. Alonso, L. Martinelli, and A. Jameson. Multigrid unsteady Navier-Stokes calculations with aeroelastic applications. *AIAA paper 95-0048*, AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1995.

[25] A. Belov, L. Martinelli, and A. Jameson. Three-dimensional computations of time-dependent incompressible flows with an implicit multigrid-driven algorithm on parallel computers. Monterey, CA, June 1996. Proceedings of the 15th International Conference on Numerical Methods in Fluid Dynamics.

[26] J. J. Alonso. *Parallel Computation of Unsteady and Aeroelastic Flows using an Implicit Multigrid-Driven Algorithm.* PhD thesis, Princeton University, Dept. of Mechanical and Aerospace Engineering, June 1997.

[27] J. Reuther, J. J. Alonso, J. C. Vassberg, A. Jameson, and L. Martinelli. An efficient multiblock method for aerodynamic analysis and design on distributed memory systems. *AIAA Paper 97-1893.*

[28] F. Liu and X. Zheng. A strongly coupled time-marching method for solving the navier-stokes and $k - \omega$ turbulence model equations with multigrid. *J. of Computational Physics*, pages 289–300, August 1996.

[29] F. Liu and A. Jameson. Multigrid navier-stokes calculations for three-dimensional cascade. *AIAA Journal*, 32(8), August 1994.

[30] F. Liu, A. Jameson, and I.K. Jennions. Computation of turbomachinery flow by a convective-upwind-split-pressure (cusp) scheme. *AIAA Paper 98-0969*, January 1998.

[31] S. Tatsumi, L. Martinelli, and A. Jameson. A new high resolution scheme for compressible viscous flows with shocks. *AIAA paper*, AIAA 33nd Aerospace Sciences Meeting, Reno, Nevada, January 1995.

[32] N. D. Melson, M. D. Sanetrik, and H. L. Atkins. Time-accurate Navier-Stokes calculations with multigrid acceleration. Copper Mountain, CO, July 1993. Proceedings of the Sixth Copper Mountain Conference on Multigrid Methods.

[33] A. Belov, L. Martinelli, and A. Jameson. A new implicit algorithm with multigrid for unsteady incompressible flow calculations. *AIAA paper 95-0049*, AIAA 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1995.

[34] E. Arad and L. Martinelli. Large eddy simulation of compressible flow using a parallel, multigrid driven algorithm. *AIAA paper 96-2065*, AIAA 27th Fluid Dynamics Conference, New Orleans, LA, 1996.

[35] R. L. Davis, T. Shang, J. Buteau, and R. H. Ni. Prediction of 3-d unsteady flows in multistage turbomachinery using an implicit dual

time-step approach. *AIAA Paper 96-2565*, July 1996.

[36] A. Arnone, M. S. Liou, and L. A. Povinelli. Multigrid time-accurate integrations of Navier-Stokes equations. *AIAA paper 93-3361*, AIAA 24th Fluid Dynamics Conference, Orlando, FL, July 1993.

[37] W. D. Bachalo and D. A. Johnson. Transonic turbulent boundary-layer separation generated on an axisymmetric flow model. *AIAA Journal*, 24:437–443, 1986.

[38] T. Barber, D. Choi, G. S. McNulty, and E. J. Hall. Preliminary findings in certification of adpac. *AIAA Paper 94-2240*, 1994.

[39] H. Schlichting. *Boundary Layer Theory*. McGraw Hill, 1955.

[40] K. R. Kirtley, M. G. Turner, and S. Saeidi. An average passage closure model for general meshes. *ASME Paper 99–GT-077*, 1999.

[41] Edward J. Hall. Aerodynamic modelling of multistage compressor flowfields - part 1: Analysis of rotor/stator/rotor aerodynamic interaction. Technical report, Allison Engine Company, Indianapolis, Indiana.

[42] Edward J. Hall. Aerodynamic modelling of multistage compressor flowfields - part 1: Modelling deterministic stresses. Technical report, Allison Engine Company, Indianapolis, Indiana.