JOURNAL OF COMPUTATIONAL PHYSICS **128,** 289–300 (1996)
ARTICLE NO. 0211

A Strongly Coupled Time-Marching Method for Solving the Navier–Stokes and k-ω Turbulence Model Equations with Multigrid

FENG LIU AND XIAOQING ZHENG*

Department of Mechanical and Aerospace Engineering, University of California, Irvine, California 92697

Received May 1, 1995; revised March 18, 1996

Many researchers use a time-lagged or loosely coupled approach in solving the Navier–Stokes equations and two-equation turbulence model equations in a time-marching method. The Navier–Stokes equations and the turbulence model equations are solved separately and often with different methods. In this paper a strongly coupled method is presented for such calculations. The Navier–Stokes equations and two-equation turbulence model equations, in particular, the k-ω equations, are considered as one single set of strongly coupled equations and solved with the same explicit time-marching algorithm without time-lagging. A multigrid method, together with other acceleration techniques such as local time steps and implicit residual smoothing, is applied to both the Navier–Stokes and the turbulence model equations. Time step limits due to the source terms in the k-ω equations are relieved by treating the appropriate source terms implicitly. The equations are also strongly coupled in space through the use of staggered control volumes. The method is applied to the calculation of flows through cascades as well as over isolated airfoils. Convergence rate is greatly improved by the use of the multigrid method with the strongly coupled time-marching scheme.

1. INTRODUCTION

Most Navier–Stokes codes incorporate an algebraic model initially to demonstrate their capability of solving high Reynolds number viscous flows [1–6]. With the development of efficient numerical methods and powerful computers, more complicated turbulence models are being used for better simulation of practical flows. Among the most used turbulence models today, two-equation eddy viscosity models appear to be favored for the reason that they are more general than algebraic models and affordable with current available computer resources.

However, investigators using two-equation models seem to have been more concerned with the solution of the Navier–Stokes equations. Less attention is paid to the solution method for the turbulence model equations, particularly their coupling with the Navier–Stokes equations due to perhaps the fact that the turbulence model equations are solved only to obtain the eddy viscosity and also the convenience of simply adding separate routines to an existing Navier–Stokes code. Consequently, many codes use a time-lagged or loosely-coupled approach in solving the Navier–Stokes and two-equation turbulence model equations [7, 8].

In a typical iteration of a loosely coupled approach, the Navier–Stokes equations are first solved with fixed eddy viscosity and then the k-ε or k-ω equations are solved with the newly updated flow field. Different solution methods are often used for the Navier–Stokes and turbulence model equations. To some extent, the model equations look simpler than the Navier–Stokes equations, particularly after the convection velocities are frozen in a loosely coupled algorithm. However, this does not appear to yield an easier task for numerical solution. On the contrary, results seem to show slow convergence or incomplete convergence due to possible reasons such as numerical stiffness for some models, not well-defined boundary conditions, troublesome source terms, imposed limiters on k, ε, or ω, and the fact that the Navier–Stokes and turbulence model equations are not strongly coupled in the numerical scheme. It appears that the solution of the turbulence model equations has a significant effect on the final convergence of the complete system. This is particularly true for methods that use very fine grids and integrate the model equations to the wall. Not well-solved model equations might greatly slow down the convergence of the Navier–Stokes equations because of the strong nonlinear interaction between the two sets of equations. Kunz and Lakshminarayana [8] had to march up to 10,000 time steps to reduce the residuals for the Navier–Stokes and the k-ε equations by four orders of magnitude and the convergence seems to hang at that residual level. In [9], Lin *et al.* were able to reduce the residual by six orders of magnitude in about 6000 steps for 2D transonic flows by using a variant biconjugate gradient method.

In this paper, an efficient multigrid algorithm is developed to solve a k-ω two-equation turbulence model pro-

* Current address: Computational Mathematics, University of Colorado, Denver, CO 80217.

0021-9991/96 $18.00

posed by Wilcox [10]. The Navier–Stokes and k-ω turbulence model equations are treated as a single set of strongly coupled equations and solved with the same multistage explicit time-stepping scheme. With proper construction of the residuals and suitable implicit treatment of the source terms, a multigrid method is applied to both the Navier–Stokes and the k-ω equations, giving excellent convergence properties. With the multigrid method the residuals of both the Navier–Stokes and the k-ω model equations can be reduced by 10 to 13 orders of magnitude in a few hundred cycles.

In the following section of this paper we will first outline the basic governing equations including the k-ω turbulence model equations. The numerical method is presented in Section 3. Section 4 shows the computational results for a low pressure turbine cascade and an airfoil.

2. GOVERNING EQUATIONS

The Favre-averaged Navier–Stokes equations for a compressible turbulent flow with a k-ω model by Wilcox [10] can be summarized as follows:

Mass conservation,

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j) = 0; \tag{1}$$

Momentum conservation,

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_j u_i) = -\frac{\partial p}{\partial x_i} + \frac{\partial \hat{\tau}_{ji}}{\partial x_j}; \tag{2}$$

Mean energy conservation,

$$\frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_j}(\rho u_j H) = \frac{\partial}{\partial x_j}\left[u_i \hat{\tau}_{ij} + (\mu + \sigma^* \mu_T)\frac{\partial k}{\partial x_j} - q_j\right]; \tag{3}$$

Turbulent mixing energy,

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_j}(\rho u_j k) = \tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta^* \rho \omega k + \frac{\partial}{\partial x_j}\left[(\mu + \sigma^* \mu_T)\frac{\partial k}{\partial x_j}\right]; \tag{4}$$

Specific dissipation rate,

$$\frac{\partial}{\partial t}(\rho \omega) + \frac{\partial}{\partial x_j}(\rho u_j \omega) = (\alpha \omega / k)\tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta \rho \omega^2 + \frac{\partial}{\partial x_j}\left[(\mu + \sigma \mu_T)\frac{\partial \omega}{\partial x_j}\right]; \tag{5}$$

where t is time, x_i is the position vector, u_i is the Favre-averaged velocity vector, ρ is density, p is pressure, μ is the molecular viscosity, k is the turbulent mixing energy, ω is the specific dissipation rate. The total energy and enthalpy are $E = e + k + u_i u_i/2$ and $H = h + k + u_i u_i/2$, respectively, with $h = e + p/\rho$, and $e = p/(\gamma - 1)\rho$; γ is the ratio of specific heats. The other quantities are defined

$$\mu_T = \alpha^* \frac{\rho k}{\omega} \tag{6}$$

$$S_{ij} = \frac{1}{2}\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] \tag{7}$$

$$\tau_{ij} = 2\mu_T\left[S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right] - 2/3\,\rho k \delta_{ij} \tag{8}$$

$$\hat{\tau}_{ij} = 2\mu\left[S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right] + \tau_{ij} \tag{9}$$

$$q_j = -\left(\frac{\mu}{\mathrm{Pr_L}} + \frac{\mu_T}{\mathrm{Pr_T}}\right)\frac{\partial h}{\partial x_j}, \tag{10}$$

where $\mathrm{Pr_L}$ and $\mathrm{Pr_T}$ are the laminar and turbulent Prandtl numbers, respectively. The closure coefficients are

$$\beta = \tfrac{3}{40}, \quad \beta^* = \tfrac{9}{100}, \quad \alpha = \tfrac{5}{9}$$
$$\alpha^* = 1, \qquad \sigma = \tfrac{1}{2}, \quad \sigma^* = \tfrac{1}{2}. \tag{11}$$

3. NUMERICAL METHOD

Based on our previous work [11, 12], the above-described equations are discretized by using a staggered finite volume scheme. This scheme strongly couples the k-ω and Navier–Stokes equations and maintains a small stencil for the diffusion terms. In this paper, a correction term is introduced to remove the possibility of an odd–even decoupling mode that may still be present in the discretization of the diffusion terms. Through this correction the scheme for diffusion terms becomes a compact one. A semi-loosely coupled algorithm was used for integrating in time the discrete finite-volume equations in our previous work [11, 12]. We present here a new strongly coupled approach for the Navier–Stokes and the k-ω equations with multigrid. The basic staggered finite volume discretization originally proposed in [11] is outlined in Subsection 1. The correction for the discretization of diffusion terms is presented in

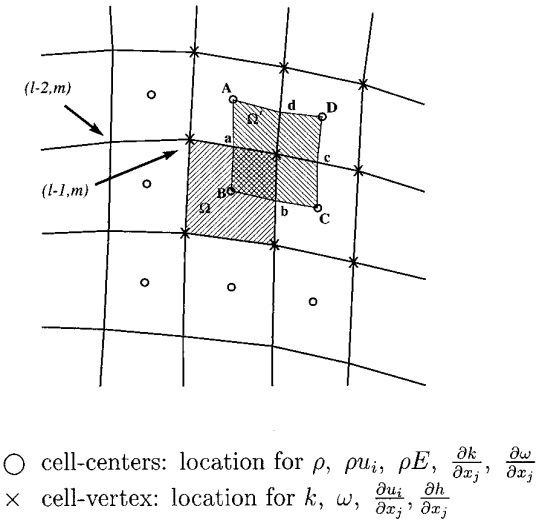

FIG. 1. Staggered Control Volumes for a Mixed Cell-center and Cell-vertex Scheme.

Subsection 2. Subsection 3 describes the strongly coupled multigrid time-marching algorithm for solving the Navier–Stokes and the k-ω equations.

3.1. *Staggered Finite Volume Scheme*

The computational domain is discretized into a number of quadrilateral cells in two dimensions or hexahedral cells in three dimensions. Consider a computational mesh in two dimensions. The governing equations are applied to each of the cells in integral form. With a cell-centered scheme the flow variables ρ, ρu_i, and ρE are defined at the cell centers marked by the circles in Fig. 1. Both the convective and diffusive fluxes in the Navier–Stokes equations have to be estimated over the four cell faces of a control volume, for example, the cell Ω shown in Fig. 1. The convective fluxes can be easily estimated by taking the averages of the flow variables on either side of a cell face, yielding a five-point stencil for the total Euler flux balance. To estimate the diffusion terms, a staggered auxiliary control volume Ω' was formed by connecting the cell-centers A, B, C, D and the mid-points of the cell faces a, b, c, and d as shown in Fig. 1. Since the flow variables are defined at the vertices of this auxiliary cell, Gauss's formula can then be applied as in a vertex scheme to calculate the velocity and temperature gradients at the center of the auxiliary cell, which in fact is the vertex of the original cell Ω. Once the stresses are known at the cell vertices of Ω, the diffusive fluxes in the Navier–Stokes equations can then be easily evaluated over the cell faces by trapezoidal rule as in a vertex scheme. This yields a discretization stencil involving nine points with minimum spatial extent as shown by the circles in Fig. 1.

In order to solve the k-ω equations one could either define k and ω at the cell centers or the cell vertices. If k and ω were defined at the cell centers, one would have to interpolate the strain tensor calculated at the cell vertices to the cell center of Ω so that the production terms for the control volume can be evaluated. On the other hand, the eddy viscosity μ_T calculated from the k and ω at the cell centers must be translated to the cell vertices in order to calculate the turbulent stresses there. This double averaging process would broaden the final discretization stencil for the coupled Navier–Stokes and the k-ω equations and thus reduce the accuracy and increase the likelihood of uninhibited growing modes.

Alternatively, one can define k and ω at the cell vertices and use the staggered control volume Ω' to integrate the k-ω equations. The discretization is done in a similar fashion as that for the Navier–Stokes equations, but in the reverse order of using the original and the auxiliary cells. Since the variables k and ω are defined at the cell vertices marked by the crosses, we will no longer need the excessive averaging steps for the strain tensor and the eddy viscosity. The production terms are evaluated at exactly the same locations, namely the cell vertices, where the stress and strain tensors are calculated, and the eddy viscosity calculated from the k and ω at these cell vertices are directly used to calculate the turbulent stress tensors. In this way, the Navier–Stokes equations and the k-ω equations in their discrete forms are coupled as closely as possible. The discretization of each set of the equations involves a stencil of only nine points: those for the Navier–Stokes equations are shown by the circles and those for the k-ω equation are shown by the crosses in Fig. 1. The staggered finite volume approach proposed here for the conservative flow variables and the k and ω in the turbulence model equations much resembles the staggering of the pressure and velocities used in the MAC and SIMPLE types of methods [13, 14]. The velocity gradient calculated at the cell vertices with the tightest possible stencil directly drives the solution of the turbulence model equations, just as the pressure gradient calculated at a cell face directly drives the momentum equation in the MAC and SIMPLE schemes.

The scheme as presented above reduces to a centered difference scheme for the convective terms in both the Navier–Stokes equations and the k-ω equations. In regions outside the boundary layer where the grid size is too large to render the physical viscosity effective, dissipation terms of fourth-order differences need to be added to eliminate odd-and-even decoupling modes for the convective terms and a second-difference dissipation is needed for capturing shocks. The blended second- and foruth-order difference formulation by Jameson [15] is used for the Navier–Stokes equations. For subsonic flow the second-difference dissipation is turned off completely.

The staggered finite-volume approach may also be com-

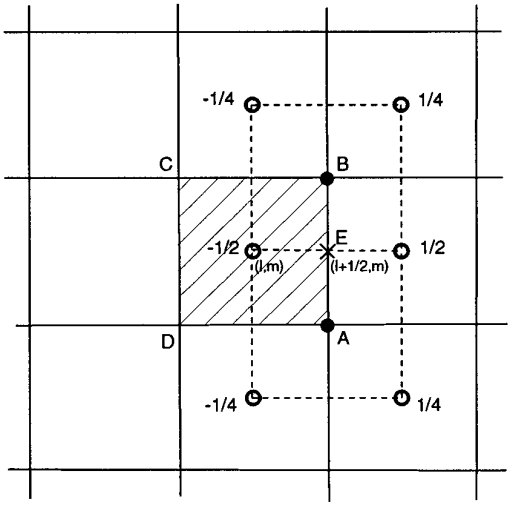

FIG. 2. Finite-volume discretization stencil for $\partial u/\partial x$.

bined with upwind-type schemes. For instance, schemes using second- or third-order MUSCL interpolation [16] and Roe's approximate Riemann solver [17] are implemented by the present authors in [12]. The flow over an airfoil presented later in this paper is done with this upwind type scheme for the Navier–Stokes equations.

In principle, the artificial dissipation formulation of blended second- and fourth-order differences may also be used for the k-ω equations. However, it is noted that the k and ω equations have very simple wave structures which consist of essentially the flow convective velocities in the three coordinate directions. Therefore, upwind schemes of various orders can be easily formed, based on the local convective velocity at the interface of the control volume Ω'. For instance, if the estimated normal convective velocity on the interface AaB in Fig. 1 is positive, a second-order upwind interpolation formula may be used to obtain the values k and ω at the interface mid-point a:

$$(k)_a = \tfrac{1}{2}\,[3(k)_{l-1,m} - (k)_{l-2,m}] \tag{12}$$

$$(\omega)_a = \tfrac{1}{2}\,[3(\omega)_{l-1,m} - (\omega)_{l-2,m}]. \tag{13}$$

These values are then used to form the convective fluxes through the interface AaB. This is similar to the MUSCL-type scheme for the Euler equations used by Anderson, Thomas, and Van Leer [16]. In this way no explicit artificial dissipation is required for the k-ω equations, since the upwinding is simply a way of introducing dissipation implicitly.

3.2. *Improvement on the Discretization of the Diffusion Terms*

As pointed out by Liu and Zheng [11], the 9-point scheme for the diffusion terms in the Navier–Stokes equations shown by the circles in Fig. 1 does not completely rule out the possibility of an odd–even decoupled mode. Consider the Laplacian $\nabla^2 u = \partial^2 u/\partial x^2 + \partial^2 u/\partial y^2$, which is the viscous diffusion term in the x-momentum equation for an incompressible fluid. On a uniform cartesian grid shown in Fig. 2, $\partial u/\partial x$ are calculated by using Gauss formula over the staggered finite volume around the filled circles (points A and B). In order to calculate the viscous flux through the cell interface for the shaded finite volume centered at point (l, m), $\partial u/\partial x$ at the center of the cell interface $(l + 1/2, m)$ (point E) is obtained through averaging the values of $\partial u/\partial x$ at the cell vertices $(l + \frac{1}{2}, m - \frac{1}{2})$ and $(l + \frac{1}{2}, m + \frac{1}{2})$ (points A and B), resulting in a 6-point stencil shown in Fig. 2 by the open circles. The numbers beside the circles stand for the coefficients for those points in the discretization. If we use the subscript *averaged* for this $\partial u/\partial x$ so obtained, we get

$$\left(\frac{\partial u}{\partial x}\right)_{E,\text{averaged}} = \frac{1}{4}\frac{u_{l+1,m-1} - u_{l,m-1}}{\Delta x} + \frac{1}{2}\frac{u_{l+1,m} - u_{l,m}}{\Delta x} + \frac{1}{4}\frac{u_{l+1,m+1} - u_{l,m+1}}{\Delta x}. \tag{14}$$

Notice that the right-hand side of Eq. (14) is, in fact, a weighted averaging of the finite difference formulas for $\partial u/\partial x$ at $(l + \frac{1}{2}, m - 1)$, $(l + \frac{1}{2}, m)$, and $(l + \frac{1}{2}, m + 1)$. This value of $\partial u/\partial x$ is then used to calculate the diffusion flux through the cell face at $(l + \frac{1}{2}, m)$, which is in turn used to calculate the total flux balance for the cell (l, m). If this is carried out for all the cell faces and also $\partial u/\partial y$, we get a discretization stencil for $\nabla^2 u$ shown in Fig. 3, which can be written as

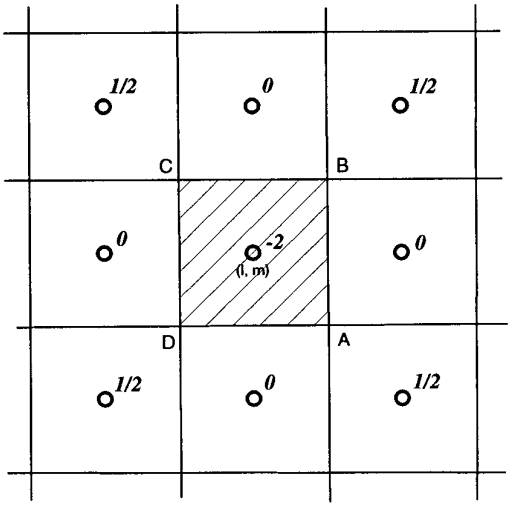

FIG. 3. Finite-volume discretization stencil for $\nabla^2 u$.

$$(\nabla^2 u)_{\text{averaged}} = \frac{\frac{1}{2}(u_{l+1,m+1} + u_{l+1,m-1} + u_{l-1,m+1} + u_{l-1,m-1}) - 2u_{l,m}}{h^2}, \tag{15}$$

where $h = \Delta x = \Delta y$.

If we apply this to a Fourier component $u = e^{I(\omega l\Delta x + \omega m\Delta y)}$, where $I = \sqrt{-1}$ and ω is the wave frequency, the Fourier symbol of the finite-difference operation is

$$Z = \frac{1}{h^2}\left\{\frac{1}{2}[e^{(I\omega\Delta x + I\omega\Delta y)} + e^{(+I\omega\Delta x - I\omega\Delta y)} + e^{(-I\omega\Delta x + I\omega\Delta y)} + e^{(-I\omega\Delta x - I\omega\Delta y)}] - 2\right\} u$$
$$= -\frac{2}{h^2}[1 - \cos(\omega\Delta x)\cos(\omega\Delta y)]u.$$

This implies that the scheme is insensitive to the Fourier mode $u = e^{I(\pi l + \pi m)}$ corresponding to the highest frequency $\omega\Delta x = \omega\Delta y = \pi$, which is an odd–even decoupled mode that can be easily identified in Fig. 3. The reason that this scheme is insensitive to this odd–even decoupled mode is due to the averaging of the terms $\partial u/\partial x$ in Eq. (14). If we directly take

$$\left(\frac{\partial u}{\partial x}\right)_E = \frac{u_{l+1,m} - u_{l,m}}{\Delta x}, \tag{16}$$

we will then obtain the usual five-point finite difference stencil for $\nabla^2 u = \partial^2 u/\partial x^2 + \partial^2 u/\partial y^2$ shown in Fig. 4. We will call it the compact form for $\nabla^2 u$

$$(\nabla^2 u)_{\text{compact}} = \frac{u_{l+1,m} + u_{l-1,m} + u_{l,m+1} + u_{l,m-1} - 4u_{l,m}}{h^2}. \tag{17}$$

The Fourier symbol of this operator is

$$Z = -\frac{2}{h^2}[(1 - \cos(\omega\Delta x)) + (1 - \cos(\omega\Delta y))]u$$

which does not allow any odd–even decoupled modes. Clearly, in order to get the compact scheme (17), one must evaluate $\partial u_i/\partial x_j$ at the center of the cell interfaces in a finite-volume method. Since there are twice as many (three times in 3D) cell faces as cell vertices for typical quadrilateral grids, directly evaluating and storing the stress tensor at the center of cell faces will require extra storage and computational time.

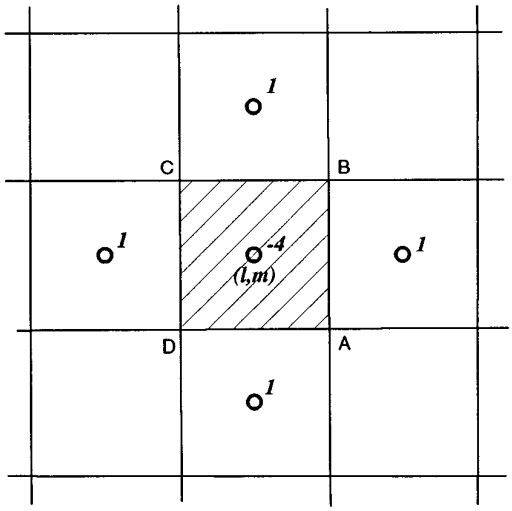

FIG. 4. Five-point compact finite-difference stencil for $\nabla^2 u$.

We here present a simple alternative, which is analogous to the approach used by Jameson and Caughey [18] in their finite-volume method for the transonic potential equation. We continue evaluating and storing the stress tensor at cell vertices as we do in our staggered finite-volume approach, but we add a correction term to Eq. (14) to recover the compact form given by Eq. (16). This correction term can be easily identified as

$$-\frac{1}{4}\left[\left(\frac{\partial u}{\partial x}\right)_{l,m+1} - 2\left(\frac{\partial u}{\partial x}\right)_{l,m} + \left(\frac{\partial u}{\partial x}\right)_{l,m-1}\right] \approx -\frac{1}{4}\frac{\partial^2}{\partial y^2}\left(\frac{\partial u}{\partial x}\right)_{l,m}\Delta y^2. \tag{18}$$

In other words, we can write

$$\left(\frac{\partial u}{\partial x}\right)_{\text{compact}} = \left(\frac{\partial u}{\partial x}\right)_{\text{averaged}} - \frac{1}{4}\frac{\partial^2}{\partial y^2}\left(\frac{\partial u}{\partial x}\right)_{l,m}\Delta y^2. \tag{19}$$

With a curvilinear grid as shown in Fig. 5, we have

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial \xi}\xi_x + \frac{\partial u}{\partial \eta}\eta_x. \tag{20}$$

This is equivalent to using Gauss formula, provided the metric coefficient ξ_x and η_x are appropriately interpreted. The problem with our staggered finite-volume approach comes from the η direction averaging of the $\partial/\partial\xi$ derivatives. A correction term is needed to make $\partial u/\partial\xi$ compact:

$$\left(\frac{\partial u}{\partial \xi}\right)_{\text{compact}} = \left(\frac{\partial u}{\partial \xi}\right)_{\text{averaged}} - \frac{1}{4}\frac{\partial^2}{\partial \eta^2}\left(\frac{\partial u}{\partial \xi}\right)_{l,m}\Delta\eta^2.$$

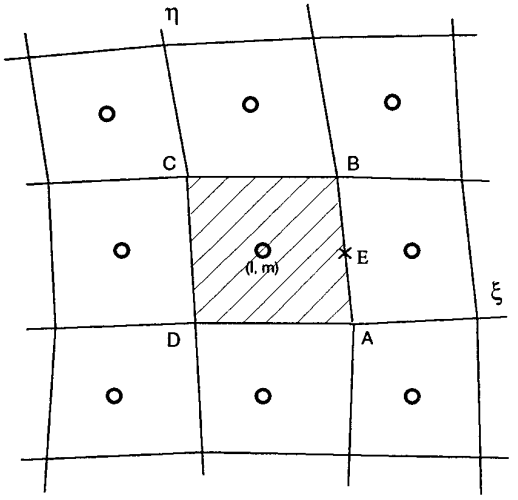

FIG. 5. Finite-volume discretization stencil on a curvilinear grid.

Note that the $\partial u/\partial \eta$ term does not cause odd–even decoupling on the computational domain. Consequently, no correction is needed. Thus we use the following to estimate the partial derivatives at the cell surfaces in order to calculate the diffusive flux balance over a control volume,

$$\frac{\partial u}{\partial x} = \left(\frac{\partial u}{\partial x}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \xi}\frac{\partial^2 u}{\partial \eta^2}\right)_{l+1/2,m} \frac{S_x^\xi}{\text{Vol}} \Delta\eta^2, \quad (21)$$

where S_x^ξ is the x component of the cell surface area vector **S**, Vol is the average volume of the cells on either side of the cell interface.

Similarly, we get

$$\frac{\partial u}{\partial y} = \left(\frac{\partial u}{\partial y}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \xi}\frac{\partial^2 u}{\partial \eta^2}\right)_{l+1/2,m} \frac{S_y^\xi}{\text{Vol}} \Delta\eta^2 \quad (22)$$

$$\frac{\partial v}{\partial x} = \left(\frac{\partial v}{\partial x}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \xi}\frac{\partial^2 v}{\partial \eta^2}\right)_{l+1/2,m} \frac{S_x^\xi}{\text{Vol}} \Delta\eta^2 \quad (23)$$

$$\frac{\partial v}{\partial y} = \left(\frac{\partial v}{\partial y}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \xi}\frac{\partial^2 v}{\partial \eta^2}\right)_{l+1/2,m} \frac{S_y^\xi}{\text{Vol}} \Delta\eta^2. \quad (24)$$

Since the correction terms involve only simple differences on the computational plane, minor computational effort is required. In the computer program, the $\partial^2 u/\partial \eta^2$ in the above equations is calculated by central differencing at the cell centers first, then their first differences normal to the cell face, $\partial/\partial\xi$, are calculated and appended to the diffusion fluxes during the assemblage of the diffusion terms. Notice that the correction terms are only needed in evaluating the diffusive fluxes through the cell faces. Since they are uniquely defined at each cell interface for the two cells on either side, they preserve the conservativeness of the overall finite-volume scheme. Their function is to convert our 9-point finite-volume discretization into approximately a compact 5-point finite difference scheme for the diffusion terms so that no odd–even decoupled modes will occur. It must also be pointed out that the correction terms do not change the second-order accuracy of the scheme and there are no free parameters involved.

If we consider the cell-interface in the η direction, we get

$$\frac{\partial u}{\partial x} = \left(\frac{\partial u}{\partial x}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \eta}\frac{\partial^2 u}{\partial \xi^2}\right)_{l,m+1/2} \frac{S_x^\eta}{\text{Vol}} \Delta\xi^2 \quad (25)$$

$$\frac{\partial u}{\partial y} = \left(\frac{\partial u}{\partial x}\right)_{\text{averaged}} - \frac{1}{4}\left(\frac{\partial}{\partial \eta}\frac{\partial^2 u}{\partial \xi^2}\right)_{l,m+1/2} \frac{S_y^\eta}{\text{Vol}} \Delta\xi^2 \quad (26)$$

and similar equations for $\partial v/\partial x$ and $\partial v/\partial y$.

Similar corrections are also formed for the diffusion terms in the k-ω equations. As mentioned in Liu and Zheng [11], our experience shows that in most cases such correction terms are not needed. It seems that the added artificial dissipation for the convective terms or the intrinsic dissipation in an upwind scheme is enough to damp out the probable odd–even decoupled mode for the diffusive terms. However, it appears that problems may arise in regions of small shear where oscillating shearing forces may appear. This situation was found in our RAE airfoil test case. In the wake of the airfoil right after the trailing edge, one can observe sawtooth-like distribution of the shearing force. Because of the use of the trapezoidal rule in evaluating the diffusive fluxes, such sawtooth variations of shearing forces were undetected by the viscous diffusivity. After implementing the correction terms the saw-tooth variations were completely eliminated.

3.3. *Multigrid Algorithm for the k-ω Equations*

After discretized in space, the governing equations are reduced to a set of ordinary differential equations in time, which can be solved by using a hybrid multistage scheme as proposed by Jameson, Schmidt, and Turkel [19]. Residual smoothing and multigrid acceleration can be applied to the Navier–Stokes equations as described in Jameson [15], Martinelli and Jameson [2], and Liu and Jameson [6]. The time integration for the k-ω equations needs some special attention. The semi-discrete k-ω equations can be written as

$$\frac{\partial}{\partial t}(\rho k) + R_k(\rho k, \rho\omega) = 0 \quad (27)$$

$$\frac{\partial}{\partial t}(\rho\omega) + R_\omega(\rho k, \rho\omega) = 0, \quad (28)$$

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

$$R_k(\rho k, \rho\omega) = \frac{1}{\Omega'}(C_k - D_k) - S_k \tag{29}$$

$$R_\omega(\rho k, \rho\omega) = \frac{1}{\Omega'}(C_\omega - D_\omega) - S_\omega. \tag{30}$$

C_k and C_ω are the discrete forms of the convective terms in the k and ω equations, respectively, 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 in a lesser nonlinear form than in Eqs.(4) and (5) as

$$S_k = \mu_t P_d - \frac{2}{3}\left(\nabla\cdot\mathbf{u}\right)(\rho k) - \frac{\beta^*}{\rho}(\rho\omega)(\rho k) \tag{31}$$

$$S_\omega = \alpha\alpha^*\rho P_d - \alpha\frac{2}{3}\left(\nabla\cdot\mathbf{u}\right)(\rho\omega) - \frac{\beta}{\rho}(\rho\omega)^2. \tag{32}$$

where

$$P_d = \tfrac{1}{2}(\varepsilon_{11}^2 + \varepsilon_{22}^2 + \varepsilon_{33}^2) + \varepsilon_{12}^2 + \varepsilon_{13}^2 + \varepsilon_{23}^2$$

$$\varepsilon_{ij} = 2\left(S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right);$$

S_{ij} is the velocity strain rate as defined in Eq. (7).

The $\mu_t P_d$ and $\alpha\,\alpha^* P_d$ terms are the major parts of production for k and ω and are always positive. The $-\frac{2}{3}(\nabla\cdot\mathbf{u})(\rho k)$ and $-\frac{2}{3}\sigma(\nabla\cdot\mathbf{u})(\rho\omega)$ terms are two minor parts for the production of k and ω, which, however, may be either positive or negative. When the flow is undergoing an expansion, $\nabla\cdot\mathbf{u} > 0$, they dissipate k or ω. Conversely, when the flow is undergoing compression, they produce k or ω. The $-(\beta^*/\rho)(\rho\omega)(\rho k)$ and the $-(\beta/\rho)(\rho\omega)^2$ are the dissipation terms which are always negative and thus annihilate k and ω. The larger these terms, the faster k and ω decay, but the system, however, becomes more stiff because of the larger negative eigenvalues. The explicit time-marching formula for the k and ω equations within each stage of a multistage time-stepping scheme can be modified to treat parts of the source terms implicitly so that time steps are not too severely restricted due to the stiffness of the k-ω equations. This in general will affect the time accuracy of a multistage time-stepping scheme. But since we are interested in reaching a steady state solution, the time accuracy is of lesser concern than obtaining a scheme with faster convergence to steady state. If we define

$$\Delta^+ = \mathrm{Max}(0, \tfrac{2}{3}\nabla\cdot\mathbf{u}), \tag{33}$$

the negative contribution of the source terms in the k and ω equations can be moved to the left-hand side of Eq. (27) and (28) to form an implicit time-marching formula within each stage of the multistage scheme. Thus, we have

$$[1 + \Delta t\,\Delta^+][(\rho k)^{n+1} - (\rho k)^n] + \Delta t\frac{\beta^*}{\rho}[(\rho k)^{n+1}(\rho\omega)^{n+1} - (\rho k)^n(\rho\omega)^n] = -R_k^n\,\Delta t \tag{34}$$

$$[1 + \Delta t\alpha\,\Delta^+][(\rho\omega)^{n+1} - (\rho\omega)^n] + \Delta t\frac{\beta^*}{\rho}[(\rho\omega)^{n+1}(\rho\omega)^{n+1} - (\rho\omega)^n(\rho\omega)^n] = -R_\omega^n\,\Delta t. \tag{35}$$

There are several ways to solve the above two nonlinear equations. Notice that Eq. (35) is independent of $(\rho k)^{n+1}$ and can be solved exactly by using the root formula for quadratic equations. After obtaining $(\rho\omega)^{n+1}$, $(\rho k)^{n+1}$ can be obtained by Eq. (34). Alternatively one may linearize Eq. (35) to avoid solving the quadratic equation for $(\rho\omega)^{n+1}$. Yet, another method, which is used in the current calculations, is to linearize both Eqs. (34) and (35) and write the solution in the following delta form

$$\delta(\rho\omega)^n = \Delta t\frac{-R_\omega^n}{1 + \Delta t(\alpha\Delta^+ + 2\beta\omega^n)} \tag{36}$$

$$\delta(\rho k)^n = \Delta t\frac{-R_k^n - \beta^*(k)^n\,\delta(\rho\omega)^n}{1 + \Delta t(\Delta^+ + \beta^*\omega^n)}, \tag{37}$$

where

$$\delta(\rho k)^n = (\rho k)^{n+1} - (\rho k)^n \tag{38}$$

$$\delta(\rho\omega)^n = (\rho\omega)^{n+1} - (\rho\omega)^n. \tag{39}$$

In a loosely coupled approach, the Navier–Stokes equations and the k-ω equations would be marched in time separately. When the Navier–Stokes equations were marched in time, the values of k, ω, and μ_t would be frozen. When the k-ω equations were marched, the flow variables ρ, ρu_i, and ρE would be fixed. In Liu and Zheng [11] a semi-loosely coupled approach was used. A five-stage time-stepping scheme with three evaluations of the viscous terms was employed for the Navier–Stokes equations. The k-ω equations were marched separately, one or more time steps, with the same five-stage scheme at the first, third, and fifth stages of the five-stage time stepping for the Navier–Stokes equations when the viscous terms were evaluated. Multigrid and residual smoothing were applied to the Navier–Stokes equations (see [6, 15, 20]) but not to the k-ω equations. As such, it was found that the convergence of the k-ω equations usually lagged behind the Navier–Stokes equations. Consequently, Liu and Zheng [11]

marched the k-ω equation four time steps for each of the three updates of the k-ω equations within each time step for the Navier–Stokes equations in order to obtain good convergence for the overall system.

Just as in the case of spatial discretization, the coupling between the Navier–Stokes equations and the two-equation turbulence model equations has significant effect on the convergence of the complete system. It is anticipated that a strongly coupled approach would result in faster convergence. Therefore, the Navier–Stokes equations and the k-ω equations are here marched in time simultaneously with the same five-stage time-stepping scheme. Neither the k and ω values, nor the flow variables in the Navier–Stokes equations are frozen in calculating the residuals of the Navier–Stokes and the turbulence model equations within the multistage time stepping. Residual smoothing and multigrid are applied uniformly to both the Navier–Stokes and the turbulence model equations. In this way the governing equations (1)–(5) are treated truly as a single system of coupled equations.

There are, however, two modifications. First, one may have the option to update the eddy viscosity by Equation (6) only at the end of each time step although the flow variables ρ, ρu_i, ρE, and k and ω are updated within each stage of the time step. Second, even though the Navier–Stokes and the k-ω equations are marched simultaneously, one may still have the option to use different time steps to reach steady state. The time step limit for the k-ω equations can be estimated by the following equations with the Courant number CFL also possibly different from that for the Navier–Stokes equations,

$$\Delta t = \frac{\mathrm{CFL} \cdot \Omega'}{\Sigma \, |\mathbf{u} \cdot \mathbf{S}\| + \sigma \Sigma \, (S^2(\mu + \mu_t)/\rho\Omega'}, \tag{40}$$

where Ω' is the cell volume of the staggered cell shown in Fig. 1, $\mathbf{S}$ is the face area vector of the cell in each coordinate direction, S is the cell face area, and the summation is over all three directions in a three-dimensional problem.

In a very recent paper, Mohammadi and Pironneau [21] presented an implicit treatment of the source terms of the incompressible k-ε equation in a multistep method in which the convection and diffusion operators are split and marched in time separately. They also proved that their implicit treatment guarantees positivity of k and ε with a Lagrangian finite element method under certain conditions. It is likely that the implicit treatment of the source terms in the compressible k-ω equations proposed in this paper may also preserve positivity under certain conditions. However, even if the numerical scheme may guarantee the positivity of k and ω, the computation may lead to low levels of ω that is not physical, resulting in excessively large values of eddy viscosity. In order to prevent this, Zheng and Liu [12] derived a lower limit for ω based on physical reasons. This is expressed in the following equation taken from [12]:

$$(\rho\omega)_{\min} = \alpha\alpha^* \rho \sqrt{P_d}. \tag{41}$$

In a single grid application, this limit can be directly imposed on ω at every time step. In a multigrid application, direct application of the above limit appears to hinder the effectiveness of multigrid. To avoid this problem, this limit is imposed by limiting the residuals of the ω equation calculated on the fine grids before they are passed to the coarse grids as

$$R_\omega^* = R_\omega - \mathrm{Max}\left(0, \frac{(\rho\omega)_{\min} - (\rho\omega)_E}{\Delta t}\right) \tag{42}$$

$$(\rho\omega)_E = (\rho\omega) - \frac{\Delta t R_\omega}{1 + \Delta t(\alpha\Delta^+ + 2\beta\omega)}, \tag{43}$$

where R_ω^* is the limited residual, $(\rho\omega)_E$ is the predicted $\rho\omega$, based on the original residual R_ω.

During the multigrid cycle, the residuals on a fine grid are passed down to the next coarse grid as forcing terms that are used to drive the update on the coarse grid. Since k and ω are defined at cell vertex, their values on the coarse grid are transferred directly from the corresponding vertex on the fine grid. As discussed in the previous section, the production term P_d has an important role on the solutions of k and ω. As such it is only calculated on the finest mesh to preserve its accuracy. The values calculated on the finest grid are then passed down to the coarse grids and used as source terms to drive the solution of k and ω. The correction calculated on each grid is passed back to the next finer grid by bilinear interpolation. To further accelerate the solution, implicit residual smoothing is also used for the k-ω equations. With the help of this technique, the allowable time steps are increased significantly. CFL values of around 7 can be used.

4. BOUNDARY CONDITIONS

H-type meshes are used for cascade flows and C-type meshes are used for airfoil flows. Boundary conditions for the Navier–Stokes equations are set in the same manner as outlined in [11]. Appropriate boundary conditions for the k-ω equations must be imposed in the far field and on solid walls. In the far field a small value of the turbulent kinetic energy is specified. In our calculations $k = 10^{-6}$. The freestream values of ω is estimated by using the following equation as Menter proposed [22]

$$\omega_\infty = \mathcal{O}\left(10 \frac{U_\infty}{L}\right). \tag{44}$$

At solid walls $k = 0$. The specific dissipation rate ω does not have a natural boundary condition. Its asymptotic behavior is specified as

$$\omega \to \frac{6\nu_w}{\beta y^2} \quad \text{as the wall distance } y \to 0. \tag{45}$$

In our calculations, the boundary condition of ω is imposed at the first point away from the wall by using the above equation. Theoretically, ω is infinite at the wall. One could simply set a large value there. However, since what we really need is to impose the asymptotic behavior specified by Eq. (45) and the value at the wall is really not useful except to obtain the value of ω at the cell interface for the third grid point from the wall through the use of Eq. (13), the ω value at the wall can be set to ensure that the interpolated value $(\omega)_a$ at the cell interface also satisfies the asymptotic solution specified in Eq. (45) for a positive normal convective velocity from the wall. Thus, the ω at wall is set to be

$$\omega_0 = \frac{19}{9}\frac{6\nu_w}{\beta y_1^2}, \tag{46}$$

where y_1 is the distance from the wall of the first grid point.

In the near wall region, the ω equation is dissipation dominant. As shown in Eq. (45), ω decays very rapidly as one moves away from the wall. During the multigrid cycle, the coarse grids cannot resolve such great variation. Therefore ω values at the wall and the first grid point are passed down without updating on each grid level.

On outlet boundaries where the flow velocity in the outer normal of the boundary is positive, only the pressure is specified. All other variables are extrapolated.

For airfoil flows, one-dimensional Riemann invariants are used to form nonreflection boundary conditions if the flow is subsonic (see Jameson, Schmidt, and Turkel [19] and Jameson [15]). For supersonic flows, all the flow quantities are set to the free stream values at the inflow. They are extrapolated from the interior at the outflow part of boundary.

5. COMPUTATIONAL RESULTS

5.1. *Cascade Flows*

To demonstrate the efficiency of the multigrid algorithm for the k-ω equations, we recalculate the turbine cascade flows that we did in Ref. [11]. The cascade was tested by Hodson and Dominy [23–25]. At its design condition this cascade has an exit isentropic Mach number of 0.7 and an incidence angle of 38.8°. The isentropic Mach number, often used by the turbomachinery community, is defined as the Mach number calculated from local static pressure

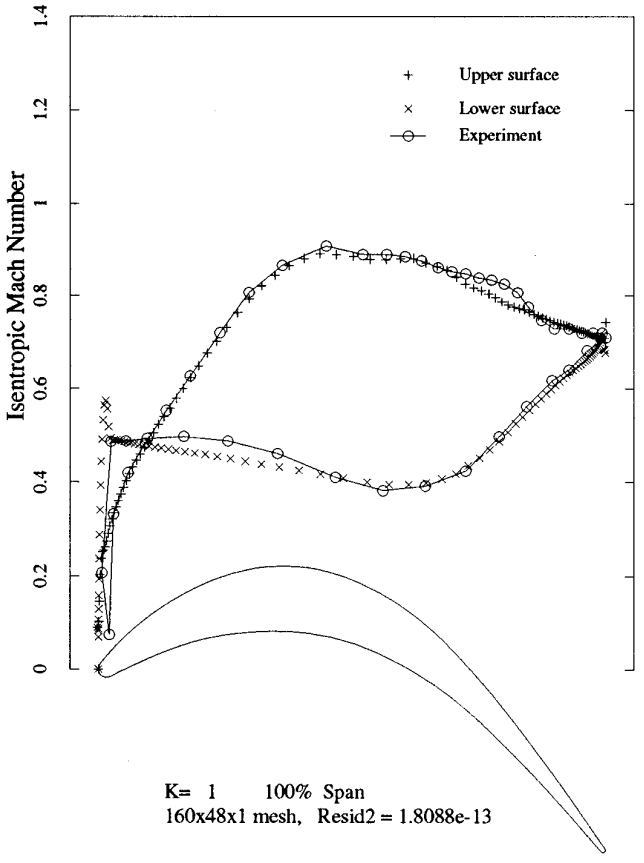

FIG. 6. Isentropic Mach Number Distribution over a Turbine Cascade at an Off-design Condition.

by assuming a constant total pressure equal to the upstream total pressure. The experimental Reynolds number based on exit velocity and blade chord length for this test case is 2.9×10^5. Although the blade is linear, the side walls have a 6° divergence. Therefore, a purely two-dimensional calculation would underpredict the isentropic Mach number on the forward part of the blade for the same exit Mach number. To avoid that, the stream tube thickness correction as described in Liu and Zheng [11] is used. The computational mesh, which contains 161×49 grid points, and all flow conditions used in the current calculation are the same as in [11]. The only difference is that we now use the new strongly coupled time integration with multigrid and residual smoothing for both the Navier–Stokes and turbulence model equations.

A difficult test condition for this cascade is when the incoming flow has a negative incidence angle of 20.3° relative to the design condition. In this case there is a large separation bubble on the pressure surface. Usually this causes slow convergence. Figure 6 shows the isentropic Mach number distribution over the cascade. The flat region of the isentropic Mach number distribution on the pressure

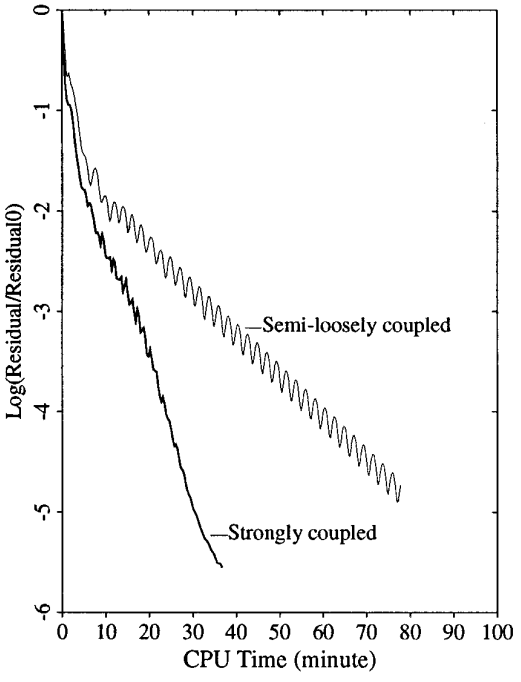

FIG. 7. Comparison of Convergence histories against CPU time for the semi-loosely coupled and the strongly coupled methods.

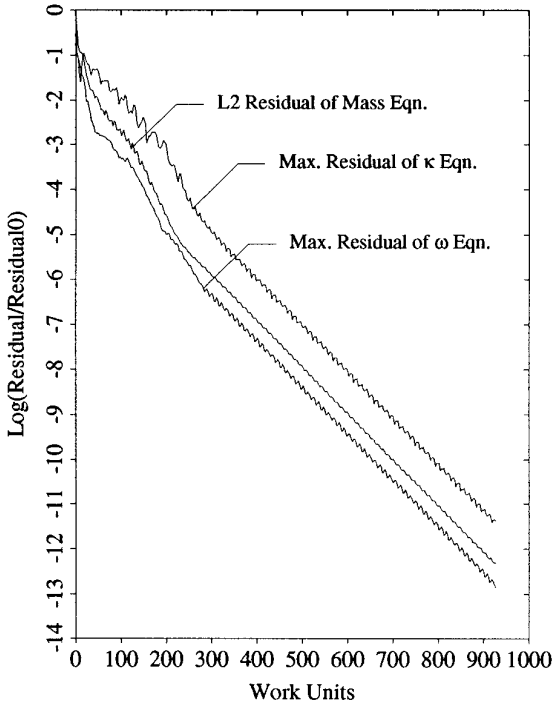

FIG. 8. Convergence history for the off-design cascade case.

side of the blade signifies the large separation bubble in the flow. The computed flow field as shown in Fig. 6 has no difference from that obtained by the semi-loosely coupled method in [11]. Figure 7 shows the comparison of the convergence history against CPU time by the loosely coupled algorithm and that by the strongly coupled algorithm. The implicit treatment of the source terms and the correction of the diffusive operators are used in both computations. It is seen that the computational time is reduced by more than half with the strongly coupled multigrid method. If the calculations are continued, the residuals keep going down continuously. As shown in Fig. 8, the residuals of mass conservation, k and ω equations are driven down more than 11 orders of magnitude in less than 1000 work units.

At the design conditions, the flow through this cascade does not have the large separation on the pressure side of the blade. Figure 9 shows the computed isentropic Mach number against experimental data. The flat isentropic Mach number distribution on the pressure surface is no longer present. Because of the absence of the large separation, the convergence of the computation is even better. As shown in Fig. 10, the residuals of each equation is driven to machine zero within 700 work units.

5.2. *Transonic Airfoil Flow*

This method is extended to compute the airfoil flows. The flow over the RAE airfoil is calculated with an upwind

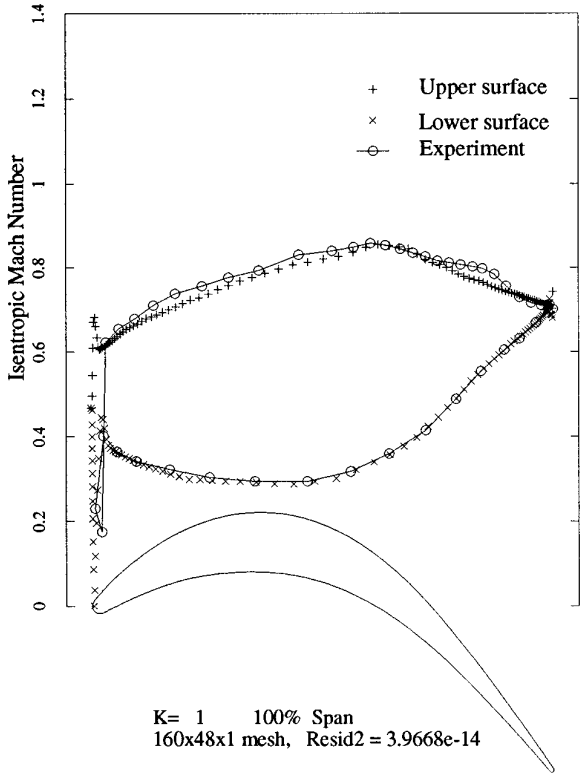

FIG. 9. Isentropic Mach number distribution over a turbin cascade at its design condition.

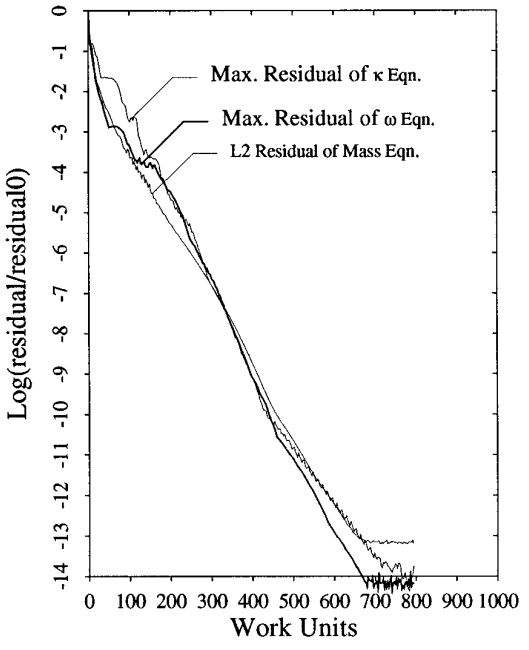

FIG. 10. Convergence history for the cascade case at design condition.

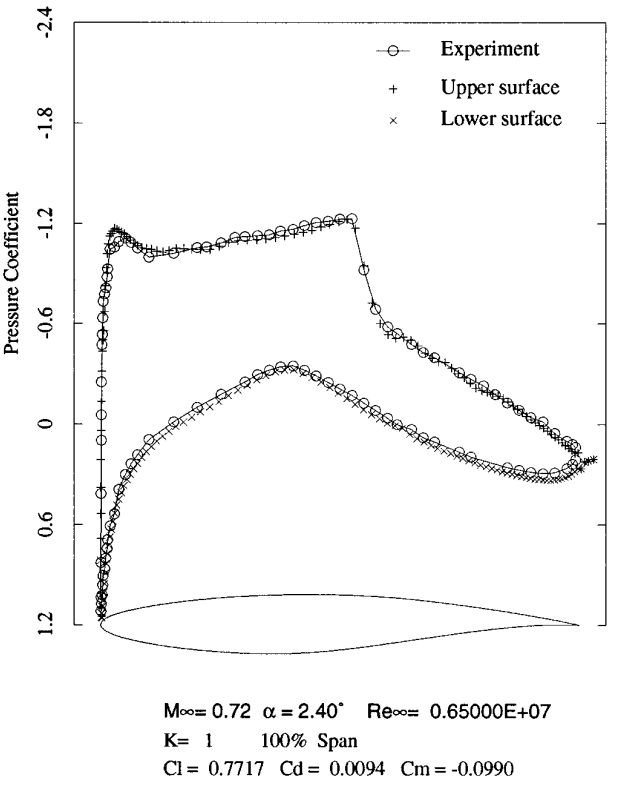

FIG. 11. Pressure Distribution for the RAE2822 airfoil case #6.

version of our program using a MUSCL type second-order interpolation [16] and Roe's approximate Riemann solver [17] for the convective terms (see Zheng and Liu [12]). Test data of this airfoil were reported by Cook, McDonald, and Firmin in [26]. Computation of the test case number 6 in [26] is presented here. The free stream mach number of the flow is 0.725 for this case. The Reynolds number is 6.5×10^6 based on chord length. The nominal experimental angle of attack is 2.92° but is adjusted to 2.4° to account for wall interference. This adjustment is the same as used by Martinelli and Jameson [2]. Figure 11 shows the calculated pressure coefficient compared with experimental data. The shock wave is captured almost exactly, showing very good promise of the k-ω model in the simulation of transonic airfoil flows. The experimental normal force coefficient, pitching moment coefficient around 0.25 chord, and the drag coefficient are $c_N = 0.743$, $c_m = -0.095$, and $c_d = 0.0127$, respectively. The computational results give c_N=0.770, $c_m = -0.098$, and $c_d = 0.0163$ which includes both wave and skin-friction drag.

Figure 12 shows the convergence history for 300 time steps with three levels of multigrid for both the semi-loosely coupled approach and the strongly coupled approach. Again it is seen that the latter approach yields a better convergence rate. However, the overall convergence rate for this case is slower, compared to the cascade flow calculations. This is due to the higher grid aspect and stretching ratios at the specified Reynolds number and the larger extent of the computational domain for this case. The far field boundary is 18 chord lengths away from the

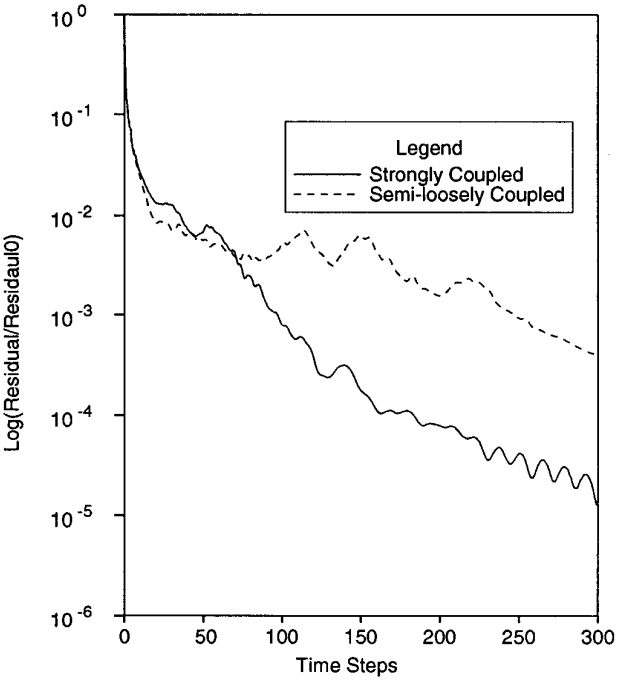

FIG. 12. Convergence history for the RAE2822 airfoil case #6.

airfoil for this calculation. Nevertheless, the convergence rate is seen to be comparable to that shown by Martinelli and Jameson [2] with a multigrid code using central differencing, scalar artificial disspation, and the Baldwin–Lomax algebraic turbulence model. Further increase of convergence for multigrid methods may hinge on proper preconditioning to relieve the stiffness due to grid stretching and large cell aspect ratios.

6. SUMMARY

A strongly coupled multigrid algorithm is developed for solving the Navier–Stokes and Wilcox's k-ω two-equation turbulence model equations. The Navier–Stokes and the turbulence model equations are treated as a single system of equations and marched in time by the same multistage scheme with multigrid. Source terms in the turbulence equations are treated implicitly within each stage of the multistage time stepping. Results for a turbine cascade shows that the method greatly increases the computational efficiency compared to a semi-loosely coupled algorithm. Residuals of both the Navier–Stokes equations and the k-ω equations can be reduced to machine zero in less than 1000 work units.

A correction method for removing potential odd–even decoupled modes in the finite-volume discretization of diffusion terms is also presented. This method does not require storing and calculating the stress tensor for every cell face. Thus, it needs less memory and computer time. Yet, it can convert a 9-point finite-volume discretization stencil into approximately a compact 5-point finite difference stencil. Such corrections may be needed to prevent computational results of oscillating shear stresses.

ACKNOWLEDGEMENT

This research was supported in part by the California Space Institute under Grant CS-35-92, the National Science Foundation under grant CTS-9410800 and the Academic Senate Committee on Research at UC Irvine. Computer time was provided on the Convex-C240 computer by the Office of Academic Computing at UC Irvine and on the Cray-YMP by the San Diego Super Computer Center.

REFERENCES

1. J. L. Steger, *AIAA J.* **16**(7), 679 (1978).
2. L. Martinelli and A. Jameson, AIAA Paper-88-0414, January 1988 (unpublished).
3. D. J. Mavriplis, *AIAA J.* **29**(12), 2086 (1991).
4. R. R. Varma and D. A. Caughey, AIAA Paper 91-1571, June 1991 (unpublished).
5. M. Giles, *J. Turbomach. Trans. ASME* **115**(1), 110 (1993).
6. F. Liu and A. Jameson, *AIAA J.* **31**(10), 1785(1993).
7. M. G. Turner and I. K. Jennions, ASME 92-GT-308, 1992 (unpublished).
8. R. F. Kunz and B. Lakshminarayana, *AIAA J.* **30**(1), 13 (1992).
9. H. Lin, D. Y. Yang, and C. C. Chieng, AIAA Paper 93-3316, July 1993; *11th AIAA Computational Fluid Dynamics Conference.*
10. D. C. Wilcox, *AIAA J.* **26**(11), 1299 (1988).
11. F. Liu and X. Zheng, *AIAA J.* **32**(8), 1589 (1994).
12. X. Zheng and F. Liu, *AIAA J.* **33**(6), 991 (1995).
13. F. H. Harlow and J. E. Welch, *Phys. Fluids* **8,** 2182 (1965).
14. S. V. Patanka, *Numerical Heat Transfer and Fluid Flow* (Hemisphere, Washington, DC 1980).
15. A. Jameson, MAE Report 1651, Department of Mechanical and Aerospace Engineering, Princeton University, 1983 (unpublished).
16. W. K. Anderson, J. L. Thomas, and B. Van Leer, *AIAA J.* **24**(9), 1453 (1986).
17. P. L. Roe, *J. Comput. Phys.* **43**(7), 357 (1981).
18. A. Jameson and D. A. Caughey, AIAA Paper 77-635, 1977 (unpublished).
19. A. Jameson, W. Schmidt, and E. Turkel, AIAA Paper 81-1259, 1981 (unpublished).
20. F. Liu, Ph.D. dissertation, Department of Mechanical and Aerospace Engineering, Princeton University, June 1991 (unpublished).
21. B. Mohammadi and O. Pironneau, *Int. J. Numer. Methods Fluids* **20,** 819 (1995).
22. F. R. Menter, *AIAA J.* **30**(6), 1657 (1992).
23. H. P. Hodson and R. G. Dominy, *J. Eng. Gas Turbines Power* **107** (1985).
24. H. P. Hodson and R. G. Dominy, *J. Turbomach.* **109**(2), 177 (1987).
25. H. P. Hodson and R. G. Dominy, *J. Turbomach.* **109**(2), 201 (1987).
26. M. A. McDonald, P. H. Cook, and M. C. P. Firmin, AGARD Advisory Report No. 138, May 1979 (unpublished).