INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2004; **46**:59–84 (DOI: 10.1002/fld.742)

A finite volume unstructured multigrid method for efficient computation of unsteady incompressible viscous flows

Chin Hoe Tai and Yong Zhao*,†

School of Mechanical and Production Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore

SUMMARY

An unstructured non-nested multigrid method is presented for efficient simulation of unsteady incompressible Navier–Stokes flows. The Navier–Stokes solver is based on the artificial compressibility approach and a higher-order characteristics-based finite-volume scheme on unstructured grids. Unsteady flow is calculated with an implicit dual time stepping scheme. For efficient computation of unsteady viscous flows over complex geometries, an unstructured multigrid method is developed to speed up the convergence rate of the dual time stepping calculation. The multigrid method is used to simulate the steady and unsteady incompressible viscous flows over a circular cylinder for validation and performance evaluation purposes. It is found that the multigrid method with three levels of grids results in a 75% reduction in CPU time for the steady flow calculation and 55% reduction for the unsteady flow calculation, compared with its single grid counterparts. The results obtained are compared with numerical solutions obtained by other researchers as well as experimental measurements wherever available and good agreements are obtained. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: unstructured grid; multigrid; high-order scheme; incompressible flow; unsteady flow

1. INTRODUCTION

In recent years, many advanced Computational Fluid Dynamics (CFD) methods have been developed for accurate and efficient simulation of fluid flows over complex geometries. The steady-state numerical solution of a non-linear system of partial-differential equations, such as the Navier–Stokes equations, is often found through iterative methods. Improving the rate of convergence of such iterative methods is crucial for practical applications of CFD simulation, which would result in more cost-effective design and analysis. Multigrid techniques

*Correspondence to: Yong Zhao, School of Mechanical and Production Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore.
†E-mail: myzhao@ntu.edu.sg

Contract/grant sponsor: Nanyang Technological University
Contract/grant sponsor: Agency for Science, Technology and Research (A*STAR)

Published online 29 June 2004
Copyright © 2004 John Wiley & Sons, Ltd.
Received October 2000
Revised 5 April 2004

have been demonstrated as an efficient means for obtaining steady-state numerical solutions to both the compressible Euler and Navier–Stokes equations on unstructured meshes in two and three dimensions [1–4], which find their applications mainly in aerospace engineering. Recent years also saw an increasing interest in applying the above methods to steady incompressible flow simulation [5–8] through the use of the artificial compressibility method proposed by Chorin [9], a field which has been traditionally dominated by the pressure-based method such the SIMPLE-type methods and the projection method. For most real life fluid flow problems, unsteady flow is the rule and steady flow is the exception. However, it is still very time consuming to simulate unsteady flows, especially unsteady incompressible viscous flows due to their elliptic nature, which means that global iterations are required to achieve the divergence free condition. To simulate unsteady incompressible flows with the artificial compressibility method, a dual time stepping scheme is necessary. While physical time integration can usually be obtained with a three-point second-order implicit scheme, time stepping in pseudo time can be explicit [10–12] or implicit [8, 13, 14]. Liu *et al.* [11] have developed a multigrid method using the dual time stepping scheme on structured grids for the computation of unsteady incompressible viscous flows while Lin [12] has attempted the unstructured multigrid method for calculating unsteady inviscid flows. In this study, we aim to extend the high-order characteristics-based finite-volume scheme for unstructured grids in Reference [15] to simulate unsteady incompressible viscous flows by introducing an unstructured multigrid method. A multistage Runge–Kutta time-stepping scheme in pseudo time is adopted and coupled with the unstructured multigrid method. The ultimate aim of this study is to use this combination to produce an highly efficient and flexible method for simulating unsteady flows with complex geometries.

2. MATHEMATICAL FORMULATION

The two-dimensional Navier–Stokes equations for incompressible unsteady flows, modified by the artificial compressibility method, can be written in vector form with dimensionless parameters:

$$\mathbf{C}\frac{\partial \mathbf{W}}{\partial \tau}+\mathbf{K}\frac{\partial \mathbf{W}}{\partial t}+\nabla\cdot\boldsymbol{F}_{\mathrm{c}}=\nabla\cdot\boldsymbol{F}_{\mathrm{v}} \tag{1}$$

where

$$\mathbf{W}=\begin{bmatrix}p\\u\\v\end{bmatrix},\quad \boldsymbol{F}_{\mathrm{c}}=\begin{bmatrix}\boldsymbol{U}\\u\boldsymbol{U}+(p/\rho)\boldsymbol{i}\\v\boldsymbol{U}+(p/\rho)\boldsymbol{j}\end{bmatrix},\quad \boldsymbol{F}_{\mathrm{v}}=\begin{bmatrix}0\\\dfrac{1}{Re}\left(\dfrac{\partial u}{\partial x}\boldsymbol{i}+\dfrac{\partial u}{\partial y}\boldsymbol{j}\right)\\\dfrac{1}{Re}\left(\dfrac{\partial v}{\partial x}\boldsymbol{i}+\dfrac{\partial v}{\partial y}\boldsymbol{j}\right)\end{bmatrix}$$

$$\mathbf{K}=\begin{bmatrix}0&0&0\\0&1&0\\0&0&1\end{bmatrix},\quad \mathbf{C}=\begin{bmatrix}1/\beta&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

Copyright © 2004 John Wiley & Sons, Ltd.

where $\mathbf{W}$ is the flow field variable vector, $\boldsymbol{F}_{\mathbf{c}}$ and $\boldsymbol{F}_{\mathbf{v}}$ are the convective and viscous flux vectors, respectively, β being a constant called artificial compressibility. $\mathbf{K}$ is the unit matrix, except that the first element is zero and $\mathbf{C}$ is a preconditioning matrix.

Equation (1) can be recast in an integral form as follows:

$$\mathbf{C}\frac{\partial}{\partial\tau}\iint_S \mathbf{W}\,\mathrm{d}S + \mathbf{K}\frac{\partial}{\partial t}\iint_S \mathbf{W}\,\mathrm{d}S + \iint_S \nabla\cdot(\boldsymbol{F}_{\mathbf{c}} - \boldsymbol{F}_{\mathbf{v}})\,\mathrm{d}S = 0 \tag{2}$$

Equation (2) is equivalent to the following equation:

$$\mathbf{C}\frac{\partial}{\partial\tau}\iint_S \mathbf{W}\,\mathrm{d}S + \mathbf{K}\frac{\partial}{\partial t}\iint_S \mathbf{W}\,\mathrm{d}S + \oint_l (\boldsymbol{F}_{\mathbf{c}} - \boldsymbol{F}_{\mathbf{v}})\,\mathrm{d}\boldsymbol{l} = 0$$

Once the artificial steady state is reached, those derivative terms with respect to τ become zero and the above equation reduces to the following equation:

$$\mathbf{K}\frac{\partial}{\partial t}\iint_S \mathbf{W}\,\mathrm{d}S + \oint_l (\boldsymbol{F}_{\mathbf{c}} - \boldsymbol{F}_{\mathbf{v}})\,\mathrm{d}\boldsymbol{l} = 0 \tag{3}$$

Equation (3) shows that the preconditioning matrix does not affect the solution and the original unsteady incompressible Navier–Stokes equations are fully recovered.

3. NUMERICAL METHODS

A new unstructured-grid high-order characteristics-based upwind finite-volume algorithm [17] is used to solve the governing equations, the outline of the method is described in the following sections.

3.1. Finite volume formulation

The 2D equations in (2) are discretized on an unstructured grid and a cell-vertex scheme is adopted here, i.e. all computed variables in vector $\mathbf{W}$ are stored at vertices of the triangular cells. For every vertex, as shown in Figure 1, a control volume is constructed by joining the centres of the cells to the centres of the edges using the median dual of the triangular grid.

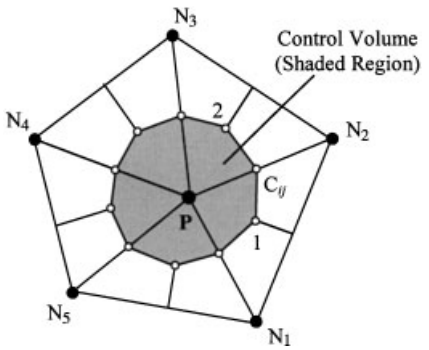

Figure 1. Construction of control volume for node P.

Copyright © 2004 John Wiley & Sons, Ltd.

Spatial discretization is performed by using the integral form of the conservation equations over the control volume surrounding node or vertex P, as shown in the following equation:

$$\mathbf{C}\frac{\partial}{\partial\tau}\iint_{S_{cv}}\mathbf{W}_p\,dS+\mathbf{K}\frac{\partial}{\partial t}\iint_{S_{cv}}\mathbf{W}_p\,dS+\iint_{S_{cv}}\nabla\cdot\boldsymbol{F}_c\,dS=\iint_{S_{cv}}\nabla\cdot\boldsymbol{F}_v\,dS \tag{4}$$

In order to introduce the upwind scheme using an edge-based procedure, the convective term is transformed into a summation as in the following equation:

$$\iint_{S_{cv}}\nabla\cdot\boldsymbol{F}_c\,dS=\oint_{L_{cv}}\boldsymbol{F}_c\cdot\boldsymbol{n}\,dl=\sum_{k=1}^{\text{nbseg}}[(\boldsymbol{F}_c)_{ij}^k\cdot\boldsymbol{n}\Delta l_k] \tag{5}$$

where nbseg is the number of the edges connected to node P, $(\boldsymbol{F}_c)_{ij}^k$ is the convection flux through the part of control volume boundary (similar to $1-C_{ij}-2$ in Figure 1). The length of the boundary is Δl_k and its effective outward normal unit vector is $\boldsymbol{n}$. Therefore, all the fluxes are calculated for the edges and then collected at the two ends of each edge for updating of flow variables using the time-marching scheme.

The viscous term is calculated using a cell-based method, which is shown in the following equation:

$$\iint_{S_{cv}}\nabla\cdot\boldsymbol{F}_v\,dS=\oint_{L_{cv}}\boldsymbol{F}_v\cdot d\boldsymbol{l}=\sum_{i=1}^{\text{ncell}}(\boldsymbol{F}_v\cdot\Delta\boldsymbol{l}_c)_i=\frac{1}{2}\sum_{i=1}^{\text{ncell}}(\boldsymbol{F}_v\cdot\Delta\boldsymbol{l}_p)_i \tag{6}$$

where $\Delta\boldsymbol{l}_{ci}$ is the part of control volume boundary in cell C_i, and $\Delta\boldsymbol{l}_{pi}$ is the edge vector of the cell edge opposite to node P of the triangle under consideration. Here the $(\boldsymbol{F}_v)_i$ is calculated at the centre of the triangular cell, which can be obtained by using the Green's Theorem and the variables at the three vertices of the triangle. Here ncell is the number of cells surrounding node P. The viscous flux in Equation (6) is actually calculated in a cell-by-cell manners and then collected at the nodes of the cells for the calculation of the residuals at all the nodes.

3.2. Upwind-biased interpolation

Here an edge-based method for calculating the total inviscid flux is adopted by calculating and storing the flux integrals based on the associated edges. The left and right state vectors $\mathbf{W}_L$ and $\mathbf{W}_R$ on both sides of a control volume surface associated an edge ij can be evaluated using an upwind-biased interpolation scheme along the edge, which are shown as follows:

$$\mathbf{W}_L=\mathbf{W}_i+\frac{1}{4}[(1-\kappa)\Delta_i^-+(1+\kappa)\Delta_i^+] \tag{7a}$$

$$\mathbf{W}_R=\mathbf{W}_i-\frac{1}{4}[(1-\kappa)\Delta_j^++(1+\kappa)\Delta_j^-] \tag{7b}$$

where

$$\Delta_i^+=\Delta_j^-=\mathbf{W}_j-\mathbf{W}_i$$

$$\Delta_i^-=\mathbf{W}_i-\mathbf{W}_{i-1}=2\boldsymbol{ij}\cdot\nabla\mathbf{W}_i-(\mathbf{W}_j-\mathbf{W}_i)=2\boldsymbol{ij}\cdot\nabla\mathbf{W}_i-\Delta_i^+$$

$$\Delta_j^+=\mathbf{W}_{j+1}-\mathbf{W}_j=2\boldsymbol{ij}\cdot\nabla\mathbf{W}_j-(\mathbf{W}_j-\mathbf{W}_i)=2\boldsymbol{ij}\cdot\nabla\mathbf{W}_j-\Delta_j^-$$

Copyright © 2004 John Wiley & Sons, Ltd.

Therefore, substituting the above equations into Equations (7a) and (7b), the final formulas based on the upwind-biased interpolation scheme are obtained:

$$\mathbf{W}_{\mathrm{L}} = \mathbf{W}_i + \frac{1}{2}[(1-\kappa)\boldsymbol{ij} \cdot \nabla \mathbf{W}_i + \kappa \Delta_i^+] \tag{8a}$$

$$\mathbf{W}_{\mathrm{R}} = \mathbf{W}_j - \frac{1}{2}[(1-\kappa)\boldsymbol{ij} \cdot \nabla \mathbf{W}_j + \kappa \Delta_j^-] \tag{8b}$$

where κ is set to $\frac{1}{3}$, which corresponds to a nominally third-order accuracy. The two values will then be used in the characteristics-based method, which will be introduced in the following section.

3.3. The upwind characteristics-based method

Similar to the approach in Reference [16] for compressible flow and that in Reference [17] for incompressible flow on structured grids, a high-order characteristics-based scheme for incompressible flow and heat transfer on arbitrary unstructured grids has been developed in Reference [15]. A 2D version of this is briefly shown below. Suppose that ξ is a new co-ordinate outward normal to the boundary of a control volume that surrounds a particular vertex. In order to extend the method of characteristics to the unstructured grid solver, it is assumed that flow in ξ direction is approximately one-dimensional. In the $\tau - \xi$ space as shown in Figure 2, flow variable $\mathbf{W}$ at pseudo time level $m+1$ can be calculated along a characteristics k using a Taylor series expansion and the initial value at pseudo time level $m(\mathbf{W}^k)$,

$$\mathbf{W} = \mathbf{W}^k + \mathbf{W}_\xi \xi_\tau \Delta\tau + \mathbf{W}_\tau \Delta\tau \tag{9}$$

and

$$\mathbf{W}_\tau = \frac{\mathbf{W} - \mathbf{W}^k}{\Delta\tau} - \mathbf{W}_\xi \xi_\tau \tag{10}$$

A wave speed λ^k is introduced

$$\xi_\tau = \lambda^k \sqrt{\xi_{x_i} \xi_{x_i}}$$

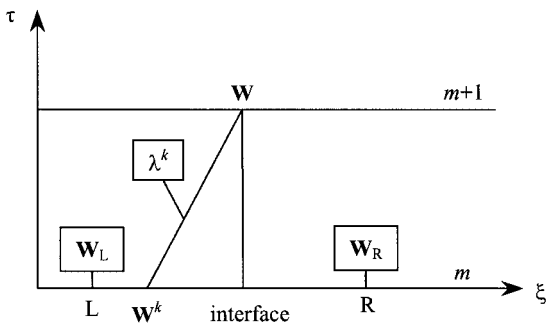

Figure 2. τ-ξ co-ordinate.

Copyright © 2004 John Wiley & Sons, Ltd.

and the unit normal vector components are

$$n_{x_j} = \frac{\xi_{x_j}}{\sqrt{\xi_{x_i}\xi_{x_i}}}$$

The flow parameters at $(m+1)$ pseudo time level are then calculated using the characteristics-based method along the three characteristics:

$$u = fn_x + u^0 n_y^2 - v^0 n_x n_y \tag{11}$$

$$v = fn_y + v^0 n_x^2 - u^0 n_y n_x \tag{12}$$

$$p = p^1 - \lambda^1[(u-u^1)n_x + (v-v^1)n_y] \tag{13}$$

where

$$f = \frac{1}{2C}[p^1 - p^2 + n_x(\lambda^1 u^1 - \lambda^2 u^2) + n_y(\lambda^1 v^1 - \lambda^2 v^2)]$$

$$C = \sqrt{(\lambda^0)^2 + \beta}; \quad \lambda^0 = un_x + vn_y; \quad \lambda^1 = \lambda^0 + C; \quad \lambda^2 = \lambda^0 - C$$

Flow quantities at $m+1$ pseudo time level obtained from the above equations on the characteristics are then used to calculate convection fluxes at the control volume interface. While those on different characteristics at m time level are approximately evaluated by an upwind scheme using the signs of the characteristics as suggested in Reference [17].

$$\mathbf{W}^j = \frac{1}{2}[(1+\mathrm{sign}(\lambda^j))\mathbf{W}_\mathrm{L} + (1-\mathrm{sign}(\lambda^j))\mathbf{W}_\mathrm{R}] \tag{14}$$

where $\mathbf{W}_\mathrm{L}$ and $\mathbf{W}_\mathrm{R}$ are obtained by the high-order upwind-biased interpolation as introduced in the previous section.

3.4. Dual time-stepping algorithm

Finally, for a given node P, the spatially discretized equations form a system of coupled ordinary differential equations, which can be reformulated as

$$\mathbf{C}\frac{\partial}{\partial\tau}(\Delta S_\mathrm{cv}\mathbf{W}_\mathrm{p}) + \mathbf{K}\frac{\partial}{\partial t}(\Delta S_\mathrm{cv}\mathbf{W}_\mathrm{p}) = -\left\{\sum_{k=1}^{\mathrm{nbseg}}[(\boldsymbol{F}_\mathbf{c})_{ij}^k \cdot \boldsymbol{n}\Delta l_k] - \frac{1}{2}\sum_{i=1}^{\mathrm{ncell}}(\boldsymbol{F}_\mathbf{v}\cdot\boldsymbol{n}\Delta l_P)_i\right\}$$
$$= -R(\mathbf{W}_\mathrm{P}) \tag{15}$$

where $R(\mathbf{W}_\mathrm{P})$ represents the residual which includes the contributions of the convective and diffusive fluxes and ΔS_cv is the control volume of node P.

Copyright © 2004 John Wiley & Sons, Ltd.

An implicit scheme is adopted to approximate Equation (15) in physical time and the semi-discrete equations are

$$\mathbf{C}\frac{\partial}{\partial\tau}(\Delta S_{\mathrm{cv}}\mathbf{W}_{\mathrm{p}})+\mathbf{K}\frac{\partial}{\partial t}(\Delta S_{\mathrm{cv}}^{n+1}\mathbf{W}_{\mathrm{p}}^{n+1})=-R(\mathbf{W}_{\mathrm{P}}^{n+1}) \tag{16}$$

The superscript $(n+1)$ denotes the physical time level $(n+1)\Delta t$ and all the variables are evaluated at this time level. The time dependent term in Equation (16) can be discretized by a three-point second-order difference scheme, thus it becomes

$$\begin{aligned}\mathbf{C}\frac{\partial}{\partial\tau}(\Delta S_{\mathrm{cv}}\mathbf{W}_{\mathrm{p}}) &= -R(\mathbf{W}_{\mathrm{P}}^{n+1})-\mathbf{K}\left(\frac{1.5\Delta S_{\mathrm{cv}}^{n+1}\mathbf{W}_{\mathrm{p}}^{n+1}-2.0\Delta S_{\mathrm{cv}}^{n}\mathbf{W}_{\mathrm{p}}^{n}+0.5\Delta S_{\mathrm{cv}}^{n-1}\mathbf{W}_{\mathrm{p}}^{n-1}}{\Delta t}\right)\\ &= \tilde{R}(\mathbf{W}_{\mathrm{p}}^{n+1})\end{aligned} \tag{17}$$

where $\tilde{R}(\mathbf{W}_{\mathrm{p}}^{n+1})$ is the new modified residual which contains both the time derivative and flux vectors. The advantage of the above implicit scheme is that the physical time step size is not restricted by numerical stability, but only by numerical accuracy. This is especially useful in unsteady flow simulation. The derivative with respect to a fictitious pseudo time τ is discretized as

$$\Delta S_{\mathrm{cv}}\frac{\mathbf{W}_{\mathrm{p}}^{n+1,m+1}-\mathbf{W}_{\mathrm{p}}^{n+1,m}}{\Delta\tau}=\tilde{R}(\mathbf{W}_{\mathrm{p}}^{n+1,m}) \tag{18}$$

whose solution is sought by marching to a pseudo steady state in τ. Here m and $(m+1)$ denote the initial and final pseudo time levels. Once the artificial steady state is reached, the derivative of $\mathbf{W}_{\mathrm{p}}$ with respect to τ becomes zero, and the solution will satisfy $\tilde{R}(\mathbf{W}_{\mathrm{p}}^{n+1})=0$. Hence, the original unsteady Navier–Stokes equations are fully recovered. Therefore, instead of solving each time step in the physical time domain (t), the problem is transformed into a sequence of steady-state computations in the artificial time domain (τ). Equation (18) is integrated in pseudo time by the five-stage Runge–Kutta time stepping scheme.

4. ACCELERATION TECHNIQUES

In this work, time-dependent calculations require the convergence of the Navier–Stokes equations to the steady state in pseudo time for each real time step. The methods used to accelerate convergence to steady state in pseudo time are local time stepping, implicit residual smoothing and multigrid. Since these convergence acceleration techniques are only used to accelerate convergence in pseudo time, therefore they will not directly affect the accuracy in real time. In the dual time-stepping algorithm, global time stepping is used to advance the solution in real or physical time whereas local time stepping is used to advance the solution in pseudo time. In this work, the large variation in grid size in the unstructured mesh will restrict the time step used and the smallest control volume dictates the maximum time step. In order to overcome the above problems, each control volume can be advanced in pseudo time by its own maximum local time step, which greatly enhances the convergence rate. The local time step size can be estimated via the Courant–Friedrichs–Lewy (CFL)

Copyright © 2004 John Wiley & Sons, Ltd.

stability condition

$$\Delta\tau = \mathrm{CFL} \cdot \frac{\Delta l}{|\boldsymbol{U}| + c} \tag{19}$$

where Δl is the length scale associated with a node under consideration. Normally, it is taken as the smallest height of all the cells sharing the node. For global time stepping, Δl is taken as the smallest value of all the nodes in the domain. And c is the speed of sound. Another condition that limits the time step in τ is $\Delta\tau \leqslant \frac{2}{3}\Delta t$.

4.1. Implicit residual smoothing

In order to speed up the convergence rate, an implicit residual smoothing scheme developed for unstructured grids [15] is employed. The idea behind this is to replace the residual at one point of the flow field with a smoothed or weighted average of the residuals at the neighbouring points. The averaged residuals are calculated implicitly in order to increase the maximum CFL number, thus increasing the convergence rate. Normally this procedure allows the CFL number to be increased by a factor of 2 or 3. The smoothing equation for a vertex k can be expressed as follows:

$$\bar{R}_k = R_k + \varepsilon \nabla^2 \bar{R}_k \tag{20}$$

where R is the original residual, $\bar{R}$ is smoothed residual and ε is the smoothing coefficient, which can be defined as,

$$\varepsilon = \max\left\{\frac{1}{4}\left[\left(\frac{\mathrm{CFL}}{\mathrm{CFL}^*}\right)^2 - 1\right], 0\right\} \tag{21}$$

where CFL^* is the maximum CFL number of the basic scheme. The solution to the above equations can be obtained on an unstructured grid by using Jacobi iterative method as follows,

$$\bar{R}_k^{(m)} = R_k^{(0)} + \varepsilon \sum_{i=1}^{\mathrm{numnod}(k)} [\bar{R}_i^{(m)} - \bar{R}_k^{(m)}]$$

i.e.

$$\bar{R}_k^{(m)} = \frac{R_k^{(0)} + \varepsilon \sum_{i=1}^{\mathrm{numnod}(k)} \bar{R}_i^{(m-1,m)}}{1 + \varepsilon \cdot \mathrm{numnod}(k)} \tag{22}$$

where $\mathrm{numnod}(k)$ is the number of neighbouring nodes of vertex k.

4.2. Multigrid method

The multigrid method was originally developed for elliptic equations by Fedorenko [18] and the full potential of the multigrid approach was demonstrated by Brandt [19]. Later, Jameson *et al.* [20, 21] applied the multigrid method to the solution of the Euler equations for compressible flows. The philosophy of the multigrid method is to carry out early iterations on a fine grid and then progressively transfer these flow field variables and residuals to a series of coarser grids. On the coarser grids, the low frequency errors become high frequency ones

Copyright © 2004 John Wiley & Sons, Ltd.

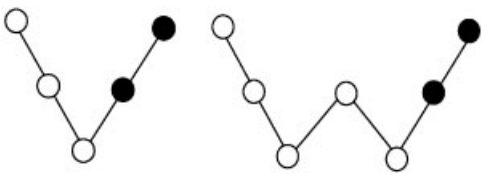

Figure 3. V- and W-cycle for three grid levels. ○ denotes flow field computation on a particular level and ● denotes interpolation of corrections of flow field variables.

and they can be easily eliminated by a time stepping scheme. The flow is then solved on the coarser grids and the corrections are then interpolated back to the fine grid. The process is repeated over a sufficient number of times until satisfactory convergence on the fine grid is achieved.

For unstructured meshes, there are two approaches to generating the multi-level coarse grids, one being the agglomeration method which agglomerates the fine mesh cells or control volumes [1–3], and the other being the non-nested mesh method using independently generated non-nested (or overset) coarse meshes [4, 7]. In this study the latter approach is adopted for ease of implementation. Two different cycle strategies have been investigated in the present work, which are V- and W-cycles. Figure 3 depicts the difference between the two strategies.

The solution on the coarse grid $(h+1)$ is initialized by transferring the flow field variables from the fine grid (h) using the transfer operator T_h^{h+1},

$$\mathbf{W}_{h+1}^{(0)} = T_h^{h+1}\mathbf{W}_h \tag{23}$$

where $\mathbf{W}_{h+1}^{(0)}$ is the initial values transferred from the fine grid and $\mathbf{W}_h$ is the solution from the fine grid. In order to drive the coarser grid solution using the fine grid residual, a forcing function is calculated at the first stage of the Runge–Kutta scheme and subsequently, this forcing term is added to the residual on the coarse grid. The forcing function on the coarse grid is

$$P_{h+1} = Q_h^{h+1}R(\mathbf{W}_h) - R(\mathbf{W}_{h+1}^{(0)}) \tag{24}$$

where Q_h^{h+1} is the residual transfer operator based on the weighted average of the values of the fine nodes. After calculating the variables on the coarsest grid, the corrections are evaluated and interpolated back to the fine grid. The correction is the difference between the newly computed value on the coarse grid, $\mathbf{W}_{h+1}^{+}$, and the initial value that was transferred from the finer grid, $\mathbf{W}_{h+1}^{(0)}$, and this correction is transferred to the fine grid and added to the solution on that grid, i.e.

$$\mathbf{W}_h^{+} = \mathbf{W}_h + I_{h+1}^{h}(\mathbf{W}_{h+1}^{+} - \mathbf{W}_{h+1}^{(0)}) \tag{25}$$

where I_{h+1}^{h} is an *interpolation* operator from the coarse grid to the fine grid and $\mathbf{W}_h^{+}$ is the updated solution. In order to improve the efficiency for the simulation of viscous flows, the viscous terms are only evaluated on the fine grid and not evaluated at the coarser grids. Since the coarser grids are only used to cancel the dominating low frequency errors, this treatment does not affect the accuracy of the solution. The upwind-biased interpolation scheme is set to first-order at the coarser levels where the left-right states of Equations (8a) and (8b) are

Copyright © 2004 John Wiley & Sons, Ltd.

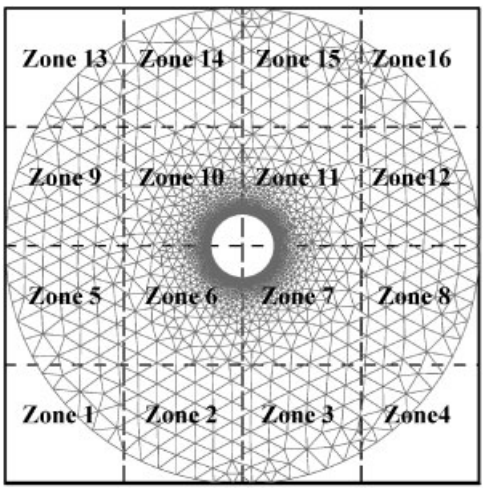

Figure 4. Zoning of flow field domain.

taken as the values of the two nodes of the edge to calculate the flux associated with the edge.

4.2.1. Zoning of flow field domain. In order to reduce pre-processing time, the flow field domain is decomposed into a number of square zones, in which searching for a particular node is only done within the related square zones, instead of searching the whole flow field. Figure 4 shows a flow field domain being decomposed into 4×4 zones. The pre-processing work in term of CPU time can be reduced according to the following equation with the assumption of having the same number of nodes in each zone:

$$\text{Total pre-processing work (CPU Time)} = \frac{\text{Number of nodes}}{\text{NZ}} \tag{26}$$

where NZ represents the number of zones created in the flow domain.

The idea of this algorithm is that if any node of a triangular cell is mapped onto a particular zone as shown in Figure 5(a), then this cell is considered mapped to this zone. This algorithm will yield a negative mapping when, for example, zone 11 of Figure 5(a) will not be considered to contain this cell, which is not true. Another negative mapping encountered by this algorithm is that the area of the zones may be smaller than the area of the largest element as shown in Figure 5(b). Both of the above-mentioned problems will directly affect the accuracy of the flow solution. In order to eliminate the negative mapping, the corner co-ordinates of the zone are also used to test whether a cell is mapped to the zone. The algorithm used to test for this mapping is depicted in Section 4.2.2. If the corner point of the zone is found to be within the cell, then this triangular cell is also considered to be mapped to this zone.

4.2.2. Inter-connectivity relationship between meshes. Before the flow field variables and residuals are transferred from the fine grid to the coarse grid or the corrections are interpolated

Copyright © 2004 John Wiley & Sons, Ltd.

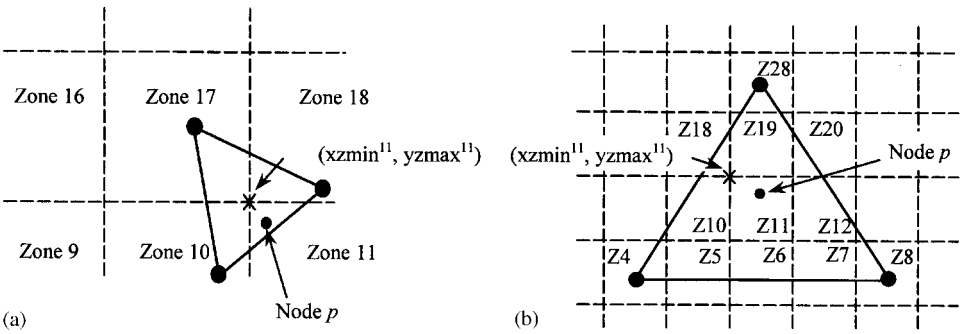

Figure 5. (a) Negative mapping; and (b) area of the zones smaller than the largest element.

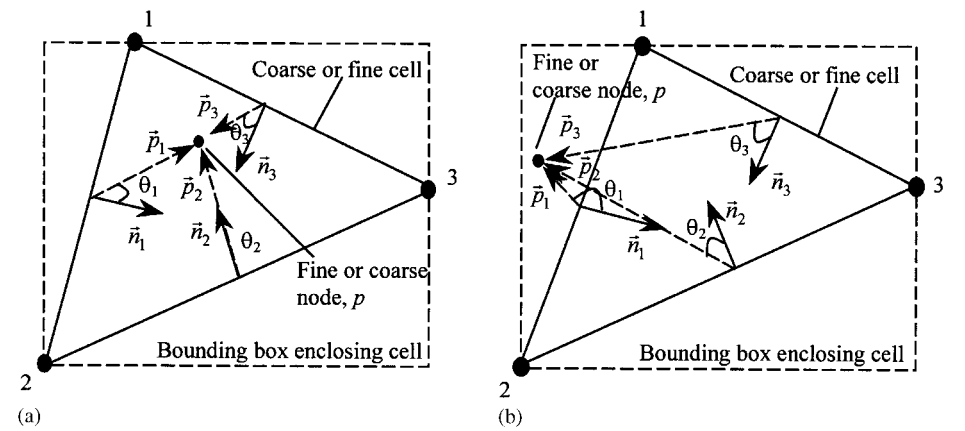

Figure 6. (a) Node falls within a cell using dot product; and (b) node does not fall within a cell.

from the coarse grid back to the fine grid, it is necessary to determine in which coarse cell each fine node is located and *vice versa.*

The algorithm for inter-connectivity relationship between meshes is based on the concept of dot product of two vectors: the unit normal vector oriented *inward* from an edge, $\boldsymbol{n}$, and the vector $\boldsymbol{p}$ which points from the centre of the edge to a node under consideration. The dot product of these two vectors, $(p_n)^{\text{nedg}}$, for the three edges is shown as follows:

$$(p_n)^{\text{nedg}} = \{\boldsymbol{p}\} \cdot \{\boldsymbol{n}\} = \{p_x \;\; p_y\} \cdot \begin{Bmatrix} n_x \\ n_y \end{Bmatrix} = (p_x n_x + p_y n_y)^{\text{nedg}}, \quad \text{nedg} = 1\text{–}3 \tag{27}$$

where the superscript nedg is the edge number of the cell.

With reference to Figure 6(a), the dot product of these two vectors must be positive for all the three edges if the node falls within the cell. If one of the dot product of these two vectors is less than zero, then the node being tested is not within the cell, as depicted in Figure 6(b).

Copyright © 2004 John Wiley & Sons, Ltd.

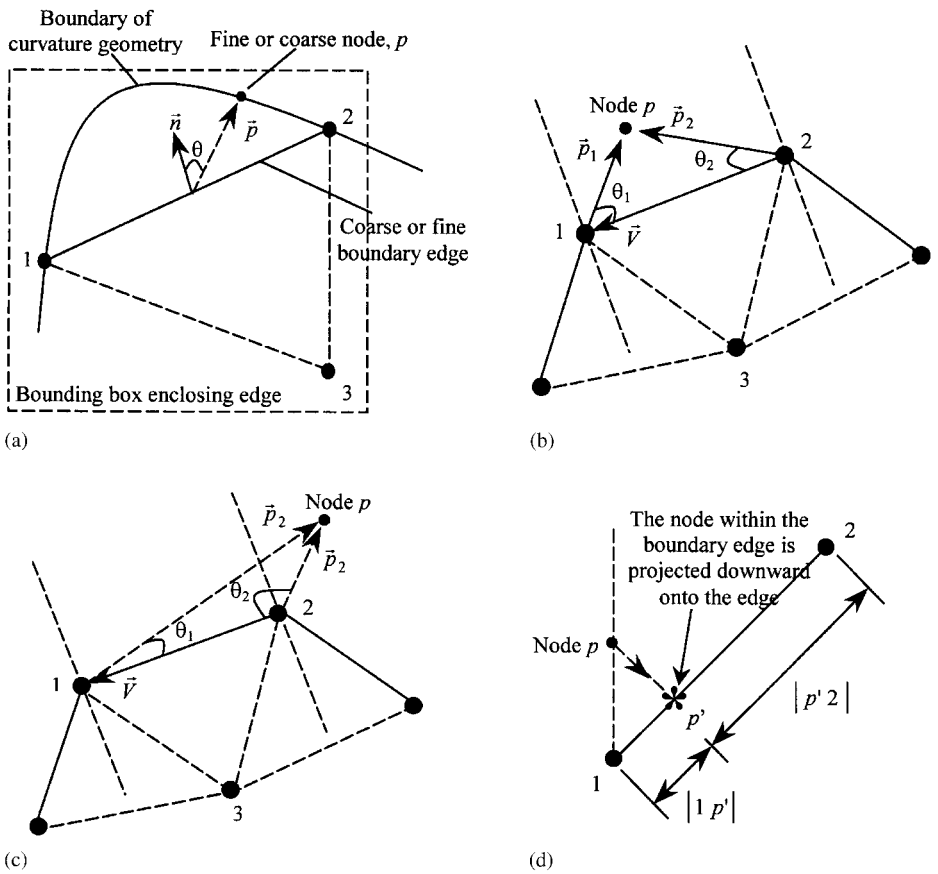

Figure 7. (a) Fine node within the coarse boundary edge; (b) different sign computed between $(p_v)^1$ and $(p_v)^2$; (c) node belonging to the neighbouring edge (i.e. $(p_v)^1 < 0$ and $(p_v)^2 < 0$); and (d) projected lengths for the transfer operator algorithms.

The following equation shows the criteria for the node to be within the cell:

$$\{(p_n)^1 \quad .\text{and.} \quad (p_n)^2 \quad .\text{and.} \quad (p_n)^3\} \geqslant 0 \tag{28}$$

In order to reduce the pre-processing CPU time further more, those nodes that fall within a particular cell are marked as 1 and they will not be tested for the next cell in the same or next zone.

Basically, the search for nodes within the boundary edges will continue after the search for nodes within the cells are performed if the user specifies that the boundary is curved. Two criteria are used to determine if a node is within the boundary edge. The first criterion is depicted in Figure 7(a) where $\boldsymbol{n}$ is the *outward* normal vector of the boundary edge and $\boldsymbol{p}$ is a vector pointing from the centre of the boundary edge to the node considered. If the product is greater than or equal to zero, then the node is considered to be within the edge, as shown

Copyright © 2004 John Wiley & Sons, Ltd.

in the following:

$$(p_n) = \{\boldsymbol{p}\} \cdot \{\boldsymbol{n}\} = \{p_x \quad p_y\} \cdot \begin{Bmatrix} n_x \\ n_y \end{Bmatrix} = (p_x n_x + p_y n_y) \geqslant 0 \tag{29}$$

In order to ensure that vector $\boldsymbol{p}$ is not pointing to a node that is very far away from the boundary edge, the boundary edge will be enclosed within a bounding box. The second criterion is depicted in Figure 7(b) where the dot product of both $\boldsymbol{p}_1 \cdot \boldsymbol{v}$ and $\boldsymbol{p}_2 \cdot \boldsymbol{v}$ must be different in sign. The purpose of this second criterion is to ensure that the nodes tested is within the boundary edge. Figure 7(c) shows that the sign for these two dot products are the same, i.e. both dot products are less than zero. If the node being tested fulfils these two criteria, then the node is projected downward onto the boundary edge and the projected lengths between this projected node and the two edge nodes are computed for the transfer operator algorithms, as illustrated in Figure 7(d).

4.2.3. Mesh-to-mesh transfer operators. Following the approach presented [12] for data transfer within the domain and incorporating the new technique developed here for curved boundary, there are two classes of mesh-to-mesh transfer operators being implemented in the present study, which are *restriction* operators and *prolongation* operators.

4.2.3.1. Flow-field-variable transfer operators. The *restriction* transfer operator, T_h^{h+1}, that transfers the flow field values from the fine grid to the coarse grid is given as follows:

$$\mathbf{W}_1 = \frac{\mathbf{A}_a \mathbf{W}_a + \mathbf{A}_b \mathbf{W}_b + \mathbf{A}_c \mathbf{W}_c}{\mathbf{A}_a + \mathbf{A}_b + \mathbf{A}_c} \tag{30}$$

Lower case letters denote fine grid nodes and Arabic numbers denote coarse grid nodes for all the figures in this section. The flow field value at the coarse node 1, which is contained in the fine cell formed by nodes a, b and c, is a weighted average of the values at those nodes

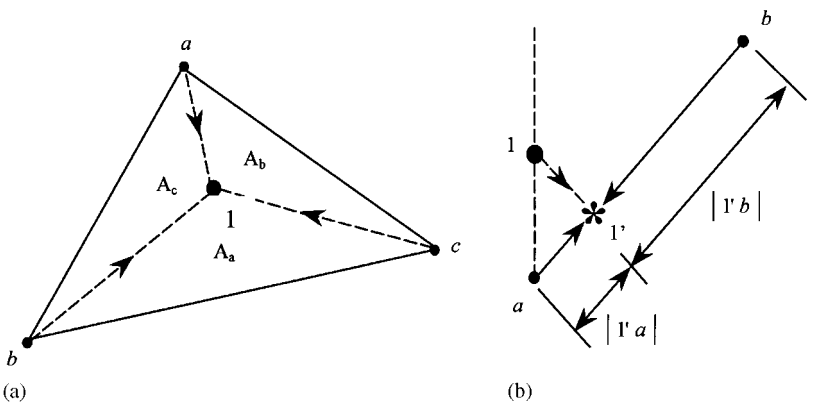

Figure 8. (a) Transfer of flow field values from the fine mesh to the coarse mesh; and (b) transfer of variables from fine nodes to the coarse node at the boundary.

Copyright © 2004 John Wiley & Sons, Ltd.

as shown in Figure 8(a). A_a,A_b, and A_c are the areas of the corresponding triangles opposite to the nodes.

On the curved boundary, the flow field values are transferred from the fine nodes of the edge formed by vertices a and b onto the coarse node $1'$. The *restriction* transfer operator, T_h^{h+1}, that transfers the flow field values from the fine grid to the coarse grid is based on the following formula:

$$\mathbf{W}_1 = \frac{|1'b|\mathbf{W}_a + |1'a|\mathbf{W}_b}{|ab|} \tag{31}$$

The flow field values at the coarse node $1'$, which is projected downward onto the fine edge enclosed by nodes a and b, is a weighted average of the values at those nodes as shown in Figure 8(b). $|1'a|$ and $|1'b|$ are the projected lengths between the fine nodes and the projected coarse node. $|ab|$ is the length of the fine edge.

4.2.3.2. Residual transfer operators. The transfer of residuals from the fine nodes to the coarse nodes is based on an area-weighted contribution calculation as depicted in Figure 9(a). The final transferred residual at a coarse grid node will be the summation of the contributions from all the fine nodes located within all the coarse cells surrounding this particular coarse grid node. For the three nodes of a coarse grid cell, the residual contributions they receive from a single fine grid node inside the cell are calculated as

$$R_1 = R_1 + \frac{A_1 R_a}{A_1 + A_2 + A_3}$$

$$R_2 = R_2 + \frac{A_2 R_a}{A_1 + A_2 + A_3}$$

and

$$R_3 = R_3 + \frac{A_3 R_a}{A_1 + A_2 + A_3} \tag{32}$$

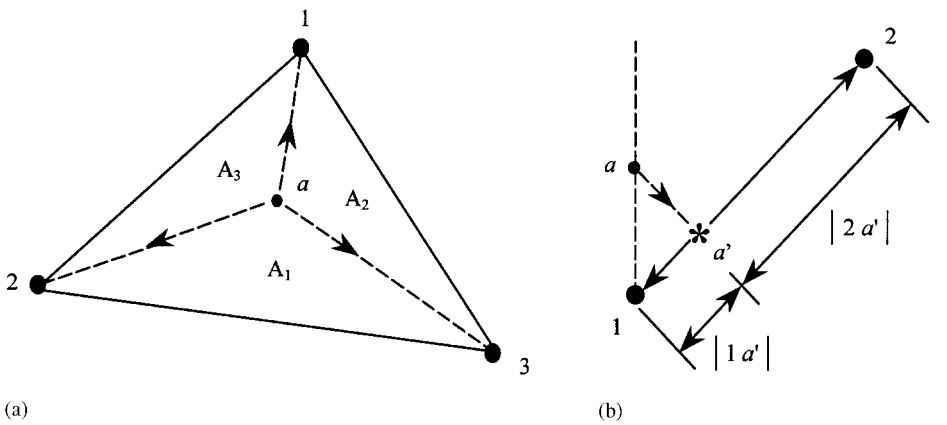

Figure 9. (a) Transfer of residuals from the fine mesh to the coarse mesh; and (b) transfer of residuals from the fine node to the coarse nodes at the boundary.

Copyright © 2004 John Wiley & Sons, Ltd.

where A_1, A_2 and A_3 are the areas of the corresponding triangles opposite to the nodes. It is easy to show that this transfer is conservative in the sense that the total fine mesh residuals are equal to the coarse mesh ones.

On a curved boundary the residual contributions from fine node a', which is projected onto a coarse edge enclosed by coarse nodes 1 and 2, are distributed to the coarse nodes 1 and 2 using a length-weighted contribution calculation. The transfer operator, Q_h^{h+1} is given as

$$\begin{aligned} R_1 &= R_1 + \frac{|2a'|R_{a'}}{|12|} \\ R_2 &= R_2 + \frac{|1a'|R_{a'}}{|12|} \end{aligned} \tag{33}$$

This is shown in Figure 9(b) for a fine node within a coarse boundary edge. It can be seen that the final residual at the coarse nodes 1 or 2 is actually the summation of all the contributions from those fine nodes projected onto the two edges that share the coarse node. This also ensures the residual transfer is conservative on the boundary.

4.2.3.3. Correction transfer operators. Prolongation operators are used to transfer corrections of the flow field variables from the coarse mesh to the fine mesh, which is illustrated in Figure 10. The corrections on a coarse mesh are calculated as follows:

$$\mathrm{d}\mathbf{W}_{h+1} = \mathbf{W}_{h+1}^{+} - \mathbf{W}_{h+1}^{(0)} \tag{34}$$

The corrections are then transferred to the fine mesh by the *prolongation* operator, I_{h+1}^{h}

$$v_h = I_{h+1}^{h}\mathrm{d}\mathbf{W}_{h+1}$$

According to Figure 10(a), the correction of the flow field variables transferred from coarse nodes 1, 2 and 3 to fine node a is a weighted average of the corrections at 1, 2 and 3 and

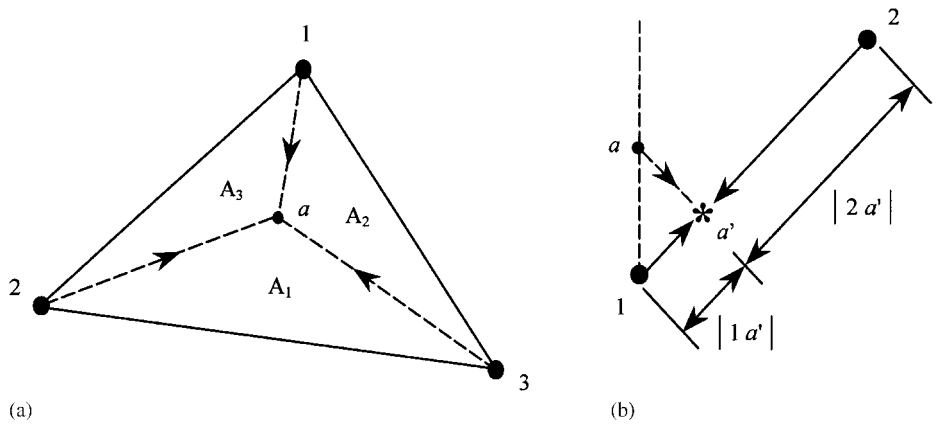

Figure 10. (a) Transfer of corrections from the coarse mesh to the fine mesh; and (b) transfer of corrections from the coarse nodes to the fine node at the boundary.

Copyright © 2004 John Wiley & Sons, Ltd.

the expression for the transferred correction is

$$(v_a)_h = \frac{A_1(\mathrm{d}\mathbf{W}_1)_{h+1} + A_2(\mathrm{d}\mathbf{W}_2)_{h+1} + A_3(\mathrm{d}\mathbf{W}_3)_{h+1}}{A_1 + A_2 + A_3} \tag{35}$$

where A_1, A_2 and A_3 are the areas of the corresponding triangles opposite to the nodes 1, 2 and 3, respectively.

The corrections of the flow field variables are transferred from the coarse nodes of the edge formed by vertices 1 and 2 to the fine node a' as shown in Figure 10(b). The transfer operator, I_{h+1}^h, that transfers the corrections from the coarse nodes to a fine one is derived as follows:

$$(v_{a'})_h = \frac{|2a'|(\mathrm{d}\mathbf{W}_1)_{h+1} + |1a'|(\mathrm{d}\mathbf{W}_2)_{h+1}}{|12|} \tag{36}$$

Thus the correction at the fine node a', which is projected downward onto the coarse edge formed by nodes 1 and 2, is also a weighted average of the values at the two coarse nodes. $|1a'|$ and $|2a'|$ are the projected lengths between the coarse nodes and the projected fine node. $|12|$ is the length of the coarse edge.

5. BOUNDARY AND INITIAL CONDITIONS

At the solid wall, a *no-slip* condition is imposed for viscous flow by setting the flow velocity equal to that of the body. A uniform velocity profile is given as the free stream boundary condition and the pressure at the free stream is calculated while pressure at the down stream boundary is fixed at a constant value and the velocity at the down stream boundary is calculated. The flow field values are set to the free-stream values at the start of the computation.

6. COMPUTATIONAL RESULTS

In this section, the validation of the incompressible multigrid flow solver is described and its performance evaluated. The test case is viscous flow over a circular cylinder at different Reynolds numbers.

6.1. Low-Reynolds-number steady flow

The Reynolds number specified in the computations was 41.0, the pseudo-compressibility coefficient β was set to 1.0 for fast convergence and the physical time step was set to a large value. The third-order characteristics-based scheme was used in both single grid and multigrid computations. A three-level multigrid was used to compute the flow. There are 12 556 nodes and 24 763 elements in the fine grid, 1571 nodes and 3009 elements in the coarse grid, and 212 nodes and 404 elements in the coarsest grid, all of which are shown in Figure 11.

The results obtained from the multigrid solver for both V- and W-cycle are used to make qualitative comparisons with experimental results and that of single-grid computation. Figure 12 shows the streamline plots for flow over a circular cylinder obtained from experimental measurement, single grid, multigrid V- and W-cycle computations.

Copyright © 2004 John Wiley & Sons, Ltd.

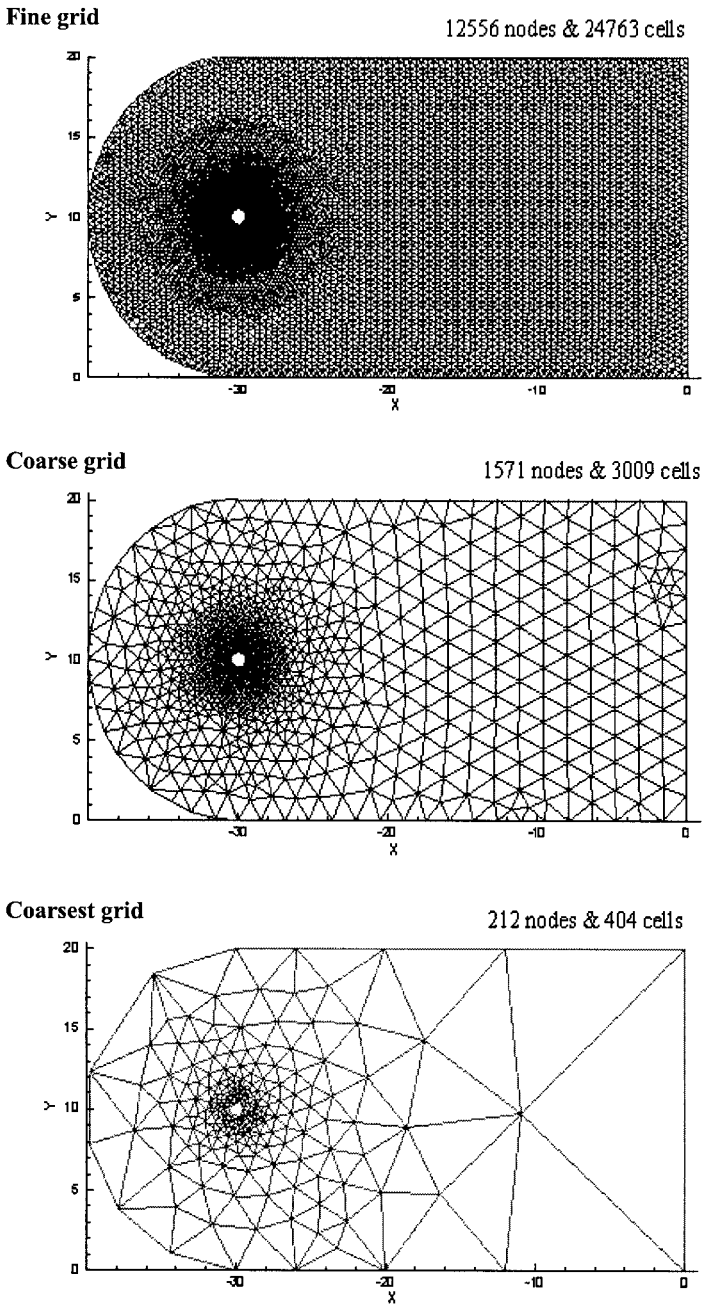

Figure 11. The sequence of grids used in the three-level multigrid computations.

Copyright © 2004 John Wiley & Sons, Ltd.

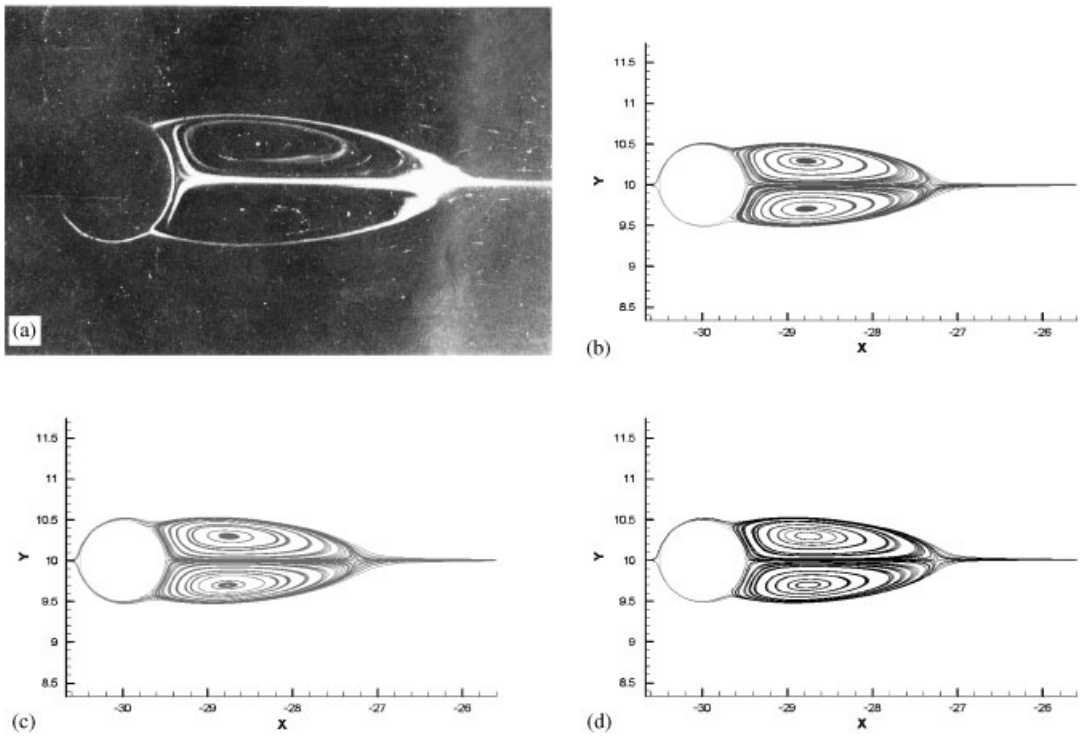

Figure 12. Streamlines plot for flow over a circular cylinder for: (a) Experimental measurements [22]; numerical computation for (b) single gird; (c) MG, V-cycle; and (d) MG, W-cycle ($Re = 41.0$).

Based on visual comparison with the experimental and single-grid results, the results obtained for both multigrid strategies show a similar trend, where the separation point occurs at the same location, the separation bubble is of the same size and the reattachment point is at the same location. Since the main purpose of the multigrid method is to accelerate the convergence rate, a more realistic way of comparing the results obtained is to make a quantitative comparison between the single-grid and the multigrid computations in term of the order of magnitude reduction in residuals vs number of time steps and CPU time. This comparison is given in Figure 13.

The steady flow calculations with both the single-grid and multigrid were performed on a DEC AlphaServer 8400 5/440 machine. Figure 13(a) indicates that it takes more time steps for the single-grid computation to reduce the residuals to 10^{-4} as compared with three-level multigrid (MG). The MG solver shows a reduction of 83% in time steps required for MG compared with the single grid. Figure 13(b) shows a significant saving in the CPU time using the MG method. From the figure, it can be seen that it takes about 200 min to compute the same problem using MG method, as compared to the single grid, which takes about 800 min. The multigrid solver shows a reduction of 75% in CPU time when the three-level multigrid V-cycle is used while the three-level W-cycle leads to a 88% reduction in CPU time, compared with the single grid computation. The CPU time of the W-cycle is found to be less than that of the V-cycle on three levels of grids since the number of time steps for W-cycle is less

Copyright © 2004 John Wiley & Sons, Ltd.

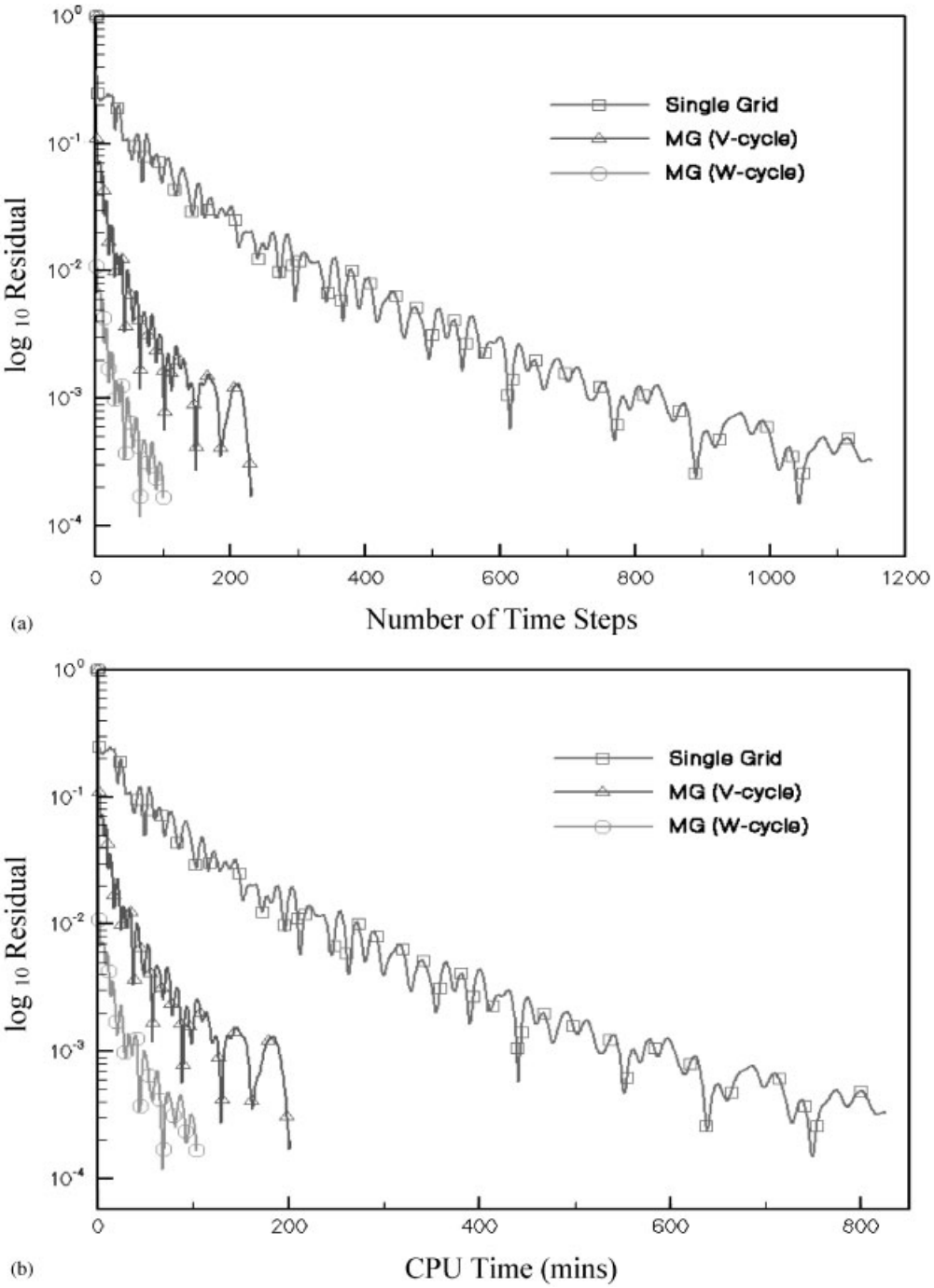

Figure 13. Convergence history plot for: (a) Residual vs physical time steps; and (b) residual vs CPU time.

Copyright © 2004 John Wiley & Sons, Ltd.

than that of the V-cycle, which indicates that the W-cycle is more efficient than the V-cycle for this particular case.

From the results obtained, it can be concluded that the multigrid method is a faster and more efficient technique than the baseline single-grid method for low-Reynolds-number steady flows. The W-cycle requires approximately 90% more CPU time than a single-grid cycle, while the V-cycle requires 75% more CPU time. However, both multigrid strategies provide close to an order of magnitude increase in convergence rate, thus greatly outweighing their increased cost per cycle.

6.2. *High-Reynolds-number unsteady flow*

The test case considered for high-Reynolds-number unsteady flow is the viscous flow over a circular cylinder at $Re = 200$, which is one of the most thoroughly investigated unsteady flow. The parameters and schemes for both the single-grid and multigrid in the computations were the same as before, except that the number of pseudo sub-iterations and number of multigrid cycles per step were set differently. The non-dimensional physical time step was set to be 0.09 for better temporal resolution. The third-order characteristics-based scheme was also used in both single-grid and multigrid computations together with the dual-time stepping scheme and the second-order temporal discretization. The pseudo sub-iterations per time step were set to 200 for single grid, 30 V-cycles per time step and 15 W-cycles per time step. A three-level multigrid was used to compute the flow. There are 24 226 nodes and 48 103 elements in fine grid, 8646 nodes and 17 102 elements in the coarse grid, and 3139 nodes and 6139 elements in the coarsest grid. Especially in the fine grid the wake region is further refined in order to accurately capture the fine details of the vortex shedding phenomenon (see Figure 14).

The high-Reynolds-number-flow calculations using both single grid and multigrid were performed on a SGI OCTANE workstation. The flow was started from stationary conditions and the simulation was run until periodic shedding of vortices occurred. Figure 15 shows the computed lift and drag coefficients on the cylinder vs non-dimensional time with both single grid and multigrid, respectively. A pronounced asymmetric wake began to appear at

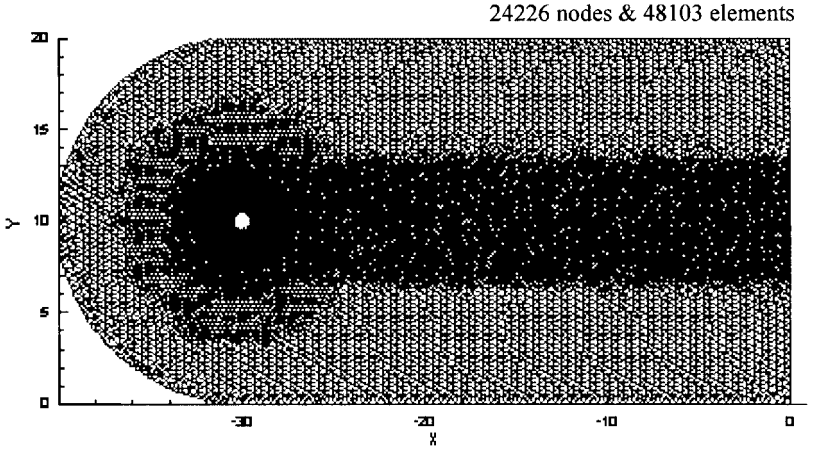

Figure 14. Fine mesh with grid refinement in the wake region for unsteady flow calculation.

Copyright © 2004 John Wiley & Sons, Ltd.

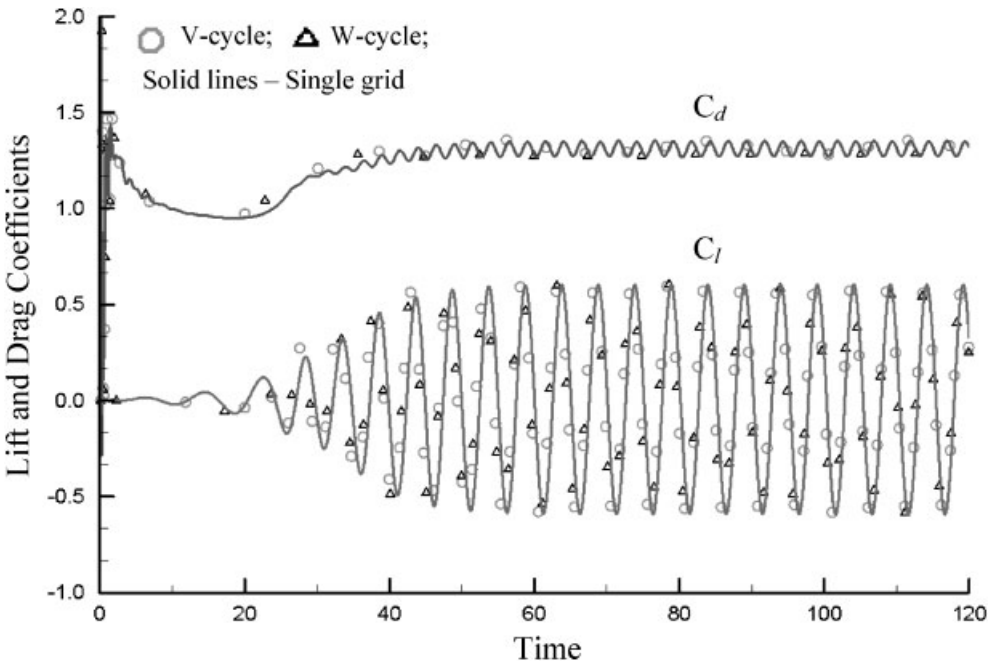

Figure 15. Lift and drag coefficients vs time for flow over a circular cylinder, $Re = 200$.

non-dimensional time of 20 for both single grid and multigrid. The flow became completely periodic at a time of 55 for both methods. Although both methods took the same time to obtain fully periodic flow, the number of sub-iterations or multigrid cycles is much less than that of the single grid. This signifies that the multigrid method takes shorter time than the single grid to produce the vortex shedding phenomenon, thus lesser CPU time is needed for the flow to become fully periodic. The period of the flow as shown in the figure is 5.13.

Figure 16 illustrates a comparison between the CPU time taken for the same non-dimensional time for both the single grid and multigrid methods. From the figure, it is found that 30 V-cycles and 15 W-cycles took 55 and 53%, respectively, lesser CPU time than the single grid to compute the same flow problem. Comparing the CPU time taken for both V- and W-cycle, it is observed that both of them almost took the same CPU time to reach the periodic/cyclic stage, even though W-cycle uses lesser number of cycles per time step. This is mainly due to the additional operations on the coarser grid for W-cycle.

Figure 17 shows the contours of vorticity obtained by the multigrid method, which basically outlines the von Kármán vortex street phenomenon. The figure shows that the vortices with opposite signs are shed from upper and lower surfaces alternatively and, thus forming the von Kármán vortex street phenomenon.

Table I tabulates the results for $Re = 200$ from both numerical studies and experimental measurement. In this table, C_l is the lift coefficient, C_d is the drag coefficient, and S_t the Strouhal number. The computed coefficients and Strouhal number from the multigrid solver are found to produce excellent agreement with those by the single-grid method. In comparison with other researchers' results, the value of C_l for the multigrid method obtained produces

Copyright © 2004 John Wiley & Sons, Ltd.

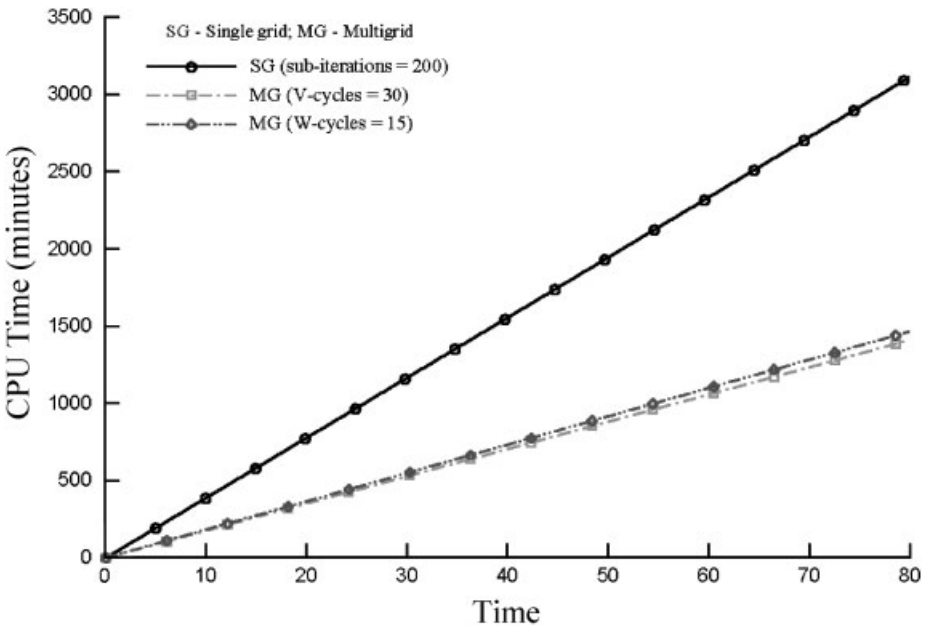

Figure 16. Comparison of CPU time between the single grid and multigrid method.

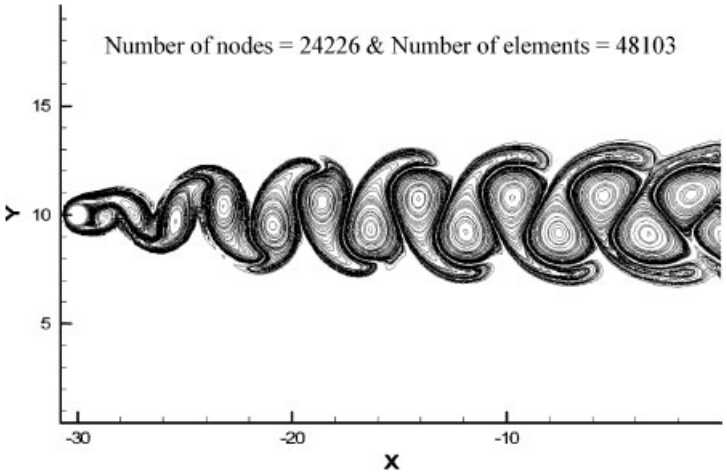

Figure 17. Vorticity contours plot for multigrid, $Re = 200$.

quite a good agreement with the results by Chan *et al.* [14] and Belov *et al.* [10], but deviates slightly from that of Liu *et al.* [11]. The average value of 1.31 for C_d obtained is in good agreement with that in Reference [11] and the experimental results obtained by Wille [23], but the amplitude deviates slightly by 8×10^{-3}. The Strouhal number obtained has excellent agreement with the experimental data obtained by Kovaznay [24] and Roshko [25], and the numerical results by most of the researchers.

Copyright © 2004 John Wiley & Sons, Ltd.

Table I. Comparison of results for unsteady flow over a cylinder, $Re = 200$.

Reference	C_l	C_d	S_t
Present (multigrid)	± 0.64	1.31 ± 0.041	0.195
Present (single-grid)	± 0.64	1.31 ± 0.041	0.195
Liu *et al.* [11]	± 0.69	1.31 ± 0.049	0.192
Chan *et al.* [14]	± 0.63	1.48 ± 0.05	0.183
Belov *et al.* [10]	± 0.64	1.19 ± 0.042	0.193
Rogers *et al.* [13]	± 0.65	1.23 ± 0.05	0.185
Miyake *et al.* [10]	± 0.67	1.34 ± 0.043	0.196
Rosenfeld *et al.* [27]	± 0.69	1.46 ± 0.05	0.211
Lecointe *et al.* [26]	± 0.50	1.58 ± 0.0035	0.194
Kovaznay (expt.) [24]	—	—	0.19
Roshko (expt.) [25]	—	—	0.19
Wille (expt.) [23]	—	1.30	—

6.3. *Effect of sub-iterations and multigrid cycles*

Three different numbers of sub-iterations per time step were used, i.e. 100, 200 and 300, for single-grid computation in order to determine the optimum number of sub-iterations necessary to obtain accurate results. It can be seen from Figure 18(a) that the results obtained with 100 sub-iterations show significant deviations from those obtained using 200 and 300 sub-iterations, while the curves for 200 and 300 sub-iterations coincide with each other. Thus, the optimum number of sub-iterations to achieve accurate solution for this flow problem was considered to be 200.

The performance of the multigrid code strongly depends on the number of multigrid cycles necessary to achieve steady state in pseudo-time. Since one pseudo sub-iteration is equivalent to one multigrid cycle on single grids, computations were performed on the multigrid using 10, 20, 30 and 100 V-cycles per time step, and 10, 15 and 20 W-cycles per time step, in order to determine the optimum number of cycles necessary to obtain good results. Figures 18(b) and (c) show the. results of different multigrid cycles per time step for V- and W-cycle, respectively. It was found that the optimum number of multigrid cycles to achieve the same solution as the single grid was 30 multigrid V-cycles and 15 multigrid W-cycles. Thus, the optimum number of multigrid cycles to achieve accurate solution for this flow problem was 30 V-cycles and 15 W-cycles per time step.

7. CONCLUSIONS

An unstructured non-nested multigrid method has been successfully developed and implemented into an existing finite volume Navier–Stokes solver using higher-order characteristics-based upwind scheme and dual time stepping for the study of unsteady incompressible flows. The computational results obtained agree well with numerical solutions obtained by other researchers as well as experimental measurements. The multigrid method with three levels of grids results in a 75% reduction in CPU time for the steady flow and 55% reduction for the unsteady flow compared with the single-grid method. The co-relation between the CPU time required and the non-dimensional time to simulate was obtained. The V-cycle is found to be more efficient than the W-cycle in multigrid computation of steady, but their efficiency seems

Copyright © 2004 John Wiley & Sons, Ltd.

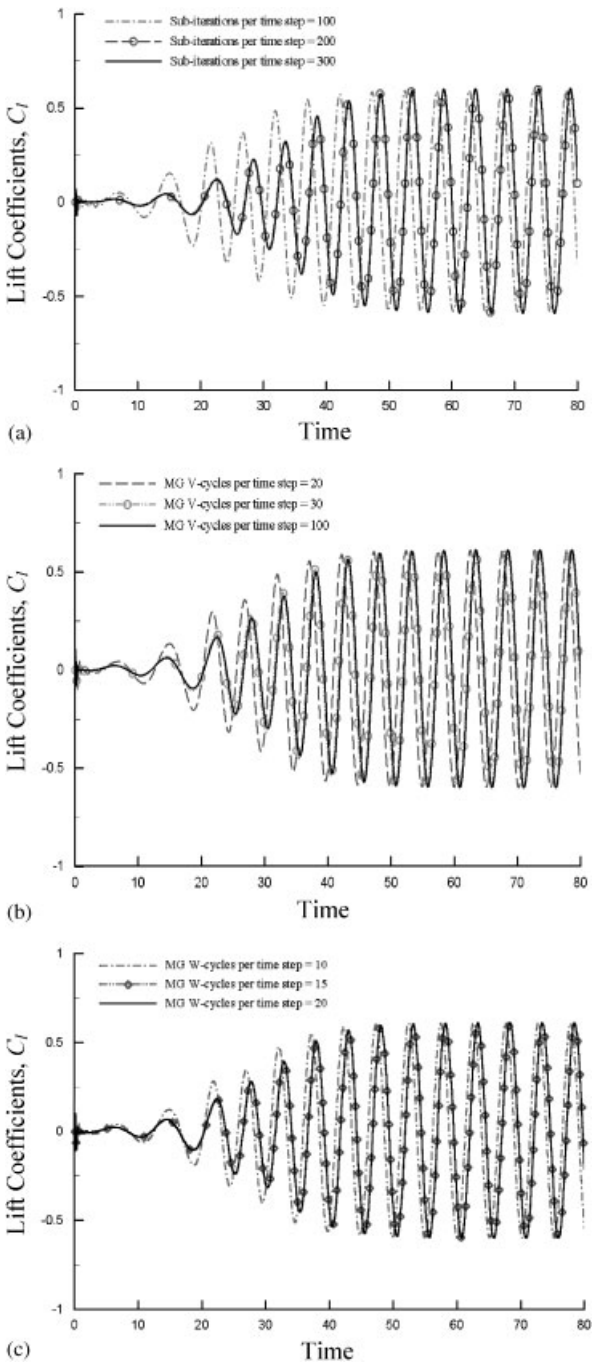

Figure 18. Effect of varying the number of sub-iterations per time step: C_l vs time with: (a) Single-grid; (b) MG V-cycle; and (c) MG W-cycle.

Copyright © 2004 John Wiley & Sons, Ltd.

to be almost the same in unsteady flow simulation if optimum numbers of sub-iterations per time step are found. The effects of sub-iteration on numerical solutions were studied and optimum numbers of sub-iterations for different multigrid as well single-grid iterations were obtained.

NOMENCLATURE

C_l	lift coefficient
C_d	drag coefficient
CFL	Courant–Friedrichs–Lewy number
$\boldsymbol{F}_\mathbf{c}$	convective flux vectors
$\boldsymbol{F}_\mathbf{v}$	viscous flux vectors
I_{h+1}^{h}	*prolongation* operator from the coarse grid $h+1$ to the fine grid h
MG	multigrid
$\boldsymbol{n}$	unit normal vector
$\mathbf{P}_{h+1}$	forcing function in the time stepping scheme for the multigrid method
Q_h^{h+1}	residual transfer operator from fine grid h to the coarse grid $h+1$
Re	Reynolds number
$R(\mathbf{W}_\mathrm{p})$	residual
$\tilde{R}(\mathbf{w}_\mathrm{p}^{n+1})$	modified residual in implicit time stepping
S_t	Strouhal number
t	time
Δt	real time step
$\Delta\tau$	pseudo-time step
T_h^{h+1}	solution transfer operator from fine grid h to the coarse grid $h+1$
$\mathbf{W}$	vector of conserved flow variables
$\mathbf{W}_h$	solution from the fine grid
$\mathbf{W}_h^+$	updated variables of the solution on the fine grid h
$\mathbf{W}_{h+1}^{(0)}$	initial values transferred from the fine grid
$\mathbf{W}_{h+1}^+$	newly computed value on the coarse grid $h+1$
$\mathrm{d}\mathbf{W}_{h+1}$	correction from coarse grid $h+1$

Greek symbols

τ	pseudo-time
α	stage coefficients for Runge–Kutta time integration
ε	smoothing coefficient
β	artificial compressibility parameter
κ	weight used in MUSCL interpolation

ACKNOWLEDGEMENTS

This research work was supported by a research scholarship provided by Nanyang Technological University (NTU) and Agency for Science, Technology and Research (A*STAR), Singapore. The provision of computing facilities by Nanyang Centre for Supercomputing and Visualisation (NCSV), School of MPE, NTU is acknowledged.

Copyright © 2004 John Wiley & Sons, Ltd.

REFERENCES

1. Lallemand M, Steve H, Dervieux A. Unstructured multigridding by volume agglomeration. *Computers and Fluids* 1992; **21**:397–433.
2. Mavriplis D. Three-dimensional multigrid for the Euler equations. *AIAA Journal* 1992; **30**:1753–1761.
3. Mavriplis D, Venkatakrishnan V. Agglomeration multigrid for two dimensional viscous flows. *Computers and Fluids* 1995; **24**:553–570.
4. Peraire J, Morgan K, Peiro J. Multigrid solution of the 3D compressible Euler equations on unstructured tetrahedral grids. *International Journal for Numerical Methods in Engineering* 1993; **36**:1029–1044.
5. Farmer J, Martinelli L, Jameson A. Fast multigrid method for solving incompressible hydrodynamic problems with free surface. *AIAA Journal* 1994; **32**:1175–1182.
6. Rizzi A, Erikson L. Computation of inviscid incompressible flow with rotation. *Journal of Fluid Mechanics* 1985; **153**:275–312.
7. Peraire J, Morgan K, Peiro J. The simulation of 3D incompressible flows using unstructured grids. In *Frontier of Computational Fluid Dynamics*, Caughey DA, Hafez MM (eds). John Wiley & Sons: New York, 1994; 281–301.
8. Anderson WK, Rausch RD, Bonhaus DL. Implicit/Multigrid algorithm for incompressible turbulent flows on unstructured grids. *Journal of Computational Physics* 1996; **128**:391–408.
9. Chorin A. A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics* 1967; **2**:12–26.
10. Belov A, Martinelli L, Jameson A. A new implicit algorithm with multigrid for unsteady incompressible flow calculations. *AIAA* 95-0049, January 9–12, 1995.
11. Liu C, Zheng X, Sung CH. Preconditioned multigrid methods for unsteady incompressible flows. *Journal of Computational Physics* 1998; **139**:35–57.
12. Lin PT. Implicit time dependent calculations for compressible and incompressible flows on unstructured meshes. *M.Sc. Thesis*, Department of Mechanical and Aerospace Engineering, Princeton University, 1994.
13. Rogers SE, Kwak D. Upwind differencing scheme for the time-accurate incompressible Navier–Stokes equations. *AIAA Journal* 1990; **28**(2):253.
14. Chan CT, Anastasiou K. Solution of incompressible flows with or without a free surface using the finite volume method on unstructured triangular meshes. *International Journal for Numerical Methods in Fluids* 1995; **29**:35–37.
15. Zhao Y, Zhang BL. A high-order characteristics upwind FV method for incompressible flow and heat transfer simulation on unstructured grids. *Computer Methods in Applied Mechanics and Engineering* 2001; **25**(6):523–536.
16. Eberle A. Three-dimensional Euler calculations using characteristics flux extrapolation. *AIAA Paper* 85-0119, 1985.
17. Drikakis D, Govatsos PA, Papantonis DE. A characteristic-based method for incompressible flows. *International Journal for Numerical Methods in Fluids* 1994; **11**:953–976.
18. Fedorenko RP. The speed of convergence of one iterative process. *U.S.S.R. Computational Mathematics and Mathematical Physics* 1964; **4**(3):227–235.
19. Brandt A. Guide to multigrid development. In *Multigrid Methods*. Hackbusch W, Trottenberg U (eds), Lecture Notes in Mathematics, vol. 960. Springer-Verlag: Berlin, 1982.
20. Jameson A. Solution of the Euler equations for two dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation* 1983; **13**:327–335.
21. Jameson A. Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA Paper* 91-1596, June 1991.
22. Van Dyke, Milton D. *An Album of Fluid Motion*. Parabolic Press: Stanford, California, USA, 1982.
23. Wille R. Karman vortex streets. *Advances in Applied Mechanics*, vol. 6. Academic Press: New York, 1960, 273–287.
24. Kovasznay LSG. Hot-wire investigation of the wake behind cylinders at low Reynolds numbers. *Proceedings of The Royal Society of London*, *Series A*, vol. 198, No. 1053, August 1949; 174–190.
25. Roshko A. On the development of turbulent wakes from vortex streets. *NASA Report* 1191, 1954.
26. Lecointe Y, Piquet J. On the use of several compact methods for the study of unsteady incompressible viscous flow round a circular cylinder. *Computers and Fluids* 1984; **12**(4):255–280.
27. Rosenfeld M, Kwak D, Vinokur M. A solution method for the unsteady and incompressible Navier–Stokes equations in generalized coordinate systems. *AIAA Paper* 88-0718, 1988.

Copyright © 2004 John Wiley & Sons, Ltd.