INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2003; **41**:931–949 (DOI: 10.1002/fld.474)

Convergence of steady and unsteady formulations for inviscid hovering rotor solutions

C. B. Allen*,†

Department of Aerospace Engineering, University of Bristol, Bristol, BS8 1TR, U.K.

SUMMARY

An upwind Euler solver is presented, and applied to multibladed lifting hovering rotor flow. These flows can be simulated as a steady case, in a blade-fixed rotating co-ordinate system. However, forward flight simulation will always require an unsteady solution. Hence, as a stepping stone in the development of a forward flight simulation tool, both explicit steady and implicit unsteady simulations of the same hovering case are presented. Convergence of the two approaches is examined and compared, in terms of residual history, cost, and solution evolution, as a means of both validating the unsteady formulation and considering implications for forward flight simulation. Consideration of the solution evolution and wake capturing shows that for hovering rotor cases, the unsteady and steady solutions are the same, but the unsteady solution is more expensive in terms of CPU time. It is also shown that for hover, the fewer real time-steps taken per revolution the more efficient the implicit scheme is. However, this is a characteristic of the case, which results in smooth solution variation between time steps. It is also demonstrated that for rotary flow simulation, the global residual is not a useful quantity to assess convergence. The residual reaches a very low (constant in the implicit case) value while the solution is still evolving. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: computational fluid dynamics; unsteady flows; euler equations; structured grid generation; rotor flows

1. INTRODUCTION

One of the most demanding problems for numerical methods is the numerical simulation of the flow about a helicopter rotor. The complexity of the flow—non-linear, highly three-dimensional, and unsteady—is increased due to each blade moving into a fluid which has already been disturbed by a previous blade. However, the accurate capture of the vortical wake is vital, as the loading on a blade is affected significantly by this wake, and particularly the tip vortex, shed from the previous one. Hence, not only is the flow-field more complex than a fixed-wing case, but must be captured over a larger distance away from the solid surface.

* Correspondence to: C.B. Allen, Department of Aerospace Engineering, University of Bristol, Queen's Building, University Walk, Bristol BS8 1TR, U.K.
† E-mail: c.b.allen@bristol.ac.uk

Received February 2002
Revised 26 August 2002

This leads to different requirements of the numerical mesh for a rotary-wing flow than a fixed-wing flow, since the vortical wake capture requires a much higher grid density away from the surface than for a fixed wing case, where far-field mesh and solution are normally of little significance. Hence, rotary-wing flow computations require finer meshes than fixed-wing flows, and a large numerical integration time for the wake to develop over several turns, and this leads to impractical run times. These run-times can be reduced by parallelizing the code [1] or, if a parallel platform is not available, a multigrid approach [2]. However, it has been shown [2, 3] that multigrid acceleration is not as effective for hovering rotor flows as it is for fixed-wing flows. One reason for this is the very slow development of the vortical wake. It was also shown in Reference [2] that the residual level does not give an accurate representation of a rotor flow convergence. The residual tends to level off, even though the solution has not converged. This paper explores this further, by considering the effects of the time-stepping scheme on the solution evolution, convergence and wake capturing, during steady and unsteady simulation of lifting hovering rotor flow.

Hovering flight can be simulated as a steady problem, in a blade-fixed rotating co-ordinate system. However, the next stage of this research is to simulate forward flight, and this will always require a full unsteady simulation. Hence, steady explicit and unsteady implicit methods have been developed and applied to lifting hovering flows, as a stepping stone in the development of a forward flight simulation tool. The convergence of the two approaches is examined and compared, in terms of residual history, cost, solution evolution and wake capturing, as a means of both validating the unsteady formulation and considering implications for forward flight simulation.

Numerical solutions for hovering rotors have been considered previously, for inviscid flows, see for example References [1, 4–14], and for viscous flows, see for example References [15–17]. It has been demonstrated previously [9] that the numerical dissipation used by central-difference schemes has a significant effect on the solution, in terms of wake capturing and hence convergence rate. An upwind Euler solver is used in the current work, since this accurately models the physics of the flow, in terms of characteristic behaviour, and so is naturally dissipative. A finite-volume solver is presented, based on the flux-vector splitting of Van-Leer [18].

Structured grids are used, in conjunction with a periodic transformation [19], for rotary-wing grid generation for hovering flight.

In this paper the steady and unsteady forms of the flow-solver are presented, followed by grid generation aspects of single block grids. Then solution evolution, convergence, and wake capturing are considered for computations of lifting hovering multi-bladed rotor flows, using steady and unsteady computations. Finally, the implications of the presented results are discussed for forward flight simulations.

2. EULER SOLVER

The computation of flows about a rotor blade in hover requires a rigid but rotating grid. However, there are then two options: the flow can be computed as a time-dependent flow using an unsteady solver, or the equations can be transformed to a blade-fixed rotating reference frame. In this frame the hover case is then a steady problem, and a steady solver can be used.

2.1. Steady approach

The equations may be transformed in terms of absolute or relative velocities. Absolute velocities are used here. If the frame rotates with angular velocity $\boldsymbol{\omega} = [\Omega_x, \Omega_y, \Omega_z]^{\mathrm{T}}$, and the absolute velocity vector in the rotating frame is denoted by $\mathbf{q}_{\mathrm{r}} = [u_{\mathrm{r}}, v_{\mathrm{r}}, w_{\mathrm{r}}]^{\mathrm{T}}$, the resulting Euler equations in integral form are then

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_{\mathrm{r}}} \mathbf{U}_{\mathrm{r}}\,\mathrm{d}V_{\mathrm{r}} + \int_{\partial V_{\mathrm{r}}} \mathbf{F}_{\mathrm{r}}.\mathbf{n}_{\mathrm{r}}\,\mathrm{d}S_{\mathrm{r}} + \int_{V_{\mathrm{r}}} \mathbf{G}_{\mathrm{r}}\,\mathrm{d}V_{\mathrm{r}} = 0 \tag{1}$$

where

$$\mathbf{U}_{\mathrm{r}} = \begin{bmatrix} \rho \\ \rho u_{\mathrm{r}} \\ \rho v_{\mathrm{r}} \\ \rho w_{\mathrm{r}} \\ E \end{bmatrix}, \quad \mathbf{F}_{\mathrm{r}} = \begin{bmatrix} \rho[\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})] \\ \rho u_{\mathrm{r}}[\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})] + P\mathbf{i}_{\mathbf{r}} \\ \rho v_{\mathrm{r}}[\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})] + P\mathbf{j}_{\mathrm{r}} \\ \rho w_{\mathrm{r}}[\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})] + P\mathbf{k}_{\mathrm{r}} \\ E[\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})] + P\mathbf{q}_{\mathrm{r}} \end{bmatrix}, \quad \mathbf{G}_{\mathrm{r}} = \begin{bmatrix} 0 \\ \rho(\boldsymbol{\omega} \times \mathbf{q}_{\mathrm{r}}).\mathbf{i}_{\mathbf{r}} \\ \rho(\boldsymbol{\omega} \times \mathbf{q}_{\mathrm{r}}).\mathbf{j}_{\mathrm{r}} \\ \rho(\boldsymbol{\omega} \times \mathbf{q}_{\mathrm{r}}).\mathbf{k}_{\mathrm{r}} \\ 0 \end{bmatrix} \tag{2}$$

Here $\mathbf{G}_{\mathrm{r}}$ is the source term resulting from the transformation, and $\mathbf{r} = [x_{\mathrm{r}}, y_{\mathrm{r}}, z_{\mathrm{r}}]^{\mathrm{T}}$ is the co-ordinate vector. This is a steady case, as the grid speeds are constant, i.e. the co-ordinate vector is constant in the rotating frame. The equation set is closed by

$$P = (\gamma - 1)\left[E - \frac{\rho}{2}\mathbf{q}^2\right] \tag{3}$$

2.2. Upwind difference scheme

A finite-volume upwind scheme is used to solve the integral form of the Euler equations (Equation (1)), since by correctly modelling the characteristic behaviour of the flow upwind schemes are naturally dissipative. The flux-vector splitting of Van-Leer [18, 20] is used.

For each cell face a local orthogonal co-ordinate system (ξ, η, ζ) is adopted, where the principal co-ordinate direction ξ is normal to the cell face. The unit normal to each cell face is defined as the unit vector in the ξ direction, $\mathbf{n}_{\xi}$. Unit vectors in two directions, lying in the cell face, $\mathbf{n}_{\eta}$ and $\mathbf{n}_{\zeta}$, are then defined to form an orthogonal axis system.

To compute the flux in the principal direction, i.e. the total flux across the face, the cartesian velocity components in the local cell face axis system are required,

$$\bar{u} = \mathbf{q}_{\mathrm{r}}.\mathbf{n}_{\xi} \tag{4}$$

$$\bar{v} = \mathbf{q}_{\mathrm{r}}.\mathbf{n}_{\eta} \tag{5}$$

$$\bar{w} = \mathbf{q}_{\mathrm{r}}.\mathbf{n}_{\zeta} \tag{6}$$

and the contravariant velocity normal to the face

$$\bar{U} = [\mathbf{q}_{\mathrm{r}} - (\boldsymbol{\omega} \times \mathbf{r})].\mathbf{n}_{\xi} \tag{7}$$

The general flux function in the principal direction, $\bar{\mathbf{F}}$, is then

$$\bar{\mathbf{F}} = \begin{Bmatrix} \rho\bar{U} \\ \rho\bar{U}\bar{u} + P \\ \rho\bar{U}\bar{v} \\ \rho\bar{U}\bar{w} \\ E\bar{U} + P\bar{u} \end{Bmatrix} \tag{8}$$

and the total flux across the face is simply $\bar{\mathbf{F}}A$, where A is the cell face area.

However, since quantities are evaluated at cell face centres, care must be taken when evaluating the rotational terms. To avoid introducing error the condition

$$\sum_{k=1}^{6} (\boldsymbol{\omega} \times \mathbf{r}).\mathbf{n}_{\xi} A_k = 0 \tag{9}$$

where k represents the six cell faces, must be satisfied. To satisfy this condition the area moment vector is defined for each cell face

$$\mathbf{M} = \int_{\partial V_r} \mathbf{r} \times \mathbf{n}\, \mathrm{d}S_r \tag{10}$$

and is evaluated as in Reference [21]. If this vector is normalized by cell face area

$$\bar{\mathbf{M}} = \frac{\mathbf{M}}{A} \tag{11}$$

the contravariant velocity can be expressed as

$$\bar{U} = \mathbf{q}_r.\mathbf{n}_{\xi} - \boldsymbol{\omega}.\bar{\mathbf{M}} \tag{12}$$

The general flux vector is split into a forward part, $\bar{\mathbf{F}}^{+}$, associated with positive moving waves only, i.e. all eigenvalues of $\partial\bar{\mathbf{F}}^{+}/\partial\mathbf{U} \geqslant 0$, and a backward part, $\bar{\mathbf{F}}^{-}$, associated with negative moving waves only, all eigenvalues of $\partial\bar{\mathbf{F}}^{-}/\partial\mathbf{U} \leqslant 0$. At each cell face a pair of states are thus defined and a single numerical flux derived from this pair. The split flux components are,

$$\bar{\mathbf{F}}^{\pm} = \begin{Bmatrix} f^{\pm}_{\text{mass}} \\ f^{\pm}_{\text{mass}}.\left[\frac{(-\bar{U} \pm 2a)}{\gamma} + \bar{u}\right] \\ f^{\pm}_{\text{mass}}.\bar{v} \\ f^{\pm}_{\text{mass}}.\bar{w} \\ f^{\pm}_{\text{energy}} \end{Bmatrix} \tag{13}$$

where

$$f^{\pm}_{\text{mass}} = \pm\frac{\rho a}{4}(\bar{M} \pm 1)^2 \tag{14}$$

$$f^{\pm}_{\text{energy}} = f^{\pm}_{\text{mass}} \left\{ \frac{[(\gamma-1)\bar{U} \pm 2a]^2}{2(\gamma^2-1)} - \frac{\bar{U}^2}{2} + \frac{\mathbf{q}^2}{2} + (\boldsymbol{\omega}.\bar{\mathbf{M}}) \frac{(-\bar{U} \pm 2a)}{\gamma} \right\} \tag{15}$$

and the Mach number normal to the cell face is defined as

$$\bar{M} = \frac{\bar{U}}{a} \tag{16}$$

$$a = \sqrt{\frac{\gamma P}{\rho}} \tag{17}$$

The above splitting is only valid for $|\bar{M}| \leqslant 1$. Otherwise

$$\left.\begin{aligned} \bar{\mathbf{F}}^{+} &= \bar{\mathbf{F}} \\ \bar{\mathbf{F}}^{-} &= 0 \end{aligned}\right\} \bar{M} > 1 \tag{18}$$

$$\left.\begin{aligned} \bar{\mathbf{F}}^{+} &= 0 \\ \bar{\mathbf{F}}^{-} &= \bar{\mathbf{F}} \end{aligned}\right\} \bar{M} < -1 \tag{19}$$

The values of the conserved variables used in the split fluxes must be consistent with the splitting, i.e. the positive vector must be evaluated using information from upstream (in the principal direction) of the cell face only, and the negative vector using information from downstream only. Hence the flux vector is split by

$$\bar{\mathbf{F}} = \bar{\mathbf{F}}^{+}(\mathbf{U}^{+}_{\text{r}}) + \bar{\mathbf{F}}^{-}(\mathbf{U}^{-}_{\text{r}}) \tag{20}$$

with the upwind interpolations given by a third-order spatial interpolation [22]. High order schemes suffer from spurious oscillations in regions of high flow quantity gradients, and so a flux limiter is required, and the continuously differentiable one due to Anderson *et al.* [22] was chosen.

Once $\bar{\mathbf{F}}$ has been split into its components the resulting flux must be rotated back to the original co-ordinate system. This is achieved by

$$\mathbf{F}_{\text{r}}.\mathbf{n}_{\text{r}} = [R]^{-1}[\bar{\mathbf{F}}^{+}(\mathbf{U}^{+}_{\text{r}}) + \bar{\mathbf{F}}^{-}(\mathbf{U}^{-}_{\text{r}})] \tag{21}$$

where $[R]$ is the rotation matrix.

2.2.1. Explicit time-stepping scheme. An explicit multi-stage Runge–Kutta scheme is used to integrate the equations forward in time. However, the four-stage scheme of Jameson *et al.* [23] is not efficient for an upwind scheme, since the stability limit is greatly reduced from the $2\sqrt{2}$ value for a central difference scheme [24]. A three-stage scheme is used, which can operate at a CFL number of 1.5. The time-stepping scheme used for each computational cell to integrate from time level n to $n+1$ is

$$\mathbf{U}^{n+\alpha_j}_{\text{r}} = \mathbf{U}^{n}_{\text{r}} - \alpha_j \frac{\Delta t}{V} \left\{ V\mathbf{G}_{\text{r}}(\mathbf{U}^{n+\alpha_{j-1}}_{\text{r}}) + \sum_{k=1}^{6} [R]^{-1}_{k} [\bar{\mathbf{F}}^{+}(\mathbf{U}^{+}_{\text{r}})^{n+\alpha_{j-1}}_{k} + \bar{\mathbf{F}}^{-}(\mathbf{U}^{-}_{\text{r}})^{n+\alpha_{j-1}}_{k}] A_k \right\} \tag{22}$$

with $\alpha_{0,1,2,3} = 0, \frac{1}{4}, \frac{1}{2}, 1$. In the above, V is the cell volume, and k represents the six cell faces. The source term is a volume integral, so does not need to be upwinded.

Apart from the flux-limiter, no dissipation is required by the upwind scheme since, by accurately modelling the physics of the flow the upwind differencing is naturally dissipative. As this is a steady flow local time-stepping is used to accelerate convergence.

2.3. Unsteady approach

The Euler equations in integral form for a mesh rotating in a fixed axis system are

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \mathbf{U}\,\mathrm{d}V + \int_{\partial V} \mathbf{F}.\mathbf{n}\,\mathrm{d}S = 0 \tag{23}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} \rho[\mathbf{q} - (\boldsymbol{\omega} \times \mathbf{r}(t))] \\ \rho u[\mathbf{q} - (\boldsymbol{\omega} \times \mathbf{r}(t))] + P\mathbf{i} \\ \rho v[\mathbf{q} - (\boldsymbol{\omega} \times \mathbf{r}(t))] + P\mathbf{j} \\ \rho w[\mathbf{q} - (\boldsymbol{\omega} \times \mathbf{r}(t))] + P\mathbf{k} \\ E[\mathbf{q} - (\boldsymbol{\omega} \times \mathbf{r}(t))] + P\mathbf{q} \end{bmatrix} \tag{24}$$

The co-ordinate vector $\mathbf{r}(t)$ is time dependent in the fixed axis system, i.e.

$$\mathbf{r}(t) = [R(t)]\mathbf{r}(0) \tag{25}$$

where $[R(t)]$ is the time-dependent rotation matrix. The z-axis is taken as the rotation axis here, and so

$$[R(t)] = \begin{bmatrix} \cos(\Omega_z t) & \sin(\Omega_z t) & 0 \\ -\sin(\Omega_z t) & \cos(\Omega_z t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{26}$$

The equation set is again closed by

$$P = (\gamma - 1)\left[E - \frac{\rho}{2}\mathbf{q}^2\right] \tag{27}$$

The upwind scheme outlined in the previous section is used, with the grid speed terms now time dependent.

2.3.1. Implicit time-stepping scheme. An implicit form of the differential equation for each computational cell is considered,

$$\frac{\partial(V^{n+1}\mathbf{U}^{n+1})}{\partial t} + \mathbf{R}(\mathbf{U}^{n+1}) = 0 \tag{28}$$

where V is the time-dependent cell volume and $\mathbf{R}$ is the upwinded flux integral. The implicit temporal derivative is then approximated by a second-order backward difference, following Jameson [25]. However, the hovering rotor case is started by moving the blade into a stationary

fluid, and a small initial time-step is required to start the computation. This time-step can be increased during the computation, so a variable time-step scheme is adopted. If Δt^{n+1} is the time-step from time level n to $n+1$, and Δt^n the time-step from time level $n-1$ to n, the scheme can be written as

$$\frac{(2\Delta t^{n+1}+\Delta t^n)}{(\Delta t^{n+1}+\Delta t^n)\Delta t^{n+1}}[V^{n+1}\mathbf{U}^{n+1}]-\frac{(\Delta t^{n+1}+\Delta t^n)}{\Delta t^{n+1}\Delta t^n}[V^n\mathbf{U}^n]$$
$$+\frac{\Delta t^{n+1}}{(\Delta t^{n+1}+\Delta t^n)\Delta t^n}[V^{n-1}\mathbf{U}^{n-1}]+\mathbf{R}(\mathbf{U}^{n+1})=0 \tag{29}$$

$\mathbf{U}^{n+1}$ must now be iterated on to satisfy this equation. To achieve this a new residual $\mathbf{R}^*(\mathbf{U})$ is defined as

$$\mathbf{R}^*(\mathbf{U})=\frac{(2\Delta t^{n+1}+\Delta t^n)}{(\Delta t^{n+1}+\Delta t^n)\Delta t^{n+1}}[V^{n+1}\mathbf{U}]-\frac{(\Delta t^{n+1}+\Delta t^n)}{\Delta t^{n+1}\Delta t^n}[V^n\mathbf{U}^n]$$
$$+\frac{\Delta t^{n+1}}{(\Delta t^{n+1}+\Delta t^n)\Delta t^n}[V^{n-1}\mathbf{U}^{n-1}]+\mathbf{R}(\mathbf{U}) \tag{30}$$

and then a new differential equation can be written in terms of a fictitious time τ, (called pseudo-time)

$$V^{n+1}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\tau}+\mathbf{R}^*(\mathbf{U})=0 \tag{31}$$

This is simply time-marched to convergence in the fictitious time τ, for each real time-step. Clearly as $\mathbf{R}^*\to 0, \mathbf{U}\to\mathbf{U}^{n+1}$. For each real time-step Equations (31) are solved to convergence using a form of the multi-stage time-stepping scheme with local time-stepping that is used for steady computations. The R.H.S. is manipulated such that it is implicit and gives a 'residual' that tends to zero, see Reference [26] for more details. For example integrating from pseudotime-level m to $m+1$, the scheme would be

$$\left(1+\alpha_j\frac{\Delta\tau(2\Delta t^{n+1}+\Delta t^n)}{(\Delta t^{n+1}+\Delta t^n)\Delta t^{n+1}}\right)\mathbf{U}^{m+\alpha_j}$$
$$=\mathbf{U}^m+\alpha_j\frac{\Delta\tau(2\Delta t^{n+1}+\Delta t^n)}{(\Delta t^{n+1}+\Delta t^n)\Delta t^{n+1}}\mathbf{U}^{m+\alpha_{j-1}}-\alpha_j\frac{\Delta\tau}{V^{n+1}}\left\{\frac{(2\Delta t^{n+1}+\Delta t^n)}{(\Delta t^{n+1}+\Delta t^n)\Delta t^{n+1}}\right.$$
$$\times V^{n+1}\mathbf{U}^{m+\alpha_{j-1}}-\frac{(\Delta t^{n+1}+\Delta t^n)}{\Delta t^{n+1}\Delta t^n}V^n\mathbf{U}^n+\frac{\Delta t^{n+1}}{(\Delta t^{n+1}+\Delta t^n)\Delta t^n}V^{n-1}\mathbf{U}^{n-1}$$
$$\left.+\sum_{k=1}^{6}[R]_k^{-1}[\bar{\mathbf{F}}^+(\mathbf{U}^+)_k^{m+\alpha_{j-1}}+\bar{\mathbf{F}}^-(\mathbf{U}^-)_k^{m+\alpha_{j-1}}]A_k^{n+1}\right\} \tag{32}$$

with $\alpha_{0,1,2,3}=0,\frac{1}{4},\frac{1}{2},1$. k represents the six cell faces, and $\Delta\tau$ is the pseudotime-step. There is no limit to the size of the real time-step that can be taken and this leads to a large reduction

in CPU time compared to an explicit scheme [26–28]. The time step is limited by accuracy rather than stability, which is the reason small time-steps are used at the beginning of the computation. This approach means that the instantaneous grid positions and speeds, and the geometric quantities (normal vectors, areamoment vectors, etc.), only have to be recomputed once every real time-step, and remain constant during the pseudo-time iterations. The cell volumes are constant for this case.

To improve convergence, the initial guess of the solution at the next time-level is set using,

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \frac{\Delta t^{n+1}}{\Delta t^n}\{\mathbf{U}^n - \mathbf{U}^{n-1}\} \tag{33}$$

In fact, the same code can be used for both steady and unsteady simulations. Two angular velocities are defined relating to steady flow, Ω_{st}, and unsteady flow, Ω_{un}, and the time-step coefficients in Equation (32) labelled as

$$T^{n+1} = \frac{(2\Delta t^{n+1} + \Delta t^n)}{(\Delta t^{n+1} + \Delta t^n)\Delta t^{n+1}} \tag{34}$$

$$T^{n} = \frac{(\Delta t^{n+1} + \Delta t^n)}{\Delta t^{n+1}\Delta t^n} \tag{35}$$

$$T^{n-1} = \frac{\Delta t^{n+1}}{(\Delta t^{n+1} + \Delta t^n)\Delta t^n} \tag{36}$$

The general solution procedure is then (dropping r subscript)

$$\begin{aligned}(1 + \alpha_j \Delta\tau T^{n+1})\mathbf{U}^{m+\alpha_j} = \mathbf{U}^m + \alpha_j \Delta\tau T^{n+1}\mathbf{U}^{m+\alpha_{j-1}} - \alpha_j \frac{\Delta\tau}{V^{n+1}} \Big\{ & T^{n+1}V^{n+1}\mathbf{U}^{m+\alpha_{j-1}} - T^n V^n \mathbf{U}^n \\ & + T^{n-1}V^{n-1}\mathbf{U}^{n-1} + V^{n+1}\mathbf{G}(\mathbf{U}^{m+\alpha_{j-1}}) \\ & + \sum_{k=1}^{6} [R]_k^{-1}[\bar{\mathbf{F}}^+(\mathbf{U}^+)_k^{m+\alpha_{j-1}} + \bar{\mathbf{F}}^-(\mathbf{U}^-)_k^{m+\alpha_{j-1}}]A_k^{n+1} \Big\}\end{aligned} \tag{37}$$

with $\alpha_{0,1,2,3} = 0, \frac{1}{4}, \frac{1}{2}, 1$. k represents the six cell faces, and

$$\mathbf{G} = \begin{bmatrix} 0 \\ -\rho\Omega_{\mathrm{st}} v \\ \rho\Omega_{\mathrm{st}} u \\ 0 \\ 0 \end{bmatrix}, \qquad \mathbf{r}(t) = \begin{bmatrix} \cos(\Omega_{\mathrm{un}} t) & \sin(\Omega_{\mathrm{un}} t) & 0 \\ -\sin(\Omega_{\mathrm{un}} t) & \cos(\Omega_{\mathrm{un}} t) & 0 \\ 0 & 0 & 1 \end{bmatrix} . \mathbf{r}(0) \tag{38}$$

To run a steady simulation

$$\begin{aligned} \Omega_{\mathrm{st}} &= \Omega_z \\ \Omega_{\mathrm{un}} &= 0 \\ T^{n-1} &= T^n = T^{n+1} = 0 \end{aligned} \tag{39}$$

the number of real timesteps = 1, and m represents n and $\Delta\tau$ represents Δt in the timestepping scheme. For an unsteady simulation

$$\begin{aligned} \Omega_{\mathrm{st}} &= 0 \\ \Omega_{\mathrm{un}} &= \Omega_z \end{aligned} \tag{40}$$

Hence, for a steady simulation all geometric quantities are fixed (in the blade-fixed coordinate system), while for an unsteady case the source term vanishes, and all geometric terms are time-dependent.

2.4. Boundary conditions

At upper, lower, and spanwise farfield boundaries, characteristic based conditions are applied. At periodic planes, the conserved variables required in neighbouring cells (i.e. the cells adjacent to the periodic planes are treated as internal cells) are evaluated using the following transformations:

$$\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}_{\mathrm{Downstream}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos(2\pi/N_{\mathrm{b}}) & \sin(2\pi/N_{\mathrm{b}}) & 0 & 0 \\ 0 & -\sin(2\pi/N_{\mathrm{b}}) & \cos(2\pi/N_{\mathrm{b}}) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}_{\mathrm{Upstream}} \tag{41}$$

$$\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}_{\mathrm{Upstream}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos(2\pi/N_{\mathrm{b}}) & -\sin(2\pi/N_{\mathrm{b}}) & 0 & 0 \\ 0 & \sin(2\pi/N_{\mathrm{b}}) & \cos(2\pi/N_{\mathrm{b}}) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}_{\mathrm{Downstream}} \tag{42}$$

where N_{b} is the number of blades.

3. PERIODIC GRID GENERATION

A transfinite interpolation method, originally described by Gordon and Hall [29], is used to generate structured grids.

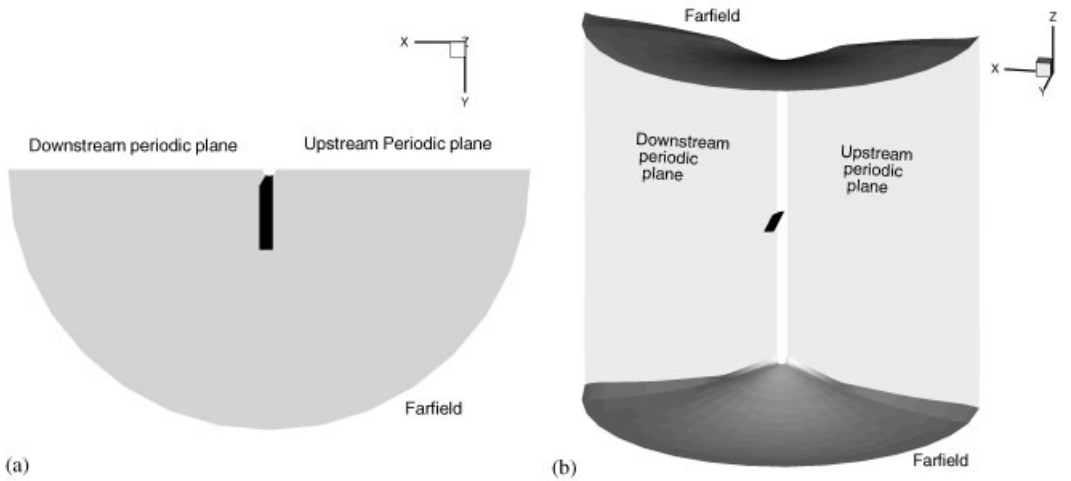

Figure 1. (a) and (b) Two views of computational domain and blade.

For N-bladed hovering rotor calculations only one blade, and $1/N$ of the cylindrical domain, need be considered, and the up- and downstream boundary planes can be treated as periodic boundaries. See References [4, 19] for details of the grid generation. To improve grid quality an elliptic smoothing scheme is also applied, see References [30, 19]. The geometry considered is that of the well-known Caradonna–Tung two-bladed rotor [31]. This is a rotor with no twist or taper, aspect ratio six, with a constant NACA0012 section. The incidence is 8°. In the hover case only one blade, and half the physical domain, need be considered. The axis system used is z vertical, y spanwise (from root to tip) and x to give an orthogonal system. Hence the rotation vector is $\boldsymbol{\omega} = [0, 0, \Omega_z]$, and the $y = 0$ plane is the periodic plane.

The mesh used in the computations is an O–H mesh of dimensions 161(chord) $\times$113(span)$\times$ 65(vertical) points, with 65 spanwise sections on the blade. Figure 1 shows the extent of the computational domain. Figure 1(a) is a view from above, showing the positions of the blade, hub, and periodic and spanwise farfield planes, and (b) shows the positions of the upper and lower farfield planes. Figure 2(a) shows the blade surface and the hub boundary plane, and (b) the blade near the tip, showing an O-plane variation.

4. RESULTS

The Caradonna–Tung hovering rotor test case with a tip Mach number of 0.794 was run using the explicit and implicit scheme. The implicit scheme was run with 30, 60, 120 and 360 real time-steps per rotation, i.e.

$$\Delta t = \frac{2\pi}{N_T \Omega_z} \tag{43}$$

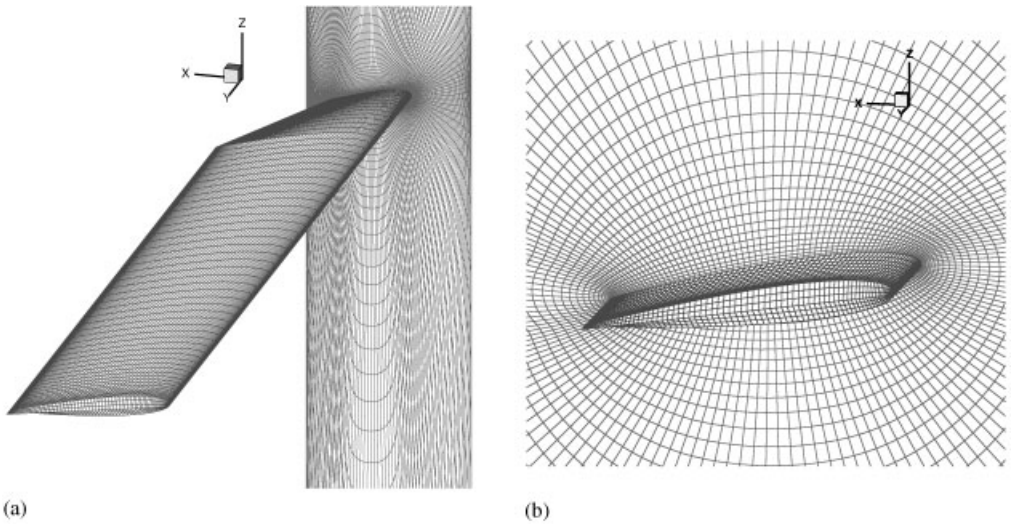

Figure 2. (a) Blade surface mesh and hub plane, (b) blade tip region.

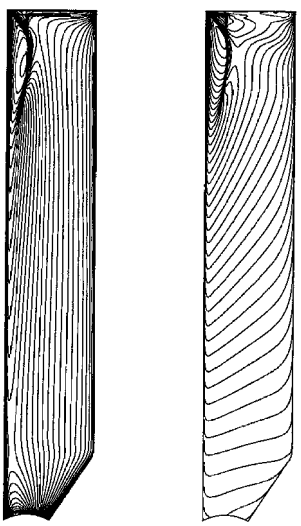

Figure 3. Upper surface pressure coefficient and mach contours.

where $N_T = 30, 60, 120$ or 360. The initial time-step was $1/20$ of this, and was increased to the constant value over the first twenty time-steps. The converged explicit and implicit solutions were identical. The upper surface pressure coefficient and Mach contours are shown in Figure 3 (50 contour levels are plotted between -1.0 and 1.5 for C_P and between 0.0 and 1.55 for Mach number).

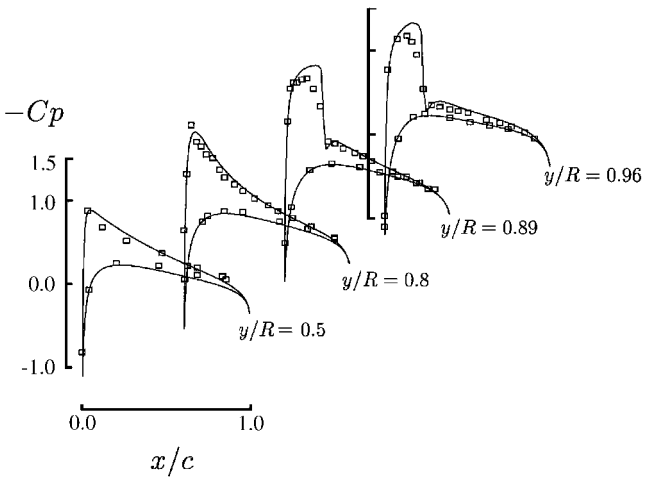

Figure 4. Computed and experimental pressure coefficient at radial stations.

Also shown, in Figure 4, are surface pressure coefficient variations at several spanwise stations compared with experimental results (plotted as squares). The results compare well, except for the computed result showing stronger shock strength. This is expected since viscosity will weaken shock strength compared to an inviscid result.

The solution evolution was investigated by considering the vorticity contours in the downstream periodic plane (90° behind the blade) during the computations. Figure 5 shows vorticity contours during the explicit computation. Every 5000 time-steps are shown. Hence, almost 30000 time-steps are required for convergence.

Figure 6 shows vorticity contours in the downstream periodic plane during the implicit computation. The solutions after 1, 3, 5, 8, 12 and 20 revolutions are shown. Almost 20 revolutions are required for convergence.

Figure 7 shows the convergence history for the explicit scheme and the implicit scheme with 60 real time-steps per revolution. The revolution count during the unsteady computation is also shown for comparison. The convergence of the implicit scheme with other values of N_T was similar except for a scaling in the revolution count, see below. The residual is defined as

$$RES = \sqrt{\frac{1}{ni.nj.nk}\sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{k=1}^{nk}\left\{\left(\frac{\Delta\rho}{\Delta t}\right)^2+\left(\frac{\Delta\rho u}{\Delta t}\right)^2+\left(\frac{\Delta\rho v}{\Delta t}\right)^2+\left(\frac{\Delta\rho w}{\Delta t}\right)^2+\left(\frac{\Delta E}{\Delta t}\right)^2\right\}_{i,j,k}} \tag{44}$$

(for the implicit scheme this is computed using $\Delta\tau$). The pseudotime-stepping scheme was run either until the residual reduced to below 10^{-6} or a maximum number of pseudotime-steps were performed. A minimum number of pseudo time-steps was also set. As the implicit residual jumps on the first pseudotime-step of each real time-step, then converges, clarity

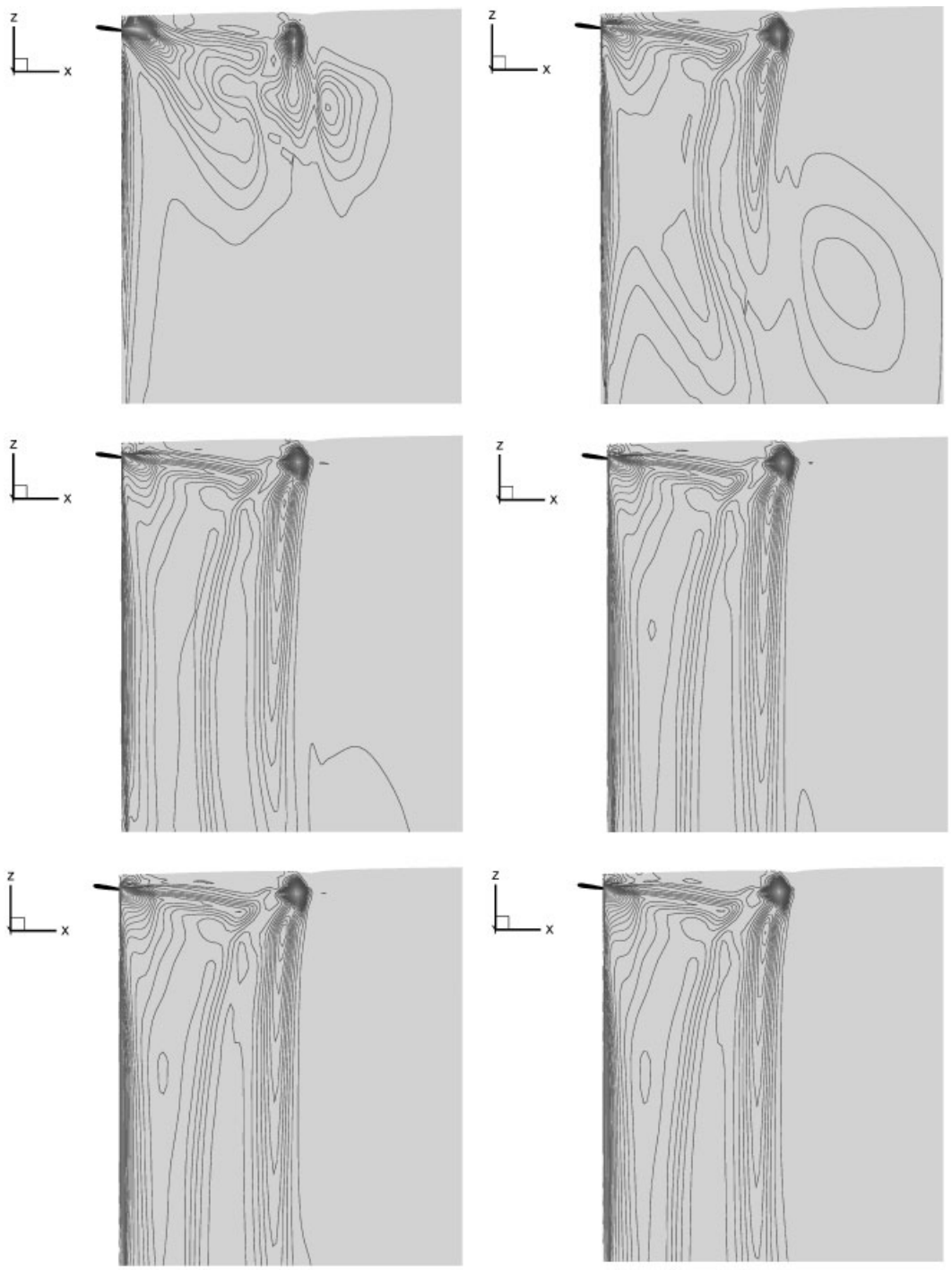

Figure 5. Vorticity contours on downstream periodic plane. Explicit solution, every 5000 time-steps.

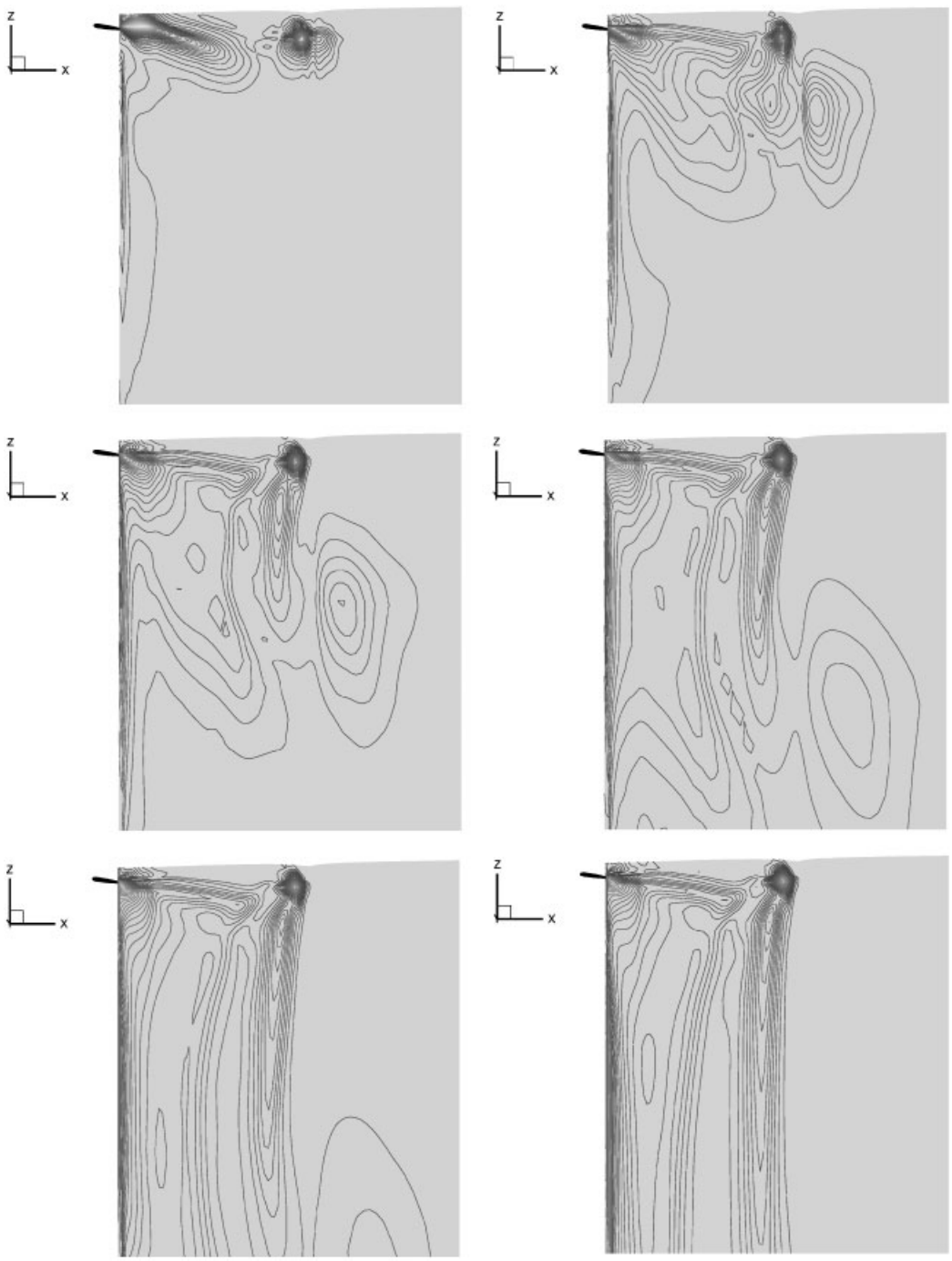

Figure 6. Vorticity contours on downstream periodic plane. Implicit solution after 1, 3, 5, 8, 12, and 20 revolutions.

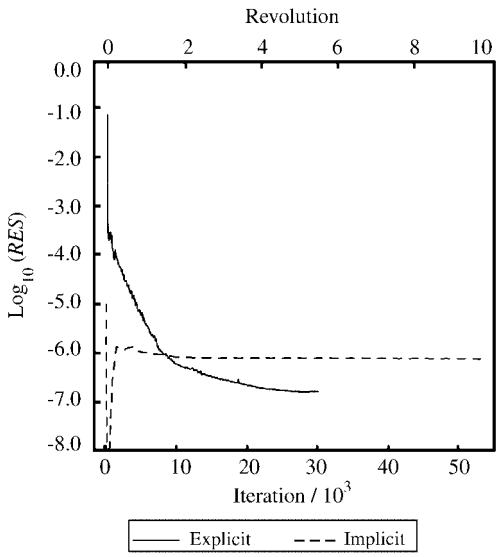

Figure 7. Explicit and implicit convergence history.

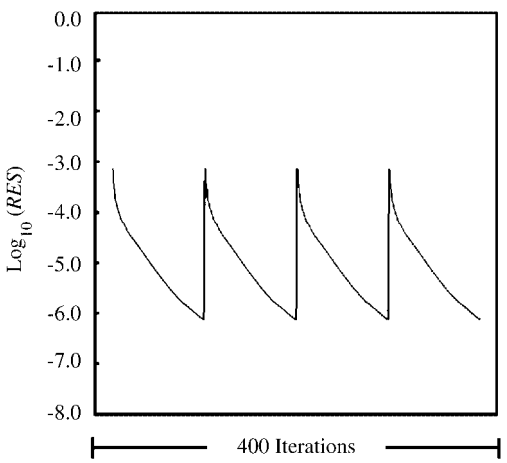

Figure 8. Implicit convergence over first four time-steps of fifth revolution.

prevents every iteration being plotted. Hence, only the residual on the final iteration every real time-step is plotted.

Figure 8 shows every iteration over the first four time-steps of the fifth revolution, for the 60 real time-steps per revolution solution. This shows how, once the solution becomes periodic,

Table I. Time-step count for unsteady solutions with varying real time-steps per revolution.

Number of real time-steps per revolution (N_T)	Average number of pseudotime-steps per real time-step	Total number of pseudotime-steps for 20 revolutions/10^3
30	119	71
60	91	109
120	69	166
360	43	310

the convergence each time-step becomes identical. Hence, even though the solution is still evolving there is no evidence of this in the residual. Examination of the blade loads, or detailed examination of the flowfield is required to ascertain whether the solution has converged. The other implicit solutions show identical behaviour. Table I presents the average number of pseudotime-steps per real time-step, and total number of pseudotime-steps for a 20 revolution simulation, for varying numbers of real time-steps per revolution. This shows that the fewer real time-steps per revolution are used the more efficient the implicit scheme is.

5. IMPLICATIONS FOR FORWARD FLIGHT

Forward flight is a fundamentally different flow to hovering flight. In hover the solution varies effectively linearly between adjacent time levels due to constant rotation rate. For hovering simulation two conclusions may be drawn: the fewer real time-steps per revolution the cheaper the implicit scheme is, and the implicit scheme is always significantly more expensive than the explicit.

Clearly these conclusions have serious implications for forward flight simulations. The unsteady scheme must be used here, but more real time-steps per revolution will be required than for hover, to accurately capture the unsteady wake and particularly to simulate BVI effects. However, it is likely that fewer revolutions will be needed in forward flight, since the wake is swept downstream and capture of many turns is not required. Hence, although moving from hover to forward flight the number of real time-steps per revolution will need to be increased for accuracy, the number of revolutions is likely to be significantly lower than the twenty required to converge the hover case here.

It must also be remembered, however, that for forward flight the complete rotor disk must be solved, so both the grid density and distribution will need to be modified from the hovering case.

The real interest in forward flight, and main long-term application of the current research, is aeroelastic simulation. This requires the fully synchronised time-accurate coupling of CFD code with structural dynamics code in the time domain. The implicit scheme adopted here is ideal for such applications, since it involves solving a ‘steady-type’ problem for each real time step. Hence, the solution of the structural equations of the rotor blade fit naturally into the time stepping scheme, and synchronisation of both codes in time is simple to guarantee.

Algebraic grid motion schemes applicable to elastically deforming surfaces have already been developed, see for example Reference [32].

6. CONCLUSIONS

An upwind Euler solver has been presented, and applied to multibladed lifting hovering rotor flow. Both an explicit steady formulation and an implicit unsteady formulation have been presented, and applied to the same hovering case, as a means of both validating the unsteady code and considering implications for forward flight. Comparison of the two formulations, in terms of residual history, cost, and solution evolution, has shown that for hovering rotor cases, the unsteady and steady solutions are the same, but the unsteady solution is more expensive in terms of CPU time. It is also shown that for hover, the fewer real time-steps taken per revolution the more efficient the implicit scheme is. However, this is a characteristic of the case, which results in smooth solution variation between time steps. The unsteady scheme is essential for a forward flight case and, furthermore, more real time-steps will be needed, since the unsteady wake must be represented accurately to simulate BVI effects. However, it is likely that fewer revolutions will need to be computed in this case as the wake is swept downstream away from the blades, so the slow wake development demonstrated here for hover will not be a problem. Hence, forward flight simulation may not be significantly more expensive than hover for a similar grid density, but it must also be remembered that the entire rotor disk must be solved in forward flight, so the grid density and distribution will need to be changed.

It has also been demonstrated that for rotary flow simulation, the global residual is not a useful quantity to assess convergence. The residual reaches a very low (constant in the implicit case) value while the solution is still evolving. Since the residual as defined in Equation (44) was shown not to be an accurate indicator of convergence, it may be more appropriate to use the residual of the differential equation instead.

NOMENCLATURE

a	acoustic speed
A	cell face area
c	aerofoil chord
C_T	thrust coefficient
E	total energy
$\mathbf{f}$	transfinite interpolation function
$f^{\pm}_{\text{mass}}$	split mass flux components
$f^{\pm}_{\text{energy}}$	split energy flux components
$\mathbf{F}$	flux vector
$\bar{\mathbf{F}}$	flux vector normal to cell face
$\bar{\mathbf{F}}^{\pm}$	split flux vector components normal to cell face
$\mathbf{G}$	source term vector
$\bar{M}$	Mach number normal to cell face
$\mathbf{M}$	cell face area moment vector

$\bar{\mathbf{M}}$	normalized cell face area moment vector
$\mathbf{n}$	outward cell face normal vector
N_b	number of blades in rotor disk
N_T	number of real time-steps per revolution
P	static pressure
$\mathbf{q}$	velocity vector
$\mathbf{r}$	co-ordinate vector
R	rotation matrix
$\mathbf{R}$	residual vector
R_{Tip}	blade tip radius
t	time
Δt	time step
u, v, w	velocity components
U	contravariant velocity
$\bar{u}, \bar{v}, \bar{w}$	velocity components in local cell face axis
$\bar{U}$	contravariant velocity normal to cell face
$\mathbf{U}$	conserved quantity vector
U_∞	freestream speed
V	cell volume
x, y, z	inertial co-ordinates
α_j	time-step factor
γ	ratio of specific heats
κ	spatial interpolation weighting parameter
ξ, η, ζ	parametric co-ordinates
ψ	blending function
ρ	density
σ	solidity of rotor
$\boldsymbol{\omega}$	angular velocity vector
Ω	angular velocity

Subscripts

r	rotating reference frame
st	steady flow
un	unsteady flow
i	gridpoint counter around each blade section
j	gridpoint counter spanwise direction from hub outward
k	gridpoint counter from inner boundary outward

REFERENCES

1. Allen CB, Jones DP. Parallel implementation of an upwind Euler solver for inviscid hovering rotor flows. *The Aeronautical Journal* 1999; **103**(1021):129–138.
2. Allen CB. Multigrid acceleration of an upwind Euler code for hovering rotor flows. *The Aeronautical Journal* 2001; **105**(1051):517–524.
3. Allen CB. Multigrid convergence of fixed- and rotary-wing flows. *International Journal for Numerical Methods in Fluids* 2002; **39**:121–140.
4. Allen CB. The effect of grid topology and density on inviscid hovering rotor solutions. *Institution of Mechanical Engineers Journal of Aerospace Engineering*, Part G 1999; **213**:81–95.

5. Kroll N. Computation of the flow fields of propellers and hovering rotors using Euler equations. *Paper 28*, 12th European Rotorcraft Forum, Garmisch-Partenkirchen, Germany, September 1986.
6. Boniface JC, Sides J. Numerical simulation of steady and unsteady Euler flows around multibladed helicopter rotors. *Paper C10*, 19th European Rotorcraft Forum, Cernobbio (Como), Italy, September 1993.
7. Pahlke K, Raddatz J. 3D Euler methods for multibladed rotors in hover and forward flight. *Paper 20*, 19th European Rotorcraft Forum, Cernobbio (Como), Italy, September 1993.
8. Raddatz J, Pahlke K. 3D Euler calculations of multibladed rotors in hover: investigations of the wake capturing properties. *75th AGARD Fluid Dynamics Panel Meeting and Symposium on Aerodynamics and Aeroacoustics of Rotorcraft*. Berlin, Germany, October 1994.
9. Raddatz J, Rouzard O. Calculations of multibladed rotors in hover using 3D Euler methods of DLR and Onera. *Paper 11*, 21st European Rotorcraft Forum, St. Petersburg, Russia, August 1995.
10. Boniface JC, Sides J. Extension and improvement of existing Euler code of ONERA for multibladed rotors in hover. *HELISHAPE Technical Report*, 1995.
11. Sankar LN, Wake BE, Lekoudis SG. Solution of the unsteady Euler equations for fixed and rotating wing configurations. *AIAA Journal of Aircraft* 1986; **23**(4):283–289.
12. Sankar LN, Wake BE, Lekoudis SG. Computation of rotor blade flows using the Euler equations. *AIAA Journal of Aircraft* 1986; **23**(7):582–588.
13. Strawn RC, Barth TJ. A finite-volume Euler solver for computing rotary-wing aerodynamics on unstructured meshes. In *Proceedings of the 48th Annual American Helicopter Society Forum*, Washington, DC, June 1992.
14. Renzoni P, D'Alascio A, Kroll N, Peshkin D, Hounjet MHL, Boniface J-C, Vigevano L, Morino L, Allen CB, Dubuc L, Righi M, Scholl E, Kokkalis A. EROS: A European Euler code for helicopter rotor simulations. *Journal of Progress in Aerospace Sciences* 2000; **36**:437–485.
15. Srinivasan GR, Baeder JD, Obayashi S, McCroskey WJ. Flowfield of a lifting rotor in hover. A Navier–Stokes simulation. *AIAA Journal of Aircraft* 1992; **30**(10):2371–2377.
16. Wake BE, Egolf TA. Implementation of a rotary-wing Navier–Stokes solver on a massively parallel computer. *AIAA Journal of Aircraft* 1991; **29**(1):58–67.
17. Zhong B, Qin N. Non-inertial multiblock Navier–Stokes calculation for hovering rotor flowfields using high order upwind scheme. *Proceedings Royal Aeronautical Society Aerodynamics Conference*, London, April 2000.
18. Van-Leer B. Flux-vector splitting for the Euler equations. *Lecture Notes in Physics*, vol. 170. Springer: Berlin, 1982; 507–512.
19. Allen CB. CHIMERA volume grid generation within the EROS code. *Institution of Mechanical Engineers Journal of Aerospace Engineering* Part G 2000; **412**:125–141.
20. Parpia IH. Van-leer flux-vector splitting in moving coordinates. *AIAA Journal* 1988; **26**:113–115.
21. Obayashi S. Freestream capturing for moving coordinates in three dimensions. *AIAA Journal* 1992; **30**(4): 1125–1128.
22. Anderson WK, Thomas JL, Van-Leer B. Comparison of finite volume flux vector splittings for the Euler equations. *AIAA Journal* 1986; **24**:1453–1460.
23. Jameson A, Schmidt W, Turkel E. Numerical solution of the Euler equations by finite-volume methods using Runge–Kutta time-stepping schemes. *AIAA Paper* 81-1259, 1981.
24. Turkel E, Van-Leer B. Runge–Kutta methods for partial differential equations. *ICASE Report*, 1983.
25. Jameson AL. Time-dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA Paper* 91-1596, 1991.
26. Allen CB. Grid adaptation for unsteady flow computations. *Institution of Mechanical Engineers Journal of Aerospace Engineering*, Part G4 1997; **211**:237–250.
27. Gaitonde AL. A dual-time method for the solution of the unsteady Euler equations. *The Aeronautical Journal* 1994; 283–291.
28. Allen CB. The reduction of numerical entropy generated by unsteady shockwaves. *The Aeronautical Journal* 1997; **101**(1001):9–16.
29. Gordon WJ, Hall CA. Construction of curvilinear coordinate systems and applications of mesh generation. *International Journal for Numerical Methods in Engineering* 1973; **7**:461–477.
30. Thompson JF. A general three dimensional elliptic grid generation system on a composite block-structure. *Computer Methods in Applied Mechanics and Engineering* 1987; **64**:377–411.
31. Caradonna FX, Tung C. Experimental and analytical studies of a model helicopter rotor in hover. *NASA* TM-81232, September 1981.
32. Allen CB. An algebraic grid motion technique for large deformations. *Institution of Mechanical Engineers Journal of Aerospace Engineering* 2002; **216**:51–58.

