INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2002; **39**:121–140 (DOI: 10.1002/fld.282)

Multigrid convergence of inviscid fixed- and rotary-wing flows

C. B. Allen[*,†]

Department of Aerospace Engineering, University of Bristol, Bristol, BS8 ITR, U.K.

SUMMARY

The affect of multigrid acceleration implemented within an upwind-biased Euler method is presented, and applied to fixed-wing and rotary-wing flows. The convergence of fixed- and rotary-wing computations is shown to be vastly different, and multigrid is shown to be less effective for rotary-wing flows. The flow about a hovering rotor suffers from very slow convergence of the inner blade region, where the flow is effectively incompressible. Furthermore, the vortical wake must develop over several turns before convergence is achieved, whereas for fixed-wing computations the far-field grid and solution have little significance. Results are presented for single mesh and two, three, four, and five level multigrid, and using five levels a reduction in required CPU time of over 80 per cent is demonstrated for rotary-wing computations, but 94 per cent for fixed-wing computations. It is found that a simple V-cycle is the most effective, smoothing in the decreasing mesh density direction only, with a relaxed trilinear prolongation operator. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: computational fluid dynamics; Euler equations; structured grid generation; fixed-wing flows; rotary-wing flows; multigrid acceleration

INTRODUCTION

Increases in computer processing speed and memory have allowed numerical methods to be applied to increasingly complex flows and geometries. Furthermore, growing maturity of these methods have seen them used increasingly in initial design stages. The application of CFD methods to fixed-wing analysis and design is now commonplace, but the numerical simulation of the flow about a helicopter rotor is still a very expensive problem for CFD methods. The complexity of the flow—non-linear, highly three-dimensional, and unsteady—is increased due the effect of each blade moving into a fluid that has already been disturbed by a previous blade. 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. The requirements on the numerical mesh are fundamentally different for a hovering rotor flow than

[*]Correspondence to: C. B. Allen, Department of Aerospace Engineering, University of Bristol, Queens Building, University Walk, Bristol BS8 ITR, U.K.
[†]E-mail: c.b.allen@bristol.ac.uk

Received April 2001
Revised November 2001

a fixed-wing flow, since the tip vortex 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 normally require finer meshes than fixed-wing flows. Furthermore, a large numerical integration time is required for the wake to develop over several turns. This can lead to impractical run times, and so is an ideal candidate for parallelization [1]. However, parallel architectures are not always available, so a multigrid approach can be valuable in reducing run-times. In this approach, which originates from works by Fedorenko [2], Bakhvarov [3] and later by Brandt [4, 5], errors computed on the finest mesh are transferred to progressively coarser meshes to compute the corrections to the fine mesh solution. As larger time-steps are allowed on the coarser meshes, errors are propagated out of the domain faster than is possible on the fine mesh, resulting in faster convergence, see for example References [6–10]. This paper describes the implementation of the multigrid technique to an inviscid solver. In particular, the different convergence properties of fixed- and rotary-wing flows are considered.

Numerical solutions of fixed-wing flows have been commonplace since the early 1980s, and are too numerous to cite. However, numerical solutions for hovering rotor flows are less common for inviscid flows, see for example References [1, 11–20], and for viscous flows see for example References [21–23]. It is well known that the numerical dissipation used by central-difference schemes has a significant effect on the solution in terms of wake capturing. This is more significant for rotary-wing flows, since the accurate capture of the wake is important for the surface loads. It is shown in Reference [15] that a high level of dissipation results in rapid convergence but poor wake capturing, and hence incorrect blade loads, while a low level of dissipation results in more accurate wake and hence blade load representation, but slower convergence. To avoid this problem, 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 [24].

The structured grid approach is used in the current work, and the transfinite interpolation technique adopted, as this can be used for steady and unsteady computations. Unsteady flows, for example an oscillating wing or rotor blade in forward flight, require moving surfaces and hence meshes. Transfinite interpolation has been previously shown to be well suited to structured moving grids [25–28] and, when adopted in conjunction with a periodic transformation, for helicopter rotor flows [1, 20, 29–32]. The interpolation is simple, and hence requires little CPU time, and is extremely flexible.

In this paper the flow-solver, grid generation and multigrid implementation procedures are first presented. Then results of multigrid computations of fixed-wing flow and hovering multi-bladed rotor flow are considered for varying numbers of grid levels. Reductions in required CPU times of approximately 94 per cent are demonstrated for a five-level fixed-wing computation, but only 80 per cent for a five-level rotary-wing computation. The difference is partly due to the requirement to capture the vortical wake over several turns. This requires a much longer numerical integration time than a fixed-wing case, and convergence is not simply dependent on propagating all errors away from the solid surface as quickly as possible, as is the case for fixed-wing flows. Furthermore, near the blade root the flow is at very low Mach number, and this region also suffers from slow convergence.

EULER SOLVER

Hovering rotor blade computations require a rigid but rotating grid. However, instead of solving an unsteady problem, the three-dimensional unsteady Euler equations are transformed to a blade-fixed rotating reference frame. In this frame the hover case is then a steady problem. The equations may be transformed in terms of absolute or relative velocities. Absolute velocities are used here. If the frame rotates with angular velocity $\underline{\omega} = [\Omega_x, \Omega_y, \Omega_z]^T$, and the absolute velocity vector in the rotating frame is denoted by $\underline{\mathbf{q}}_{\mathbf{r}} = [u_r, v_r, w_r]^T$, the resulting Euler equations in integral form are then

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_r} \underline{\mathbf{U}}_{\mathbf{r}}\,\mathrm{d}V_r + \int_{\partial V_r} \underline{\mathbf{F}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_{\mathbf{r}}\,\mathrm{d}S_r + \int_{V_r} \underline{\mathbf{G}}_{\mathbf{r}}\,\mathrm{d}V_r = 0 \tag{1}$$

where

$$\underline{\mathbf{U}}_{\mathbf{r}} = \begin{bmatrix} \rho \\ \rho u_r \\ \rho v_r \\ \rho w_r \\ E \end{bmatrix}, \quad \underline{\mathbf{F}}_{\mathbf{r}} = \begin{bmatrix} \rho[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] \\ \rho u_r[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] + P\underline{\mathbf{i}}_{\mathbf{r}} \\ \rho v_r[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] + P\underline{\mathbf{j}}_{\mathbf{r}} \\ \rho w_r[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] + P\underline{\mathbf{k}}_{\mathbf{r}} \\ E[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] + P\underline{\mathbf{q}}_{\mathbf{r}} \end{bmatrix}, \quad \underline{\mathbf{G}}_{\mathbf{r}} = \begin{bmatrix} 0 \\ \rho(\underline{\omega} \times \underline{\mathbf{q}}_{\mathbf{r}}) \cdot \underline{\mathbf{i}}_{\mathbf{r}} \\ \rho(\underline{\omega} \times \underline{\mathbf{q}}_{\mathbf{r}}) \cdot \underline{\mathbf{j}}_{\mathbf{r}} \\ \rho(\underline{\omega} \times \underline{\mathbf{q}}_{\mathbf{r}}) \cdot \underline{\mathbf{k}}_{\mathbf{r}} \\ 0 \end{bmatrix} \tag{2}$$

Here $\underline{\mathbf{G}}_{\mathbf{r}}$ is the source term resulting from the transformation, and $\underline{\mathbf{r}} = [x_r, y_r, z_r]^T$ is the coordinate vector. The equation set is closed by

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

For fixed-wing computations the rotation vector is simply set to zero.

Upwind difference scheme

A finite-volume upwind scheme is used to solve the integral form of the Euler equations (Equation (2)), since by correctly modelling the characteristic behaviour of the flow, upwind schemes are naturally dissipative. The flux-vector splitting of Van-Leer [24, 33] is used.

For each cell face a local orthogonal coordinate system (ξ, η, ζ) is adopted, where the principal coordinate 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, $\underline{\mathbf{n}}_\xi$. Unit vectors in two directions, lying in the cell face, $\underline{\mathbf{n}}_\eta$ and $\underline{\mathbf{n}}_\zeta$, 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 zin the local cell face axis system are required,

$$\bar{u} = \underline{\mathbf{q}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_\xi \tag{4}$$

$$\bar{v} = \underline{\mathbf{q}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_\eta \tag{5}$$

$$\bar{w} = \underline{\mathbf{q}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_\zeta \tag{6}$$

and the contravariant velocity normal to the face

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

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

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

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

Since quantities are evaluated at cell face centres, care must be taken when evaluating the geometric terms. The continuity equation means that for steady flow

$$\int_{\partial V} \rho\underline{\mathbf{q}} \cdot \underline{\mathbf{n}}\, \mathrm{d}S = 0 \tag{9}$$

must be satisfied. By considering a uniform flow it can be seen that, to avoid introducing error the conditions

$$\sum_{k=1}^{6} (\underline{\mathbf{n}}_{\xi} A)_k = \underline{\mathbf{0}} \tag{10}$$

where k represents the six cell faces, must be satisfied for every cell. This is satisfied by evaluating the cell face normals and area using the cross product of the diagonal vectors. The rotation of the axis requires satisfaction of a further condition. The continuity equation in this situation means

$$\int_{\partial V_r} \rho[\underline{\mathbf{q}}_{\mathbf{r}} - (\underline{\omega} \times \underline{\mathbf{r}})] \cdot \underline{\mathbf{n}}_{\mathbf{r}}\, \mathrm{d}S_r = 0 \tag{11}$$

must also be satisfied. By considering a rotating mesh through a uniform flow, it can be seen that the condition

$$\sum_{k=1}^{6} [(\underline{\omega} \times \underline{\mathbf{r}}) \cdot \underline{\mathbf{n}}_{\xi} A]_k = 0 \tag{12}$$

must be satisfied for every cell, otherwise errors are introduced into all equations. To satisfy this condition the area moment vector is defined for each cell face

$$\underline{\mathbf{M}} = \int_{\partial V_r} \underline{\mathbf{r}} \times \underline{\mathbf{n}}\, \mathrm{d}S_r \tag{13}$$

and is evaluated as in Reference [34]. If this vector is normalized by the cell face area

$$\underline{\bar{\mathbf{M}}} = \frac{\underline{\mathbf{M}}}{A} \tag{14}$$

the contravariant velocity can be expressed as

$$\bar{U} = \underline{\mathbf{q}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_{\xi} - \underline{\omega} \cdot \underline{\bar{\mathbf{M}}} \tag{15}$$

and Equation (12) is satisfied.

The general flux vector is split into a forward part, $\underline{\bar{\mathbf{F}}}^{+}$, associated with positive moving waves only, i.e. all eigenvalues of $[(\partial\underline{\bar{\mathbf{F}}}^{+})/(\partial\underline{\mathbf{U}})]\geqslant 0$, and a backward part, $\underline{\bar{\mathbf{F}}}^{-}$, associated with negative moving waves only, all eigenvalues of $[(\partial\underline{\bar{\mathbf{F}}}^{-})/(\partial\underline{\mathbf{U}})]\leqslant 0$. 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,

$$\underline{\bar{\mathbf{F}}}^{\pm}=\left\{\begin{array}{c} f^{\pm}_{\text{mass}} \\ f^{\pm}_{\text{mass}}\cdot[\frac{(-\bar{U}\pm 2a)}{\gamma}+\bar{u}] \\ f^{\pm}_{\text{mass}}\cdot\bar{v} \\ f^{\pm}_{\text{mass}}\cdot\bar{w} \\ f^{\pm}_{\text{energy}} \end{array}\right\} \tag{16}$$

where

$$f^{\pm}_{\text{mass}}=\pm\frac{\rho a}{4}(\bar{M}\pm 1)^2 \tag{17}$$

$$f^{\pm}_{\text{energy}}=f^{\pm}_{\text{mass}}\left\{\frac{[(\gamma-1)\bar{U}\pm 2a]^2}{2(\gamma^2-1)}-\frac{\bar{U}^2}{2}+\frac{\mathbf{q}_{\mathbf{r}}^2}{2}+(\underline{\omega}\cdot\underline{\bar{\mathbf{M}}})\frac{(-\bar{U}\pm 2a)}{\gamma}\right\} \tag{18}$$

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

$$\bar{M}=\frac{\bar{U}}{a}, \tag{19}$$

$$a=\sqrt{\frac{\gamma P}{\rho}} \tag{20}$$

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

$$\left.\begin{array}{l}\underline{\bar{\mathbf{F}}}^{+}=\underline{\bar{\mathbf{F}}} \\ \underline{\bar{\mathbf{F}}}^{-}=0\end{array}\right\}\quad \bar{M}>1 \tag{21}$$

$$\left.\begin{array}{l}\underline{\bar{\mathbf{F}}}^{+}=0 \\ \underline{\bar{\mathbf{F}}}^{-}=\underline{\bar{\mathbf{F}}}\end{array}\right\}\quad \bar{M}<-1 \tag{22}$$

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

$$\underline{\bar{\mathbf{F}}}=\underline{\bar{\mathbf{F}}}^{+}(\underline{\mathbf{U}}_{\mathbf{r}}^{+})+\underline{\bar{\mathbf{F}}}^{-}(\underline{\mathbf{U}}_{\mathbf{r}}^{-}) \tag{23}$$

with the upwind interpolations given by a third-order spatial interpolation [35]. 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.* [35] was chosen.

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

$$\underline{\mathbf{F}}_{\mathbf{r}} \cdot \underline{\mathbf{n}}_{\mathbf{r}} = R^{-1}[\underline{\bar{\mathbf{F}}}^{+}(\underline{\mathbf{U}}_{\mathbf{r}}^{+}) + \underline{\bar{\mathbf{F}}}^{-}(\underline{\mathbf{U}}_{\mathbf{r}}^{-})] \tag{24}$$

where R is the rotation matrix.

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.* [36] 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 [37]. 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

$$\underline{\mathbf{U}}_{\mathbf{r}}^{n+\alpha_j} = \underline{\mathbf{U}}_{\mathbf{r}}^{n} - \alpha_j \frac{\Delta t}{V} \left\{ V\underline{\mathbf{G}}_{\mathbf{r}}(\underline{\mathbf{U}}_{\mathbf{r}}^{n+\alpha_{j-1}}) + \sum_{k=1}^{6} R_k^{-1}[\underline{\bar{\mathbf{F}}}^{+}(\underline{\mathbf{U}}_{\mathbf{r}}^{+})_k^{n+\alpha_{j-1}} + \underline{\bar{\mathbf{F}}}^{-}(\underline{\mathbf{U}}_{\mathbf{r}}^{-})_k^{n+\alpha_{j-1}}]A_k \right\} \tag{25}$$

with $\alpha_{0,1,2,3} = 0, 1/4, 1/2, 1$. In the above, V is the cell volume, and k represents the six cell faces. The source term is a simple 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.

Boundary conditions For the hovering rotor cases farfield boundary conditions are periodic at the upstream and downstream faces (see next section). At the upper, lower, and spanwise boundaries characteristic conditions are applied, allowing for grid speeds. This is made simpler as absolute velocities are considered. In the fixed wing case, a symmetry condition is applied at the root plane, and farfield conditions are characteristic based.

In the rotary case experience has shown that convergence that can be affected if the farfield boundaries, particularly the lower one, are not at a sufficient distance. Hence, the upper, lower, and spanwise boundaries are set at 25 chords from the blade root.

GRID GENERATION

A transfinite interpolation method, originally described by Gordon and Hall [38], is used to generate structured grids. Transfinite interpolation gives the interpolated function $\underline{\mathbf{f}}(\xi,\eta,\zeta)$ throughout a defined region by a direct algebraic mapping. Control parameters can then be used to control the behaviour of the function, in terms of its derivatives normal to a line, at any of the lines where the function is known, by incorporating a set of univariate blending functions.

The general transfinite interpolation method results in a recursive algorithm, see Eriksson [39], but this can be significantly reduced. The interpolation is performed here by

$$\begin{aligned}\underline{\mathbf{f}}(\xi,\eta,\zeta) = {} & \psi_1^0(\zeta)\underline{\mathbf{f}}(\xi,\eta,0) + \psi_1^1(\zeta)\frac{\partial}{\partial\eta}\underline{\mathbf{f}}(\xi,\eta,1) \\ & + \psi_2^0(\zeta)[\psi_2^0(\zeta)\underline{\mathbf{f}}(\xi,\eta,1) + \psi_1^0(\zeta)\underline{\mathbf{f}}_2(\xi,\eta,1)]\end{aligned} \tag{26}$$

where $\underline{\mathbf{f}}(\xi,\eta,0)$ is the inner boundary, $\underline{\mathbf{f}}(\xi,\eta,1)$ the outer boundary, and $\underline{\mathbf{f}}_2(\xi,\eta,1)$ a smoothed outer boundary function to control smoothness.

The most effective blending functions have been found to be,

$$\bar{\zeta} = \left\{ \frac{e^{\zeta} - 1 - \zeta}{e - 2} \right\}^{st} \tag{27}$$

$$\psi_1^0 = 1 - \bar{\zeta} \tag{28}$$

$$\psi_1^1 = \sqrt{\bar{\zeta}} - \left\{ \frac{e^{\bar{\zeta}} - 1 - \bar{\zeta}}{e - 2} \right\} \tag{29}$$

$$\psi_2^0 = \bar{\zeta} \tag{30}$$

where *st* is a stretching exponent. The stretching exponent *st* for each grid line is computed using an iterative procedure, such that the grid cells at the inner boundary have a pre-defined aspect ratio.

After generating the volume mesh an elliptic smoothing is applied [40]. The smoothing has been coded such that boundaries are also smoothed (see Reference [32] for more details).

Periodic grid for rotary-wing computations

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. Hence, the grid must be generated with a suitable outer boundary distribution. This is achieved by initially generating the outer boundary, regardless of the grid topology, as a box. The outer boundary box is then scaled according to a local radius function. The distance between the inner and outer boundary is set to the length of the circular arc with the local radius. The transfinite interpolation algorithm is then used to compute the grid, and once the grid has been generated the entire grid, except the region over the blade, is transformed to produce a part-cylindrical domain (more details of the grid generation can be found in References [20, 32]). The geometry considered is that of the well known Caradonna–Tung two-bladed rotor [41]. 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 $\underline{\omega} = [0, 0, \Omega]$, 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, i.e. 1.18×10^6 points. These dimensions were chosen such that up to five levels of multigrid could be used, i.e. four coarsenings so the coarsest mesh is $10 \times 7 \times 4$ cells, with four spanwise cells on the blade. Figure 1 shows the blade surface mesh and the hub boundary plane. This shows how the blade has zero thickness outboard of the tip to form the tipslit. Figure 2(a) shows the inner and outer boundary (and wake plane), and (b) the outer periodic boundary. Only every other grid point in each direction is shown.

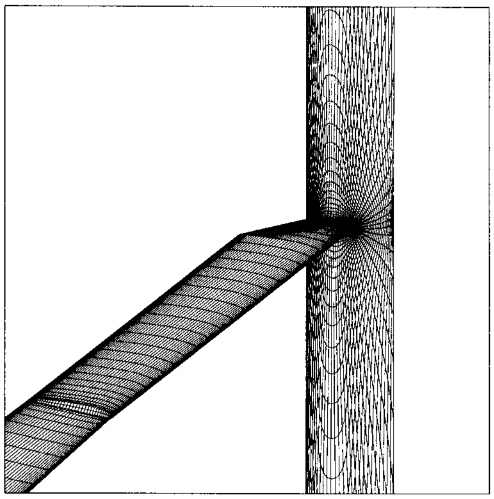


Figure 1. Blade surface mesh and hub plane.

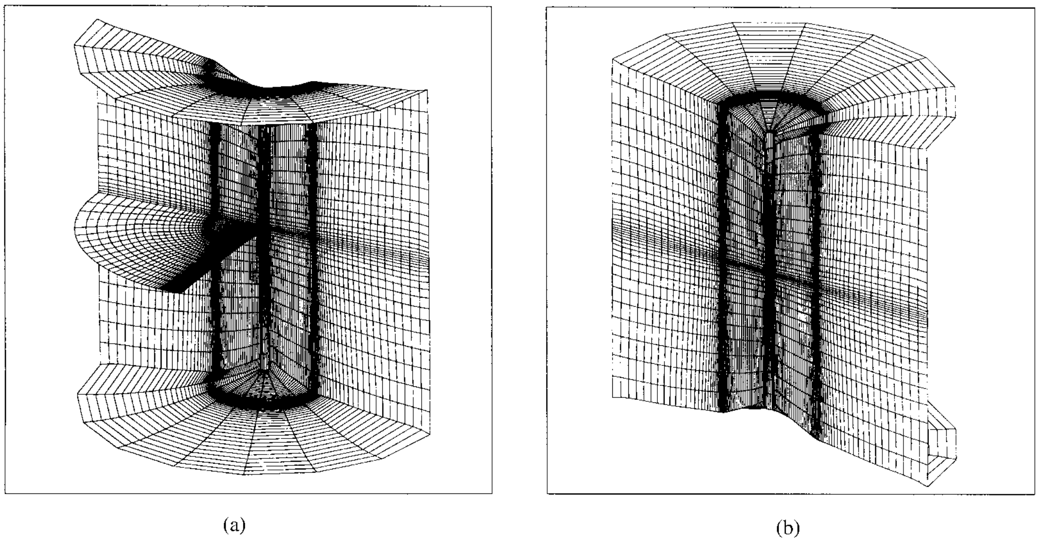
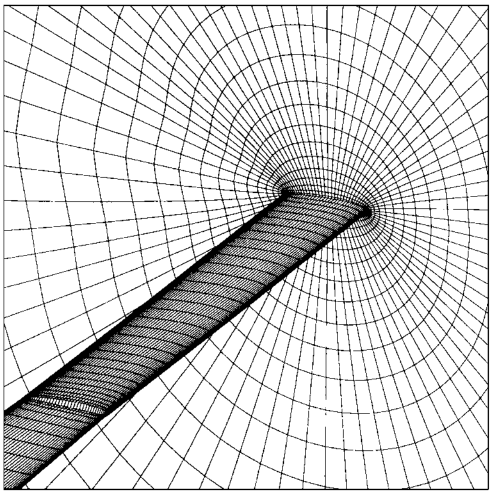


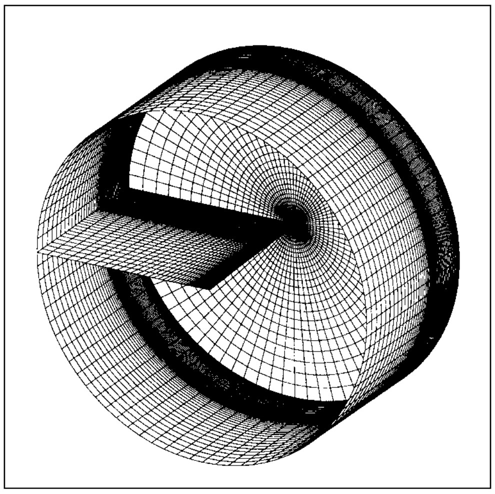
Figure 2. (a) Inner and outer boundary planes, (b) outer boundary plane.

Grid for fixed-wing computations

The convergence of fixed- and rotary-wing flows will be compared. Hence, similar geometries must be used, and so a wing of aspect ratio six, constant section of NACA0012 is used. The only difference with the rotor blade surface is at the root. Again an O–H mesh of dimensions 161(chord) $\times$ 113(span) $\times$ 65(vertical) points is used, with 65 spanwise sections on the wing.



(a)



(b)

Figure 3. (a) Wing surface and symmetry planes, (b) inner and outer boundary planes.

Figure 3(a) shows the wing surface mesh (plus part of tip-slit) and the symmetry plane, and (b) the same planes plus wake plane and outer boundary.

MULTIGRID SCHEME

It is well known that explicit time-stepping schemes lend themselves well to multigrid acceleration, see for example Reference [6]. In this approach, errors computed on the finest mesh are transferred to progressively coarser meshes to compute the corrections to the fine mesh solution. As larger time-steps are allowed on the coarser meshes, errors are propagated out of the domain faster than is possible on the fine mesh, resulting in faster convergence.

The scheme is presented here using N to represent the mesh number ($N=1$ is finest), V is the cell volume, $\underline{\mathbf{R}}$ represents the residual, $\underline{\mathbf{U}}^R$ and $\underline{\mathbf{U}}^S$ represent the restricted and smoothed (updated) solution respectively, and I_N and I^N are the number of iterations performed on mesh N in the decreasing and increasing mesh density directions (the 'r' subscript has been omitted from $\underline{\mathbf{U}}$ for clarity). For a simple 'V' cycle, the scheme can be written as

```
Perform $I_1$ iterations (smoothing sweeps) on mesh 1 (finest)
        $\underline{\mathbf{U}}_1^{n+\alpha_j} = \underline{\mathbf{U}}_1^n - \alpha_j \frac{\Delta t}{V_N} \underline{\mathbf{R}}(\underline{\mathbf{U}}_1^{n+\alpha_{j-1}}) \quad j = 1\ldots 3$
        $\Rightarrow \underline{\mathbf{U}}_1^S$
        set $\underline{\mathbf{f}}_1 = 0$
DO FOR $N = 2\ldots NMESH$
        Restrict solution to next finest mesh
                $\underline{\mathbf{U}}_N^R = \frac{\sum_{\mathrm{cell}=1}^{8} \underline{\mathbf{U}}_{N-1}^S V_{N-1}}{\sum_{\mathrm{cell}=1}^{8} V_{N-1}}$
```

```
          Evaluate forcing function
                    $\underline{\mathbf{f}}_N = \sum_{\text{cell}=1}^{8} \{\underline{\mathbf{R}}_{N-1}(\underline{\mathbf{U}}_{N-1}^S) + \underline{\mathbf{f}}_{N-1}\} - \underline{\mathbf{R}}_N(\underline{\mathbf{U}}_N^R)$
          Perform $I_N$ iterations on mesh $N$
                    $\underline{\mathbf{U}}_N^{n+\alpha_j} = \underline{\mathbf{U}}_N^n - \alpha_j \frac{\Delta t}{V_N}(\underline{\mathbf{R}}(\underline{\mathbf{U}}_N^{n+\alpha_{j-1}}) + \underline{\mathbf{f}}_N) \quad j = 1 \dots 3$
                         $\Rightarrow \underline{\mathbf{U}}_N^S$
ENDDO
DO FOR $N = NMESH - 1 \dots 1$
          Compute corrections on coarser mesh
                    $\Delta\underline{\mathbf{U}}_{N+1} = \underline{\mathbf{U}}_{N+1}^S - \underline{\mathbf{U}}_{N+1}^R$
          Pass, prolong, corrections to current mesh
                         $\Rightarrow \Delta\underline{\mathbf{U}}_N$
                    $\underline{\mathbf{U}}_N^S = \underline{\mathbf{U}}_N^R + \Delta\underline{\mathbf{U}}_N$
          Perform $I^N$ iterations on mesh $N$
                    $\underline{\mathbf{U}}_N^{n+\alpha_j} = \underline{\mathbf{U}}_N^n - \alpha_j \frac{\Delta t}{V_N}(\underline{\mathbf{R}}(\underline{\mathbf{U}}_N^{n+\alpha_{j-1}}) + \underline{\mathbf{f}}_N) \quad j = 1 \dots 3$
                         $\Rightarrow \underline{\mathbf{U}}_N^S$
ENDDO
```

The prolongation step, i.e. passing corrections up to next finest mesh, can be performed using several methods. The simplest way is to use direct injection, wherein all eight cells surrounding each coarser mesh cell are given the same correction. More commonly a trilinear interpolation scheme is used. As an upwind scheme is used here, an upwind solution correction interpolation can be used. However, although theoretically an upwind-biased prolongation results in a more robust and stable scheme, it has been shown that this does not have a significant effect on convergence [42, 43]. Hence, the extra computational requirement of using such a prolongation does not seem justified and a trilinear interpolation scheme is used here.

RESULTS

The Caradonna–Tung hovering rotor test case with a tip Mach number of 0.794 was run using a single mesh (finest), and as multigrid with two, three, four, and five mesh levels. The upper surface pressure coefficient and Mach contours are shown in Figure 4 (50 contour levels are plotted between -1.0 and 1.5 for C_P and between 0.0 and 1.55 for Mach number).

Also shown, in Figure 5, 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 fixed-wing case was also run with a constant wing incidence of 8°. To obtain a similar solution, i.e. a single shock with comparable peak pressure coefficient, an onflow Mach number of 0.7 was used. Figure 6 shows upper surface pressure coefficient and Mach contours. The surface pressure coefficient at several spanwise stations is shown in Figure 7. This clearly shows how the shock strength diminishes rapidly towards the tip.

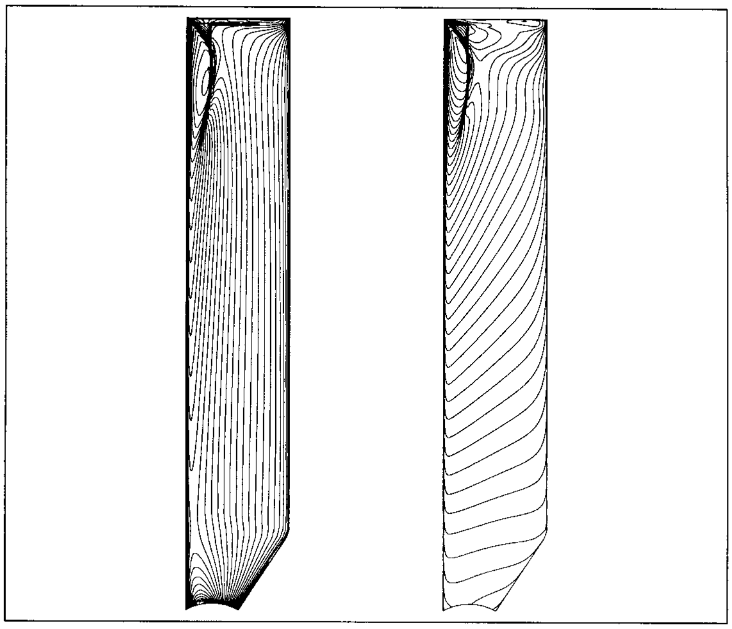


Figure 4. Upper surface pressure coefficient and Mach contours.

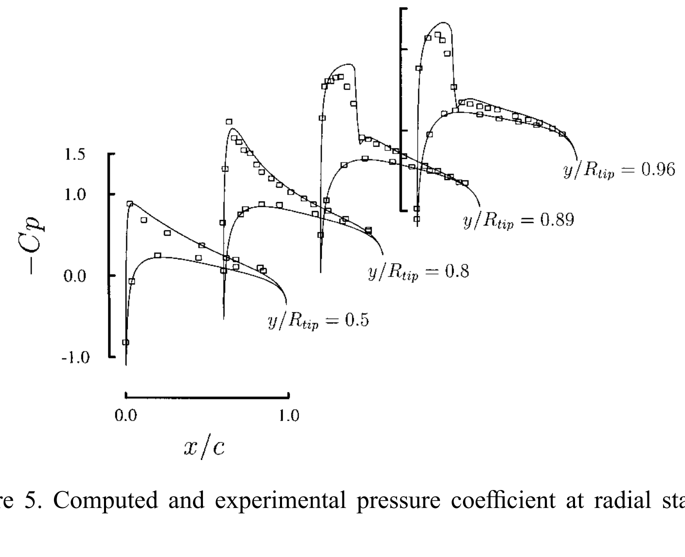


Figure 5. Computed and experimental pressure coefficient at radial stations.

Effect of multigrid scheme for fixed-wing computation

The convergence history, in terms of average global residual level against iteration number for various numbers of meshes will be considered initially. The residual is defined as

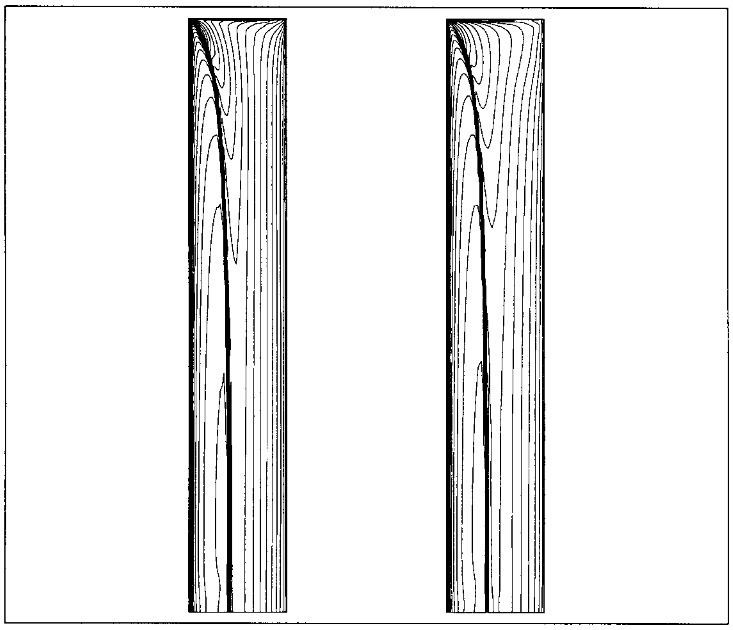


Figure 6. Upper surface pressure coefficient and Mach contours.

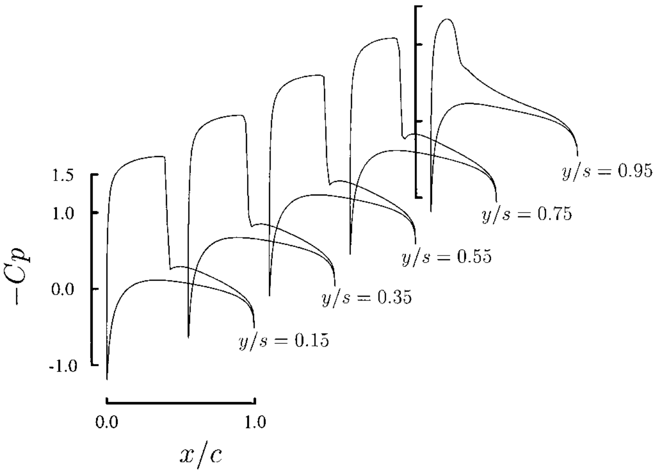


Figure 7. Computed and experimental pressure coefficient at radial stations.

$$\mathrm{RES} = \sqrt{\frac{1}{ni.nj.nk}\sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{k=1}^{nk}\left\{\left(\frac{\Delta\rho}{\Delta t}\right)^2 + \left(\frac{\Delta\rho u}{\Delta t}\right)^2 + \left(\frac{\Delta\rho v}{\Delta t}\right)^2 + \left(\frac{\Delta\rho w}{\Delta t}\right)^2 + \left(\frac{\Delta E}{\Delta t}\right)^2\right\}_{i,j,k}} \tag{31}$$

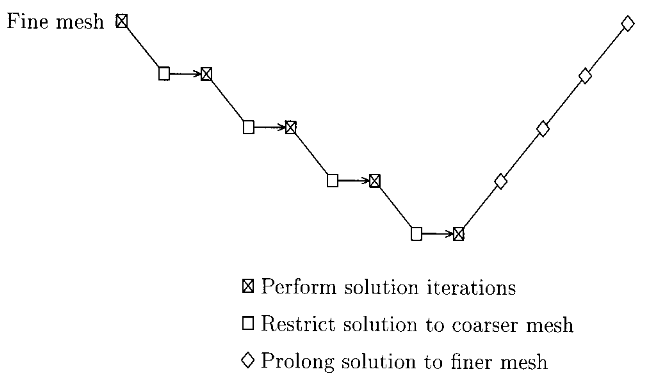

Fine mesh
⊠ Perform solution iterations
□ Restrict solution to coarser mesh
◇ Prolong solution to finer mesh

Figure 8. Multigrid cycle.

The fixed-wing case was run on a single mesh (finest), and with two, three, four and five levels of multigrid. Many combinations of the number of solution iterations on each mesh level on both the way up and down were considered. It was found that the most effective scheme was to use a simple V-cycle solving on the way down only, i.e. the 'sawtooth' cycle as shown in Figure 8. Although also smoothing on the way up resulted in fewer multigrid cycles required for convergence, in fact more total iterations and hence CPU time were required. It was found that the fastest convergence was achieved smoothing only on the way down, with a relaxed prolongation operator. When passing the error from mesh $N+1$ to mesh N the correction is relaxed by

$$\underline{\mathbf{U}}_N^S = \underline{\mathbf{U}}_N^R + \Theta \Delta \underline{\mathbf{U}}_N \tag{32}$$

where $\Theta = 0.8$ was found to be the optimum value regardless of number of mesh levels.

The multigrid approach means errors are propagated out of the domain at a rate dependent on the local grid size, and fixed-wing solutions are primarily dependent on propagating errors away from the solid surface. Hence, it seems logical, since coarser mesh iterations are so much 'cheaper' than fine mesh ones, to increase the number of iterations when moving in the decreasing density direction, as this gives the maximum signal propagation 'distance' per unit of time. This was found to improve convergence, and the fastest convergence was achieved using the following sequence

$$I_{1,2,3,4,5} = 2, 4, 8, 16, 32 \tag{33}$$

$$I^{1,2,3,4,5} = 0, 0, 0, 0, 0 \tag{34}$$

The coarse mesh solution is effectively converged at every time step using this scheme.

Figure 9 shows the convergence history for two, three, four and five mesh levels, using the iterations scheme above. It should be noted here that one 'iteration' in any multigrid scheme takes the same amount of CPU time. For example one cycle of a three-level calculation, with two iterations per mesh, would take the equivalent of $(2 + 2/8 + 2/64 +$ communication overhead) iterations on a single mesh. Hence, the CPU requirement can also be plotted on the same graph.

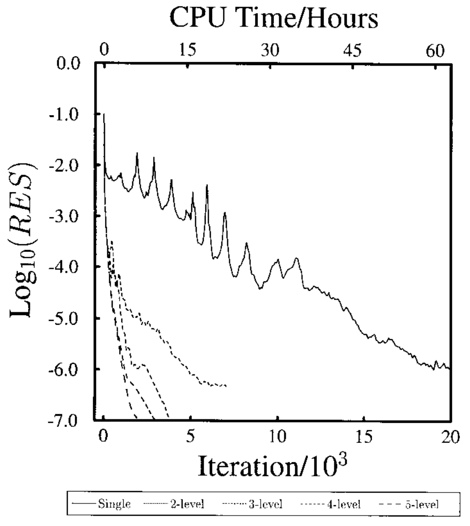


Figure 9. Multigrid convergence history.

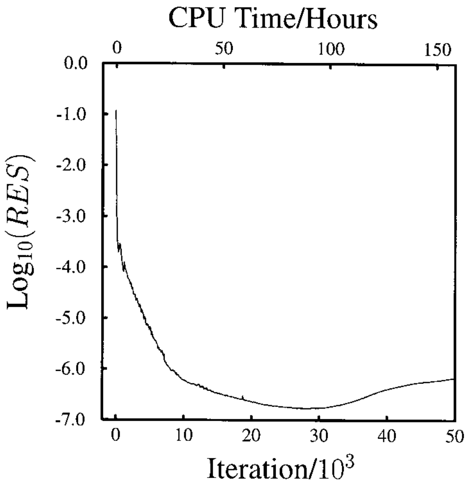


Figure 10. Single mesh convergence history.

Using an average global residual level of 10^{-6} as a measure, the multigrid scheme results in speed-ups of 4× for two levels, around 11× for three levels, around 16× for four levels, and 17× for five levels. Hence, a CPU reduction of 94 per cent is achieved using five mesh levels.

Effect of multigrid scheme for rotary-wing computation

Figure 10 shows the convergence history for the single mesh computation of the Caradonna–Tung case. It is not clear from this whether convergence has been achieved. The case was also

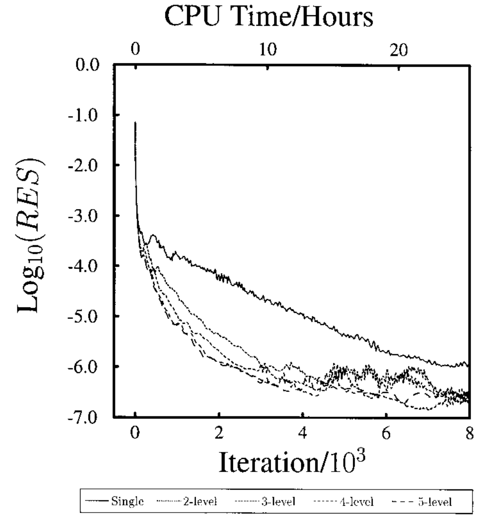


Figure 11. Multigrid convergence history.

run with two, three, four and five levels of multigrid, and Figure 11 shows the convergence histories. Only up to 8000 iterations are shown as there are no changes in the multigrid residual levels beyond this point.

However, the residual history is not a particularly useful quantity when assessing convergence for rotary flows. The vortical wake takes many iterations to develop, and the development does not have a huge effect on the global residual level. Furthermore, it is not clear whether there is some inherent unsteadiness in these hovering cases. Figure 11 implies that all four multigrid solutions have converged in around 8000 iterations or less, while Figure 10 seems to show that even after 50 000 iterations the single grid solution has still not settled, apparently resulting in a large multigrid speed-up. Analysing the solution shows that this is not the case however.

Global convergence of the solution is limited by both the development of the wake, and the slow convergence of the near-hub area where the solution is effectively incompressible. Hence the thrust coefficient for the inner 20 per cent of the blade has been considered, as this is a more sensitive convergence measure. The thrust coefficient is defined as

$$C_T = \frac{\text{Lift for all blades}}{\rho(\Omega R_{\text{Tip}})^2 \pi R_{\text{Tip}}^2} \tag{35}$$

and this is usually normalized by the solidity of the rotor, σ,

$$\sigma = \frac{N_{\text{blades}} R_{\text{Tip}} c}{\pi R_{\text{Tip}}^2} \tag{36}$$

Figure 12 shows the variation of the inner blade thrust coefficient for single mesh, and multigrid with two, three, four and five levels. This shows that the single mesh solution converges to a constant value of inner blade thrust coefficient in around 30 000 iterations, two-level multigrid in around 13 000, three-level in under 10 000, four-level in just over 7000,

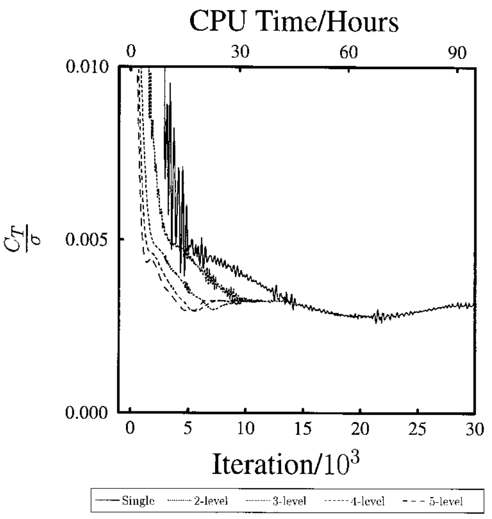


Figure 12. Multigrid thrust coefficient history.

and five-level in around 6000. Hence, the speed-ups are approximately 2.3× for two levels, 3.1× for three levels, 4.2× for four levels and 5.0× for five levels.

The optimum iteration scheme was also analysed for the rotary-wing case. The requirement for the wake to develop means that convergence is not solely dependent on propagating errors away from the solid surface. Hence, increasing the number of iterations on the coarser meshes was not efficient here. It was found that the optimum scheme was

$$I_{1,2,3,4,5} = 2,2,2,2,2 \tag{37}$$

$$I^{1,2,3,4,5} = 0,0,0,0,0. \tag{38}$$

The relaxation parameter value $\Theta = 0.8$ was again found to be optimum, regardless of the number of mesh levels used.

The speed-ups obtained are nowhere near as large as for the fixed-wing case and, as mentioned previously, there are two reasons for this. The first is the slow convergence of the inner blade region, where the flow is effectively incompressible. The second is the requirement of capturing the tip vortex over several turns in the rotary case. Neither of these phenomena are present in a fixed-wing flow. The complexity of the rotary-wing flowfield is demonstrated in Figure 13, which shows vorticity contours for the downstream periodic plane, i.e. the $y = 0$ plane, 90° behind the blade. The vorticity in each cell is defined as

$$\omega = |\underline{\nabla} \times \underline{\mathbf{q}}_{\mathbf{r}}| \tag{39}$$

and 40 contour levels are plotted between 0 and 0.5. This shows that three turns of the tip vortex have been captured. This compares well with previously published results [14].

The vortical wake capturing requirement is unique to rotary flows, but the problem of incompressible flow near the hub is a more common problem. One obvious solution would be to apply a preconditioning algorithm, see for example References [44, 45], which are proven

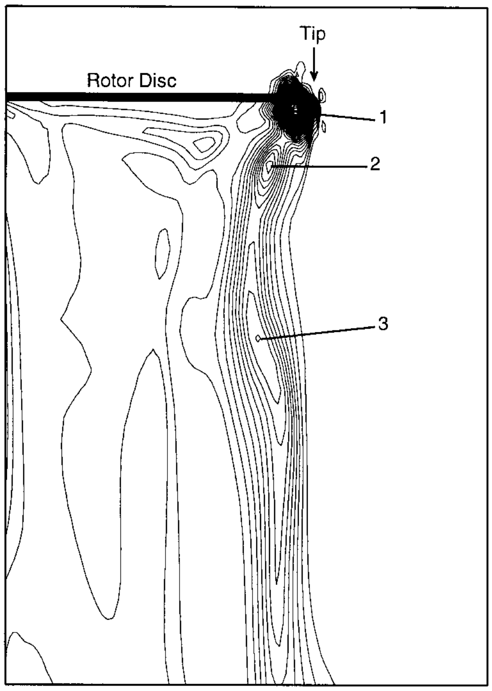


Figure 13. Vorticity contours in plane 90° behind blade.

to improve convergence for low speed flows. However, preconditioning is not considered here as it is the fundamental difference in convergence properties of fixed- and rotary-wing flows which is of interest.

CONCLUSIONS

An upwind Euler solver has been presented, with multigrid convergence acceleration, and applied to both fixed-wing and multibladed hovering rotary-wing flows. Using five levels of multigrid, a reduction in required CPU time of over 80 per cent has been demonstrated for rotary-wing computations, and 94 per cent for fixed-wing computations. Hence, the convergence of fixed-wing and rotary-wing computations are vastly different. The rotary-wing flow suffers from very slow convergence of the inner blade region, where the flow is effectively incompressible. Furthermore, the vortical wake of a hovering rotor must develop over several turns before convergence is achieved, whereas for fixed-wing computations the far-field grid and solution have little significance.

The most effective convergence has been achieved using a simple V-cycle, solving when moving in the decreasing mesh density direction only. However, due to the different solution evolution, the optimum number of iterations on each mesh level is different. For the fixed-wing case errors simply need to be propagated out of the domain as quickly as possible, so the number of iterations is increased as the grid density decreases. Convergence of the rotary-wing flow requires the wake to develop, so increasing the number of iterations on the coarser meshes is not efficient, and two solution iterations on each level is found to give the fastest convergence. A trilinear prolongation operator is employed with a relaxation parameter which has an optimum value of 0.8, regardless of the number of mesh levels employed or flow type.

NOTATION

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
$\underline{\mathbf{F}}$	flux vector
$\underline{\bar{\mathbf{F}}}$	flux vector normal to cell face
$\underline{\bar{\mathbf{F}}}^{\pm}$	split flux vector components normal to cell face
$\underline{\mathbf{G}}$	source term vector
$\bar{M}$	Mach number normal to cell face
$\underline{\mathbf{M}}$	cell face area moment vector
$\underline{\bar{\mathbf{M}}}$	normalized cell face area moment vector
$\underline{\mathbf{n}}$	outward cell face normal vector
P	static pressure
$\underline{\mathbf{q}}$	velocity vector
$\underline{\mathbf{r}}$	coordinate vector
R	rotation matrix
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
$\underline{\mathbf{U}}$	conserved quantity vector
U_{∞}	freestream speed
V	cell volume
x, y, z	inertial co-ordinates

Greek characters

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
$\underline{\omega}$	angular velocity vector
Ω	angular velocity

Subscripts

r	rotating reference frame

REFERENCES

1. Allen CB, Jones DP. Parallel implementation of an upwind Euler solver for inviscid hovering rotor flows. *Aeronautical Journal* 1999; **103**(1021):129–138.
2. Fedorenko RP. The speed of convergence of one iterative process. *Zh. Vych. Mat.* 1964; **4**(5):559–564. (*USSR Computational Mathematics and Mathematical Physics* 1964; **4**:227–235.)
3. Bakhvalov NS. On the convergence of a relaxation method with natural constraints of the elliptic operator. *Zh. Vych. Mat.* 1966; **6**(5):861–885. (*USSR Computational Mathematics and Mathematical Physics* 1966; **6**:101–135.)
4. Brandt A. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.* 1977; **31**(138):333–390.
5. Brandt A. Guide to multigrid development. In Hackbusch and Trottenberg (eds), *Multigrid Methods*. Springer-Verlag: New York; *Lecture Notes in Mathematics* 1982: 960.
6. Jameson A. Transonic flow calculations. *Report MAE 1751*, Princeton University, 1984.
7. Jameson A. Time-dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA Paper 91-1596*, 1991.
8. Ni RH. A multiple-grid scheme for solving the Euler equations. *AIAA Journal* 1982; **20**(11):1565–1571.
9. Arnone A, Liou M-S, Povinelli LA. Multigrid time-accurate integration of Navier–Stokes equations. *AIAA Paper 93-3361*, 1993.
10. Mavriplis DJ. Three-dimensional multigrid for the Euler equations. *AIAA Journal* 1992; **30**:1753–1761.
11. 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.
12. 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.
13. 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.
14. 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.
15. 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.
16. Boniface JC, Sides J. Extension and improvement of existing Euler code of ONERA for multibladed rotors in Hover. *HELISHAPE Technical Report*, 1995.
17. 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.
18. Sankar LN, Wake BE, Lekoudis SG. Computation of rotor blade flows using the Euler equations. *AIAA Journal of Aircraft* 1986; **23**(7):582–588.
19. Strawn RC, Barth TJ. A finite-volume Euler solver for computing rotary-wing aerodynamics on unstructured meshes. *Proceedings of the 48th Annual American Helicopter Society Forum*, Washington, D.C., June 1992.
20. Allen CB. The effect of grid topology and density on inviscid hovering rotor solutions. *I. Mech. E. Journal of Aerospace Engineering, Part G7* 1999.
21. 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.
22. 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.
23. 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.
24. Van-Leer B. Flux-vector splitting for the Euler equations. *Lecture Notes in Physics* 1982; **170**:507–512.
25. Williams AL, Fiddes SP. Solution of the 2D unsteady Euler equations on a structured moving grid. *Bristol University Aero. Eng. Dept. Report No. 453*, 1992.

26. Allen CB. Central-difference and upwind-biased schemes for steady and unsteady Euler aerofoil computations. *Aeronautical Journal* 1995; **99**:52–62.
27. Allen CB. The reduction of numerical entropy generated by unsteady shockwaves. *Aeronautical Journal* 1997; **101**(1001):9–16.
28. Allen CB. Grid adaptation for unsteady flow computations. *I. Mech. E. Journal of Aerospace Engineering, Part G4* 1997; **211**:237–250.
29. Hounjet MHL, Allen CB, Vigevano L, Gasparini L, Pagano A. GEROS: A European grid generator for rotorcraft simulation methods. *Presented at 6th International Conference on Numerical Grid Generation in Computational Field Simulations*, London, July 1998. Proceedings, pp. 813–822.
30. D'Alascio A, Dubuc L, Peshkin D, Vigevano L, Allen CB, Pagano A, Salvatore F, Boniface J-C, Hounjet MHL, Kroll N, Scholl E, Kokkalis A, Righi M. First Results of the EROS European Unsteady Euler Code on Overlapping Grids. *Presented at ECCOMAS 98 Conference*, Athens, September 1998.
31. 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.
32. Allen CB. CHIMERA volume grid generation within the EROS code. *I. Mech. E. Journal of Aerospace Engineering, Part G* 2000.
33. Parpia IH. Van-Leer flux-vector splitting in moving coordinates. *AIAA Journal* 1988; **26**:113–115.
34. Obayashi S. Freestream capturing for moving coordinates in three dimensions. *AIAA Journal* 1992; **30**(4):1125–1128.
35. Anderson WK, Thomas JL, Van-Leer B. Comparison of finite volume flux vector splittings for the Euler equations. *AIAA Journal* 1986; **24**:1453–1460.
36. 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.
37. Turkel E, Van-Leer B. Runge–Kutta methods for partial differential equations. *ICASE Report*, 1983.
38. Gordon WJ, Hall CA. Construction of curvilinear coordinate systems and applications of mesh generation. *International Journal of Numerical Methods in Engineering* 1973; **7**:461–477.
39. Eriksson LE. Generation of boundary-conforming grids around wing-body configurations using transfinite interpolation. *AIAA Journal* 1982; **20**(10):1313–1320.
40. 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.
41. Caradonna FX, Tung C. Experimental and analytical studies of a model helicopter rotor in Hover. *NASA TM-81232*, September 1981.
42. Grasso F, Marini M. Multigrid techniques for hypersonic viscous flows. *AIAA Journal* 1993; **31**:1729–1731.
43. Grasso F, Marini M. Solutions of hypersonic viscous flows with total variation diminishing multigrid techniques. *Computers and Fluids* 1995; **24**:571–592.
44. Turkel E. Preconditioned methods for solving the incompressible and low speed compressible equations. *Journal of Computation Physics* 1987; **72**:277–298.
45. Van Leer B, *et al*. Characteristic time-stepping or local preconditioning of the Euler equations. *AIAA Paper 92-1552*, June 1991.