AIAA JOURNAL
Vol. 36, No. 8, August 1998

Solution of the Unsteady Euler Equations Using an Implicit Dual-Time Method

L. Dubuc,* F. Cantariti,* M. Woodgate,* B. Gribben,† K. J. Badcock,‡ and B. E. Richards§
University of Glasgow, Glasgow G12 8QQ, Scotland, United Kingdom

An unfactored implicit time-marching method for the solution of the unsteady two-dimensional Euler equations on deforming grids is described. The present work is placed into a multiblock framework and fits into the development of a generally applicable parallel multiblock flow solver. The convective terms are discretized using an upwind total variation diminishing scheme, whereas the unsteady governing equations are discretized using an implicit dual-time approach. The large sparse linear system arising from the implicit time discretization at each pseudotime step is solved efficiently by using a conjugate-gradient-type method with a preconditioning based on a block incomplete lower-upper factorization. Results are shown for a series of pitching airfoil test cases selected from the AGARD aeroelastic configurations for the NACA 0012 airfoil. Comparisons with experimental data and previous published results are presented. The efficiency of the method is demonstrated by looking at the effect of a number of numerical parameters, such as the conjugate gradient tolerance and the size of the global time step and by carrying out a grid refinement study. Finally, a demonstration test case for the Williams airfoil (Williams, B. R., "An Exact Test Case for the Plane Potential Flow About Two Adjacent Lifting Aerofoils," National Physical Lab., Aeronautical Research Council, Research Memorandum 3717, London, 1973) with an oscillating flap is presented, highlighting the capability of the grid deformation technique.

Nomenclature

$\boldsymbol{A}$	= left-hand-side matrix of the linear system
$\boldsymbol{b}$	= right-hand-side vector of the linear system
$\boldsymbol{C}$	= preconditioning matrix
C_l	= lift coefficient
C_m	= pitching moment coefficient
C_n	= normal force coefficient
C_p	= pressure coefficient
c	= airfoil chord
E	= specific total energy
$\boldsymbol{F}, \boldsymbol{G}$	= convective fluxes
k	= reduced frequency
M_∞	= freestream Mach number
p	= static pressure
$\boldsymbol{R}_{i,j}$	= flux residual
$\boldsymbol{R}^*_{i,j}$	= unsteady residual
t	= time
U, V	= contravariant velocities
U_∞	= freestream velocity
u, v	= Cartesian velocity components
$V_{i,j}$	= control volume
$\boldsymbol{W}$	= vector of conservative variables
$\boldsymbol{x}$	= solution vector of the linear system
x, y	= Cartesian coordinates
x_m	= moment center
x_t, y_t	= grid speeds
α	= angle of attack
α_m	= mean angle
α_0	= amplitude of oscillation
Δt	= real time step
Δt^*	= pseudotime step
ρ	= density
ω	= angular frequency

Subscripts

i, j	= computational cell
∞	= freestream conditions

Superscripts

m	= time level of the approximation in pseudotime
n	= time level of the approximation in real time

I. Introduction

UNSTEADY flows are at the center of many engineering problems, including helicopter aerodynamics, acoustics, fluid–structure interaction, and free surface problems. Owing to the cost of experiments, computational tools have an important role to play in this area. However, the application of computational fluid dynamics (CFD) methods is very complex, and its routine use for three-dimensional unsteady problems, such as the rotor environment or full aircraft simulation, is unlikely in the near future. Nevertheless, progress is being made in this direction, and studies of three-dimensional flows governed by the Reynolds-averaged Navier–Stokes equations are being reported in the literature with increasing frequency.

An important issue for a practical CFD code is the efficiency of the numerical method, especially for unsteady flow computations where a large number of time steps may be required. Explicit methods are simple to implement, but for many problems the allowable time step for stability is much smaller than that required for accuracy. For unsteady flows, where most of the acceleration techniques developed to speed up steady flow calculations cannot be used as they destroy time accuracy, this results in a very large number of time steps. Implicit schemes, on the other hand, allow much larger time steps, but the work required per time step may be large, particularly in three dimensions.

With the introduction by Jameson[1] of a dual-time approach, the use of explicit methods for unsteady flow computations has regained popularity. The method uses an implicit real-time discretization but at each real time step marches the solution in pseudotime to a steady state through an explicit time-marching scheme. The acceleration techniques of steady flow calculations can then be used because the marching process is done in pseudotime. This approach has led to considerable improvements compared with previous explicit methods based on single time discretization. Although originally developed to accelerate standard explicit time-marching schemes,

Received July 7, 1997; revision received April 16, 1998; accepted for publication April 27, 1998.

*Research Assistant, Department of Aerospace Engineering.
†Research Student, Department of Aerospace Engineering.
‡Lecturer, Department of Aerospace Engineering.
§Mechan Professor, Department of Aerospace Engineering. Associate Fellow AIAA.

the dual-time method can be used in conjunction with any efficient steady-state code.

The general approach adopted by the CFD group at the University of Glasgow is to use high-order upwind differencing schemes to provide accuracy and robustness and to use implicit methods to provide efficiency. Benefiting from the general experience gained through the development of two-dimensional and three-dimensional implicit steady and unsteady codes,[2–5] effort is now being put into the development of methods that fit into a generally applicable parallel multiblock flow solver.[5,6]

The large sparse linear system that arises from the implicit time discretization is solved efficiently by using a conjugate-gradient-type method. The preconditioning strategy is a crucial factor in the efficiency and the success of these methods. In Refs. 3 and 5, an alternating-direction implicit factorization was used as a preconditioner for the solution of the linear system at each iteration. This approach proved successful on a number of steady and unsteady airfoil test problems.[3,5,7] Following the dual-time approach, this implicit method is used in the current paper to solve for the pseudotime steady state. This provides several improvements when compared with previous work using the single time approach,[3] as will be described later. To enhance the parallel efficiency of the method, which depends largely on the interblock communication required by the solution algorithm, we use here a block incomplete lower-upper (BILU) factorization for the preconditioner that allows decoupling between the blocks for the solution of the linear system. Previous results obtained with the parallel multiblock code have been reported in Refs. 5, 6, and 8 for some standard airfoil test cases and for some demonstration cases for complex two-dimensional geometries. Encouraging results were obtained for all cases.

We propose here to demonstrate the capability of the method for a series of unsteady problems, including a demonstration test case for the Williams airfoil with an oscillating flap. The grid treatment is described in Refs. 9 and 10.

II. Two-Dimensional Governing Equations

The two-dimensional Euler equations in Cartesian coordinates (x, y) can be written in nondimensional conservative form as

$$\frac{\partial \boldsymbol{W}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} + \frac{\partial \boldsymbol{G}}{\partial y} = 0 \tag{1}$$

where

$$\boldsymbol{W} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}$$

and

$$\boldsymbol{F} = \begin{bmatrix} \rho U \\ \rho u U + p \\ \rho v U \\ U(\rho E + p) + x_t p \end{bmatrix}, \qquad \boldsymbol{G} = \begin{bmatrix} \rho V \\ \rho u V \\ \rho v V + p \\ V(\rho E + p) + y_t p \end{bmatrix}$$

In the preceding equations, the terms U and V are defined by

$$U = u - x_t, \qquad V = v - y_t$$

The equations are discretized using a cell-centered finite volume method that transforms the partial differential equations into a set of ordinary differential equations that can be written as

$$\frac{\partial}{\partial t}(V_{i,j}\boldsymbol{W}_{i,j}) + \boldsymbol{R}_{i,j}(\boldsymbol{W}) = 0 \tag{2}$$

where $\boldsymbol{R}_{i,j}(\boldsymbol{W})$ is the flux residual for the cell (i, j) that contains all of the terms arising from the spatial discretization. The convective fluxes are discretized using the Osher upwind flux difference splitting scheme together with a MUSCL variable extrapolation to provide second- or third-order accuracy in space. The Von Albada limiter is used to ensure monotonic solutions around shock waves. Finally, the far-field boundary conditions are treated by the characteristic boundary method based on the Riemann invariants.

III. Implicit Dual-Time Method

The original implicit dual-time approach was introduced by Jameson[1] and allows an implicit discretization to be used in real time with the solution at the new time level being obtained through an iteration in pseudotime. This permits the acceleration techniques of steady flow, such as local time stepping, residual smoothing, and multigrid or implicit methods, to be used to update the solution. This also allows the real time step to be chosen based on accuracy requirements alone without stability restrictions. The dual-time method can be obtained from a steady Euler solver by means of a few modifications and is therefore very attractive when extending steady flow methodologies to unsteady problems. The method has been used mostly in conjunction with multigrid algorithms and applied to both the Euler[1,11] and Navier–Stokes[12] equations, on structured and unstructured grids, using either rigid grids or general mesh deformation algorithms.[13]

In this work, we propose to apply the dual-time method in conjunction with an implicit time-stepping method for the solution of the steady-state problem in pseudotime. In Ref. 3, implicit time stepping was used without pseudotime iterations. However, the solutions obtained by this method, which uses a single time discretization, were only first-order accurate in time. By using a dual-time approach with pseudotime iterations, it is possible to improve the time accuracy to second order. Time accuracy can be further enhanced by using a third-order implicit time discretization rather than the second-order discretization described later, as shown by Ref. 14. However, this is achieved at the expense of additional storage for an extra time level of accuracy, which is not always necessary considering the good accuracy obtained in general with second-order time discretization.

We consider here the unsteady governing equations written in discrete form as shown in Eq. (2). This set of equations is then discretized in time by using a fully implicit time discretization (in real time) to give

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(V_{i,j}^{n+1}\boldsymbol{W}_{i,j}^{n+1}\right) + \boldsymbol{R}_{i,j}(\boldsymbol{W}^{n+1}) = 0 \tag{3}$$

where the superscript $n+1$ denotes the time level $(n+1)\Delta t$ of the approximation. Following Jameson,[1] the time derivative is approximated by a second-order backward difference discretization, and so Eq. (3) becomes

$$\frac{3V_{i,j}^{n+1}\boldsymbol{W}_{i,j}^{n+1} - 4V_{i,j}^{n}\boldsymbol{W}_{i,j}^{n} + V_{i,j}^{n-1}\boldsymbol{W}_{i,j}^{n-1}}{2\Delta t} + \boldsymbol{R}_{i,j}(\boldsymbol{W}^{n+1}) = 0 \tag{4}$$

This equation for $\boldsymbol{W}_{i,j}^{n+1}$ is nonlinear and therefore cannot be solved analytically. At this stage, it is convenient to define a new residual $\boldsymbol{R}^*$, referred to as the unsteady residual, by

$$\boldsymbol{R}_{i,j}^*(\boldsymbol{W}^{n+1}) = \frac{3V_{i,j}^{n+1}\boldsymbol{W}_{i,j}^{n+1} - 4V_{i,j}^{n}\boldsymbol{W}_{i,j}^{n} + V_{i,j}^{n-1}\boldsymbol{W}_{i,j}^{n-1}}{2\Delta t} + \boldsymbol{R}_{i,j}(\boldsymbol{W}^{n+1}) = 0 \tag{5}$$

This new equation can be seen as the solution of a steady-state problem that can be solved with a time-marching method by introducing a derivative with respect to a pseudotime t^*:

$$\frac{\partial \boldsymbol{W}_{i,j}^{n+1}}{\partial t^*} + \frac{1}{V_{i,j}^{n+1}}\boldsymbol{R}_{i,j}^*(\boldsymbol{W}^{n+1}) = 0 \tag{6}$$

The steady-state solution to Eq. (6) satisfies

$$\frac{\partial \boldsymbol{W}_{i,j}^{n+1}}{\partial t^*} = 0 \tag{7}$$

which means it also satisfies $\boldsymbol{R}_{i,j}^*(\boldsymbol{W}^{n+1}) = 0$ and hence is also the solution of the unsteady equation (5). The pseudotime problem can be solved by using any time-marching method designed to solve steady-state problems, utilizing any of the standard acceleration techniques.

IV. Implicit Unfactored Method

The steady-state problem in pseudotime to be solved at each global time step is rewritten as

$$\frac{\partial \boldsymbol{W}_{i,j}}{\partial t^*} + \frac{1}{V_{i,j}}\boldsymbol{R}^*_{i,j}(\boldsymbol{W}) = 0 \tag{8}$$

where t^* is the pseudotime, $\boldsymbol{W}_{i,j}$ is the approximation to $\boldsymbol{W}^{n+1}_{i,j}$, and $\boldsymbol{R}^*_{i,j}(\boldsymbol{W})$ is defined by Eq. (5) and contains the terms arising from the spatial discretization of the fluxes and those arising from the implicit discretization of the time derivative:

$$\boldsymbol{R}^*_{i,j}(\boldsymbol{W}) = \boldsymbol{R}_{i,j}(\boldsymbol{W}) + \frac{3V_{i,j}\boldsymbol{W}_{i,j}}{2\Delta t} - \frac{2V^n_{i,j}\boldsymbol{W}^n_{i,j}}{\Delta t} + \frac{V^{n-1}_{i,j}\boldsymbol{W}^{n-1}_{i,j}}{2\Delta t} \tag{9}$$

Using an implicit time discretization on the pseudotime t^*, Eq. (8) is discretized as

$$\frac{\Delta \boldsymbol{W}_{i,j}}{\Delta t^*} = \frac{\boldsymbol{W}^{m+1}_{i,j} - \boldsymbol{W}^m_{i,j}}{\Delta t^*} = -\frac{1}{V_{i,j}}\boldsymbol{R}^*_{i,j}(\boldsymbol{W}^{m+1}) \tag{10}$$

where the superscript $m+1$ denotes the time level $(m+1)\Delta t^*$. In this equation, the flux residual on the right-hand side is evaluated at the new time level $(m+1)$ and is therefore expressed in terms of the unknown solution at this new time level. To get an expression in terms of known quantities only, i.e., solution at the previous time levels, the term $\boldsymbol{R}^*(\boldsymbol{W}^{m+1})$ is linearized with respect to the time variable t^*:

$$\begin{aligned}\boldsymbol{R}^*(\boldsymbol{W}^{m+1}) &\approx \boldsymbol{R}^*(\boldsymbol{W}^m) + \frac{\partial \boldsymbol{R}^*}{\partial t}\Delta t^* \\ &\approx \boldsymbol{R}^*(\boldsymbol{W}^m) + \frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{W}}\frac{\partial \boldsymbol{W}}{\partial t^*}\Delta t^* \\ &\approx \boldsymbol{R}^*(\boldsymbol{W}^m) + \frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{W}}\Delta \boldsymbol{W}\end{aligned}$$

where $\Delta \boldsymbol{W} = \boldsymbol{W}^{m+1} - \boldsymbol{W}^m$ and, according to Eq. (9),

$$\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{W}} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{W}} + \frac{3V}{2\Delta t}\boldsymbol{I} \tag{11}$$

Substituting the preceding in Eq. (10), one implicit pseudotime step can be written as

$$\left[\left(\frac{V}{\Delta t^*} + \frac{3V}{2\Delta t}\right)\boldsymbol{I} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{W}}\right]\Delta \boldsymbol{W} = -\boldsymbol{R}^*(\boldsymbol{W}^m) \tag{12}$$

Equation (12) is the unfactored linear system that arises from the implicit time discretization in pseudotime. This linear system is similar to that obtained for a standard steady-state problem (without dual time) and can therefore be solved using the same method as that used for steady-state problems. The difference between the linear system given by Eq. (12) and that associated with a standard steady-state problem lies first in the definition of the flux residual on the right-hand side of Eq. (12), which is now given by Eq. (9), and second in the presence of the extra term $3V/2\Delta t$ on the diagonal of the Jacobian matrix on the left-hand side of Eq. (12).

If the equation for the updates is written in the form

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{13}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are given, respectively, by

$$\boldsymbol{A} = \left[\left(\frac{V}{\Delta t^*} + \frac{3V}{2\Delta t}\right)\boldsymbol{I} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{W}}\right] \tag{14}$$

$$\boldsymbol{b} = -\boldsymbol{R}^*(\boldsymbol{W}) \tag{15}$$

then the solution of the linear system given by Eq. (13) requires the inversion of the matrix $\boldsymbol{A}$.

The matrix $\boldsymbol{A}$ consists of 4×4 blocks. Its sparsity pattern is determined by the method employed for the discretization of the spatial terms in the model equations and is therefore known a priori. In CFD applications, the matrix $\boldsymbol{A}$ is generally sparse, and many of the block elements are zero matrices.

It is generally cheaper to use an iterative solution, such as a preconditioned conjugate gradient method, rather than a direct method to solve a sparse linear system. A conjugate-gradient-type method can be used to solve the preconditioned system

$$\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{C}^{-1}\boldsymbol{b} \tag{16}$$

where $\boldsymbol{C}^{-1} \approx \boldsymbol{A}^{-1}$ is the preconditioning matrix. If $\boldsymbol{C}^{-1}$ closely approximates $\boldsymbol{A}^{-1}$, then the conjugate gradient method will converge quickly due to the modified spectrum of the system. However, the effort and the cost to calculate $\boldsymbol{C}^{-1}$ must be minimized for overall efficiency.

The preconditioning is based here on a BILU factorization of $\boldsymbol{A}$ into matrices $\boldsymbol{LU}$, where $\boldsymbol{L}$ and $\boldsymbol{U}$ are block lower and upper triangular, respectively, and partitioned in blocks consistent with the sparsity pattern of $\boldsymbol{A}$. The method is closely related to the full Gauss factorization method where the matrices are constructed recursively by applying a two-by-two block partitioning of the current matrix where elimination is to take place and then computing an approximation of the Schur components. The idea of incomplete factorization methods is to reject the fill-in entries that occur in positions outside a chosen sparsity pattern. In this way, the factorization becomes approximate, or incomplete, and must therefore be coupled with an iterative solution method; i.e., it is used as a preconditioner for such a method. To minimize the solution coupling between the blocks in a multiblock grid system, the BILU factorization is decoupled between the blocks.[6,15]

In the present work, we use the generalized conjugate gradient (GCG) method to solve the system (16) as described in Ref. 16. The GCG method is an iterative solution method that solves a linear system by minimizing a quadratic functional that measures the linear solution error. The minimization takes place on a sequence of subspaces of increasing dimension that are constructed recursively by adding a new basis vector to the previous subspace. The GCG method has been written to minimize the number of matrix-vector and preconditioner-vector multiplications at the expense of some extra vector operations. At every iteration of the GCG method, the solution of the system (16) requires application of the preconditioner. This is achieved by solving two triangular systems with the matrices $\boldsymbol{L}$ and $\boldsymbol{U}$, once by a forward substitution and once by a backward substitution.

For the high-order spatial reconstruction schemes, the left-hand-side matrix is constructed using terms associated with a first-order spatial scheme only, while using a higher-order right-hand side. This not only has the advantage of reducing the storage from nine blocks per cell down to five blocks when high-order MUSCL extrapolation is used, but also it has the effect of greatly reducing the number of GCG steps per iteration and thereby increases the overall efficiency of the method.[17]

V. Results for NACA 0012 Airfoil

A. Description of the Test Cases

In this report, we present the results obtained using the dual-time method for a series of standard pitching airfoil test cases selected from the AGARD database.[18] The different test cases are listed in Table 1. For all of these test cases, the periodic motion of the airfoil is defined by the angle of attack as a function of time as

$$\alpha(t) = \alpha_m + \alpha_0 \sin(\omega t)$$

where ω is related to the reduced frequency k by

$$k = \omega c/2U_\infty$$

For all cases, the airfoil oscillates about its quarter-chord.

Table 1 AGARD test cases for the NACA 0012 airfoil that are examined in this report

Case	M_∞	α_m, deg	α_0, deg	k	x_m
CT1	0.6	2.89	2.41	0.0808	0.273
CT2	0.6	3.16	4.59	0.0811	0.273
CT5	0.755	0.016	2.51	0.0814	0.25

All of the results given in this report for the NACA 0012 airfoil, except those related to the grid refinement study, were obtained with the same grid, a C-type Euler grid consisting of 129 × 33 grid points with 97 points on the airfoil, and a far-field boundary situated at approximately 15 chords from the airfoil surface. The average spacing for the first layer of points from the airfoil surface varies between 0.0027 chord at the leading edge and 0.010 chord at the trailing edge.

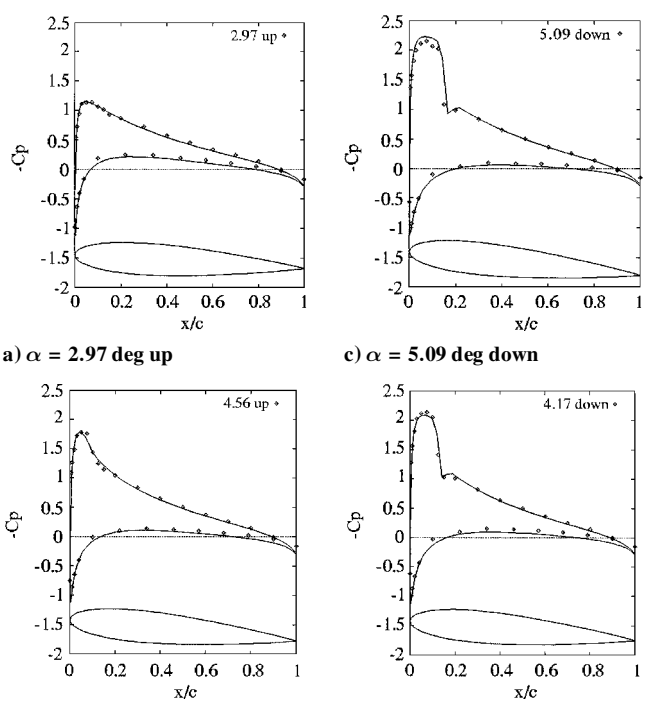
A fast deformation technique is employed here to regenerate the grid in each block at each new time step. The method is based on a modified transfinite interpolation algorithm applied to the grid displacements. This technique provides a rapid and efficient way of deforming the grid around complex geometries. It has been applied successfully to various configurations, including a multielement airfoil with an oscillating flap. A full description of the method can be found in Refs. 9 and 10.

For all unsteady calculations, an initial solution is first obtained by solving for the steady flow at the mean incidence before setting the airfoil in motion to solve the unsteady flow equations. The initial solution at each time step is obtained by linear extrapolation from the last two solutions at the previous time steps. At each global time step, the solution is marched in pseudotime until some specified tolerance for the norm of the unsteady residual is reached. Here a reduction of three orders of magnitude was found to be sufficient to achieve a converged solution at each global time step. No gain in accuracy for the solution in terms of normal force and moment coefficients was noticed when using a higher degree of convergence.

B. Results

Results are shown in terms of normal force and moment coefficient loops and instantaneous pressure distributions that are available at a number of points during the pitching cycle. For case CT5, the moment coefficient is calculated about the airfoil quarter-chord, whereas for cases CT1 and CT2 the moment coefficient is calculated about 0.273 chord. It was indicated in previous numerical studies[3,19,20] that better agreement for the moment coefficient loop was obtained by using this corrected value, suggesting a possible error in the location of the moment center quoted for the experiments.

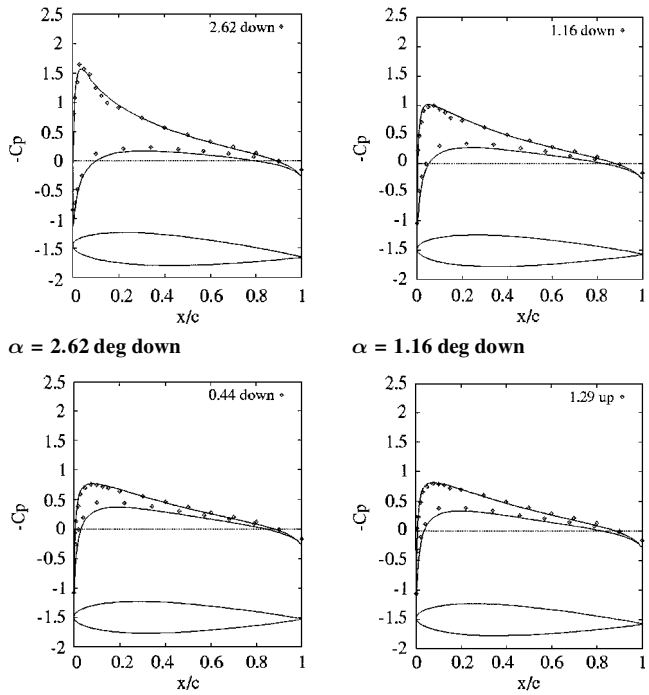
To demonstrate the good accuracy obtained by the solver for unsteady flow computations, we show in Figs. 1 and 2 the predicted pressure distributions for case CT1 at eight different angles of incidence during the cycle plotted against the measured pressure distributions as given in the AGARD report.[18] The diamonds symbols are



a) α = 2.97 deg up c) α = 5.09 deg down

b) α = 4.56 deg up d) α = 4.17 deg down

Fig. 1 Instantaneous pressure distributions for NACA 0012 airfoil.



α = 2.62 deg down α = 1.16 deg down

α = 0.44 deg down α = 1.29 deg up

Fig. 2 Instantaneous pressure distributions for NACA 0012 airfoil.

experimental data. Excellent agreement is obtained all throughout the cycle with a sharp and accurate prediction of the shock wave that develops on the upper surface of the airfoil as the incidence increases. A slight overprediction of the shock strength is observed near the maximum incidence (see Fig. 1c). Similar results were obtained for the other two cases.

Results for case CT2 are given in Fig. 3 in terms of normal force and pitching moment coefficient loops for various numbers of steps per cycle. The loops are plotted for the third cycle of the computation only. Relatively good agreement is obtained compared with the experimental values, although a slight overprediction of the normal force is observed throughout the cycle. Good agreement is obtained for the moment coefficient computed at the corrected moment center position of 0.273c. The present results compare relatively well with those given in Ref. 13 for Navier–Stokes calculations.

However, although the neglect of the viscous effects may partly explain the mismatch between prediction and experiment, it is not clear why such large discrepancies appear in the normal force coefficient loop when the pressure distributions were found to be almost in perfect agreement with the experiment. As suggested in previous numerical studies, such inconsistencies could be associated with some uncertainties in the experimental data for these test cases.[20]

Finally, results for case CT5 are given in terms of normal force and pitching moment loops for different grid sizes (Fig. 4). The flow solution for this case is characterized by the presence of a strong shock wave that develops alternatively on the upper and lower surfaces of the airfoil. Comparisons of the predicted and experimental pressure distributions for this test case have shown some discrepancies between prediction and experiment,[10] with an overprediction of the pressure jump across the shock and a slight oscillation of the pressure at the foot of the shock. Also, the results have shown that the shock was predicted slightly upstream of the experiment, which is consistent with most results (Euler or Navier–Stokes) given in previous publications.[3,14,19,21,22] However, the present results showed significant improvement compared with those obtained in Ref. 3 with a thin-layer Navier–Stokes solver.

Figure 4, which shows the normal force coefficient, indicates that the C_n is clearly underestimated all throughout the cycle. This is probably due to the overprediction of the shock strength and its location a little too far upstream, resulting in an underestimation of the pressure coefficient compared with the experiment. Some of the discrepancies can be associated with the neglect of the viscous terms. The results obtained for the C_m are relatively good, although

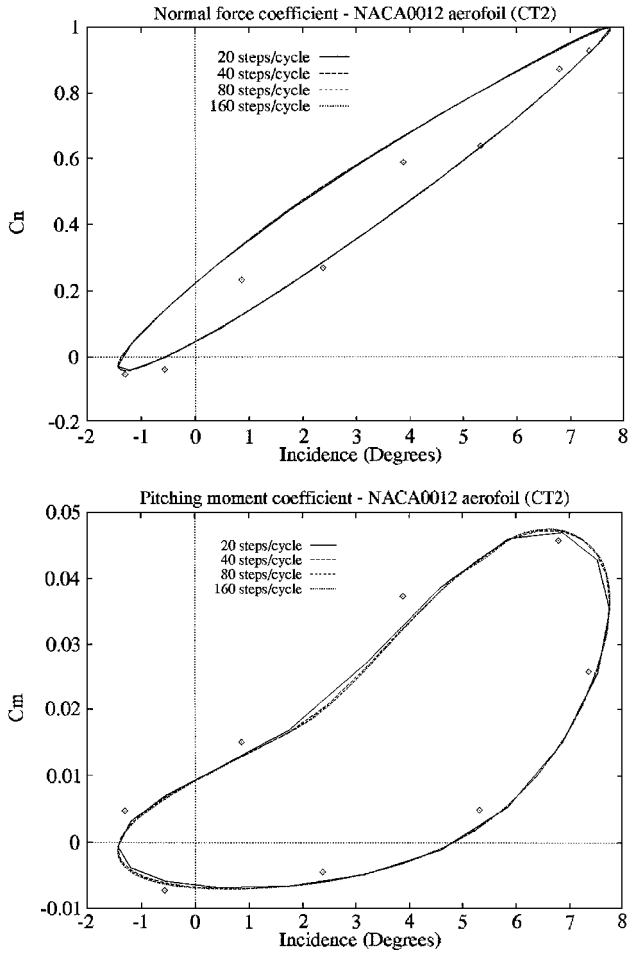


Fig. 3 Effect of the size of the time step: integrated normal force coefficient and pitching moment coefficient for case CT2.

it is clear that its value is also affected by the wrong prediction of shock location.

C. Effect of the Size of the Time Step

The size of the global time step is a numerical parameter that affects the efficiency of the conjugate gradient (CG) algorithm as well as the time accuracy of the solution. However, the dual-time method provides here a second-order discretization in time compared with the first-order discretization associated with all single time discretization methods based on time linearization. This also has the advantage of allowing larger time steps to be used. Results obtained with a single time discretization method for similar studies were given in Ref. 3. It was observed therein that, the smaller the time step, the easier the linear system is to solve at each iteration at the cost of an increased number of steps for each cycle for the solution of the flow equations.

Here we investigate the effect of the size of the time step on the efficiency and the accuracy of the flow solver. The results of this investigation are given in Table 2 for the three AGARD test cases in terms of the number of pseudotime steps and CPU time required to compute one cycle when varying the size of the time step, i.e., the number of steps per cycle.

For all cases, we note that, as the size of the global time step increases, although the average number of implicit iterations per time step increases, the overall number of implicit iterations per cycle decreases, leading to a significant reduction in the overall CPU time. Here the CPU time required to compute one cycle is expressed in terms of work units, where one work unit is the time required for one residual evaluation.

The reduction in CPU time for case CT2 is not as significant as for the other two cases. The high angles of incidence encountered during the cycle result in a linear system that is harder to solve, especially for large time steps, therefore requiring more CG iterations at each pseudotime step.

Table 2 Effect of the size of the time step on the flow solver efficiency

Case	Number of time steps per cycle	Number of pseudotime steps per cycle	Average number of pseudotime steps per real time step	CPU/cycle, work units
CT1	160	634	3.9	2643
	80	503	6.3	1740
	40	389	9.2	1649
	20	274	13.7	1250
	10	189	18.9	915
CT2	160	917	5.7	3860
	80	667	8.3	2746
	40	463	11.6	2275
	20	506	25.3	2783
CT5	160	901	5.6	3814
	80	617	7.7	3028
	40	437	10.9	2201
	20	292	14.6	1160
	10	198	19.8	956

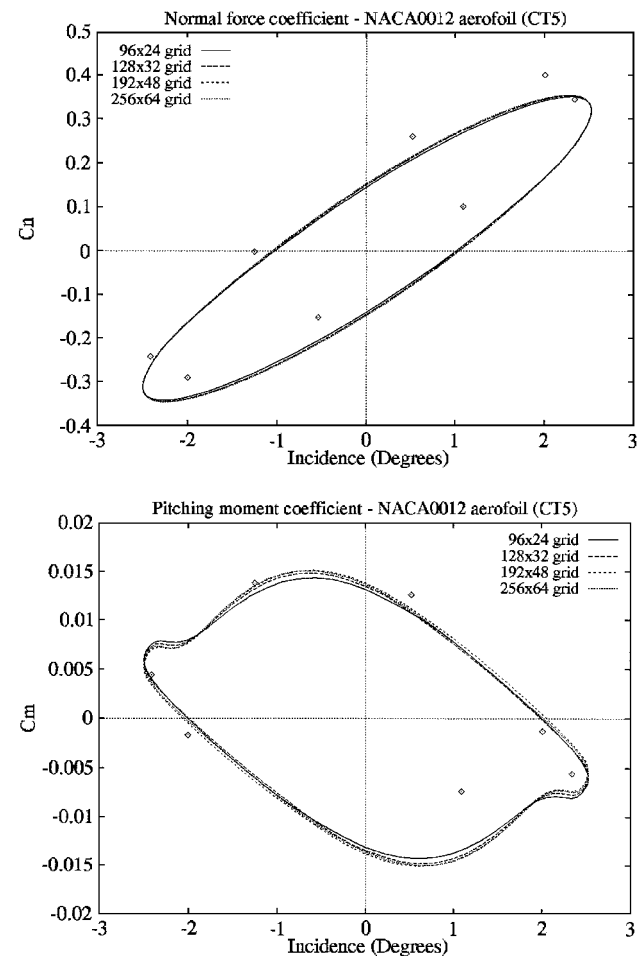


Fig. 4 Grid refinement study: integrated normal force and pitching moment coefficient for case CT5.

The results of the present method using dual time compare favorably with those given in Ref. 3 for the Approximate Factorisation-Conjugate Gradient Squared (AF-CGS) code using a single time discretization approach. For the same cases run on similar grids, the AF-CGS code required at least 800 time steps per cycle for case CT1 and at least 570 for case CT5. This was mainly due to the severe restrictions associated with the time linearization and the conditioning of the linear system. By comparison, the present method allows very large time steps to be used, resulting in the number of time steps per cycle being as small as 20 and even 10 in some cases.

The loops for the normal force and moment coefficients for case CT2 are given in Fig. 3 for a varying number of steps per cycle. The loops are plotted for the third cycle of the computation only. All three cases present similar behavior with respect to the size of

Table 3 Effect of the CG tolerance on the flow solver efficiency

Case	CGS tolerance	Number of pseudotime steps per cycle	CPU/cycle, work units
CT1	0.001	458	2235
	0.01	503	2177
	0.05	600	2361
	0.1	694	3054
CT2	0.001	620	2950
	0.01	667	2794
	0.05	772	3574
	0.1	874	3738
CT5	0.001	590	2736
	0.01	617	2584
	0.05	674	2971
	0.1	752	3035

Table 4 Grids used for the grid refinement study

Name	Size	Number of cells on airfoil surface	Number of cells along wake line	Average distance of first cells
1	96×24	72	12	0.0039–0.015
2	128×32	96	16	0.0027–0.010
3	192×48	144	24	0.0018–0.0071
4	256×64	192	32	0.0014–0.0054

the global time step, with a very good agreement for the normal force coefficient even for relatively large time steps, and a more sensitive behavior for the moment coefficient, characterized by a slight deterioration of the C_m prediction as the time step increases. However, the benefit of using a second-order discretization in time appears clearly when looking at the good accuracy obtained at large time steps.

D. Effect of the CG tolerance

Another numerical parameter that has an influence on the efficiency of the CG method is the CG tolerance, i.e., the convergence criterion used for the solution for the linear system. Table 3 shows the results obtained for different tolerances varying from 0.001 to 0.1. The results given here were obtained on a 128×32 grid for 80 steps per cycle.

For all three cases considered here, we note that a stricter tolerance results in a larger number of CG steps at each implicit iteration but an overall reduction in the number of implicit iterations per cycle, whereas a less strict tolerance requires fewer CG steps per iterations but more iterations per cycle to converge to the solution. This can be explained by the fact that larger, i.e., less strict, tolerances result in a linear system that converges quicker but presumably to a less accurate solution, leading to degraded pseudotime convergence. Looking at the CPU time required, the optimum convergence criterion for the CG algorithm is here 0.01. This value was used for all of the results presented in this report.

E. Effect of Grid Refinement

Finally, we examine the effect of mesh refinement by carrying out comparisons of the average number of implicit iterations per cycle and the CPU time per cycle and per grid point for different grid sizes. Four grids are considered here for this grid dependence study (Table 4). All four grids were extracted from the same grid by interpolating the stretching functions to preserve the good quality of the initial grid in terms of smoothness and grid line orthogonality. The spacing of the first layer of grid points from the airfoil surface increases from the leading edge to the trailing edge of the airfoil. The spacing at the leading edge and trailing edge for each grid is also given in Table 4. Also, all grids consist of three blocks. Note that the 128×32 grid, i.e., grid 2, is the grid used to obtain all of the other results given in this report.

The results are given in Table 5 and were obtained for 80 steps per cycle. As expected, the number of pseudotime iterations required

Table 5 Effect of grid refinement on flow solver efficiency

Case	Mesh	Number of pseudotime steps per cycle	CPU/cycle, work units	CPU/cycle/grid point, work units
CT1	96×24	472	1941	0.84
	128×32	503	2176	0.53
	192×48	613	2708	0.29
	256×64	638	3176	0.19
CT2	96×24	620	3065	1.33
	128×32	667	3281	0.80
	192×48	783	3489	0.38
	256×64	869	4310	0.26
CT5	96×24	546	2353	1.02
	128×32	617	2590	0.63
	192×48	779	3285	0.35
	256×64	888	4807	0.29

to compute one cycle increases as the mesh is refined, leading to a significant increase in the overall CPU time.

However, it is interesting to note that the number of pseudotime steps per cycle does not increase dramatically. Also, the CPU time per cycle and per grid point decreases as the size of the mesh increases due to a reduction in the number of CG iterations per grid point, suggesting no deterioration of the efficiency of the method on fine meshes. This represents an improvement compared with the results obtained in Ref. 3, for which it was observed that the performance of the CG method degraded as the size of the problem increased, therefore requiring smaller time steps on finer meshes for the linear solver to converge within the specified number of CG steps.

Figure 4 shows the normal force and moment obtained for case CT5 for the various grids considered here. Perfect matching is obtained with all grids for the normal force coefficient, whereas the moment coefficient exhibits a higher sensitivity to the size of the grid. However, these results suggest that no significant gain in accuracy is obtained when using fine grids and that sufficient accuracy can be obtained on a relatively coarse grid (typically 128×32) for Euler computations at similar flow conditions.

VI. Multielement Demonstration: Williams Airfoil[23]

To demonstrate the performance of the method for more complex motions, we consider here a demonstration case for the Williams airfoil (configuration B)[23] with an oscillating flap. Several test cases were considered for this configuration with various amplitudes for the flap deflection up to 14 deg. The mesh consists of 15 blocks with 11,228 grid points (10,367 mesh cells). The blocks that are allowed to deform are selected by the user by means of a graphical interface. The choice is made to retain the overall quality of the grid. For this particular problem, a minimum of 4 blocks for very small flap deflections and up to a maximum of 10 blocks for larger amplitudes (over 10 deg) were considered. It was found, in all cases, that better grids were obtained when allowing a large number of blocks to be deformed, allowing the distortion of the mesh to be reduced and spread over a larger flow region.

The flow conditions used for the steady computation about the Williams airfoil (configuration B) are a freestream Mach number of 0.58 and a zero incidence angle, which results in a supercritical flow. Previous computations of this test case[24,25] indicate a strong shock wave on the upper surface of the main airfoil situated at approximately 50% chord of the airfoil, with a local Mach number reaching 1.5 just ahead of the shock.

Comparison has been made with the steady version of the code for the Williams airfoil (configuration B)[23] and compared with previous results obtained with an analytical full potential solution and other computations. Close agreement with the full potential solution[25] was obtained, as shown in Fig. 5. The results also agree well with those of Stolcis and Johnston[24] obtained for Euler computations.

We present here the results obtained for an oscillating flap, where the periodic motion of the flap is defined by its deflection angle as

$$\alpha(t) = \alpha_m + \alpha_0 \sin(\omega t)$$

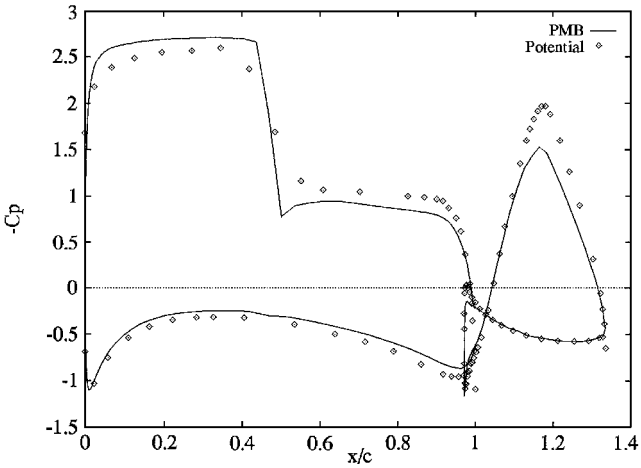


Fig. 5 Pressure coefficient for Williams airfoil (configuration B).[23]

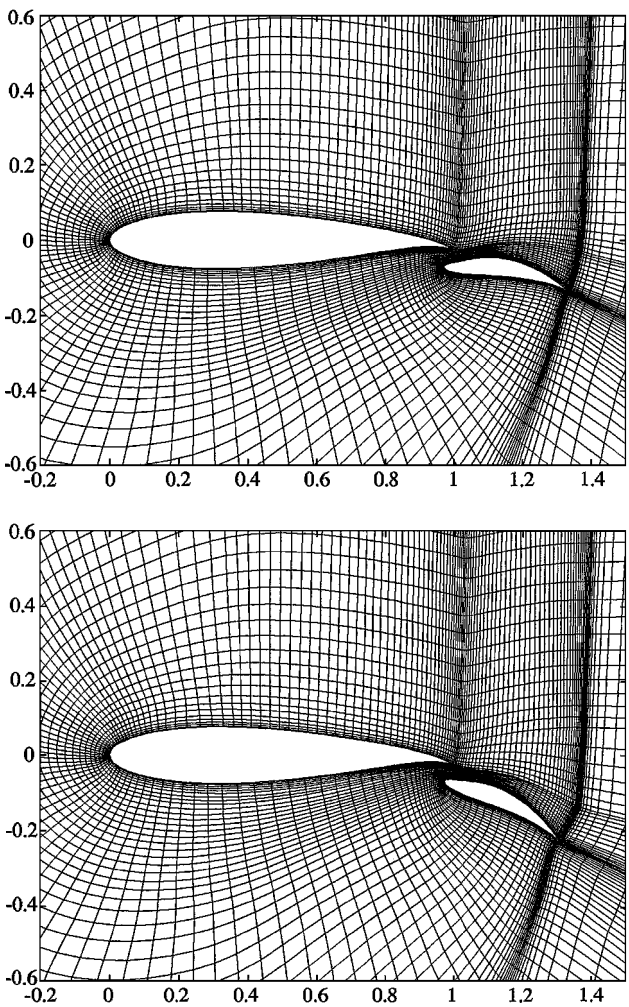


Fig. 6 Multiblock grid for the original Williams airfoil (top)[23] and 15-deg flap deflection case (bottom).

where $\alpha_m = 7$ deg is the mean deflection angle, $\alpha_0 = 7$ deg is the amplitude of the flap oscillation, and $k = \omega c/2U_\infty = 0.0814$ is the reduced frequency. The deflection of the flap is counted with respect to the position of the flap of the original Williams airfoil (configuration B), which is positive when the flap is deflected downward and negative when deflected upward. The flap is rigidly rotated about the point situated near its leading edge with coordinates $x = 0.98$ and $y = -0.07$. A view of the original and deformed grid around the flap is shown in Fig. 6.

An initial solution was first obtained by solving a steady-state problem at the mean deflection angle, and the flap was then set in motion to solve the unsteady problem for the oscillating flap. The calculations are continued until a periodic solution is obtained, usually after two or three cycles. For this test case, which involves large deformation with a flap deflection reaching 14 deg with respect to the original position of the flap, i.e., undisturbed grid, it was found that a reduction in the implicit Courant–Friedrichs–Lewy (CFL) number (down to approximately 50) was necessary to obtain the initial steady-state solution, after which the implicit CFL number was increased up to 250 for the rest of the unsteady computation.

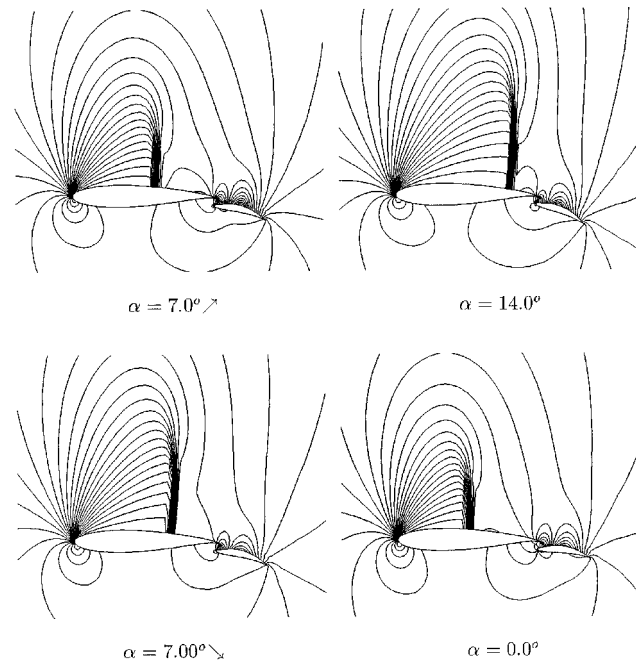


Fig. 7 Pressure contours for Williams airfoil[23] with oscillating flap.

Figure 7 shows some pressure contours at four different angles, showing clearly the displacement of the strong shock wave on the upper surface of the main airfoil. As the flap is deflected downward, the shock wave located on the upper surface of the main airfoil moves downstream and the pressure plateau ahead of the shock increases slightly, resulting in a significant increase in lift. At maximum flap deflection (14 deg), the shock is located at approximately 70% of the main airfoil chord, whereas at minimum deflection (corresponding to a zero deflection angle), the shock is situated at about 50% chord. It is also interesting to note that the extra lift generated by the flap itself increases significantly as the flap moves downward. For this particular test case, the lift coefficient varies about $\pm 20\%$ during the cycle.

VII. Conclusions

The capability of an unfactored implicit method for the solution of the two-dimensional Euler equations on moving meshes has been demonstrated. The results obtained confirm the applicability of the current time-stepping strategy, which combines a dual-time approach to discretize the unsteady equations with an implicit time-stepping method for the solution of the steady-state problem in pseudotime. The method is based on a general conjugate gradient solution of the linear system with a BILU factorization used for the preconditioner. This preconditioning strategy allows decoupling of the blocks for the solution of the linear system, resulting in a very efficient parallel implementation of the method.

The method was incorporated within a multiblock environment and was used in conjunction with a general moving grid technique that allows rapid and efficient deformation of the grid in the case of complex geometries. Very encouraging results were obtained for a demonstration test case consisting of a two-element airfoil with an oscillating flap. The new strategy employed here, based on a transfinite interpolation of the grid displacements, is not restricted to rigid-body motion and simple geometries, and it therefore provides a useful tool for computing aeroelastic problems for complex geometries.

Results were presented for several pitching airfoil flows (AGARD test cases), and good agreement was noted with both experiment and previous computations reported in the literature. Improved performance was obtained, due not only to the improved performance of the CG solver and its preconditioner but also to the use of the dual-time approach for the solution of the unsteady equations.

A grid refinement study was carried out and showed no deterioration in the efficiency of the method as the size of the problem is increased. Large time steps can be used, corresponding to a minimum of 10 or 20 steps per cycle for some cases.

The validation of the present method for unsteady Euler computations represents only an intermediate step in the development of a general flow solver capable of tackling turbulent viscous flows for complex geometries, and an immediate area of future work is the implementation of a two-equation $k-\omega$ turbulence model into the dual-time framework.

Acknowledgments

This work has been partially sponsored by British Aerospace Contract SPO S 0104458, Defence Evaluation and Research Agency (DERA) Contract FRN1c/407, Engineering and Physical Sciences Research Council/Ministry of Defence Grant GR/K55455, and Office of Science and Technology LINK Award Grant DERA ASF/2351U.

References

[1]Jameson, A., "Time Dependent Calculations Using Multigrid with Applications to Unsteady Flows Past Airfoils and Wings," AIAA Paper 91-1596, June 1991.

[2]Badcock, K. J., Xu, X., Dubuc, L., and Richards, B. E., "Preconditioners for High Speed Flows in Aerospace Engineering," *Numerical Methods for Fluid Dynamics V*, Oxford Univ. Press, Oxford, England, UK, 1996, pp. 287–294.

[3]Badcock, K. J., and Gaitonde, A. L., "An Unfactored Implicit Moving Mesh Method for the Two-Dimensional Unsteady Navier–Stokes Equations," *International Journal for Numerical Methods in Fluids*, Vol. 23, No. 6, 1996, pp. 607–631.

[4]Badcock, K. J., Richards, B. E., and Woodgate, M. A., "Factored-Unfactored Method for 3D Transonic Flows," *Computational Fluid Dynamics '96*, edited by J.-A. Desideri, P. Le Tallec, E. Onate, J. Periaux, and E. Stein, Wiley, New York, 1996, pp. 605–610.

[5]Dubuc, L., Badcock, K. J., Woodgate, M. A., and Richards, B. E., "Implicit Navier–Stokes Simulations of Unsteady Flows," *Unsteady Aerodynamics*, Royal Aeronautical Society, London, 1996, pp. 11.1–11.9.

[6]Badcock, K. J., Porter, S., and Richards, B. E., "Unfactored Multiblock Methods—Part 1: Initial Method Development," Univ. of Glasgow, Aero Rept. 9511, Glasgow, Scotland, UK, Sept. 1995.

[7]Badcock, K. J., Glover, I. C., and Richards, B. E., "Convergence Acceleration for Viscous Airfoil Flows Using an Unfactored Method," *Computational Fluid Dynamics '94*, Wiley, Chichester, England, UK, 1994, pp. 333–341.

[8]Gribben, B. J., "Progress Report: Application of the Multiblock Method in Computational Aerodynamics," Univ. of Glasgow, Aero Rept. 9621, Glasgow, Scotland, UK, Dec. 1996.

[9]Dubuc, L., Cantariti, F., Woodgate, M. A., Gribben, B., Badcock, K. J., and Richards, B. E., "Solution of the Euler Unsteady Equations Using Deforming Grids," Univ. of Glasgow, Aero Rept. 9704, Glasgow, Scotland, UK, Jan. 1997.

[10]Dubuc, L., Cantariti, F., Woodgate, M., Gribben, B. J., Badcock, K. J., and Richards, B. E., "Solution of the Euler Unsteady Equations Using Deforming Grids," Univ. of Glasgow, Aero Rept. 9704, Glasgow, Scotland, UK, Feb. 1997.

[11]Belov, A. A., Martinelli, L., and Jameson, A., "A Novel Fully Implicit Multigrid Driven Algorithm for Unsteady Incompressible Flow Calculations," *Computational Fluid Dynamics '94*, Wiley, Chichester, England, UK, 1994, pp. 663–670.

[12]Arnone, A., Liou, M.-S., and Povinelli, L. A., "Multigrid Time Accurate Integration of Navier–Stokes Equations," AIAA Paper 93-3361, July 1993.

[13]Gaitonde, A. L., "A Dual-Time Method for the Solution of the 2D Unsteady Navier–Stokes Equations on Structured Moving Meshes," Dept. of Aerospace Engineering, Rept. 716, Univ. of Bristol, Bristol, England, UK, May 1995.

[14]Heinrich, R., Pahlke, K., and Bleecke, H., "A Three Dimensional Dual-Time-Stepping Method for the Solution of the Unsteady Navier–Stokes Equations," *Unsteady Aerodynamics*, Royal Aeronautical Society, London, 1996, pp. 5.1–5.12.

[15]Badcock, K. J., McMillan, W., Woodgate, M. A., Gribben, B. J., Porter, S., and Richards, B. E., "Integration of an Implicit Multiblock Code into a Workstation Cluster Environment," *Parallel Computational Fluid Dynamics: Algorithms and Results Using Advanced Computers*, edited by P. Schiano, A. Ecer, J. Periaux, and N. Satofuka, Elsevier, Amsterdam, 1996, pp. 408–415.

[16]Axelsson, O., *Iterative Solution Methods*, Cambridge Univ. Press, Cambridge, England, UK, 1994.

[17]Cantariti, F., Dubuc, L., Gribben, B. J., Woodgate, M., Badcock, K. J., and Richards, B. E., "Approximate Jacobians for the Euler and Navier–Stokes Equations," Univ. of Glasgow, Aero Rept. 9705, Glasgow, Scotland, UK, Jan. 1997.

[18]"Compendium of Unsteady Aerodynamic Measurements," AGARD R-702, 1982.

[19]Allen, C. B., "Central-Difference and Upwind-Biased Schemes for Steady and Unsteady Euler Aerofoil Computations," *Aeronautical Journal*, Vol. 99, Feb. 1995, pp. 52–62.

[20]Gaitonde, A. L., "A Dual-Time Method for the Solution of the Unsteady Euler Equations," *Aeronautical Journal*, Vol. 98, Oct. 1994, pp. 283–291.

[21]Allen, C. B., "Adaption by Grid Motion for Unsteady Euler Aerofoil Flows," *77th Fluid Dynamics Panel Meeting and Symposium, Progress and Challenges in CFD Methods and Algorithms* (Seville, Spain), AGARD, 1995, pp. 36.1–36.10 (AGARD Paper 35).

[22]Venkatakrishnan, V., and Mavriplis, D. J., "Implicit Method for the Computation of Unsteady Flows on Unstructured Grids," AIAA Paper 95-1705, June 1995.

[23]Williams, B. R., "An Exact Test Case for the Plane Potential Flow About Two Adjacent Lifting Aerofoils," National Physical Lab., Aeronautical Research Council, Research Memorandum 3717, London, 1973.

[24]Stolcis, L., and Johnston, L. J., "Solution of the Euler Equations on Unstructured Grids for Two-Dimensional Compressible Flow," *Aeronautical Journal*, Vol. 94, June/July 1990, pp. 181–195.

[25]Suddhoo, A., and Hall, I. M., "Inviscid Compressible Flow Past a Multiple Element Aerofoil," AGARD CP-365, 1984.

A. Plotkin
Associate Editor