AIAA 2003-0072

Implementation of Preconditioned Dual-Time Procedures in OVERFLOW

Shishir A. Pandya *
NASA Ames Research Center
Moffett Field, CA 94035

Sankaran Venkateswaran †
University of Tennessee
Tullahoma, TN 37388

Thomas H. Pulliam ‡
NASA Ames Research Center
Moffett Field, CA 94035

NAS-03-003

Abstract

Preconditioning methods have become the method of choice for the solution of flowfields involving the simultaneous presence of low Mach number flow and transonic flow regions. It is well known that these methods are important for insuring accurate numerical discretization as well as convergence efficiency over various operating conditions such as low Mach number, low Reynolds number and high Strouhal numbers. For unsteady problems, the preconditioning is introduced within a dual-time framework wherein the physical time-derivatives are used to march the unsteady equations and the preconditioned time-derivatives are used for purposes of numerical discretization and iterative solution. In this paper, we describe the implementation of the preconditioned dual-time methodology in the OVERFLOW code. To demonstrate the performance of the method, we employ both simple and practical unsteady flowfields, including vortex propagation in a low Mach number flow, flowfield of an impulsively started plate (Stokes' first problem) and a cylindrical jet in a low Mach number crossflow with ground effect. All the results demonstrate that the preconditioning algorithm improves both the numerical accuracy and convergence efficiency and, thereby, enables low Mach number unsteady computations to be performed at a fraction of the cost of traditional time-marching methods.

Introduction

Accurate and efficient numerical simulation of unsteady fluid flows is one of the remaining challenges for the resolution of engineering problems. For high speed flows, time-marching methods developed by the aerodynamics' CFD community have been very successful for both Euler and Navier-Stokes problems [1, 2]. Likewise, for incompressible flows, unsteady algorithms based on projection methods [3, 4] or artificial compressibility [5, 6, 7, 8] approaches have been widely applied. However, many engineering problems involve the co-existence of both compressible and incompressible flow regimes in the same flow field. For such cases, generalized preconditioned dual-time approaches have been developed which allow the same computer code to be applied in all speed regimes [9, 10].

An example of a flow field that simultaneously involves both low speed and high speed regions is the Harrier aircraft in near-hover (landing approach) condition [11]. In this situation, the aircraft's forward velocity is approximately 0.04 Mach. However, at the same time, the aircraft is kept in the air by four high speed jet exhausts directed toward the ground. The interaction of the jets with the cross-wind and the ground effect renders this flow field extremely time-dependent. A purely incompressible method would be

*Aerospace Engineer, Member AIAA
†Research Professor, Member AIAA
‡Senior Research Scientist, Associate Fellow AIAA
This paper is declared a work of the U. S. Government and is not subject to copyright protection in the United States.

appropriate for the low-speed free-stream region, but not for the high speed jet plumes. On the other hand, a transonic flow method would be appropriate for the jets, but is inefficient for the free-stream. Thus, numerical procedures for the solution of such problems must be capable of simultaneously handling both transonic and low speed flow regimes.

The difficulty that transonic algorithms face at low speeds arises due to the poor conditioning of the eigenvalues of the time-marching system. This lack of conditioning leads to poor solution accuracy as well as poor convergence properties. Over the last decade, preconditioning methods have become increasingly popular as a means of assuring that the time-marching algorithm is well-conditioned in terms of both accuracy and efficiency at all speeds [12, 13]. For steady state problems, these methods involve the introduction of pseudo-time derivatives in lieu of the physical time derivative terms in the time-marching system. For unsteady flows, the formulation is typically cast within a dual-time-stepping strategy, wherein the physical-time derivatives are employed to follow the physical transients and the pseudo-time derivatives serve as an iterative device. Proper definition of the pseudo-time derivatives serves to optimize the numerical solution procedure, making the performance of the algorithm commensurate with traditional transonic methods at high speeds and with incompressible methods at low speeds.

In this paper, the implementation of the preconditioned dual-time algorithm in the OVERFLOW code [1] is discussed. The OVERFLOW code is a compressible Navier-Stokes code that uses the Chimera overset grid approach [14] for simulating complex-body configurations such as the afore-mentioned Harrier aircraft. The main solution algorithm in the code is the diagonalized approximate-factorization procedure [15]. The implementation of the preconditioned dual-time scheme in the diagonalized approximate factorization framework has been described by Buelow et al. [16] and we follow the same approach in this paper. Specifically, this involves a modification of the traditional diagonalization procedure to include both the physical and preconditioned time-derivatives, thereby avoiding the introduction of block matrices in the implicit operator.

The paper is organized as follows. We start by presenting the preconditioned dual-time algorithm and its implementation in the diagonalized framework of the OVERFLOW code. Following this, we present several computational examples to verify the accuracy and efficiency gains in various flow regimes of interest. The example problems include both simple test problems, such as the propagation of a Lamb vortex and Stokes' first problem, as well as more practical situations, such as a round-jet in a low-Mach number cross-flow with ground-effect. The latter problem is representative of the Harrier flow fields mentioned earlier. Finally, we summarize the current status of the unsteady modeling capability in OVERFLOW and point out the areas for future research and development.

Preconditioned Dual-Time Algorithm

Equations of Motion

The Euler equations in generalized coordinates can be written as follows:

$$\frac{\partial \hat{Q}}{\partial \hat{t}} + \frac{\partial \hat{E}}{\partial \xi} + \frac{\partial \hat{F}}{\partial \eta} + \frac{\partial \hat{G}}{\partial \zeta} = 0 \tag{1}$$

Here,

$$\hat{Q} = J^{-1}Q = (\rho, \rho u, \rho v, \rho w, e)/J,$$

$$\hat{E} = J^{-1}\begin{pmatrix} \rho U \\ \rho u U + p \\ \rho v U \\ \rho w U \\ (e+p)U \end{pmatrix},$$

$$\hat{F} = J^{-1}\begin{pmatrix} \rho V \\ \rho u V \\ \rho v V + p \\ \rho w V \\ (e+p)V \end{pmatrix} \quad and$$

$$\hat{G} = J^{-1}\begin{pmatrix} \rho W \\ \rho u W \\ \rho v W \\ \rho w W + p \\ (e+p)W \end{pmatrix}$$

where $J^{-1} = x_\xi y_\eta z_\zeta + x_\zeta y_\xi z_\eta + x_\eta y_\zeta z_\xi - x_\xi y_\zeta z_\eta - x_\eta y_\xi z_\zeta - x_\zeta y_\eta z_\xi$ and U, V, and W are the contravariant velocities.

To carry out the iterations at each physical time step, an artificial time(τ) term is explicitly introduced

$$\frac{\partial \hat{Q}}{\partial \tau} + \frac{\partial \hat{Q}}{\partial \hat{t}} + \frac{\partial \hat{E}}{\partial \xi} + \frac{\partial \hat{F}}{\partial \eta} + \frac{\partial \hat{G}}{\partial \zeta} = 0 \tag{2}$$

Convergence of the pseudo-time(sub-iterations) at each physical time step is important for obtaining an accurate transient solution. To optimize the performance of the pseudo-iterations, the pseudo-time term is written in terms of the primitive variable vector $\hat{Q}_p = (P, u, v, w, T)/J$ and the preconditioning matrix Γ_p.

$$\Gamma_p \frac{\partial \hat{Q}_p}{\partial \tau} + \frac{\partial \hat{Q}}{\partial \hat{t}} + \frac{\partial \hat{E}}{\partial \xi} + \frac{\partial \hat{F}}{\partial \eta} + \frac{\partial \hat{G}}{\partial \zeta} = 0 \tag{3}$$

Here Γ_p takes the form discussed in Ref. [9] and is given in the appendix.

Discretization

Discretizing Eqn. 3 with first order finite difference for artificial time and second order backward difference for the physical time terms results in

$$\Gamma_p \frac{\hat{Q}_p^{k+1} - \hat{Q}_p^k}{\Delta\tau} + \frac{3\hat{Q}^{k+1} - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} + \delta_\xi \hat{E}^{k+1} + \delta_\eta \hat{F}^{k+1} + \delta_\zeta \hat{G}^{k+1} = 0 \tag{4}$$

Here, k is the pseudo-iteration counter, n is the time step counter and δ represents spatial differences in the direction indicated by the subscript. After linearization,

$$\Gamma_p \frac{\Delta\hat{Q}_p}{\Delta\tau} + \frac{3(\hat{Q}^k + \Gamma_e \Delta\hat{Q}_p) - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t} + \delta_\xi(\hat{E}^k + \hat{A}_p^k \Delta\hat{Q}_p) + \delta_\eta(\hat{F}^k + \hat{B}_p^k \Delta\hat{Q}_p) + \delta_\zeta(\hat{G}^k + \hat{C}_p^k \Delta\hat{Q}_p) = 0 \tag{5}$$

where $\Delta\hat{Q}_p = \hat{Q}_p^{k+1} - \hat{Q}_p^k$, A_p, B_p, and C_p are the flux Jacobians with respect to $\hat{Q}_p$, and $\Gamma_e = \frac{\partial Q}{\partial Q_p}$.

Rewriting this equation such that all terms evaluated at sub-iteration k or time steps n and $n-1$ are on the right hand side and all terms multiplying $\Delta\hat{Q}_p$ are on the left hand side

$$\left[\Gamma_p + \Gamma_e \frac{3\Delta\tau}{2\Delta t} + \Delta\tau(\hat{A}_p^k \delta_\xi + \hat{B}_p^k \delta_\eta + \hat{C}_p^k \delta_\zeta)\right] \Delta\hat{Q}_p = R^k \tag{6}$$

where

$$R^k = -\Delta\tau \left(\frac{3\hat{Q}^k - 4\hat{Q}^n + \hat{Q}^{n-1}}{2\Delta t}\right) - \Delta\tau(\delta_\xi \hat{E}^k + \delta_\eta \hat{F}^k + \delta_\zeta \hat{G}^k) \tag{7}$$

The key step in the derivation of a diagonalized scheme rests in combining the pseudo- physical time-derivative terms on the left hand side into a single matrix. Accordingly, we define $S_p = \Gamma_p + \frac{3}{2}\frac{\Delta\tau}{\Delta t}\Gamma_e$ (see Appendix),

$$\left(S_p + \Delta\tau \hat{A}_p^k \delta_\xi + \Delta\tau \hat{B}_p^k \delta_\eta + \Delta\tau \hat{C}_p^k \delta_\zeta\right) \Delta\hat{Q}_p = R^k \tag{8}$$

Multiplying through by Γ_e and converting back to the conservative system, we get

$$\left(I + \Delta\tau\Gamma_e S_p^{-1} \hat{A}^k \delta_\xi + \Delta\tau\Gamma_e S_p^{-1} \hat{B}^k \delta_\eta + \Delta\tau\Gamma_e S_p^{-1} \hat{C}^k \delta_\zeta\right) \Delta\hat{Q} = \Gamma_e S_p^{-1} R^k \tag{9}$$

where A, B, and C are the flux Jacobians with respect to $\hat{Q}$.

Approximate Factorization and Diagonalization

Applying approximate factorization,

$$\left(I + \Delta\tau\Gamma_e S_p^{-1} \hat{A}^k \delta_\xi\right)\left(I + \Delta\tau\Gamma_e S_p^{-1} \hat{B}^k \delta_\eta\right)\left(I + \Delta\tau\Gamma_e S_p^{-1} \hat{C}^k \delta_\zeta\right) \Delta\hat{Q} = \Gamma_e S_p^{-1} R^k \tag{10}$$

Now, $\Gamma_e S_p^{-1} \hat{A}$ has the same eigenvalues as $S_p^{-1}\hat{A}_p$ and the eigenvectors of $\Gamma_e S_p^{-1}\hat{A}$ are the columns of $\Gamma_e X_\xi$ where X_ξ are the eigenvectors of $S_p^{-1}\hat{A}_p$. Thus,

$$\Gamma_e S_p^{-1} \hat{A} = \Gamma_e S_p^{-1} \hat{A}_p \Gamma_e^{-1} = \Gamma_e X_\xi \Lambda_\xi X_\xi^{-1} \Gamma_e^{-1} \tag{11}$$

The scheme can be diagonalized using Eqn. 11 and takes the form

$$\Gamma_e X_\xi \left(I + b\Delta\tau\delta_\xi \tilde{\Lambda}_\xi\right) X_\xi^{-1} X_\eta \left(I + b\Delta\tau\delta_\eta \tilde{\Lambda}_\eta\right) X_\eta^{-1} X_\zeta \left(I + b\Delta\tau\delta_\zeta \tilde{\Lambda}_\zeta\right) X_\zeta^{-1} \Gamma_e^{-1} \Delta\hat{Q} = \Gamma_e S_p^{-1} R^k \tag{12}$$

where $\frac{1}{b} = 1 + \frac{3}{2}\frac{\Delta\tau}{\Delta t}$. The eigenvalues $\tilde{\Lambda}$ are discussed in the appendix.

Artificial Dissipation

The diagonalized approximate factorization scheme employed in OVERFLOW uses central differences to discretize the spacial directions. It has been shown by many researchers that, to maintain uniform accuracy across different flow regimes, it is important to formulate the artificial dissipation terms for the preconditioned system. In the dual-time context, this means that the artificial dissipation terms must be constructed using the pseudo-time derivatives rather than the physical time-derivatives. Accordingly, a typical second order dissipation term can be written as

$$\epsilon_e \frac{\partial}{\partial\xi}\left(\Gamma_p \sigma(\Gamma_p^{-1} A_p) \frac{\partial Q_p}{\partial\xi}\right) \tag{13}$$

where $\sigma(A)$ denotes the spectral radius of the matrix A.

In the implicit scheme, the second-order dissipation terms appear in the implicit LHS operator as well. For the purposes of diagonalization, however, it is necessary to modify the spectral radius term $\sigma(\Gamma_p^{-1}A_p)$ with $\sigma(S_p^{-1}A_p)$. This approximation appears only on the LHS and has been shown by Buelow et al. [16] to have little or no effecct on the convergence efficiency. We further point out that the modification does not impact the accuracy of the discrete formulation.

Preconditioner Definition

The definition of the preconditioning matrix (Γ_p) is determined by the parameter M_p (see Eqn. 17 in

the appendix). In the present case, this parameter is defined as

$$M_p^2 = MIN(MAX(M_i^2, M_u^2, M_{min}^2), 1) \qquad (14)$$

where M_i is the local Mach number scale conventionally used in steady preconditioning [17] and M_{min}^2 is a cutoff value usually set to $3 \times M_\infty^2$. M_u is a new unsteady scale given by

$$M_u = \frac{L_u}{\pi \Delta t a} \qquad (15)$$

where L_u is a user specified unsteady length scale [16], and a is the speed of sound. Note that when $M_p = 1$, preconditioning is not applied.

Note that the unsteady preconditioning varies the choice of the preconditioning parameter between the inviscid choice and the no preconditioning choice. For acoustic problems, the physical time-step size is small since it is based upon the acoustic CFL number. The unsteady preconditioning parameter then approaches unity (equivalent to the no preconditioning choice). On the other hand, for vortex propagation problems, the physical time step is large as it is selected based upon the particle CFL number. The corresponding unsteady preconditioning parameter then approaches the Mach number (equivalent to the steady preconditioning choice). For intermediate physical time step sizes, the unsteady choice provides preconditioning values between these two limits. As discussed in Ref. [10], the unsteady preconditioning choice optimizes convergence efficiency of the pseudo-time iterations regardless of the choice of physical time step. In most cases, it also insures accurate scaling of the artificial dissipation terms. The exceptions to this statement are discussed further in the Results section.

Results

Vortex propagation

Vortex propagation is used as the first example of an unsteady flow field. Trends for this problem have been previously established in Ref. [10] based on stability analysis. Specifically, a Lamb vortex is defined with the following velocity distribution.

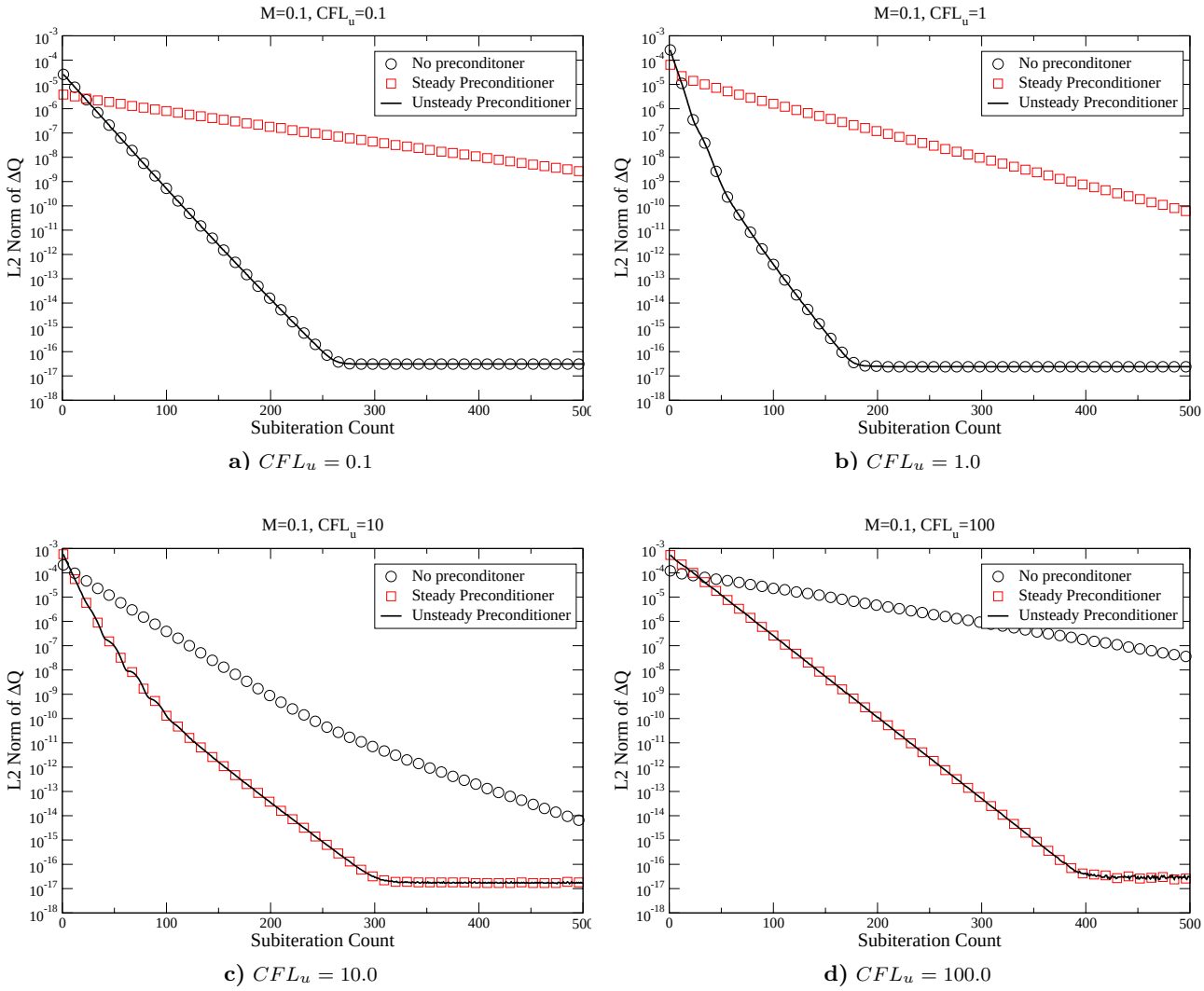
$$V_r(r, \theta) = 0, \qquad V_\theta = \Gamma \left(\frac{1 - \exp \frac{-r^2}{\phi^2}}{r} \right) \qquad (16)$$

where Γ is the vortex strength and ϕ is a characteristic radius. The initial conditions are specified using this velocity distribution and the vortex is propagated using the dual-time-stepping scheme described in the previous section. The present study is conducted at free stream Mach numbers of 0.1 and 0.001. To study the performance of the scheme with and without preconditioning, the sub-iteration convergence is studied for several time steps. The time step is characterized by $CFL_u = \frac{u \Delta t}{\Delta x}$, where u is the speed of propagation, Δt is the physical time step and Δx is the grid size. Ideally, for efficient time-marching of vortex propagation problems, this CFL number should be of order unity.

Here as well as in the other test cases presented in this paper, we compare the convergence of the sub-iterations at each physical time-step for three different choices of the preconditioning matrix, namely, no preconditioning, steady preconditioning and unsteady preconditioning. The first choice corresponds to the traditional time-marching algorithm with dual time-stepping, which is the base scheme in the OVERFLOW code. The second choice adopts the standard steady state preonditioning choice used in steady low Mach computations. Finally, the third choice concerns the present implementation and is given in Eqn. 14. We note that the length scale, L_u, is typically selected to be the height of the domain. which is in accordance with the suggested value in Ref. [10].

Figure 1 shows the results for $M = 0.1$ for four values of CFL_u. For the smaller values of CFL_u (0.1 and 1), the no preconditioning and unsteady preconditioning cases converge rapidly, taking about 200 iterations to reach machine zero. On the other hand, the steady preconditioning choice is observed to yield poor performance. At the higher CFL numbers (10 and 100), the steady and unsteady preconditioning choices converge well, taking about 300 to 400 iterations to reach machine zero, while the no preconditioning choice now yields poorer performance. It is clear from these results that the unsteady preconditioning choice essentially behaves like the no preconditioning choice for small time steps and like the steady preconditioning choice for large time steps. Importantly, the unsteady choice yields the best performance for all physical time step sizes. We point out however that for efficient and accurate solutions, the optimal physical time step choice would correspond to $CFL_u = 1$.

The advantage of using the unsteady preconditioner is observed to be more significant at even lower speeds as Fig. 2 shows for a free stream Mach number of 0.001. At this speed, for low values of CFL_u the steady preconditioner does not converge in the sub-iteration. The unsteady preconditioner is observed to follow the nonpreconditioned case as expected (see Fig. 2(a)). At $CFL_u = 0.1$, however, the unsteady preconditioner outperforms the non-preconditioned case. At $CFL_u = 1$, the steady preconditioner is convergent, but like the nonpreconditioned case it displays poor performance. Again the unsteady preconditioner



a) $CFL_u = 0.1$ **b)** $CFL_u = 1.0$

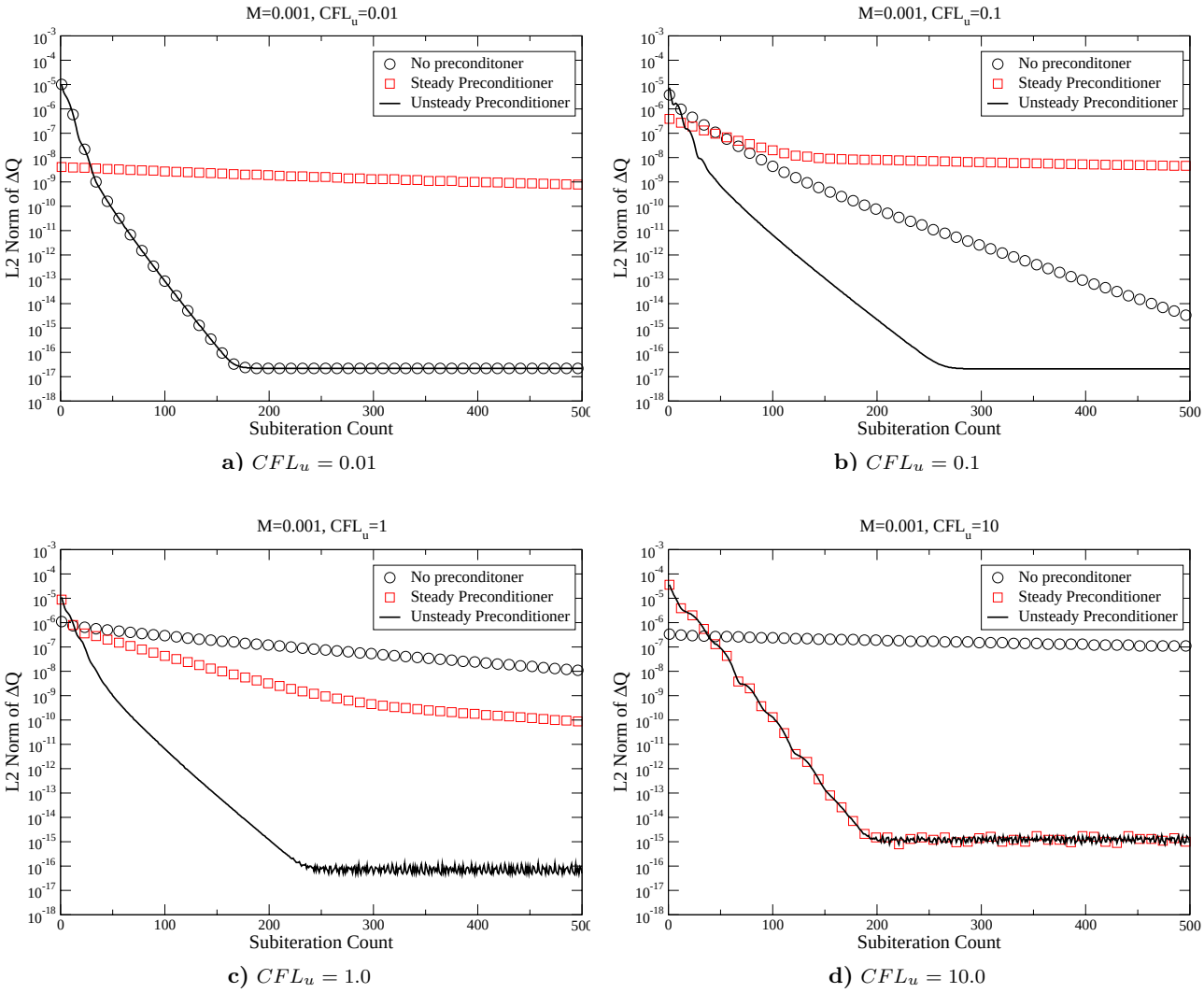
c) $CFL_u = 10.0$ **d)** $CFL_u = 100.0$

Fig. 1 Comparisons of residual convergence with and without preconditioner for a Lamb vortex at $M = 0.1$

outperforms the other choices. At an even higher $CFL_u = 10$ the steady preconditioner performs as well as its unsteady counterpart. It is thus clear that in all cases the unsteady preconditioning provides optimal performance of the pseudo-time convergence. Note that these trends are similar to those presented in Ref. [10].

The subiteration convergence is important to study because it has an impact on the accuracy of the solution. If the subiterations are not properly converged, the solution will be inaccurate. Of course, practical computations do not require that the sub-iterations be converged to machine zero. Therefore, to establish the amount of work required to obtain a solution, the number of sub-iterations required for 4-order convergence is plotted in Fig. 3 against CFL_u. For smaller values of CFL_u where the unsteady preconditioner performs as well as the no preconditioning choice, a relatively small number of sub-iterations are required. However, at these small CFL_u, a large number of time steps is needed to march the solution a specified amount of time. Accordingly, the CPU time required for marching the vortex one rotation is plotted in Fig. 4. Here at low values of CFL_u, the CPU time required is prohibitively high. For $CFL_u = O(1)$, where the vortex is marching approximately 1 grid point per time step, the unsteady preconditioner requires the least number of sub-iterations and makes one rotation in the least amount of CPU time. At larger values of CFL_u, the solution is no longer correct as the time step is simply too high to accurately capture the vortex convection. From these results, it is clear that the preconditioned dual-time scheme provides substantial savings in CPU time for low Mach flowfields.

We next turn our attention to computational accuracy. Vortex computations are notoriously difficult



a) $CFL_u = 0.01$ **b)** $CFL_u = 0.1$

c) $CFL_u = 1.0$ **d)** $CFL_u = 10.0$

Fig. 2 Comparisons of residual convergence with and without preconditioner for a Lamb vortex at $M = 0.001$

and the second-order schemes considered here are especially prone to excessive numerical damping. However, we are primarily concerned with a comparative assessment of the different preconditioning choices. Figure 5 shows vorticity contours of the initial condition (Fig. 5a) and the solution after one rotation through the (periodic) domain for the various preconditioning choices (Figs. 5b, c and d). The non-preconditioned result (Fig. 5b) is wrong, with the correct vortex location but strength significantly damped out. Note that this is true even though the inner iterations are fully converged. Thus, the discrepancy is due to the poor accuracy of the numerical discretization (i.e. excessive artificial dissipation) and not due to lack of convergence. In contrast, the steady (Fig. 5c) and unsteady preconditioning (Fig. 5d) results predict the correct location of the vortex and a more respectable vortex strength. These results clearly point to the benefit of preconditioning in limiting excessive numerical dissipation and hence preserving the accuracy of the numerical formulation as well as computational efficiency.

Stokes' first problem

For the next test case, we consider the solution of an impulsively started plate, which involves the unsteady development of a viscous boundary layer. Commonly referred to as Stokes' first problem, there is a closed form self-similar solution, which makes this an ideal test case for verifying both convergence and accuracy properties of the preconditioned scheme considered here.

To systematically study the computational results, we use an isotropic mesh with a fixed aspect ratio of

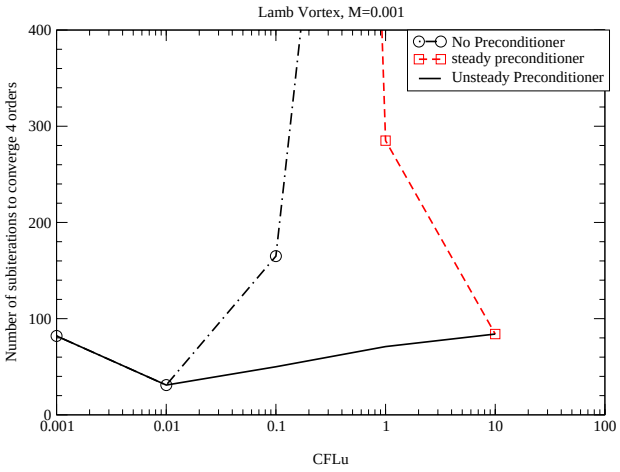


Fig. 3 Number of sub-iterations required for 4 **order convergence for a Lamb vortex at** $M = 0.001$

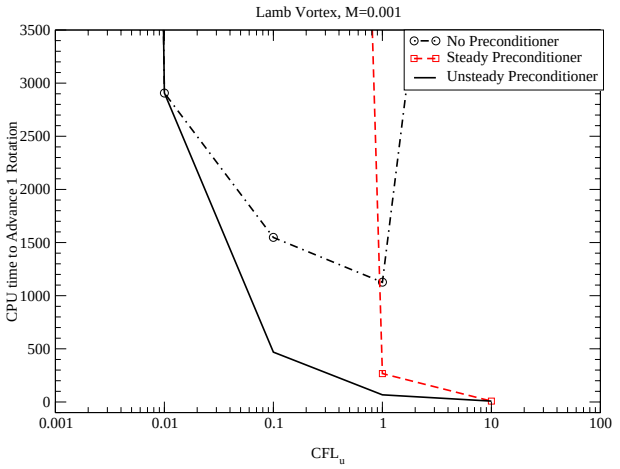


Fig. 4 CPU time required for 4 **order converged(sub-iteration) Lamb vortex to make one rotation at** $M = 0.001$

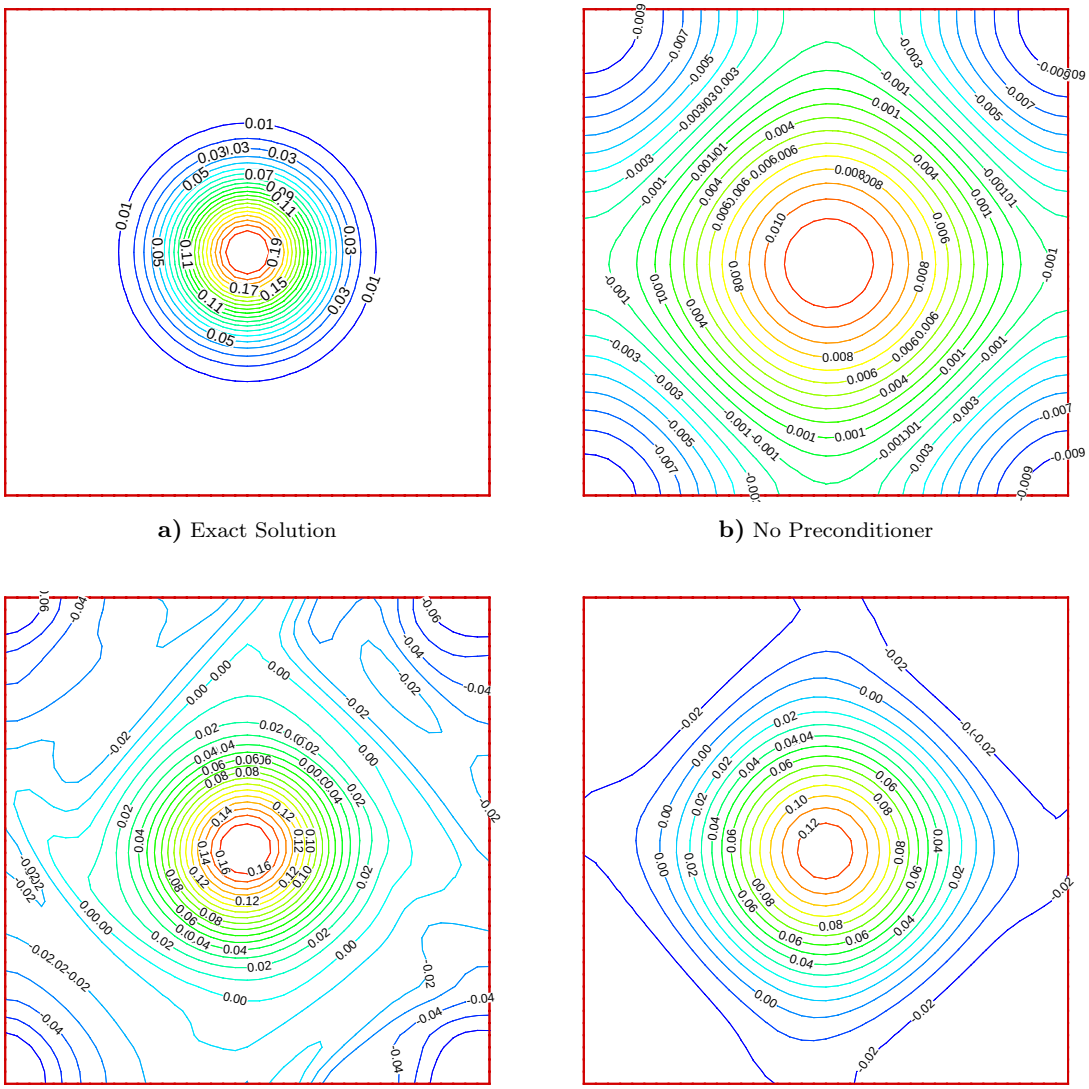
unity. We use 11 grid points in the self-similar direction and 101 grid points in the wall-normal direction. The domain height is 10^{-4} for a $\Delta y = 10^{-6}$. On this uniform Cartesian grid, the initial boundary layer is selected to extend to approximately 51 points off the surface. The initial condition is obtained from the analytical solution and the computations are carried out to follow the continued evolution of the boundary layer.

The characteristic length (L_u) that appears in the definition of the unsteady preconditioning parameter (Eqn. 15) is typically chosen to be the length or width of the computational domain. Here, the relevant length scale is the height of the domain, which is given by 10^{-4}. Figure 6 shows the sensitivity of the sub-iteration convergence to the selection of this parameter for a von Neumann number of $VNN = 1.67$ ($VNN = \frac{\nu \Delta t}{\Delta y^2}$). It is clear that values of 10^{-4} (and less) yield the best convergence rates. Figure 6 also shows the convergence rates with no preconditioning and the steady preconditioning choices. It is clear that these choices yield substantially poorer sub-iteration convergence.

To accurately capture the boundary layer growth, an appropriate physical time step for the numerical marching process must be chosen. The viscous time step or Von Neumann number (VNN) is the characteristic time-step of interest in this problem. It is desirable to use a von Neumann number of 1 which corresponds approximately to a boundary layer growth of 1 grid point per time step. To assess the convergence behavior of the low Mach preconditioner with respect to VNN, the number of sub-iterations required to converge the problem 4 orders of magnitude is plotted for a given time step in Fig. 7. For small values of VNN, the no preconditioning choice shows good convergence behavior. However, for higher values of VNN, the convergence is extremely slow and the solution can not be obtained in a reasonable number of sub-iterations. Using steady preconditioning, the convergence hangs before getting to 4 orders for $VNN < 0.5$. For higher values of VNN proper convergence can be obtained, but very large time steps can lead to poor resolution of the transient evolution of the solution. The unsteady preconditioner exhibits good sub-iteration convergence behavior regardless of the choice of VNN.

The CPU time required to get four orders of sub-iteration convergence is shown in Fig. 8. It is clear that the unsteady preconditioning formulation yields the best CPU times for all choices of VNN. As mentioned earlier, optimal efficiency and accuracy is obtained for VNN of order unity. Under these conditions, the no preconditioning case does not provide good convergence, while the steady and unsteady preconditioning choices provide good sub-iteration convergence. Indeed, Figure 7 indicates that the dual time preconditioned formulation provides significant CPU savings over the traditional (non-preconditioned) formulation.

As mentioned earlier, the choice of preconditioning also influences the accuracy of the computations through the definition of the artificial dissipation terms. Table 1 compares the solutions obtained at a single point in the flowfield with the exact solution for several choices of VNN and preconditionings. It can be observed that the no preconditioning choice is not convergent for large VNN, while the steady preconditioning choice is not convergent for small VNN. The unsteady preconditioning, in contrast, is always convergent. The most accurate solutions are obtained by the steady and unsteady preconditioning choices when the VNN is order unity. It can also be observed that



a) Exact Solution

b) No Preconditioner

c) Steady Preconditioner

d) Unsteady Preconditioner

Fig. 5 Comparison of solution with and without preconditioner for a Lamb vortex at $M = 0.001$

the no preconditioning result for small VNN is somewhat inaccurate even though the solution is converged. This is because the poor scaling of the artificial dissipation terms yields a more diffusive formulation. For a more detailed explanantion, the reader is referred to Ref. [9]. Interestingly, the unsteady preconditioning result for small VNN is also somewhat in error. This is because the unsteady preconditioning approaches the no preconditioning formulation at small VNN and the corresponding artificial dissipation terms become poorly scaled. This difficulty may be readily circumvented by separating the preconditioning formulation used for the time-derivative (which is responsible for convergence effficiency) and that used for the artificial dissipation terms (responsible for accuracy). Such a formulation may be implemented within a multiple pseudo-time framework as discussed in Ref. [18], but is beyond the scope of the present article.

Table 1 Solution comparison for Stokes' first problem with respect to various values of VNN

VNN	No Precond.	Steady Precond.	Unsteady Precond.	Exact Solution
0.017	0.37151	0.37617*	0.37465	0.37622
0.167	0.37177*	0.37617*	0.37606	0.37622
1.67	0.37645*	0.37618	0.37619	0.37622

$*$ Not converged

Cylindrical jet in cross flow with ground effect

A complex unsteady flow field akin to the problem of a Harrier in ground effect is a jet in cross flow with

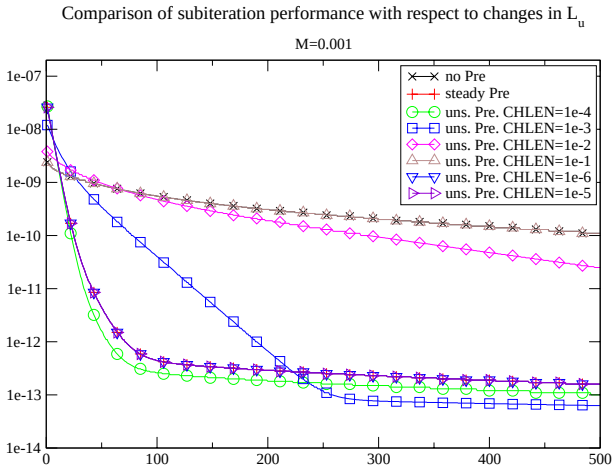


Fig. 6 Parameter study to determine best L_u for $M = 0.001$ on a grid of $AR = 1$ based on sub-iteration residual convergence of the Stokes' first problem

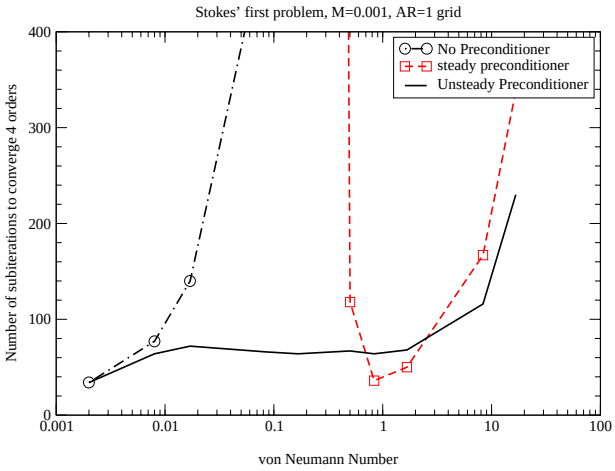


Fig. 7 Number of sub-iterations required 4 order convergence for Stokes' first problem at $M = 0.001$ on a grid of $AR = 1$

ground effect. This problem has been studied experimentally by Cimbala et. al. [19] with a jet at Mach 0.13 issuing out of a 3 in. cylindrical tube in the center of a wind tunnel. The crosswind Mach number is 0.013. The jet subsequently impinges on the wind tunnel floor 9 in. away from the tube exit.

For the present simulation a single grid with cylindrical topology is used with stretching to assure proper viscous mesh spacing near the tube walls and ground. The mesh consists of approximately 250,000 grid points.

The experimental results of Cimbala et al. [19] reveal that the jet impinges on the ground and splays in all directions. The cross flow pushes the jet creating a horseshoe shaped vortex which surrounds the jet on

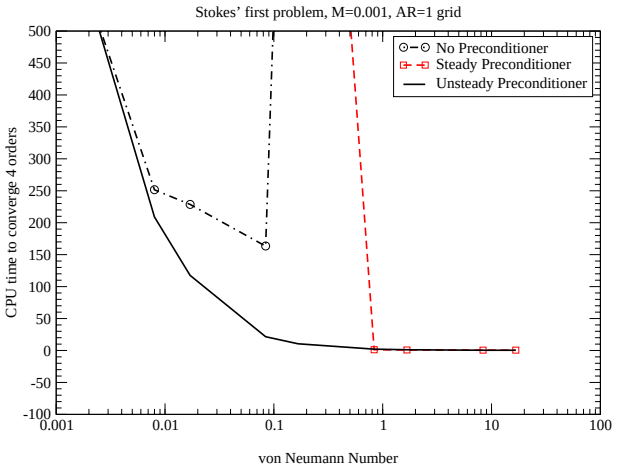


Fig. 8 CPU time required for 4 order converged(sub-iteration) Stokes' first problem boundary layer to grow 10 grid points at $M = 0.001$ on a grid of $AR = 1$

3 sides. The ground vortex in turn is observed to pulsate with a frequency of about 1-5 Hz, a result of the interaction between the jet with the low speed crossflow. The general features of the flowfield are observed in Fig. 9, which shows a particle trace issued from the jet at a fixed point in time. The jet is seen to impinge the ground and a horseshoe vortex is seen in front of the jet.

The computation of this flowfield with a traditional dual time scheme (without preconditioning) is stable only for small time steps. This stability restriction may be attributed to the poor convergence of the sub-iterations at large time step values. Our studies indicated that the maximum physical time step that could be reliably used was 5.5×10^{-5} s. Using this time step, more than 18,000 time steps are required to resolve a frequency of 1 Hz (that is characteristic of the ground vortex unsteadiness). Clearly, this would render the computation prohibitively expensive. Moreover, the non-preconditioned choice also leads to a more dissipative numerical solution, which would also compromise the accuracy of the simulations. In contrast, when the unsteady preconditioned dual time scheme is employed, much larger time-steps may be stably employed and reliable inner iteration convergence is obtained. The computations presented in this paper were obtained with a physical time step size of 5.5×10^{-3} s., which corresponds to about 180 time steps per 1 Hz cycle. We point out however that the larger physical time steps typically necessitate the use of more inner iterations (compared with five inner iterations used in the traditional algorithm). In the present calculations, 50 inner iterations were employed at each physical time step. Thus, the preconditioned

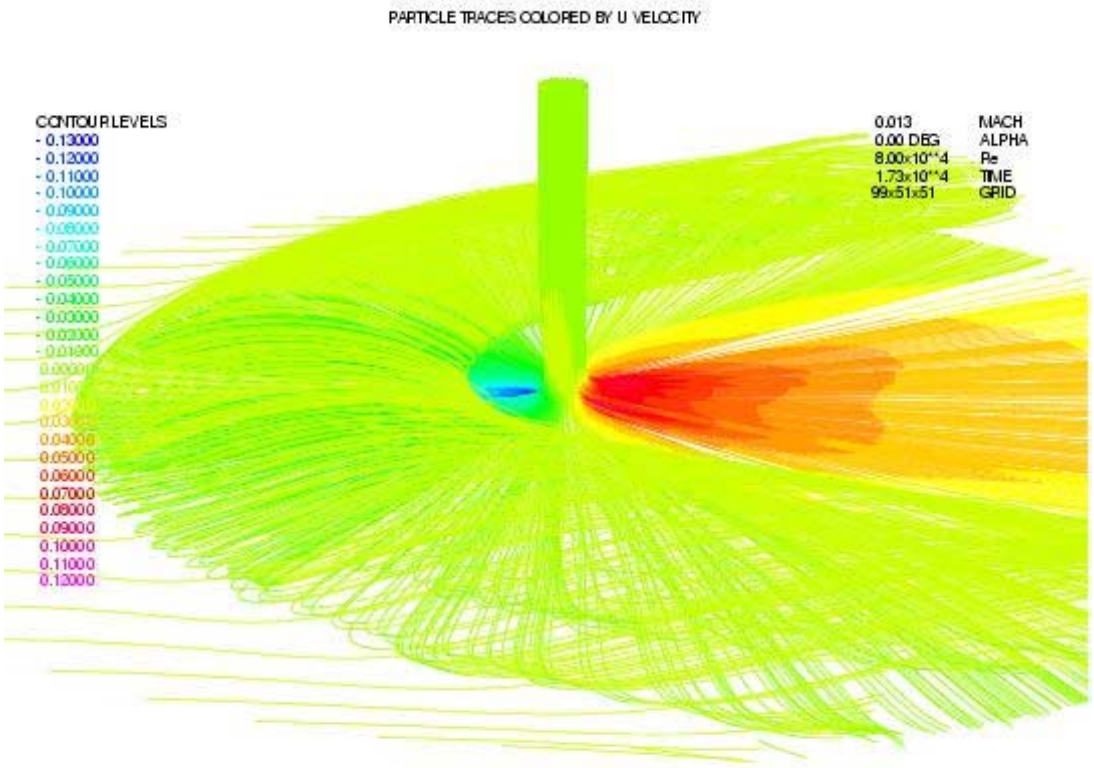


Fig. 9 The particle trace at an instance in time for a jet at $M = 0.13$ in a $M = 0.013$ cross flow

dual time scheme yields a potential CPU savings of about an order of magnitude. Figure 10 shows a plot of the L2 norm of the residual at each physical time step. The periodic unsteady nature of the flowfield is evident. The lowest frequency indicated in these results is approximately 1.8Hz. We point out that these

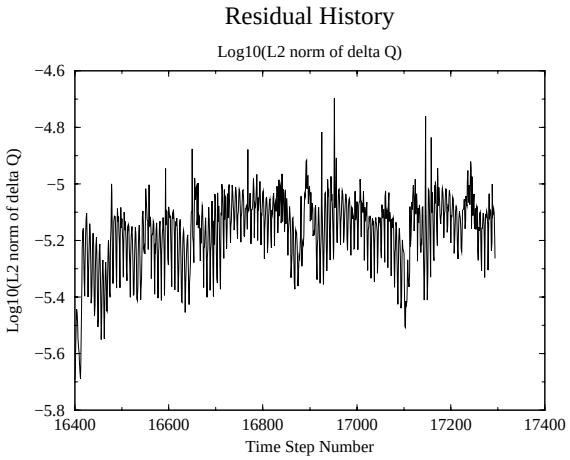


Fig. 10 The residual with respect to time step for the jet in cross flow

results represent a preliminary effort of computing this complex flowfield. Detailed evaluations are presently underway and will be the subject of a future article. Here, it is sufficient to state that the preconditioned dual-time formulation shows potential in computing complex multiple-timescale, 3-D problems with significantly improved accuracy and efficiency.

Summary

A preconditioned dual-time algorithm has been implemented in the OVERFLOW code. The approach follows the method of Buelow et al. to tailor the formulation within the diagonalized solution procedure used in the OVERFLOW code. Modifications of the preconditioning selection to account for unsteady scales have also been included. The method significantly enhances the capability of the code, enabling it to handle unsteady flows over a wide range of Mach numbers and time scales. Specifically, the preconditioned dual-time method enhances accuracy by improving the scaling of the artifical dissipation terms used in the numerical discretization procedure and improves efficiency by optimizing the number of sub-iterations required at each physical time step. For low Mach number, low

frequency problems, in particular, significant savings in CPU time are possible.

The enhancements made possible by the method are demonstrated for both simple and practical flowfields. The simple test problems include the propagation of a Lamb vortex in an straight channel and the flowfield of an impulsively started plate. Computational results reveal that the dual-time scheme with unsteady preconditioning provides optimal sub-iteration convergence for all choices of time step size and Mach number. Moreover, improved accuracy is also obtained with the method.

Preliminary computations of a round jet in a low Mach crossflow with ground effect have also been performed. Here, the preconditioned dual-time scheme enables the use of much larger physical time step sizes, which is important for efficiently resolving the low frequency pulsations of the horshoe-shaped ground vortex. More detailed computations are currently being performed and will be compared with experimental trends. The eventual goal is to apply the validated code to the computation of Harrier flowfields.

The limitation of the present implementation is that the same preconditioning is employed for both the artificial dissipation formulation and for the time-derivative. The disadvantage of this is that the final solution is dependent upon the choice of time-step size. This difficulty may be effectively circumvented through using separate preconditioners for the dissipation and time-derivative terms. The former then controls the accuracy of the discrete formulation, while the latter controls convergence efficiency of the sub-iterations. Implementation of such a multi-level preconditioning formulation will be the subject of future work.

Acknowledgments

The authors wish to thank Mike Olsen of NASA Ames Research Center and Scott Murman of ELORET for many useful discussions. Dr. Venkateswaran was supported in part by NASA Ames Research Center (cooperative agreement No. NCC 2-5493).

References

[1]P. G. Buning, D. C. Jespersen, T. H. Pulliam, W. M. Chan, J. P. Slotnick, S. E. Krist, and K. J. Renze. OVERFLOW Users Manual. NASA.

[2]J. P. Slotnick, M. Kandula, and P. G. Buning. Navier-stokes simulation of the space shuttle launch vehicle flight transonic flowfield using a large scale chimera grid system. AIAA Paper 94-1860, June 1994.

[3]J. Kim and P. Moin. Application of a Fraction-Step Method to Incompressible Navier-Stokes Equations. J. of Computational Physics, 59:308–23, 1985.

[4]C. Kiris and D. Kwak. Numerical Solution of Incompressible Navier-Stokes Equations using a Fractional-Step Approach. Computers & Fluids, 30:829–51, 2001.

[5]A. J. Chorin. A numerical Method for Solving Incompressible Viscous Flow Problems. J. of Computational Physics, 2(12):12–26, 1967.

[6]C. L. Merkle and M. Athavale. A Time Accurate Unsteady Incompressible Algorithm based on Artificial Compressibility. In AIAA 8th Computational Fluid Dynamics Conference, Honolulu, HI, June 1987. AIAA paper 87-1137.

[7]D. Kwak, J. L. C. Chang, S. P. Shanks, and S. Chakravarthy. A Three-Dimensional Incompressible Navier-Stokes Flow Solver using Primitive Variables. AIAA Journal, 24(3):390–6, 1977.

[8]S. E. Rogers, D. Kwak, and C. Kiris. Numerical Solution of the Incompressible Navier-Stokes Equations for Steady-State and Time-Dependent Problems. AIAA paper 89-0463.

[9]S. Venkateswaran and C. L. Merkle. Efficiency and accuracy issues in contemporary cfd algorithms. Fluids 2000 Conference, Denver, CO, June 2000. AIAA Paper 2000-2251.

[10]S. Venkateswaran and C. L. Merkle. Dual Time Stepping and Preconditioning for unsteady Computation. AIAA paper 95-0078, January 1995.

[11]N. Chaderjian, S. A. Pandya, J. Ahmad, and S. Murman. Progress Toward Generation of a Navier-Stokes Database for a Harrier in Ground Effect. AIAA paper 2002-5966, November 2002.

[12]J. M. Weiss and W. A. Smith. Preconditioning Applied to Variable and Constant Density Time-Accurate Flows on Unstructured Meshes. AIAA paper 94-2209, 1994.

[13]E. Turkel. Preconditioning Techniques in Computational Fluid Dynamics. Annual Review of Fluid Mechanics, 31:385–416, 1999.

[14]J. A. Benek, P. G. Buning, and J. L. Steger. A 3-d chimera grd embedding technique. AIAA Paper 85-1523-CP, July 1985.

[15]T. H. Pulliam and D. S. Chaussee. A Diagonal form of an Implicit Approximate Factorization Algorithm. Journal of Computational Physics, 39:347–363, 1981.

[16]Philip E. O. Buelow, Douglas A. Schwer, Jinzhang Feng, and Charles L. Merkle. A Preconditioned Dual-Time, Diagonalized ADI scheme for Unsteady Computations. AIAA paper 97-2101, 1997.

[17]D. Jespersen, T. Pulliam, and P. Buning. Recent Enhancements to OVERFLOW. AIAA paper 97-0644.

[18]S. Venkateswaran, C. L. Merkle, X. Zeng, and D. Li. Influence of large-scale pressure changes on preconditioned solutions. AIAA Paper 2002-2957, June 2002.

[19]J. M. Cimbala, M. L. Billet, D. P. Gaublomme, and J. C. Oefelein. Experiments on the unsteadiness associated with a ground vortex. J. of Aircraft, 28(4):261–267, April 1991.

APPENDIX

The Preconditioning Matrix

The preconditioning matrix Γ_p has the form defined in [16].

$$\Gamma_p = \begin{bmatrix} \rho'_p & 0 & 0 & 0 & \rho_T \\ u\rho'_p & \rho & 0 & 0 & u\rho_T \\ v\rho'_p & 0 & \rho & 0 & v\rho_T \\ w\rho'_p & 0 & 0 & \rho & w\rho_T \\ (\rho h_p + h_0\rho'_p - 1) & \rho u & \rho v & \rho w & \rho h_T + h_0\rho_T \end{bmatrix} \tag{17}$$

where $\rho'_p = \frac{1}{\epsilon_p a^2}$ and a is the speed of sound. Here $\epsilon_p = \frac{M_p^2}{1+(\gamma-1)M_p^2}$. However, in practice ρ'_p is never computed directly. Instead, computing the eigenvalues of $S_p^{-1}A_p$ (Eqn. 22) involves the evaluation of the terms $\frac{\rho h_T}{d'}$ and $\frac{d}{d'}$. These terms are defined as follows.

$$\begin{aligned}\frac{\rho h_T}{d'} &= \frac{\rho h_T}{\rho h_T \rho''_p + (1-\rho h_p)\rho_T} \\ &= \frac{\rho c_p}{\rho c_p(b\rho'_p + (1-b)\rho_p) - \frac{\rho}{T}} \\ &= \frac{c_p a^2}{c_p(\frac{b}{\epsilon_p} + \gamma - \gamma b) - \gamma R} \\ &= \frac{a^2}{\frac{b}{\epsilon_p} - \gamma b + 1} \\ &= \frac{M_p^2 a^2}{b - M_p^2(b-1)}\end{aligned} \tag{18}$$

Defining $\beta' = \frac{M_p^2}{b - M_p^2(b-1)}$

$$\frac{\rho h_T}{d'} = \beta' a^2 \tag{19}$$

Also,

$$\frac{d}{d'} = \frac{\frac{\rho h_T}{d'}}{\frac{\rho h_T}{d}} = \beta' \tag{20}$$

The values of M_p^2 and b now controls the behavior of the preconditioner. The parameter b switches the behavior of the preconditioner from unsteady to steady. For steady flows, $b = 1$ and $\beta' = M_p^2$.

For unsteady flows, M_p^2 is defined by Eqn. 14.

Definition of S_p

$$S_p = \frac{1}{b}\begin{bmatrix} \rho''_p & 0 & 0 & 0 & \rho_T \\ u\rho''_p & \rho & 0 & 0 & u\rho_T \\ v\rho''_p & 0 & \rho & 0 & v\rho_T \\ w\rho''_p & 0 & 0 & \rho & w\rho_T \\ h_0\rho''_p - (1-\rho h_p) & \rho u & \rho v & \rho w & \rho h_T + h_0\rho_T \end{bmatrix} \tag{21}$$

where $\rho''_p = b\rho'_p - (b-1)\rho_p$. and $\frac{1}{b} = 1 + \frac{3}{2}\frac{\Delta\tau}{\Delta t}$

Eigenvalues of $S_p^{-1}\hat{A}_p$

$$\lambda_{1,2,3} = bU$$

$$\lambda_{4,5} = \frac{b}{2}\left[U\left(1+\frac{d}{d'}\right) \pm \sqrt{U^2\left(1-\frac{d}{d'}\right)^2 + 4\frac{\rho h_T}{d'}}\right] \tag{22}$$

Eigenvalues for $S_P^{-1}\hat{B}_p$ and $S_P^{-1}\hat{C}_p$ are similar.

Eigenvectors of $S_p^{-1}\hat{A}_p$

For the repeated eigenvalues($\lambda = bU$), many choices of the eigenvectors are possible. The following choice keeps the eigenvectors associated with the linear eigenvalues from becoming degenerate by assuring that the eigenvectors are non-vanishing independent of the geometric orientation [17].

$$\begin{bmatrix} 0 \\ 0 \\ \tilde{\xi}_z \\ -\tilde{\xi}_y \\ \tilde{\xi}_x \end{bmatrix} \begin{bmatrix} 0 \\ -\tilde{\xi}_z \\ 0 \\ \tilde{\xi}_x \\ \tilde{\xi}_y \end{bmatrix} \begin{bmatrix} 0 \\ \tilde{\xi}_y \\ -\tilde{\xi}_x \\ 0 \\ \tilde{\xi}_z \end{bmatrix} \tag{23}$$

where $\tilde{\xi}_{x,y,z} = \frac{\xi_{x,y,z}}{|\nabla\xi|}$

The eigenvectors corresponding to the eigenvalues of λ^+ and λ^- are

$$\begin{bmatrix} -\frac{\lambda^- - bU\frac{d}{d'}}{\lambda^+ - \lambda^-} \\ \frac{b\tilde{\xi}_x}{\rho(\lambda^+ - \lambda^-)} \\ \frac{b\tilde{\xi}_y}{\rho(\lambda^+ - \lambda^-)} \\ \frac{b\tilde{\xi}_z}{\rho(\lambda^+ - \lambda^-)} \\ -\frac{1-\rho h_p}{\rho h_T}\frac{\lambda^- - bU\frac{d}{d'}}{\lambda^+ - \lambda^-} \end{bmatrix} ; \begin{bmatrix} \frac{\lambda^+ - bU\frac{d}{d'}}{\lambda^+ - \lambda^-} \\ -\frac{b\tilde{\xi}_x}{\rho(\lambda^+ - \lambda^-)} \\ -\frac{b\tilde{\xi}_y}{\rho(\lambda^+ - \lambda^-)} \\ -\frac{b\tilde{\xi}_z}{\rho(\lambda^+ - \lambda^-)} \\ \frac{1-\rho h_p}{\rho h_T}\frac{\lambda^+ - bU\frac{d}{d'}}{\lambda^+ - \lambda^-} \end{bmatrix} \tag{24}$$

For a perfect gas, the left and right eigenvector matrices can be written as follows.

$$X_\xi = \begin{bmatrix} 0 & 0 & 0 & \frac{\lambda^+ - bU}{\lambda^+ - \lambda^-} & \frac{\lambda^- - bU}{\lambda^+ - \lambda^-} \\ 0 & -b\tilde{\xi}_z & b\tilde{\xi}_y & \frac{b\tilde{\xi}_x}{\rho(\lambda^+ - \lambda^-)} & -\frac{b\tilde{\xi}_x}{\rho(\lambda^+ - \lambda^-)} \\ b\tilde{\xi}_z & 0 & -b\tilde{\xi}_x & \frac{b\tilde{\xi}_y}{\rho(\lambda^+ - \lambda^-)} & -\frac{b\tilde{\xi}_y}{\rho(\lambda^+ - \lambda^-)} \\ -b\tilde{\xi}_y & b\tilde{\xi}_x & 0 & \frac{b\tilde{\xi}_z}{\rho(\lambda^+ - \lambda^-)} & -\frac{b\tilde{\xi}_z}{\rho(\lambda^+ - \lambda^-)} \\ b\tilde{\xi}_x & b\tilde{\xi}_y & b\tilde{\xi}_z & \frac{\gamma-1}{\rho\gamma}\frac{\lambda^+ - bU}{\lambda^+ - \lambda^-} & -\frac{\gamma-1}{\rho\gamma}\frac{\lambda^- - bU}{\lambda^+ - \lambda^-} \end{bmatrix} \tag{25}$$

where

$$\begin{aligned}\lambda^+ - \lambda^- &= b\sqrt{U^2(1-\frac{d}{d'})^2 + 4\frac{\rho h_T}{d'}|\nabla\xi|^2} \\ &= b\sqrt{U^2(1-\beta')^2 + 4a^2\beta'|\nabla\xi|^2}\end{aligned} \tag{26}$$

$$X_\xi^{-1} = \frac{1}{b}\begin{bmatrix} -\frac{\gamma-1}{\gamma\rho}\tilde{\xi}_x & 0 & \tilde{\xi}_z & -\tilde{\xi}_y & \tilde{\xi}_x \\ -\frac{\gamma-1}{\gamma\rho}\tilde{\xi}_y & -\tilde{\xi}_z & 0 & \tilde{\xi}_x & \tilde{\xi}_y \\ -\frac{\gamma-1}{\gamma\rho}\tilde{\xi}_z & \tilde{\xi}_y & -\tilde{\xi}_x & 0 & \tilde{\xi}_z \\ 1 & A_1\tilde{\xi}_x & A_1\tilde{\xi}_y & A_1\tilde{\xi}_z & 0 \\ 1 & A_2\tilde{\xi}_x & A_2\tilde{\xi}_y & A_2\tilde{\xi}_z & 0 \end{bmatrix} \tag{27}$$

where $A_1 = -\rho(\lambda^- - bU)$, $A_2 = -\rho(\lambda^+ - bU)$, and

ξ can be replaced with η or ζ to obtain the eigenvector matrices for $S_P^{-1}\hat{B}_p$, and $S_P^{-1}\hat{C}_p$.