ASSESSMENT OF LOCAL PRECONDITIONERS FOR STEADY STATE AND TIME DEPENDENT FLOWS

Veer N. Vatsa*
NASA Langley Research Center, Hampton,VA
Eli Turkel†
Tel-Aviv University, Israel and NIA, Hampton, VA

Abstract

Preconditioners for hyperbolic systems are numerical artifices intended to accelerate the convergence path to a steady state. In addition, in some cases, the preconditioner can also be included in the artificial viscosity/upwinding terms in order to improve the accuracy of the steady state solution. For time dependent problems we use a dual time stepping approach; and therefore the preconditioner affects the convergence rate and accuracy of the subiterations at each physical time step. We consider two types of local preconditioners that couple the governing equations at a node point: Jacobi and low-speed preconditioning. We consider their effectiveness for both steady state and time dependent problems with regard to the convergence rate and the numerical accuracy. We also consider the effect of the far field boundary conditions on both steady state and time dependent problems.

1 Introduction

Preconditioning methods for low-speed, steady flows have been available for almost twenty years [16]. Because such preconditioners are generally designed to modify the path to steady state, special attention is required for adapting these methods for unsteady flows to maintain temporal accuracy. Fortunately, such preconditioning techniques are easily extendable to unsteady flow applications in conjunction with dual time stepping algorithms.

We consider the hyperbolic system of unsteady Euler equations appended with pseudo time derivatives

$$w_\tau + \xi w_t + Aw_x + Bw_y + Cw_z = 0 \tag{1}$$

where the flux Jacobian matrices A, B, C are symmetric (or simultaneously symmetrizable). We are interested in $\tau \to \infty$, where τ is viewed as an artificial (pseudo) time. For physically steady state problems $\xi = 0$, while for time dependent problems ξ is set equal to 1. We discretize the physical time derivative with a backward difference formula (BDF)

$$\frac{\partial w}{\partial t} \simeq \frac{c_t}{\Delta t} w^{n+1} + \frac{Q(w^n, w^{n-1}, ...)}{\Delta t} \tag{2}$$

where, c_t is a constant that depends on the order of the BDF scheme. Define $\zeta = \frac{\xi c_t}{\Delta t}$. We Fourier transform Eq. (1) in space and replace w_t by Eq. (2). Define the amplification matrix

$$G(\omega_1, \omega_2, \omega_3) = -\zeta I + i(A\omega_1 + B\omega_2 + C\omega_3) \tag{3}$$

The condition number of this system is defined as

$$cond\# = \max_{\omega_i} \left| \frac{\lambda_{\max}(G)}{\lambda_{\min}(G)} \right| \quad . \tag{4}$$

where λ denotes an eigenvalue of the matrix while ω_i are real numbers with $\omega_1^2 + \omega_2^2 + \omega_3^2 = 1$. Note that the eigenvalues of $i(A\omega_1 + B\omega_2 + C\omega_3)$ are pure imaginary since the matrices are symmetric. Physically the condition number (with $\xi = 0$) can

*Senior Member
†Professor, Department of Mathematics, Associate Fellow

be interpreted as the ratio of the fastest speed to the slowest speed in any direction. If viscous terms are included then we have additional negative real matrices in Eq. (3). We stress that if Δt is sufficiently small, then the condition number is close to 1, and the methods proposed here will neither be effective, nor are they needed for convergence acceleration. On the other hand, for steady flows, the condition number is approximately equal to the reciprocal of the Mach number; and hence, the proposed preconditioners are expected to be very effective for convergence acceleration.

With a local preconditioner we change the discrete equations at a single grid node without introducing new coupling between the neighboring nodes. This can be expressed on the continuous level as replacing Eq. (1) by

$$P^{-1}w_\tau + \xi w_t + Aw_x + Bw_y + Cw_z = 0. \tag{5}$$

For well-posedness the matrix P should be symmetric positive definite. We choose the matrix P so as to improve the condition number of the equations at the node point. Hence, this technique makes sense only for a system of equations. For a scalar equation local preconditioning is simply a rescaling of the time variable and has no effect on the numerical solution. This approach is distinct from incomplete-LU (ILU) decomposition based preconditioning techniques which couple all the nodes together and therefore require more expensive inversion techniques. The assumption is that the better the system is conditioned, the faster the iteration process will approach a steady state. The amplification matrix of the preconditioned system is given by $G_P = PG$. The condition number in Eq. (4) is replaced by

$$cond\# = \max_{\omega_i} \left| \frac{\lambda_{\max}(G_P)}{\lambda_{\min}(G_P)} \right| \tag{6}$$

Note that G contains both Hermitian and anti-Hermitian parts. However, since the real part is proportional to the identity matrix, the two parts commute, and so G is a normal matrix. By contrast G_P is not a normal matrix because the symmetric portion P does not commute with PA. Rather G_P is P times a normal matrix which might adversely affect the stability properties of the Runge-Kutta scheme [10].

For time dependent problems where the physical time scale is sufficiently small, and so ζ is large, preconditioning can harm the convergence rate by reducing the effective artificial time step. For such problems the preconditioning in the update stage should be turned off and should only affect the artificial viscosity or the upwinding [12, 23, 29]. In this study we consider two different local preconditioners, Jacobi and low-speed, to alleviate stiffness associated with disparate characteristic speeds and from a poor condition number. We then formulate a composite preconditioner that combines the complementary properties of the Jacobi and low-speed preconditioners in order to achieve an efficient scheme for solving viscous flows with embedded low speed flows.

2 Jacobi Preconditioning

The Jacobi preconditioning is based on adding a matrix-based artificial viscosity to a central difference scheme and then choosing P^{-1} as the terms on the diagonal (i.e. the coefficient of w_{ij}). The result for a central difference scheme is

$$P_J^{-1} = \zeta I + |A| + |B| + |C| \tag{7}$$

This approach has been proposed by Allmaras [2] and Pierce and Giles [13] for steady state flows. The good high frequency damping characteristics of the Jacobi preconditioner make it an ideal candidate for coupling with a multigrid scheme. Because this formulation connects the preconditioning with the artificial viscosity (or upwinding), the matrix P is affected by the details of the discretization. However, Eq. (7) has also been used with other artificial viscosities such as CUSP (see Caughey and Jameson, [3]). Hence, we prefer to view Eq. (7) as a matrix or characteristic inverse time step (see [24] for a similar view). A multistage, non-preconditioned Runge-Kutta scheme in artificial time uses a time step given by

$$\Delta\tau = \frac{\text{CFL}}{\zeta + \rho(A) + \rho(B) + \rho(C)} \tag{8}$$

where ρ is the spectral radius and CFL is a number chosen to achieve stability. A matrix time step for the Jacobi preconditioner replaces this by

$$\Delta\tau = \text{CFL}\,(\zeta I + |A| + |B| + |C|)^{-1} \tag{9}$$

In calculating the absolute value of the matrices one needs to cutoff the eigenvalues to prevent them from becoming too small. We do this by not allowing any eigenvalue to be less than a given percentage of the maximum eigenvalue. Within the artificial viscosity we cutoff the acoustic eigenvalues at 30% and the convective eigenvalue at 10% of the maximum [15]. Within the Jacobi preconditioning all eigenvalues are cutoff at 30% of the spectral radius.

The preconditioning techniques described here have been incorporated in the TLNS3D code [25, 26] for assessment. The standard TLNS3D code solves the generalized thin-layer Reynolds-averaged Navier-Stokes equations, and uses residual

smoothing and multigrid to accelerate the convergence to a steady state. We wish to include all of these acceleration techniques when using the Jacobi preconditioning. Though some researchers have avoided the use of residual smoothing with the Jacobi preconditioner [13], we have found no difficulty in including both the residual smoothing and the Jacobi preconditioning. In fact, they complement each other since the Jacobi preconditioning is local while the residual smoothing is global in nature due to the implicit operators in each coordinate direction. This is especially important in the presence of high aspect ratio cells which are essential for resolving boundary layers in viscous flows.

The Jacobi preconditioning for the k-th stage of a Runge-Kutta algorithm is given as

$$(\zeta I + |A| + |B| + |C|)\Delta w = \alpha_k \text{CFL} \cdot \text{Residual} \tag{10}$$

where α_k is the stage coefficient of the Runge-Kutta scheme. For a three dimensional problem each of the matrices is 5×5. As suggested by Caughey and Jameson [9] and Hosseini and Alonso [5], we transform the equation to entropy variables $Q = (\frac{dp}{\rho c}, du, dv, dw, dS)$. This has the advantage that the flux Jacobian matrices are symmetric and also that the entropy equation reduces to a scalar equation that decouples from the others. Hence, we need only to operate with a 4×4 matrix rather than a 5×5 system which results in appreciable savings in computational costs. The formulas for the absolute values are presented later. One then calculates the LU factors explicitly for a 4×4 matrix. Using a Cholesky decomposition for a symmetric matrix, fewer elements need to be calculated. However, four square roots are evaluated in this approach (Caughey and Jameson [3] reduced this to three square roots by clever programming). Computationally, we found that using the nonsymmetric LU form required about the same computer time for this small matrix, since no square roots are required. The storage is larger for the nonsymmetric decomposition. Since this matrix is not stored globally, it has an insignificant impact on memory requirements. The extra work in the Jacobi preconditioning is mainly in defining the elements of the matrix rather than the inversion, and it typically adds about 10% to the total running time for a compressible turbulent flow code.

For viscous problems, Caughey and Jameson [3] replaced the entropy variables by a transformation suggested by Abarbanel and Gottlieb [1]. An alternate approach to include viscous effects was devised by Turkel, where he suggested using a different set of physical of variables [16, 21]. Even though symmetry is preserved, the resultant absolute values constitute a full 5×5 matrix. Since this substantially adds to the computing time, we used the entropy variables instead and approximate the viscous terms by a diagonal matrix corresponding to the additional term in the time step calculation of the standard code. Hence, it is just an addition to the diagonal term in Eq. (9). For high Reynolds number flows this viscous correction is small enough that it does not justify the additional computational time required for inverting a full 5×5 matrix at all nodes.

In generalized coordinates we define the contravariant velocity $U = uS_{xx} + vS_{xy} + wS_{xz}$ where S_{ij} are the elements of the surface area tensor. In entropy variables the flux Jacobian matrix A, is given by

$$A = \begin{pmatrix} U & cS_{xx} & cS_{xy} & cS_{xz} & 0 \\ cS_{xx} & U & 0 & 0 & 0 \\ cS_{xy} & 0 & U & 0 & 0 \\ cS_{xz} & 0 & 0 & U & 0 \\ 0 & 0 & 0 & 0 & U \end{pmatrix}$$

Let

$$\begin{aligned} R_1 &= \frac{|\lambda_1| + |\lambda_2|}{2} \\ R_2 &= \frac{|\lambda_1| - |\lambda_2|}{2} \\ R_3 &= R_1 - |U| \end{aligned}$$

and $|S| = \sqrt{S_{xx}^2 + S_{xy}^2 + S_{xz}^2}$. Define the normalized surface metrics

$$\hat{S_x} = \frac{S_{xx}}{|S|} \qquad \hat{S_y} = \frac{S_{xy}}{|S|} \qquad \hat{S_z} = \frac{S_{xz}}{|S|}$$

Then the absolute value is given by (symmetric terms suppressed)

$$|A| = \begin{pmatrix} R_1 & \hat{S_x} R_2 & \hat{S_y} R_2 & \hat{S_z} R_2 & 0 \\ . & |U| + \hat{S_x}^2 R_3 & \hat{S_x}\hat{S_y} R_3 & \hat{S_x}\hat{S_z} R_3 & 0 \\ . & . & |U| + \hat{S_y}^2 R_3 & \hat{S_y}\hat{S_z} R_3 & 0 \\ . & . & . & |U| + \hat{S_z}^2 R_3 & 0 \\ . & . & . & . & |U| \end{pmatrix} \tag{11}$$

To get a better intuition of the matrix $|A|$ we consider the subsonic case with $0 \leq u, v, w \leq c$. In Cartesian coordinates we have $S_{xx}=1$, $S_{xy}=S_{xz}=0$. Then $R_1=c$, $R_2=u$, $R_3=c-u$ and

$$|A| = \begin{pmatrix} c & u & 0 & 0 & 0 \\ u & c & 0 & 0 & 0 \\ 0 & 0 & u & 0 & 0 \\ 0 & 0 & 0 & u & 0 \\ 0 & 0 & 0 & 0 & u \end{pmatrix}$$

Let d be the number of dimensions (2 for 2-D and 3 for 3-D flows). Then

$$|A| + |B| + |C| = \begin{pmatrix} d \cdot c & u & v & w & 0 \\ u & v{+}w{+}c & 0 & 0 & 0 \\ v & 0 & u{+}w{+}c & 0 & 0 \\ w & 0 & 0 & u{+}v{+}c & 0 \\ 0 & 0 & 0 & 0 & u{+}v{+}w \end{pmatrix} .$$

For $u, v, w << c$

$$(|A| + |B| + |C|)^{-1} \sim \text{diag}\frac{1}{c}\left(\frac{1}{d}, 1, 1, 1, \frac{c}{|u| + |v| + |w|}\right)$$

The inverse of the Jacobian is a diagonal matrix (up to errors of $O(M)$). On the other hand $\Delta\tau \sim \frac{1}{cd}$. So

$$\frac{(|A|+|B|+|C|)^{-1}}{\Delta\tau} \sim \text{diag}\left(1, d, d, d, \frac{cd}{|u| + |v| + |w|}\right)$$

Hence, for most of the variables (except for entropy) there is a (maximum) factor of $d \sim 2, 3$ variation in the time step.

3 Low-Speed Preconditioning

We consider the simplest low-speed preconditioning given in entropy variables by

$$P_Q = \text{diag}(\beta^2, 1, 1, 1, 1) \tag{12}$$

where $\beta^2 \sim M^2$ with a cutoff to prevent β from becoming too small [17, 21]. This cutoff is proportional to M_{ref}^2. The parameter β can be increased to account for viscous and time dependent terms [22, 23], but it is never allowed to become larger than unity. Hence, when the viscous terms become large enough or Δt is sufficiently small, then the low-speed preconditioning is turned off in the update scheme.

The low-speed preconditioning is designed to accelerate the convergence to a steady state by improving the condition number when the Mach number is small. In addition, the artificial viscosity or upwinding is adjusted to include the preconditioning. This changes the scales and improves the behavior of the artificial viscosity as the flow becomes more incompressible [16, 17, 18, 21]. When the contribution of the physical time step is large enough, then the preconditioning does not improve the convergence of the subiterations, but it is still useful for improving the accuracy of the numerical solution [18, 21]. Hence, we use different values of β for the preconditioning on the left hand side and for the artificial viscosity. In particular the effect of the physical time step is larger in the update portion than in the artificial viscosity. Similarly, the minimum cutoff differs between the preconditioner for convergence acceleration and the artificial viscosity. One can evaluate the artificial viscosity in either conservation variables or primitive variables (p, u, v, w, T), which are more appropriate for near incompressible flows (see [23] for more details).

We also combine the low-speed preconditioning with Jacobi preconditioning by starting with an artificial viscosity based on the low-speed preconditioning for increased accuracy and then forming the Jacobi preconditioning for better convergence rates [20]. Let P be the low-speed preconditioning, and let the physical time derivative be represented by Eq. (2). Then the preconditioned scheme (showing only the second-order dissipation) is given by

$$\begin{aligned} P_J^{-1} &= {P_Q}^{-1}\left(\zeta P_Q + |P_Q A| + |P_Q B| + |P_Q C|\right) \\ P_J^{-1}\Delta w &= \frac{c_t w^{n+1} + Q(w^n, w^{n-1}, ...)}{\Delta t} + f_x + g_y + h_z \\ &\quad + \frac{h}{2}\left[{P_Q}^{-1}(|P_Q A| w_x)_x + {P_Q}^{-1}(|P_Q B| w_y)_y + {P_Q}^{-1}(|P_Q C| w_z)_z\right] \equiv \text{Res} \end{aligned} \tag{13}$$

We redefine the entropy variables for the preconditioned scheme so as to include β. Then

$$Q = (\frac{dp}{\rho\beta c}, u, v, w, T) \tag{14}$$

In these Q variables the matrices P_Q and $|P_Q A| + |P_Q B| + |P_Q C|$ are symmetric. Define

$$\begin{aligned}
\lambda_+ &= \frac{(\beta^2+1)U + \sqrt{(\beta^2-1)^2U^2 + 4\beta^2c^2}}{2} \\
\lambda_- &= \frac{(\beta^2+1)U - \sqrt{(\beta^2-1)^2U^2 + 4\beta^2c^2}}{2} \\
\lambda_{\max} &= \frac{(\beta^2+1)|U| + \sqrt{(\beta^2-1)^2U^2 + 4\beta^2c^2}}{2} \\
|\Lambda_+| &= \max(|\lambda_+|, \epsilon_n\lambda_{\max}) \\
|\Lambda_-| &= \max(|\lambda_-|, \epsilon_n\lambda_{\max}) \\
|\Lambda_0| &= \max(|U|, \epsilon_l\lambda_{\max}) \\
R_1 &= \frac{(\lambda_+ - U)|\Lambda_+| - (\lambda_- - U)|\Lambda_-|}{\lambda_+ - \lambda_-} \\
S_1 &= \frac{(\lambda_+ - U)|\Lambda_-| - (\lambda_- - U)|\Lambda_+|}{\lambda_+ - \lambda_-} \\
R_2 &= \beta c\frac{|\Lambda_+| - |\Lambda_-|}{\lambda_+ - \lambda_-} \\
R_3 &= S_1 - |\Lambda_0|
\end{aligned}$$

where U is the contravariant velocity, and ϵ_n and ϵ_l are constants. Typical values are $\epsilon_n = 0.3$ and $\epsilon_l = 0.1$ for the artificial viscosity and $\epsilon_n = \epsilon_l = 0.3$ within the Jacobi preconditioning. Then

$$|P_Q A| = \begin{pmatrix}
R_1 & \hat{S_x}R_2 & \hat{S_y}R_2 & \hat{S_z}R_2 & 0 \\
. & |\Lambda_0| + \hat{S_x}^2 R_3 & \hat{S_x}\hat{S_y}R_3 & \hat{S_x}\hat{S_z}R_3 & 0 \\
. & . & |\Lambda_0| + \hat{S_y}^2 R_3 & \hat{S_y}\hat{S_z}R_3 & 0 \\
. & . & . & |\Lambda_0| + \hat{S_z}^2 R_3 & 0 \\
. & . & . & . & |\Lambda_0|
\end{pmatrix} \tag{15}$$

Similar expressions can be derived for $|P_Q B|$ and $|P_Q C|$. The update scheme, Eq. (13), can be written in generalized entropy variables, Q, as

$$\Delta w = \frac{\partial w}{\partial Q}\left(\zeta P_Q + |P_Q A| + |P_Q B| + |P_Q C|\right)^{-1} P_Q \frac{\partial Q}{\partial w} Res \tag{16}$$

4 Far Field Boundary Conditions

In the far field an artificial boundary is constructed to limit the domain to a finite size. For well-posedness it necessary to impose boundary conditions on this artificial surface. Standard theory for hyperbolic partial differential equations dictates that at an inflow boundary in two- (three-) dimensions that 3 (4) conditions need to be specified and one boundary condition should be determined from the inside. At an outflow boundary, the situation is reversed.

For the standard non-preconditioned code these boundary conditions are based on one dimensional Riemann invariants [7, 14]. One drawback with the above technique is that the Riemann invariants depend on the speed of sound c. Hence, as the Mach number goes to zero, this approach becomes less valid. Computations confirm that this type of boundary treatment does not yield convergent results for the preconditioned code at sufficiently low Mach numbers. In particular, for incompressible flow this is not a valid technique. Hence, we have used a different approach for the preconditioned code [19].

At inflow, we specify the velocity components and temperature and extrapolate the pressure.

$$u = u_0, \quad v = v_0, \quad w = w_0, \quad T = \frac{p_0}{\rho_0}, \quad p = p_e \tag{17}$$

At outflow, we instead extrapolate the velocity components and temperature and specify the pressure.

$$u = u_e, \quad v = v_e, \quad w = w_e, \quad T = \frac{p_e}{\rho_e}, \quad p = p_0 \tag{18}$$

We can then calculate the density, momentum and energy variables. This procedure is now valid for incompressible flow and is well-posed for low speed compressible flow. It will be shown in the results section that while this boundary condition treatment works nicely for the steady state calculation, it gives poor accuracy for the dual time stepping algorithm.

5 Results

5.1 RAE2822 airfoil

We consider steady state flow about a two-dimensional RAE2822 airfoil. We first consider a transonic case with an inflow Mach number of $M_\infty = 0.73$ and an angle of attack $\alpha = 2.79^\circ$. The Reynolds number is 6.5 million. The turbulent flow is simulated with a Baldwin-Lomax turbulence model using a 320×64 C grid. We use a FMG multigrid with 50 iterations on 3 coarse meshes and 300 iterations on the finest mesh. Without low-speed preconditioning we use a five stage Runge-Kutta scheme with the standard coefficients (.25, .16667, .375, .5, 1.0). With low-speed preconditioning the Runge-Kutta coefficients are chosen as (.25, .18, .40, .51, 1.0). The artificial viscosity is partially updated only on the odd stages of the multistage method using fractions of .56 and .44 on the third and fifth stages, respectively. There is also a small difference in the dependence of β on the viscous terms for the artificial viscosity [22]. All the runs are made with the same variable residual smoothing coefficients that depend on the aspect ratio [11, 30]. The low-speed preconditioner affects the artificial viscosity. Besides the artificial viscosity, the only difference caused by the preconditioners is in the update stage where the residual is multiplied by the low-speed preconditioning matrix, P, followed by either $\Delta\tau$ or else by $(|PA| + |PB|)^{-1}$. In figure 1 we compare the convergence rate for the standard code with that produced with the Jacobi preconditioning. For this transonic flow the Jacobi preconditioning results in a significant improvement in the convergence rate. In figure 2 the drag coefficient for the same case also shows improved convergence with Jacobi preconditioning. The steady state drag calculated is independent of the Jacobi preconditioning.

We next consider a low speed case with $M_\infty = 0.2$. In figure 3 we compare the impact of low-speed preconditioning without the Jacobi preconditioning on the convergence of residuals. The low-speed preconditioning improves the convergence rate significantly, with preconditioning based on primitive variables, (p, u, v, T), being the better of the two. We note that when the preconditioner is based on primitive variables, then the residual displayed is the pressure residual while for conservation variables we display the density residual. However, we have never found a fundamental change when choosing other residuals to display. In figure 4 we show the results using Jacobi preconditioner either by itself or else combined with low-speed preconditioning. The Jacobi preconditioning improves both the non-preconditioned and the low-speed preconditioned cases. The cpu time required to obtain 4 orders of magnitude reduction in residual with the preconditioned code is approximately 30% of the total run time compared to the baseline code. The gains are more significant, especially with Jacobi preconditioning, if one wishes to reduce the residual to much lower levels.

In figures 5 and 6 we show the convergence of the drag for this case. We note that the steady state drag is the same for the non-preconditioned algorithm and Jacobi preconditioning since we only change the update procedure but not the residual. However, the low-speed preconditioning is included in the artificial dissipation and so the low-speed preconditioning changes the steady state numerical solution. In [18] we show that only the low-speed preconditioned residual gives the correct solution as the Mach number approaches zero.

We conclude with results for an inflow Mach number of 0.05 presented in figures 7 and 8. We see that for this low Mach number flow that the Jacobi preconditioning by itself helps relatively little. In contrast, using only the low-speed preconditioning gives a large improvement in the residual convergence. The combined low-speed and Jacobi preconditioning gives a dramatic improvement, yielding 12 orders of magnitude decrease in the residual in 300 multigrid cycles (1500 explicit sweeps through the grid). Such rapid convergence represents a significant improvement for viscous, turbulent flow computations on high aspect ratio grids.

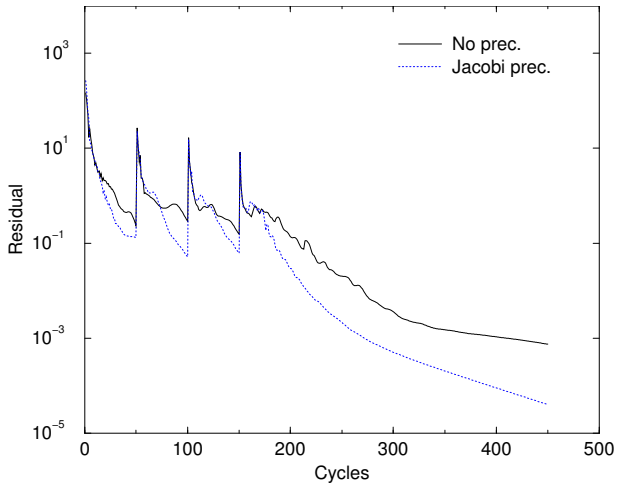

Figure 1: Residual history, RAE2822 airfoil, $\mathbf{M_\infty = 0.73}$

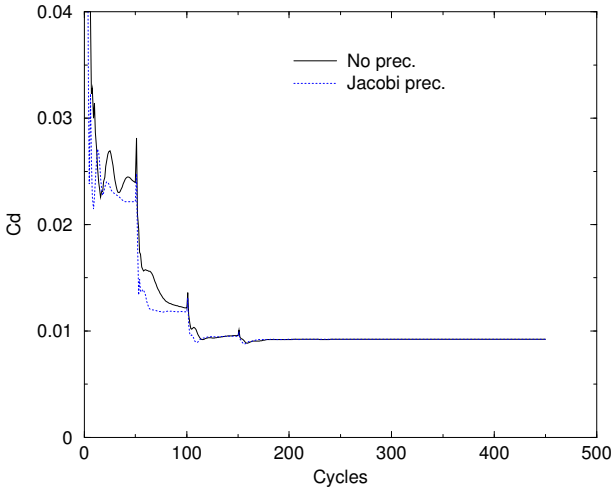

Figure 2: Drag history, RAE2822 airfoil, $\mathbf{M_\infty = 0.73}$

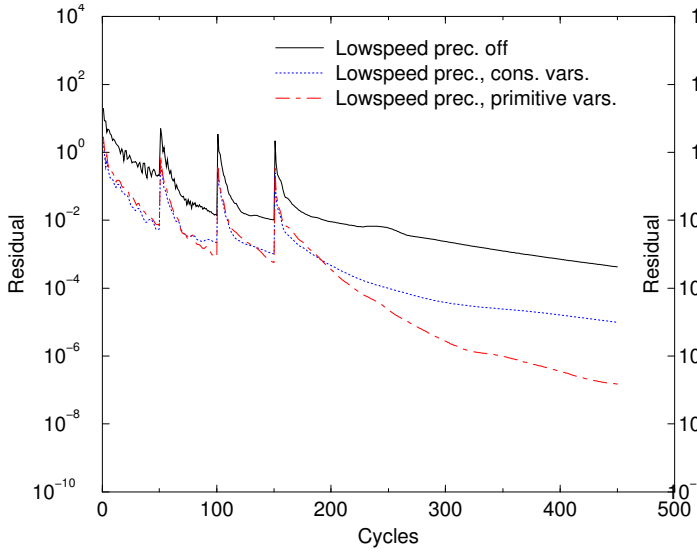

Figure 3: Residual history RAE2822 airfoil (without Jacobi preconditioner), $\mathbf{M_\infty = 0.2}$

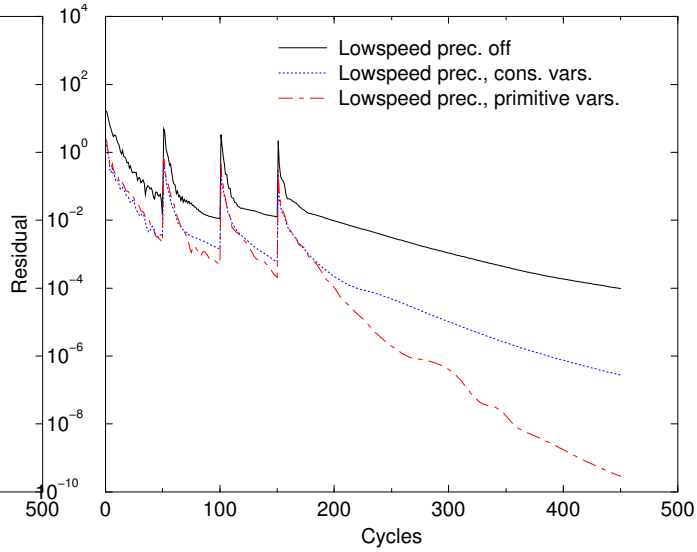

Figure 4: Residual history RAE2822 airfoil (with Jacobi preconditioner), $\mathbf{M_\infty = 0.2}$

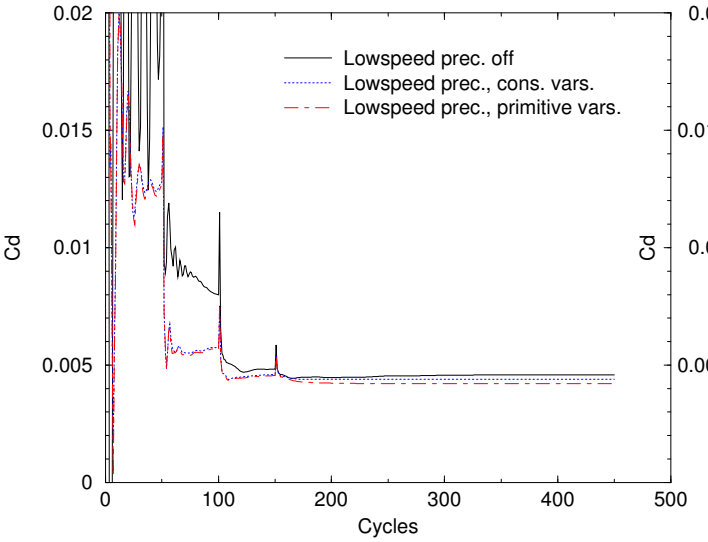

Figure 5: Drag history RAE2822 airfoil (without Jacobi preconditioner), $\mathbf{M_\infty = 0.2}$

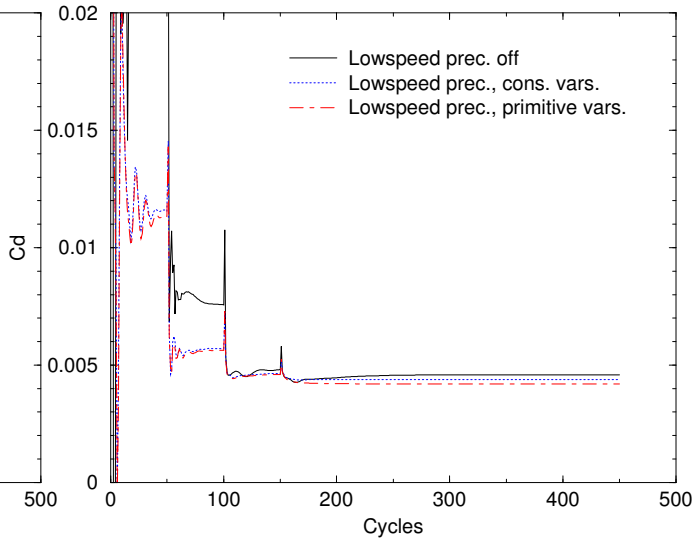

Figure 6: Drag history RAE2822 airfoil (with Jacobi preconditioner), $\mathbf{M_\infty = 0.2}$

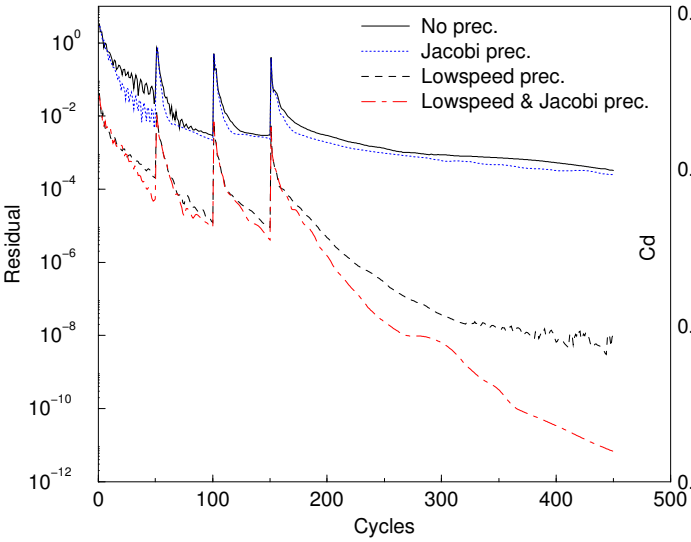

Figure 7: Residual history, RAE2822 airfoil, $\mathbf{M_\infty = 0.05}$

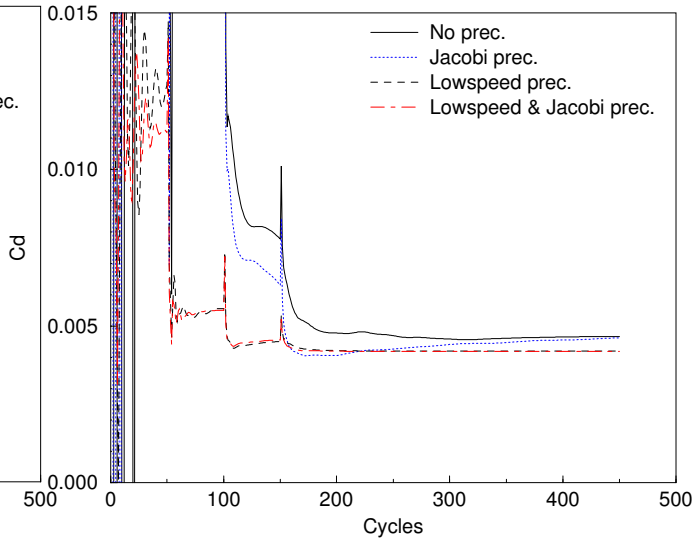

Figure 8: Drag history, RAE2822 airfoil, $\mathbf{M_\infty = 0.05}$

5.2 Vortex

To apply these concepts to a time dependent case we consider an inviscid vortex propagating in the x direction. To present the exact solution we first define several quantities where c_0 and c_1 are free parameters, and x_0 is the center of vortex at t=0.

$$c_2 = \frac{\gamma - 1}{2\gamma}\frac{\rho_0}{p_0}c_0^2 c_1^2$$
$$arg = 1 - (x - x_0 - c_0 t)^2 - y^2$$

The solution is given in [4] (with a different normalization) by

$$u(x, y, t) = c_0 \left(1 - c_1 y e^{0.5\ arg}\right)$$
$$v(x, y, t) = c_0 c_1 (x - x_0 - c_0 t) e^{0.5\ arg}$$
$$\rho(x, y, t) = \rho_0 \left(1 - c_2 e^{arg}\right)^{\frac{1}{\gamma - 1}}$$
$$p(x, y, t) = p_0 \left(1 - c_2 e^{arg}\right)^{\frac{\gamma}{\gamma - 1}}$$

Based on the nondimensionalization in TLNS3D we choose

$$c_0 = M \qquad c_1 = \frac{1}{2\pi} \qquad x_0 = 10$$
$$\rho_0 = 1 \qquad p_0 = 1 \quad \text{So} \quad c_2 = \frac{\gamma - 1}{2} c_1^2 M^2$$

We note that this vortex is both divergence free and also satisfies the compressible Euler equations. We define $\text{CFL}_{\text{phys}} = \frac{c_0 \Delta t_{\text{phys}}}{\Delta x}$. We choose $M = 10^{-3}$. A uniformly spaced Cartesian grid consisting of 97x33 nodes is used for these computations. Each physical time step is solved with 50 subiterations of the dual time-stepping. In table 1 we present the errors for the nonpreconditioned and preconditioned algorithms. The preconditioned algorithm is based on primitive variables. The preconditioned algorithm based on conservative variables gives similar accuracy but appears to be more stable for the dual time-stepping. In [23] we showed that for steady state problems the preconditionings were equally stable and the one based on primitive variables was slightly more accurate but that does not seem to be true for the dual time-stepping problem. All runs are for 16 physical time steps. Hence, as we change Δt_{phys} we change the physical time of the output. However, we are mainly concerned with comparing the errors of the non-preconditioned and preconditioned codes which are measured at the same physical time for a given Δt_{phys}.

Δt_{phys}	CFL	no precon	precon	K_τ	K_β
31.25	.5	$.105\ 10^{-4}$	$.236\ 10^{-5}$	1	2
62.5	1	$.126\ 10^{-4}$	$.296\ 10^{-5}$	1	5
125	2	$.142\ 10^{-4}$	$.375\ 10^{-5}$	1	10
250	4	$.159\ 10^{-4}$	$.653\ 10^{-5}$	15	12
312.5	5	$.168\ 10^{-4}$	$.923\ 10^{-5}$	18	15

Table 1: Vortex motion, M=.001, error in u component

The formula we use for calculating the artificial time step $\Delta\tau$ is given by

$$\frac{1}{\Delta\tau} = \frac{1}{\Delta\tau_{\text{ss}}} + \frac{1}{c_t K_\tau \Delta t_{\text{phys}}}$$

where $\Delta\tau_{\text{ss}}$ is the steady state, artificial time step which is a sum of inviscid and viscous contributions. For a CFL number above 1, the preconditioned code required increasing the K_τ in the above formula and also increasing β as a function of the physical time step. The coefficient K_τ continually increases as the CFL number increases, indicating that the relationship between $\frac{1}{\Delta\tau}$ and $\frac{1}{\Delta t}$ is not linear for the preconditioned algorithm. We note that $\Delta\tau$ is much larger for the preconditioned scheme, therefore the ratio of $\Delta\tau$ to Δt is quite different between the non-preconditioned and preconditioned cases.

The formula for β is given by

$$\beta^2 \sim M^2 + K_\beta \left(\frac{c_t \text{Vol}}{c\hat{q}\Delta t_{\text{phy}}}\right)^2 + M_{\text{ref}}^2$$

where $\hat{q}$ is a combination of velocities and face metrics [23], M is the local Mach number and M_{ref} is a reference Mach number, which is representative of free stream Mach number for low speed flows. Further time dependent applications are considered in [27].

5.3 Sensitivity to far-field boundary conditions

As described in Sec. 4, the far field boundary conditions differ between non-preconditioned and preconditioned algorithms. We first checked the importance of the change of boundary condition procedures by re-computing the RAE 2822 airfoil case described in sec. 5.1, using the preconditioned algorithm with the far field boundary conditions based on the Riemann invariants. This worked quite nicely for $M = 0.73$ and $M = 0.2$ and converged slightly faster. However, for $M = 0.05$ the code diverged without even reaching the finest grid in the full multigrid (FMG) sequence. However, when the boundary conditions given by Eqs. (17,18) are used, the preconditioned code converged nicely for this case.

We then tried these boundary conditions for the time dependent vortex flow. For the non-preconditioned algorithm the boundary conditions based on Riemann invariants worked as expected. For the preconditioned algorithm the boundary conditions of Eqs. (17,18) gave very poor accuracy, while the Riemann invariants gave the good accuracy, as shown in table 1. It is not clear why Eqs. (17,18) work well for the preconditioned scheme for steady state problems but give poor accuracy for unsteady flows. This needs to be investigated further.

6 Concluding Remarks

Jacobi and low-speed preconditioning techniques have been developed for a central difference algorithm to treat both steady state and time dependent problems. Jacobi preconditioning is shown to improve the efficiency of the baseline TLNS3D code for steady flow over a RAE 2822 airfoil at transonic Mach numbers. The efficiency gain is obtained on top of the efficiency gained by the residual smoothing and multigrid acceleration techniques which are integral parts of the TLNS3D code. For lower speed flows the convergence rate is significantly improved by combining the Jacobi and low-speed preconditionings. In addition for low Mach number flows, the low-speed preconditionings improves the accuracy of the steady state solution while the Jacobi preconditioning does not affect the steady state.

When time dependent problems are solved with a dual time step algorithm and a small physical time step, the preconditioning does not improve the convergence rate. Hence, the parameter β in the preconditioning is reduced to $\beta = 1$, i.e., the preconditioning is turned off in the update stage for sufficiently small physical time steps. However, the preconditioning does affect the artificial viscosity and does improve the accuracy for time dependent flows. Hence, we use different values of β in the update stage and in the artificial viscosity. The far field boundary condition needs to be altered for the preconditioning to work for steady state flows at low Mach numbers. However, for time dependent flows the non-preconditioned far field boundary condition seems to be more robust.

References

[1] S. Abarbanel and D. Gottlieb, *Time Splitting for Two- and Three-dimensional Navier-Stokes Equations with Mixed Derivatives*, J. Comput. Phys. 41:1-33 (1981).

[2] S. Allmaras, *Analysis of a Local Matrix Preconditioner for the 2-D Navier-Stokes Equations*, AIAA paper 93-3330 (1993).

[3] D.A. Caughey and A. Jameson, *Fast Preconditioned Multigrid Solution of the Euler and Navier-Stokes Equations for Steady Compressible Flows*, AIAA Paper 2002-0963 (2002).

[4] G. Erlebacher, M.Y. Hussaini and C.-W. Shu, *Interaction of a Shock with a Longitudinal Vortex* J. Fluid Mechanics, 337:129-153 (1997).

[5] K. Hosseini and J.J. Alonso, *Practical Implementation and Improvement of Preconditioning Methods for Explicit Multistage Flow Solvers*, AIAA paper 2004-0763 (2004).

[6] A. Jameson, W. Schmidt and E. Turkel, *Numerical Solutions of the Euler Equations by a Finite Volume Method using Runge-Kutta Time-Stepping Schemes*, AIAA Paper 81-1259 (1981).

[7] A. Jameson and T.J. Baker, *Solution of the Euler Equations for Complex Configurations*, AIAA paper 83-1929, July 1983.

[8] A. Jameson, *Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows past Airfoils and Wings*, AIAA Paper 91-1596 (1991).

[9] A. Jameson and D.A. Caughey, *How Many Steps are Required to Solve the Euler equations of Steady, Compressible Flow: In Search of a Fast Solution Algorithm*, AIAA Paper 2001-2673 (2001).

[10] D. Levy and E. Tadmor, *From Semidiscrete to Fully Discrete: Stability of Runge-Kutta Schemes by the Energy Method*, SIAM Review 40:40-73 (1998).

[11] L. Martinelli and A. Jameson, *Validation of a Multigrid Method for the Reynolds Averaged Equations*, AIAA Paper 88-0414, 1988.

[12] S.A. Pandya, S. Venkateswaran and T.H. Pulliam, *Implementation of Preconditioned Dual-Time Procedures in OVERFLOW*, AIAA paper 2003-0072 (2003).

[13] N.A. Pierce and M.B. Giles, *Preconditioned Multigrid Methods for Compressible Flow Codes on Stretched Meshes*, J. Comput. Phys. 136:425-445 (1997).

[14] J.L. Thomas and M.D. Salas, *Far-Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations*, AIAA paper 85-20, Jan. 1985.

[15] R.C. Swanson and E. Turkel E, *On Central Difference and Upwind Schemes*, J. Comput. Phys. 101:292-306 (1987).

[16] E. Turkel, *Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations*, J. Comput. Phys. 72:277 (1987).

[17] E. Turkel, *A Review of Preconditioning Methods for Fluid Dynamics*, Appl. Numer. Math., 12:257-284 (1993).

[18] E. Turkel, A. Fiterman and B. van Leer, *Preconditioning and the Limit to the Incompressible Flow Equations*, in Computing the Future: Frontiers of Computational Fluid Dynamics 1994, ed. D.A. Caughey and M.M. Hafez, Wiley Publishing, 215-234 (1994).

[19] E. Turkel, R. Radespiel and N. Kroll, *Assessment of Two Preconditioning Methods for Aerodynamic Problems*, Computers and Fluids, 26:613-634, 1997.

[20] E. Turkel, *Preconditioning-Squared Methods for Multidimensional Aerodynamics*, AIAA Paper 97-2025 (1997).

[21] E. Turkel, *Preconditioning Techniques in Computational Fluid Dynamics*, Annual Review of Fluid Mechanics, 31, 385-416 (1999).

[22] E. Turkel *Robust Preconditioning for Steady and Unsteady Viscous Flows*, AIAA paper 02-0962 (2002).

[23] E. Turkel and V.N. Vatsa, *Choice of Variables and Preconditioning for Time Dependent Problems*, AIAA Paper 2003-3692 (2003).

[24] B. van Leer, W.T. Lee and P.L. Roe, *Characteristic Time-Stepping or Local Preconditioning of the Euler Equations*, AIAA Paper 91-1552, 1991.

[25] V.N. Vatsa and B.W. Wedan, *Development of an Efficient Multigrid Code for 3-D Navier-Stokes Equations*, AIAA Paper 89-1791 (1989).

[26] V.N. Vatsa, M.D. Sanetrik and E.B. Parlette, *A Multigrid Based Multiblock Flow Solver for Practical Aerodynamic Configurations*. in Computing the Future: Frontiers of Computational Fluid Dynamics 1994, ed. D.A. Caughey and M.M. Hafez, Wiley Publishing, 414-447 (1994).

[27] V.N. Vatsa, E. Turkel and M. Carpenter, *Simulation of Synthetic Jets in Quiescent Air using URANS Equations*, AIAA paper 2004-4967.

[28] S. Venkateswaran and L. Merkle, *Dual Time Stepping and Preconditioning for Unsteady Computations*, AIAA paper 95-0078 (1995).

[29] S. Venkateswaran and L. Merkle, *Artificial Dissipation Control for Unsteady Computations*, AIAA paper 2003-3695 (2003).

[30] L.B. Wigton and R.C. Swanson, *Variable Coefficient Implicit Residual Smoothing*, 12th International Conference on Numerical Methods in Fluid Dynamics, 1990.