Eleventh International Conference on Computational Fluid Dynamics (ICCFD11), Maui, HI, USA, July 11-15, 2022

ICCFD11-2022-1702

Improving the Performance of a Compressible RANS Solver for Low and High Mach Number Flows

Sabet Seraj*, Anil Yildirim, Joshua L. Anibal, Joaquim R. R. A. Martins
*Corresponding author: sseraj@umich.edu

Department of Aerospace Engineering, University of Michigan, USA

Abstract: The aerodynamic design of aircraft often involves evaluating flow conditions that span low subsonic to transonic, or even supersonic Mach numbers. Compressible flow solvers are a natural choice for such design problems, but these solvers encounter reduced accuracy and efficiency at low Mach numbers. In addition, simulations with supersonic conditions can be challenging to converge because of large gradients in the flow field. This paper presents three contributions to address these issues in the context of an approximate Newton–Krylov solver for the Reynolds-averaged Navier–Stokes equations. First, we propose a method for scaling artificial dissipation that improves accuracy at low Mach numbers while retaining the simplicity of the original scalar dissipation scheme. Second, we show that characteristic time-stepping combined with an approximate Newton method can accelerate convergence for low Mach number flows by reducing the stiffness in the linear system for each Newton iteration. Third, we introduce a dissipation-based continuation method for flows with shocks that improves robustness and accelerates convergence without sacrificing accuracy. These methods can be used to make compressible flow solvers more accurate and efficient across low and high Mach number regimes.

Keywords: Local Preconditioning, Continuation Methods, Implicit Solvers.

1 Introduction

The aerodynamic design of aircraft often requires evaluating performance over a wide range of flow conditions. For example, the flight envelope of a supersonic transport aircraft can span Mach numbers from 0.25 to 2.0 [1]. A conventional transonic aircraft will fly through Mach numbers from 0.2 to 0.85 in a typical mission [2]. In addition, certain applications produce flow fields with a range of Mach numbers at a single operating condition. Helicopter rotors experience nearly incompressible flow at blade roots and compressible flow at the tips [3]. Nacelles in crosswind can experience supersonic flow around the inlet lips despite low subsonic flow outside the nacelle [4]. This motivates the need for CFD solvers that are accurate and efficient across a wide range of Mach numbers.

In theory, compressible flow solvers represent the true physics across all Mach numbers, making them a natural choice for aerodynamic design. In practice, these solvers commonly suffer from reduced accuracy and speed at low Mach numbers. Resolving this through low-speed preconditioning has been studied in depth by several authors for both explicit and implicit time-stepping schemes. Turkel [5] introduced a class of preconditioning matrices for low-speed flows based on analysis of the Euler equations. This approach modifies the time stepping to minimize the ratio of advective and acoustic wave speeds, which accelerates convergence. The same preconditioners can improve accuracy at low Mach numbers if the artificial dissipation in the scheme is based on preconditioned fluxes [6]. This approach has also been applied to viscous flows [7, 8, 4]. van Leer et al. [9] proposed a preconditioner for all Mach numbers that is also based on the idea of minimizing the spread in wave speeds. They interpreted this approach as using different time steps along different

characteristic directions, which led to the name "characteristic time-stepping". The authors demonstrated the preconditioner's effectiveness on the Euler equations at Mach numbers from 0.01 to 1.8 and on viscous flow at Mach 7.95. Lee [10] showed that preconditioning approaches based on the Euler equations work well for viscous flows in most practical cases.

Some work has also been done on low-speed preconditioning for Newton-based solvers. Newton-based solvers are fully implicit and offer more flexibility in addressing the accuracy and speed issues independently without introducing unwelcome stability restrictions. To speed up convergence with Newton–Krylov solvers, Knoll et al. [11] considered multiplicative Schwarz preconditioning and Weston et al. [12] used approximate block factorizations. Both of these papers take an algebraic approach to preconditioning. In contrast, Mary et al. [13] applied characteristic-based preconditioning to an approximate Newton solver. This work was limited to a maximum Reynolds number of 800 and primarily considered accuracy. These limitations suggest that there is unexplored potential for improved performance with Newton-based solvers using characteristic preconditioning methods.

In addition, high-speed flows have not been explored as extensively as low-speed cases. Kaushik et al. [14] and Olawsky et al. [15] found that Newton-based solvers can run into difficulties converging supersonic flows with second-order schemes compared to more dissipative first-order schemes. They resolved this by starting the solution with a first-order scheme before switching to a second-order scheme closer to convergence. This approach lacks reliability because it is not always clear when to switch to a second-order scheme for a given case. A related approach for improving convergence for flows with shocks is to increase the artificial dissipation in the discretization. For upwind schemes, this usually corresponds to modifying the entropy fix. For example, changing the entropy fix to be more dissipative is an option in the widely used FUN3D code [16]. The equivalent for central schemes is increasing the artificial dissipation constants, as done by Öhrman [17] for example. Increasing dissipation improves robustness in exchange for reduced accuracy. We desire an automated approach that increases robustness without sacrificing accuracy.

In this work, we introduce three methods to improve the efficiency and accuracy of Newton-based compressible flow solvers across low and high Mach number regimes. In Sec. 3, we describe a simple method for improving low-speed accuracy by scaling the artificial dissipation of a scalar dissipation scheme. This method preserves the simplicity of the original scheme and is easy to implement. We then formulate a characteristic time-stepping approach for an approximate Newton solver in Sec. 4. Characteristic time-stepping improves nonlinear convergence rates by reducing the stiffness of the linear system at each nonlinear iteration. Lastly, we present a dissipation-based continuation method in Sec. 5 that addresses nonlinear convergence difficulties that arise from high Mach number flows. This method leverages the idea that more dissipative methods are easier to converge by starting with high artificial dissipation and smoothly reducing the dissipation to the desired level as the solution converges.

2 Baseline RANS Solver

The CFD solver we use in this work is ADflow [18], a second-order finite volume code for multiblock and overset structured meshes. We solve the steady, compressible Reynolds-averaged Navier–Stokes (RANS) with the Spalart–Allmaras (SA) turbulence model [19]. The flow is assumed to be fully turbulent.

The Navier–Stokes equations can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V Q\mathrm{d}V + \oint_{\partial V} \vec{F}_c \cdot \vec{n}\mathrm{d}A - \oint_{\partial V} \vec{F}_d \cdot \vec{n}\mathrm{d}A = 0, \tag{1}$$

where $Q = [\rho, \rho u, \rho v, \rho w, \rho E]^T$ represents the conservative variables, $\vec{F}_c$ is the convective flux, and $\vec{F}_d$ is the diffusive flux. This system is augmented by the SA equation, which adds the turbulent contribution to the diffusive flux. Discretizing in space with a cell-centered finite volume method, the equation for a cell i is

$$\frac{\mathrm{d}Q_i}{\mathrm{d}t}V_i + \sum_f \hat{F}_c \Delta A_f - \sum_f \hat{F}_d \Delta A_f = 0, \tag{2}$$

where the discretized fluxes, $\hat{F}_c$ and $\hat{F}_d$, are summed over the faces of the cell. We use the Jameson–Schmidt–

Turkel (JST) scheme [20, 21] to discretize the convective flux and a central scheme with Green–Gauss gradients for the diffusive flux.

2.1 Artificial dissipation formulation

JST is a central scheme with a combination of second and fourth order artificial dissipation terms for stability. We now describe the artificial dissipation formulation in the I mesh direction. The formulation for the J and K directions follow similarly. The second-order dissipation coefficient at a face between cell i and $i+1$ is given by

$$\epsilon^{(2)}_{i+\frac{1}{2}} = \kappa_2 \Upsilon_{i+\frac{1}{2}} \hat{\Lambda}^I_{i+\frac{1}{2}}, \tag{3}$$

and the fourth-order coefficient is given by

$$\epsilon^{(4)}_{i+\frac{1}{2}} = \max\left(0, \kappa_4 \hat{\Lambda}^I_{i+\frac{1}{2}} - c_4 \epsilon^{(2)}_{i+\frac{1}{2}}\right). \tag{4}$$

By default, we use $\kappa_2 = 0.25$, $\kappa_4 = 0.0156$, $c_4 = 1$. $\Upsilon_{i+1/2}$ is the shock sensor at the face, which ensures that second-order dissipation is only applied near shocks and fourth-order dissipation is only applied away from shocks. The shock sensor at the face is computed as the maximum of the neighboring cell sensor values:

$$\Upsilon_{i+\frac{1}{2}} = \max\left(\Upsilon_i, \Upsilon_{i+1}\right). \tag{5}$$

The shock sensor value in each cell is based on the local change in entropy, s:

$$\Upsilon_i = \frac{|s_{i+1} - 2s + s_{i-1}|}{|s_{i+1} + 2s + s_{i-1}|}. \tag{6}$$

$\hat{\Lambda}^I_{i+1/2}$ is the spectral radius of the flux Jacobian in the I direction at the face and is computed as the average of the neighboring cell radii:

$$\hat{\Lambda}^I_{i+\frac{1}{2}} = \frac{1}{2}\left(\hat{\Lambda}^I_i, \hat{\Lambda}^I_{i+1}\right). \tag{7}$$

To compute the spectral radius in the I direction, we first compute the isotropic spectral radius, Λ, in each direction. The isotropic spectral radius in the I direction is

$$\Lambda^I = \vec{U} \cdot \vec{n}^I + c|\vec{n}^I|, \tag{8}$$

where $\vec{U} = [u, v, w]^T$ is the local velocity vector, c is the local speed of sound, and $\vec{n}^I$ is the face normal projected in the I direction. Similarly, for the J and K directions, we have

$$\Lambda^J = \vec{U} \cdot \vec{n}^J + c|\vec{n}^J|, \tag{9}$$

$$\Lambda^K = \vec{U} \cdot \vec{n}^K + c|\vec{n}^K|. \tag{10}$$

We then compute a scaled spectral radius that is appropriate for high aspect ratio cells as suggested by Martinelli [22]:

$$\hat{\Lambda}^I = \Lambda^I \left(1 + \left(\frac{\Lambda^J}{\Lambda^I}\right)^{0.67} + \left(\frac{\Lambda^K}{\Lambda^I}\right)^{0.67}\right). \tag{11}$$

The artificial dissipation terms are proportional to the spectral radius, so appropriately scaling the spectral radius is critical for solver convergence and solution accuracy.

2.2 Solver algorithm

To converge the semi-discrete form (Eq. 2) to a steady solution, we define the residual in each cell, R_i, as the time-derivative term and drive it to zero:

$$R_i = \frac{\mathrm{d}Q_i}{\mathrm{d}t} V_i = \sum_f \hat{F}_d \Delta A_f - \sum_f \hat{F}_c \Delta A_f = 0. \tag{12}$$

The baseline nonlinear solver algorithm we use in this work is the approximate Newton–Krylov (ANK) method with pseudo-transient continuation implemented in ADflow [23]. The solver is primarily designed for robustness and speed at transonic flow conditions. The solution update at each iteration of the ANK solver, $\Delta Q^{(n)}$, is obtained by solving the linear system

$$\left[\left(T^{-1}\right)^{(n)} + \left(\frac{\partial R}{\partial Q}\right)^{(n)} \right] \Delta Q^{(n)} = -R(Q^{(n)}), \tag{13}$$

where R is the residual vector, $Q^{(n)}$ is the state vector, and T is the time-step matrix. T is a diagonal matrix, where the diagonal terms for each cell are the time step, Δt_i, that results in the desired global CFL number. The global CFL number is initially small and is increased as the solution converges. A complete description of the ANK algorithm is given by Yildirim et al. [23].

2.3 Accuracy and cost comparisons

We compare the methods presented in this paper to the baseline solver in terms of accuracy and cost. Unless stated otherwise, all solutions are converged to a total residual of 10^{-10} relative to the freestream residual. To provide a hardware-independent measure of cost, we report total CPU times in TauBench work units [24]. We run all computations on NASA's Aitken supercomputer using Cascade Lake nodes, where one work unit is equal to 3.9078 processor seconds. We determine this by running TauBench ten times with the command `mpiexec -np 1 ./TauBench -n 250000 -s 10` and computing the average run time.

3 Scaling Artificial Dissipation to Improve Accuracy at Low Mach Numbers

Artificial dissipation is essential for nonlinear convergence but can be detrimental for accuracy. The JST scalar dissipation scheme is generally more dissipative than second-order upwind schemes [25]. Central schemes that are less dissipative than scalar dissipation include matrix dissipation [25] and the convective-upstream-split-pressure scheme [26]. However, all of these central and upwind schemes are primarily designed for transonic flows and run into accuracy issues at low Mach numbers because of improperly scaled artificial dissipation [6]. One common approach to scale the dissipation is to multiply the flux Jacobian by a local preconditioning matrix. For JST, this modifies the spectral radius and consequently the dissipation. We discuss using a local preconditioning matrix to accelerate convergence in Sec. 4. To improve low-speed accuracy, we opt to directly modify the spectral radius. We discuss some advantages of this approach over preconditioning matrices later in this section.

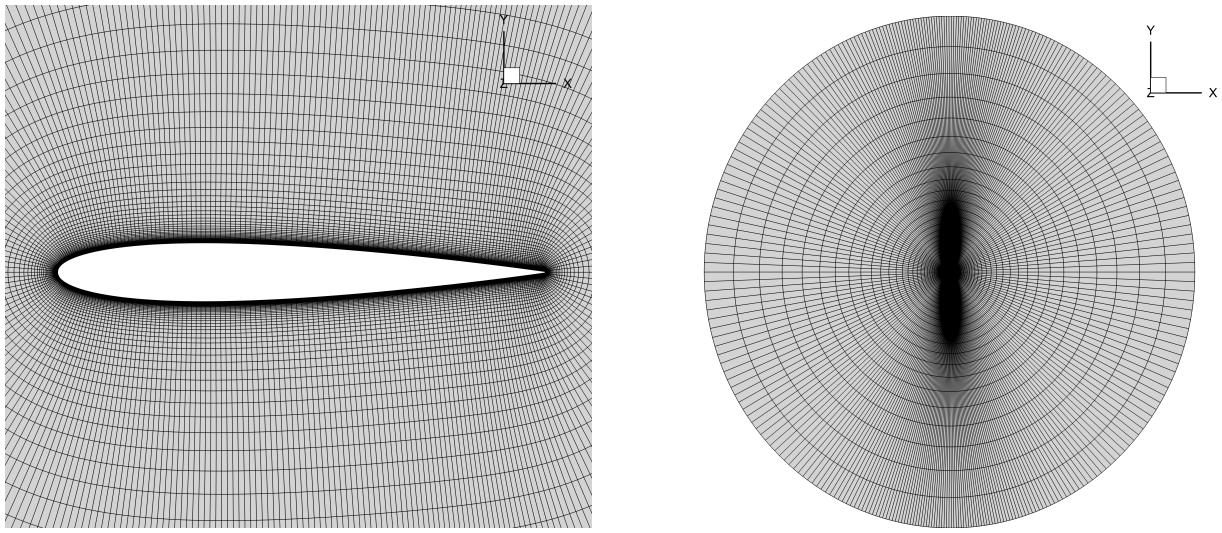
In the low Mach number limit, the advective contribution to the spectral radius (Eq. 8) is $O(1)$, whereas the acoustic contribution is $O(1/M)$ [27]. Rieper [28] showed that artificial dissipation must be independent of Mach number for accuracy in the incompressible limit. To scale the artificial dissipation for low Mach numbers, we introduce the acoustic scaling factor ζ in the isotropic spectral radius:

$$\Lambda^I = \vec{U} \cdot \vec{n}^I + \zeta c |\vec{n}^I|. \tag{14}$$

Using this formulation, we can select values for ζ that reduce or eliminate the dependence of artificial dissipation on the Mach number.

To study the effect of ζ on accuracy, we consider a NACA 0012 airfoil at a Reynolds number of 10^6, 3 deg angle of attack, and Mach numbers from 0.01 to 0.4. We use an O-mesh with 296 cells around the airfoil

and 128 cells in the offwall direction, for a total of 37888 cells (Fig. 1). The initial offwall spacing is 10^{-6} chord lengths, and the farfield is 100 chord lengths away from the airfoil.



(a) Nearfield (b) Farfield

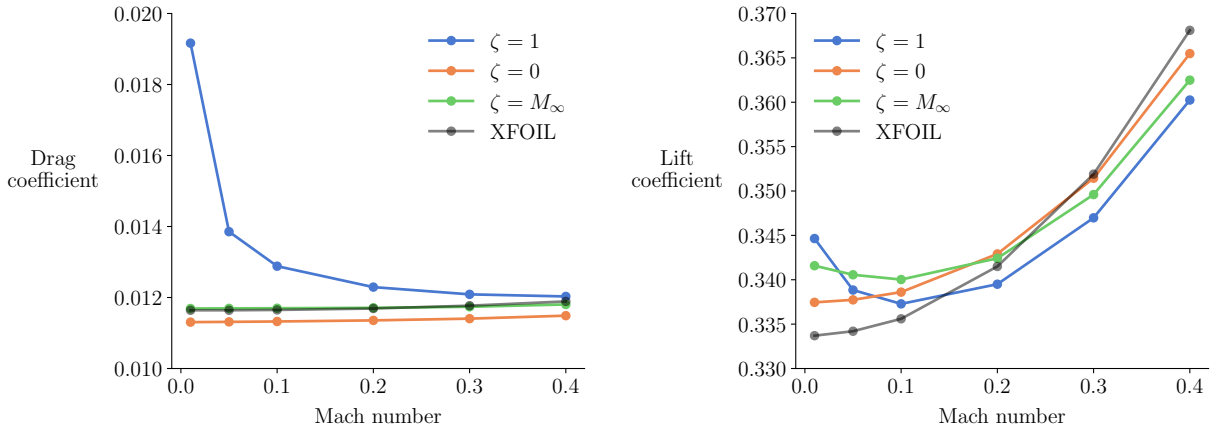
Figure 1: NACA 0012 O-mesh with 296×128 cells

To evaluate accuracy at different values of ζ, we compare the RANS results with predictions using XFOIL [29], a 2D panel method coupled with an integral boundary layer model. We use 300 panels for the inviscid discretization in the XFOIL analyses. Increasing the number of panels to 360 gives the same coefficient values to 4 significant digits. To match the fully turbulent RANS simulations, we force transition at the 0.3% chord location. This is the smallest positive value we could use for the trip location without encountering numerical difficulties.

Fig. 2a compares the drag for the baseline ($\zeta = 1$), no acoustic contribution ($\zeta = 0$), and scaling the acoustic contribution by the freestream Mach number ($\zeta = M_\infty$). With no scaling, drag diverges as the Mach number is reduced. The drag overprediction at low Mach numbers for the baseline is caused by a nonphysical suction peak on the trailing edge surface (Fig. 3). These nonphysical pressure fluctuations are a direct result of improperly scaled dissipation, as shown by Guillard and Viozat [30]. Chen et al. [31] also found similar trailing edge pressure spikes for low-speed inviscid flow. Scaling the acoustic contribution removes the trailing edge artifact and results in more accurate drag values. In addition, the change in drag flattens out as the Mach number is reduced. We find that scaling by the freestream Mach number yields the most accurate drag predictions compared to XFOIL.

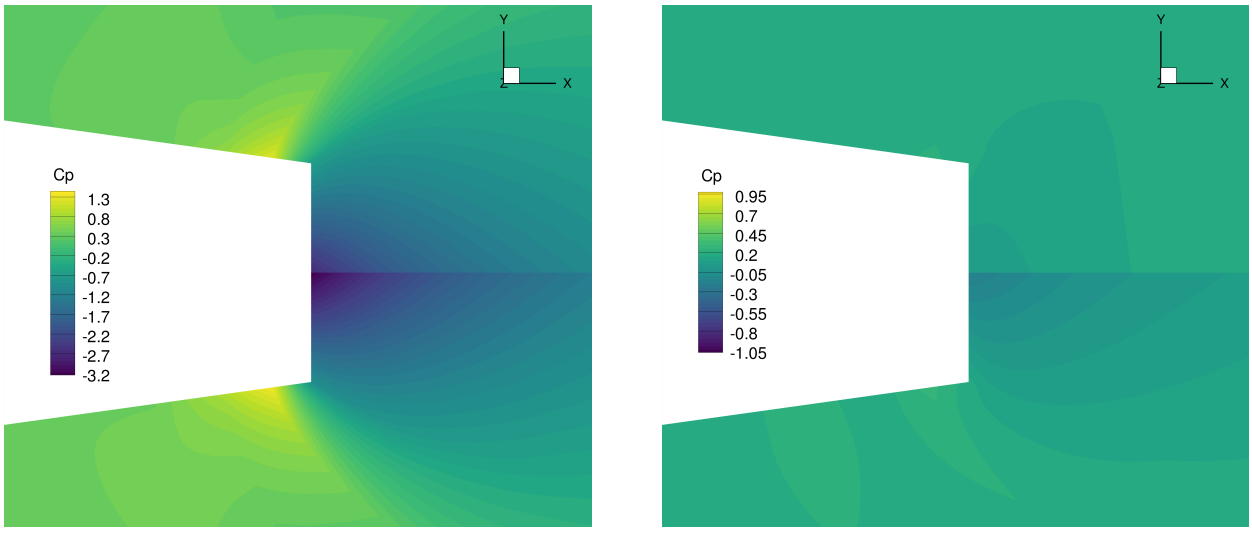
The baseline errors in lift at low Mach numbers are not as large as for drag (Fig. 2b), but the lift still diverges as the Mach number approaches zero. Reducing artificial dissipation avoids divergence. The value of ζ is not critical for lift prediction because the differences are primarily caused by discretization errors. We run a mesh refinement study at Mach 0.1 to demonstrate this. In addition to the 296×128 mesh, we use a 592×256 mesh and a 148×64 mesh for the study. Figure 4 shows the drag and lift for these three meshes, as well as the Richardson extrapolation [32, 33] using the two finest meshes. The Richardson extrapolation provides an estimate of the values we would obtain on an infinitely fine mesh. The extrapolated drag values depend on ζ. The extrapolated drag for $\zeta = 1$ is 7.5% higher than the drag for $\zeta = 0$ and 5% higher than $\zeta = M_\infty$. This means that properly scaling the dissipation is important for drag prediction even for fine meshes. On the other hand, the extrapolated lift values are within 0.5% of each other. The differences in lift with ζ are large only for the 148×64 mesh. However, such a coarse mesh is outside the asymptotic range [33] in terms of mesh size and should not be used in general. The mesh refinement study also shows that drag for $\zeta < 1$ is less sensitive to mesh size than baseline. As a result, scaling the artificial dissipation can indirectly speed up analyses by allowing for coarser meshes at the same accuracy as the baseline.

One advantage of scaling with ζ over prior approaches is that it is easy to implement. Adding the scaling factor is a minimal extension to the JST scalar dissipation formulation. Another advantage is that



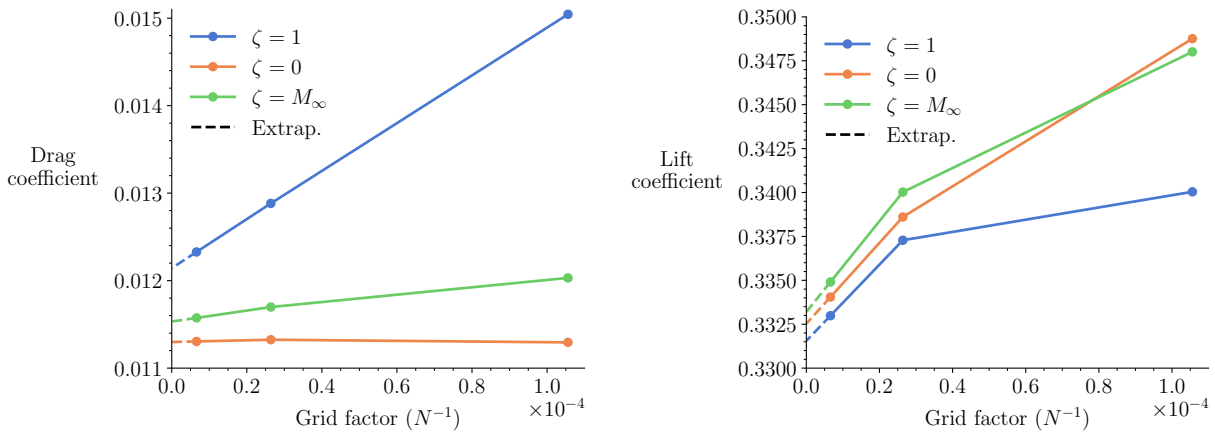
(a) (b)

Figure 2: Scaling artificial dissipation improves accuracy



(a) $\zeta = 1$ (b) $\zeta = 10^{-2}$

Figure 3: Reducing artificial dissipation removes the spurious pressure spikes at the trailing edge (Mach 0.01)



(a) (b)

Figure 4: Mesh refinement at Mach 0.1

the artificial dissipation can be varied continuously. Figure 5 shows that reducing ζ beyond 10^{-3} for Mach 0.01 incurs large penalties in cost. Depending on the case and solver, ζ lower than a certain value may not converge at all. Because ζ is a continuous parameter, the user can decide what value results in the desired combination of robustness, cost, and accuracy. This is similar to deciding the appropriate mesh size for a given problem.

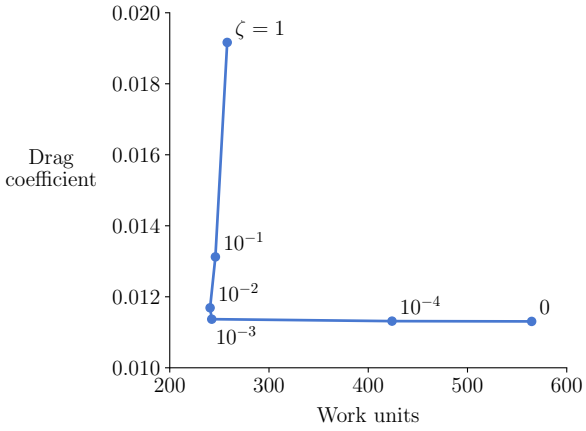


Figure 5: Reducing ζ eventually leads to an increase in cost (Mach 0.01)

In these results, we have only considered using the same value for ζ in all cells. Alternatively, we could use local values for ζ. For example, instead of the freestream Mach number, we could use the local Mach number in each cell. We find that this results in similar accuracy and convergence to using a global value of $\zeta = 0$. This is because the local Mach number near the airfoil is close to zero, and the cells closest to the airfoil dictate the overall behavior of the solver. However, local values may be more applicable for cases where a freestream value is not defined. The proposed scaling approach with either global or local ζ is easy to implement and improves the accuracy of compressible solvers at low Mach numbers.

4 Accelerating Convergence at Low Mach Numbers with Characteristic Time-Stepping

In addition to lower accuracy, compressible flow solvers face slower convergence at low Mach numbers. For the NACA 0012 case from Sec. 3 with the 296×128 mesh and $\zeta = 1$, the solution for Mach 0.01 is three times slower than Mach 0.4 (Fig. 6). The slower convergence is the result of the difference in magnitude between the advective and acoustic wave speeds. For explicit solvers, this results in a more restrictive upper bound on the time step. For Newton-based solvers, this is reflected in the stiffness of the linear system that is solved at each nonlinear iteration.

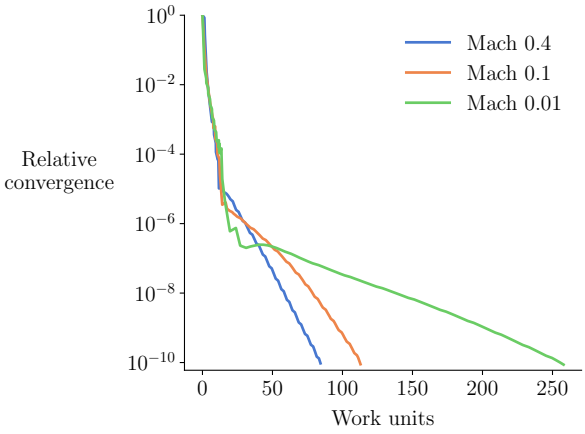


Figure 6: Cost increases as Mach number decreases (NACA 0012)

The baseline ANK solver defaults are tuned for compressible flow. However, we use settings that are beneficial for low Mach number flows for all cases with freestream Mach numbers of 0.4 or lower in this paper. The main changes are using more linear solver iterations and a more accurate preconditioner for the linear system. For higher Mach numbers and flows with shocks, solving the linear system more tightly is not as beneficial for improving nonlinear convergence.

To further accelerate convergence at low Mach numbers, we take a local preconditioning or characteristic time-stepping (CTS) approach. This approach involves modifying the time-stepping terms to equalize the advective and acoustic wave speeds. We refer to this approach as characteristic time-stepping to emphasize that we are not modifying the artificial dissipation using the same preconditioner. To apply CTS to the ANK solver, we multiply the time-step matrix by the preconditioning matrix P. The linear system with the ANK step (Eq. 13) requires the inverse of the time-step matrix, so it is more convenient to work directly with the inverse of the preconditioner, P^{-1}:

$$\left[\left(P^{-1}T^{-1}\right)^{(n)} + \left(\frac{\partial R}{\partial Q}\right)^{(n)}\right] \Delta Q^{(n)} = -R(Q^{(n)}). \tag{15}$$

We make one change to adapt the preconditioner for use with pseudo-transient continuation. When the CFL number is low, the linear system is not stiff and the baseline solver regularly outperforms CTS. To take advantage of this, we use a blended preconditioner based on the CFL number:

$$P^{-1}_{\text{blend}} = \left(\frac{\text{CFL}}{\text{CFL}_{\text{max}}}\right) P^{-1} + \left(1 - \frac{\text{CFL}}{\text{CFL}_{\text{max}}}\right) I. \tag{16}$$

With this blending, the solver initially acts like the baseline and reaches the fully preconditioned method when the CFL number reaches its maximum value.

P^{-1} is a block diagonal matrix. Each diagonal block corresponds to one cell, and we denote the pre-

conditioner for cell i as P_i^{-1}. Several different preconditioning matrices can be used in this formulation. We consider the Turkel preconditioner [5], denoted here as P_{T}, and the van Leer–Lee–Roe (VLR) preconditioner [9], denoted as P_{VLR}.

4.1 Turkel preconditioner

It is convenient to write the Turkel preconditioner in terms of the Euler symmetrizing variables,

$$\tilde{Q} = [p/(\rho c), u, v, w, p - c^2\rho]^T. \tag{17}$$

With these variables, the preconditioner for cell i is

$$\tilde{P}_{\mathrm{T},i}^{-1} = \begin{bmatrix} \frac{1}{M_{\mathrm{T}}^2} & 0 & 0 & 0 & 0 \\ \frac{\alpha u}{cM_{\mathrm{T}}^2} & 1 & 0 & 0 & 0 \\ \frac{\alpha v}{cM_{\mathrm{T}}^2} & 0 & 1 & 0 & 0 \\ \frac{\alpha w}{cM_{\mathrm{T}}^2} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{18}$$

where we use M_{T} instead of the usual symbol β to avoid confusion with the β introduced for the VLR preconditioner in Sec. 4.2. As in prior work, we define M_{T} as a truncated form of the local Mach number:

$$M_{\mathrm{T}}^2 = \min\left(1, \max(M^2, \phi M_\infty^2)\right). \tag{19}$$

For $\phi > 0$, this definition of M_{T}^2 prevents the matrix from becoming ill-conditioned in cells where the local Mach number is nearly zero. The ill-conditioning is mainly an issue at stagnation points. We find that $\phi = 10^{-4}$ provides the best performance while avoiding convergence issues. Larger values of ϕ do not show speedup compared to the baseline, whereas smaller values fail to resolve the stagnation point issue.

We also prevent M_{T}^2 from being greater than 1 and define α as

$$\alpha = 1 - M_{\mathrm{T}}^{20} \tag{20}$$

to eliminate preconditioning for locally supersonic flow. For $M \geq 1$, $M_{\mathrm{T}}^2 = 1$ and $\alpha = 0$, which is equivalent to no preconditioning. For subsonic Mach numbers, α quickly approaches 1, which results in better conditioning than $\alpha = 0$. Typically, the preconditioner with $\alpha = 0$ is used for its improved robustness despite not providing the optimal conditioning in the low Mach number limit. The difference in robustness has been attributed to $\alpha = 1$ having lower artificial dissipation than $\alpha = 0$ [6]. However, we do not use the same scaling for the dissipation as we do for the time step, so we avoid this robustness issue.

Finally, we transform the matrix to conservatives variables through

$$P_{\mathrm{T},i}^{-1} = S\tilde{P}_{\mathrm{T},i}^{-1}S^{-1}, \tag{21}$$

where S is the transformation matrix from symmetrizing to conservative variables:

$$S = \begin{bmatrix} \frac{\rho}{c} & 0 & 0 & 0 & -\frac{1}{c^2} \\ \frac{\rho u}{c} & \rho & 0 & 0 & -\frac{u}{c^2} \\ \frac{\rho v}{c} & 0 & \rho & 0 & -\frac{v}{c^2} \\ \frac{\rho w}{c} & 0 & 0 & \rho & -\frac{w}{c^2} \\ c\rho\left(\frac{M^2}{2} + \frac{1}{\gamma - 1}\right) & \rho u & \rho v & \rho w & -\frac{M^2}{2} \end{bmatrix}. \tag{22}$$

4.2 van Leer-Lee-Roe (VLR) preconditioner

The VLR preconditioner is usually expressed in terms of the Euler symmetrizing variables and in a flow-aligned coordinate frame:

$$\bar{\tilde{P}}^{-1}_{\mathrm{VLR},i} = \begin{bmatrix} \frac{\beta^2+\tau}{M^2_{\mathrm{VLR}}\tau} & \frac{1}{M} & 0 & 0 & 0 \\ \frac{1}{M} & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\tau} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\tau} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{23}$$

where

$$\beta = \begin{cases} \sqrt{1-M^2_{\mathrm{VLR}}}, M < 1 \\ \sqrt{M^2_{\mathrm{VLR}}-1}, M \geq 1 \end{cases}, \qquad \tau = \begin{cases} \sqrt{1-M^2_{\mathrm{VLR}}}, M < 1 \\ \sqrt{1-1/M^2_{\mathrm{VLR}}}+\epsilon, M \geq 1 \end{cases}. \tag{24}$$

We define M^2_{VLR} like M^2_{T} for the Turkel preconditioner but without any restrictions for supersonic flow:

$$M^2_{\mathrm{VLR}} = \max(M^2, \phi M^2_\infty). \tag{25}$$

We use $\phi = 10^{-4}$, which is the same value we use with the Turkel preconditioner. We also set $\epsilon = 10^{-4}$ to avoid dividing by zero for sonic flow. We transform the preconditioner first to Cartesian coordinates with the rotation matrix Z and then to conservatives variables with the state transformation matrix S (Eq. 22):

$$P^{-1}_{\mathrm{VLR},i} = SZ\bar{\tilde{P}}^{-1}_{\mathrm{VLR},i}Z^TS^{-1}. \tag{26}$$

The rotation matrix [34] is

$$Z = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos\alpha\cos\theta & -\sin\theta & -\sin\alpha\cos\theta & 0 \\ 0 & \cos\alpha\sin\theta & \cos\theta & -\sin\alpha\sin\theta & 0 \\ 0 & \sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{27}$$

where

$$\sin\theta = \frac{v}{\sqrt{u^2+v^2}}, \qquad \sin\alpha = \frac{w}{\sqrt{u^2+v^2+w^2}},$$

$$\cos\theta = \frac{u}{\sqrt{u^2+v^2}}, \qquad \cos\alpha = \frac{\sqrt{u^2+v^2}}{\sqrt{u^2+v^2+w^2}}.$$

4.3 The importance of dissipation scaling

We first apply CTS to the NACA 0012 case at Mach 0.01. For this section, we only show results with the Turkel preconditioner. However, we see similar trends with VLR. We find that the performance strongly depends on the choice of ζ (Fig. 7). Using $\zeta = M_\infty$ results in faster convergence compared to the baseline, whereas CTS is slower than baseline for $\zeta = 1$ and $\zeta = 0$. For $\zeta = 1$, CTS takes more than twice as long as baseline to converge and exhibits noisy convergence below a relative residual of 10^{-7}. Convergence with CTS is noisier than baseline even for cases where CTS is faster. This is related to the preconditioner becoming ill-conditioned at stagnation points. We find that the noise is worse for lower freestream Mach numbers (Fig. 8). In addition, using larger values of ϕ reduces the noise but results in slower convergence. The noise is largest for $\zeta = 1$ because of the nonphysical stagnation regions shown in Fig. 3a. We conclude that applying local preconditioning to the time step is problematic when the flow solution is locally nonphysical.

The slowdown with CTS is less severe for $\zeta = 0$ than $\zeta = 1$ but still indicates that the solver is not working as intended. As mentioned before, removing too much dissipation can be detrimental to solver robustness. In this case, the solver still converges but performs poorly.

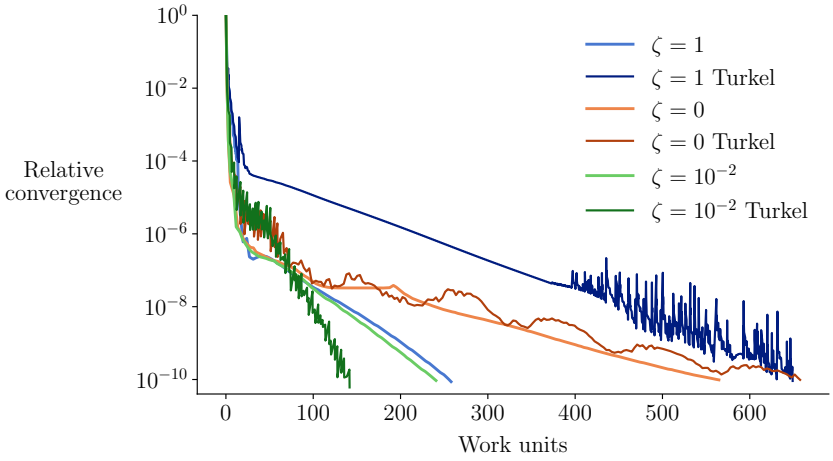


Figure 7: CTS reduces cost for $\zeta = 10^{-2}$ (NACA 0012 at Mach 0.01)

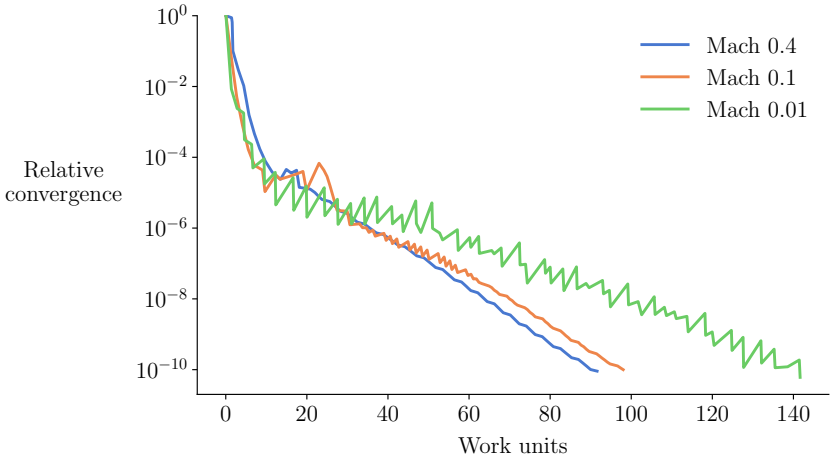
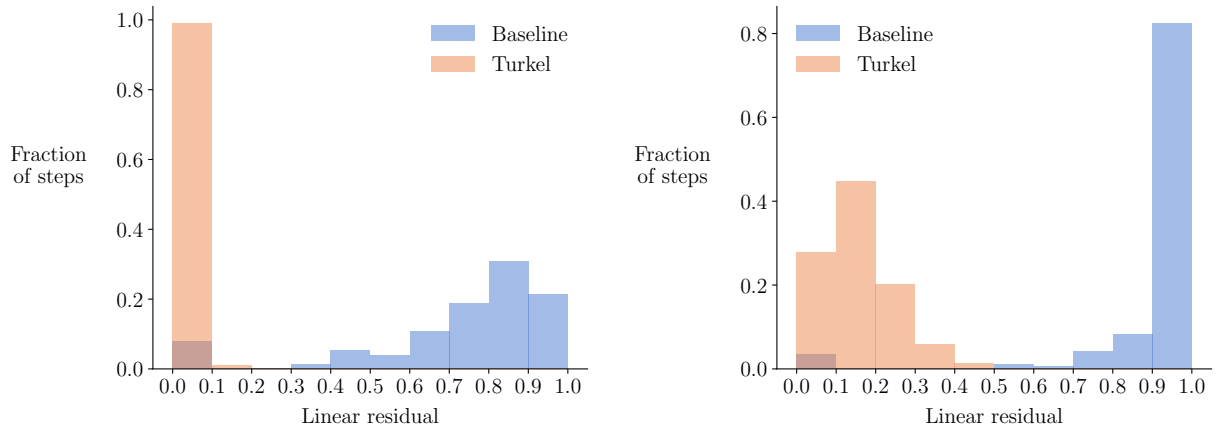


Figure 8: CTS convergence is noisier at lower Mach numbers (NACA 0012, $\zeta = M_\infty$, Turkel)

We can look at linear convergence to see how CTS achieves speedup for $\zeta = M_\infty$. The linear system for each ANK step is solved using the generalized minimum residual (GMRES) method [35]. The target linear residual is 0.05. We also set the maximum number of GMRES iterations to 100. If the iteration limit is reached before the linear residual target is met, the solver will continue with the partially converged step. We use the linear residual as an indicator of the linear system stiffness. For $\zeta = M_\infty$, CTS reduces the linear residuals such that they are all less than 0.1 (Fig. 9a). This results in faster nonlinear convergence. However, the same is not true for $\zeta = 0$ (Fig. 9b). Despite reducing the linear residuals, nonlinear convergence is not improved with CTS. As a result, we use $\zeta = M_\infty$ for all baseline and CTS cases with freestream Mach numbers of 0.4 or lower in Sec. 4.4 and Sec. 4.5.



(a) $\zeta = 10^{-2}$ (b) $\zeta = 0$

Figure 9: CTS improves linear convergence but this does not guarantee improved nonlinear convergence (NACA 0012 at Mach 0.01)

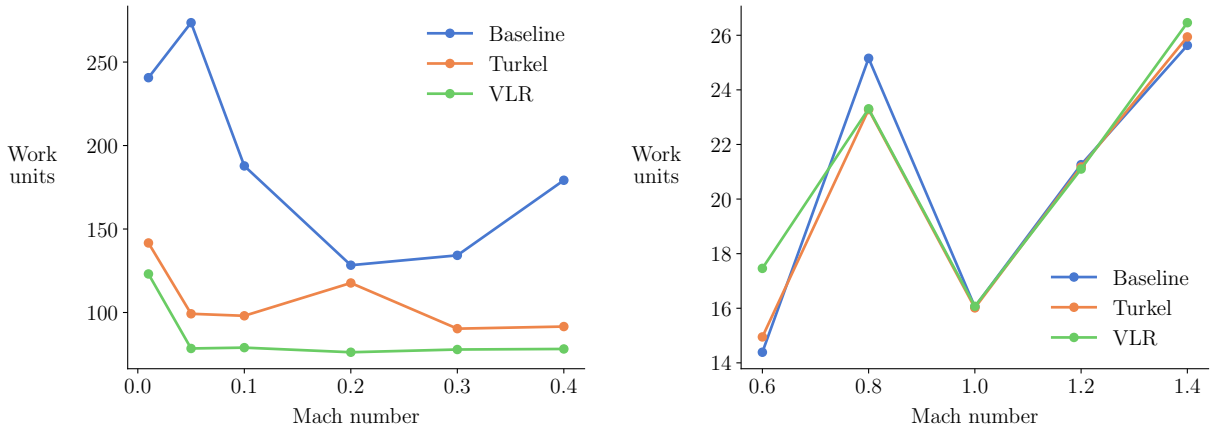
4.4 NACA 0012 Mach number sweep

We now compare Turkel and VLR preconditioning with the baseline for a range of Mach numbers. Both preconditioners converge faster than baseline for all Mach numbers we tried between 0.01 to 0.4 (Fig. 10a). The largest speedups are for Mach 0.1 and lower, where Turkel is 41%-64% faster than baseline. VLR is consistently faster than Turkel and achieves 49%-71% speedup compared to baseline in this regime. There is also a large speedup for Mach 0.4, but we do not expect to see the same speedup if the baseline solver is tuned for compressible flow.

We also compare CTS and baseline for Mach numbers between 0.6 to 1.4. We use $\zeta = 1$ for these cases because we are no longer in the low Mach number regime. We also tune the solver for compressible flow. These higher Mach number cases converge faster if we switch to a Newton–Krylov (NK) solver after converging to a relative residual of 10^{-5} with ANK. The NK solver has no time-step matrix and solves the flow and turbulence equations simultaneously. The lack of the time-step matrix means that CTS has no effect. However, this switch ensures a fair cost comparison in which the baseline solver is not restricted to less effective methods. The performance of Turkel and VLR is similar to the baseline at higher Mach numbers (Fig. 10b). This is the expected behavior for the Turkel preconditioner because we turn off preconditioning for locally supersonic flow. VLR is designed to be an all-speed preconditioner but also performs similarly to baseline for higher Mach numbers. This is because the difference in wave speeds and the resulting linear system stiffness is not an issue for Mach numbers around 1. We could try Mach numbers much higher than 1, but this runs into other challenges. We discuss these challenges and propose a solution in Sec. 5.

4.5 Delta wing

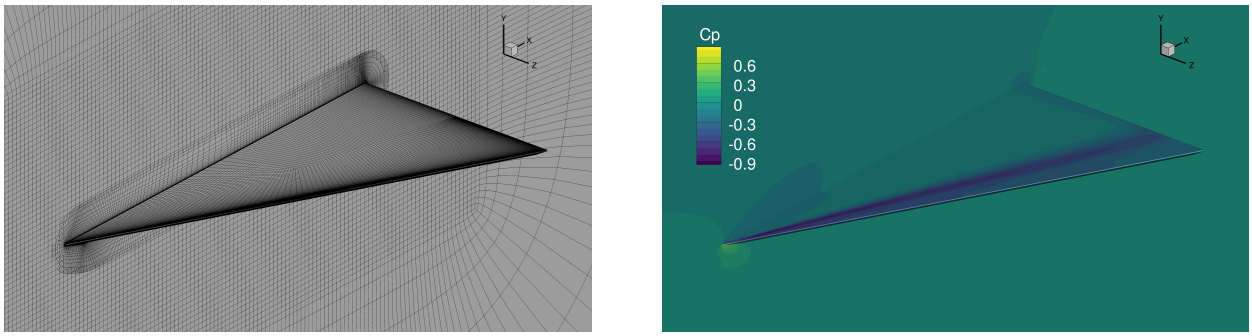
There are also low Mach number cases for which CTS does not result in any speedup. We demonstrate this on an aspect ratio 2 delta wing at a Mach number of 0.083, Reynolds number of 5.9×10^5, and angle of attack of 5 deg (Fig. 11). The geometry and flow conditions are taken from experimental work by Jarrah and Ashley [36]. We use an overset mesh with 1.9 million cells (Fig. 11a). More details on the mesh are given by Seraj and Martins [37]. For this case, stagnation point flow is restricted to a small region near the wing apex (Fig. 11b). Despite the low freestream Mach number, the flow field does not have enough cells with nearly zero local Mach number to make the global linear system stiff. As a result, CTS produces nearly identical convergence behavior as the baseline (Fig. 12). This also explains why the linear residuals are lower than the NACA 0012 baseline case. We show only Turkel results for clarity, but VLR also gives nearly identical results.



(a) Low Mach

(b) High Mach

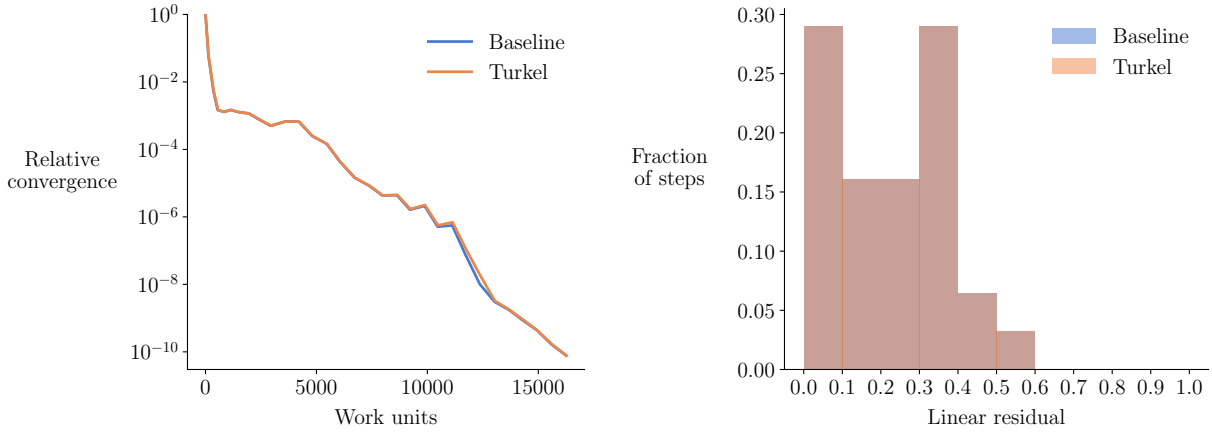
Figure 10: CTS reduces cost for low Mach numbers (NACA 0012)



(a) Overset mesh with 1.9 million cells

(b) Pressure contours at Mach 0.083, $\alpha = 5°$

Figure 11: Delta wing geometry from Jarrah and Ashley [36]



(a)

(b)

Figure 12: CTS is nearly identical to baseline for the delta wing

5 Dissipation-Based Continuation for Flows with Shocks

Flows with shocks pose different challenges for solvers than the low Mach number flows we have discussed so far. Shocks can move within the domain as the solution converges, causing large changes in cells' states. To improve robustness, the ANK solver employs a physicality check that limits the update step size such that the density and energy do not change by more than 20% in any cell. One important outcome of this approach is that the density and energy cannot become negative. Despite this added robustness, ANK can still perform poorly or fail to converge when the step sizes become small.

We demonstrate the effect of step size on convergence using transonic and supersonic flow over a NACA 0012 airfoil. At Mach 3.0, the step sizes are severely limited compared to Mach 0.75, which slows down convergence (Fig. 13). For the NACA 0012 cases in this section, we use the 296×128 mesh from Sec. 3. The Reynolds number is 10^7 and the angle of attack is 3 deg, unless stated otherwise. As with the high Mach number cases in Sec. 4.4, we use the NK solver during the final stages of convergence for all cases in this section. We switch to the NK solver after converging to a relative residual of 10^{-8} with ANK.

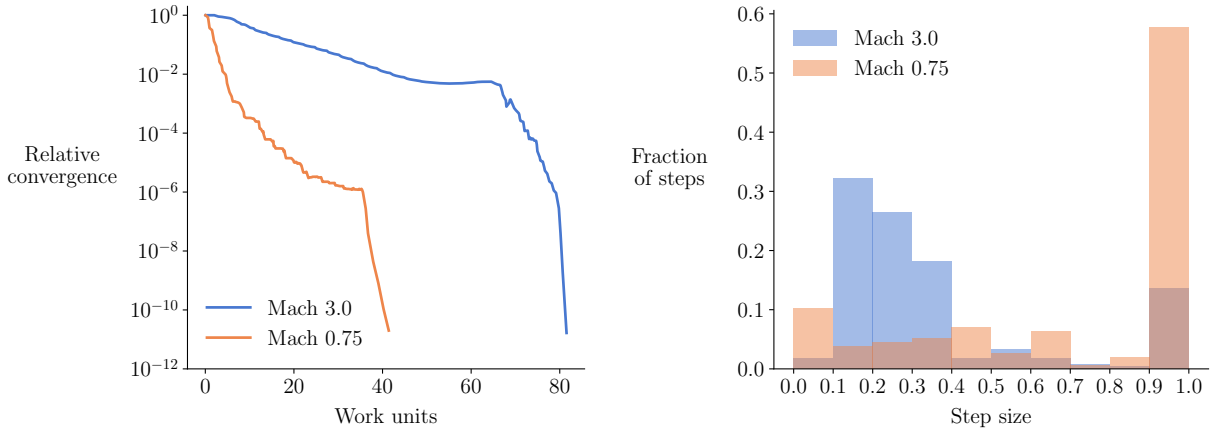


Figure 13: Limited step sizes slow down convergence for supersonic flows (NACA 0012)

One approach to accelerating convergence is to loosen the physicality check to accept larger changes in the states. However, this reduces the robustness of the solver and can often lead to stalled convergence. Another approach is to increase the second-order artificial dissipation constant, κ_2. This improves convergence by reducing the spatial gradient in the states but has the disadvantage of reducing solution accuracy. To accelerate convergence while preserving accuracy, we propose a dissipation-based continuation (DBC) method where κ_2 starts high and is continuously reduced as the solution converges.

DBC was previously explored by Hicken et al. [38, 39]. The DBC approach presented here differs from this prior work in that we do not use it as a general globalization strategy. We retain the shock sensor (Eq. 6) for the second-order dissipation, so DBC is only applied near shocks. In addition, we use DBC in combination with pseudo-transient continuation for globalization. A related but less automated approach is to start the solution with a first-order scheme and switch to a second-order scheme at some point during the solution [14, 15].

5.1 DBC formulation

We write the DBC formulation as

$$\kappa_2^{(n)} = \kappa_2 + f_c^{(n)} \kappa_2^c, \tag{28}$$

where $\kappa_2^{(n)}$ is the dissipation constant at nonlinear iteration n, κ_2 is the desired final dissipation constant, κ_2^c is the initial value of the additional dissipation, and $f_c^{(n)}$ is the continuation parameter at iteration n. We

require that the continuation parameter starts at 1 and approaches 0 as the solution converges. We choose to compute the continuation parameter using a generalized sigmoid function:

$$f_c^{(n)} = \frac{1}{1 + e^{-\sigma\left(\log_{10}\left(\eta_{\text{rel}}^{(n)}\right)+\lambda\right)}}, \tag{29}$$

where σ is the sharpness parameter, λ is the midpoint parameter, and $\eta_{\text{rel}}^{(n)}$ is the relative convergence at iteration n. The parameters σ and λ determine the shape of the sigmoid. Larger values of σ will result in a steeper drop in dissipation. We call λ the midpoint parameter because the additional dissipation is equal to $0.5\kappa_2^c$ at a relative residual of $10^{-\lambda}$. We use $\sigma = 3$ and $\lambda = 3$, which results in the continuation function shown in Fig. 14. When the solution is converged, the additional dissipation is small enough to recover the solution of the original problem. The shape parameters generally do not need to be tuned for each case. However, the appropriate value for κ_2^c depends on the Mach number. We find that $\kappa_2^c = 0.2M_\infty$ works well, and we use this value for all cases presented here.

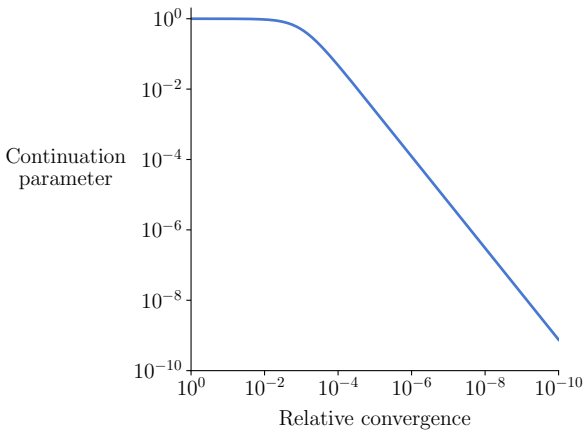


Figure 14: The additional dissipation decreases smoothly as the flow converges

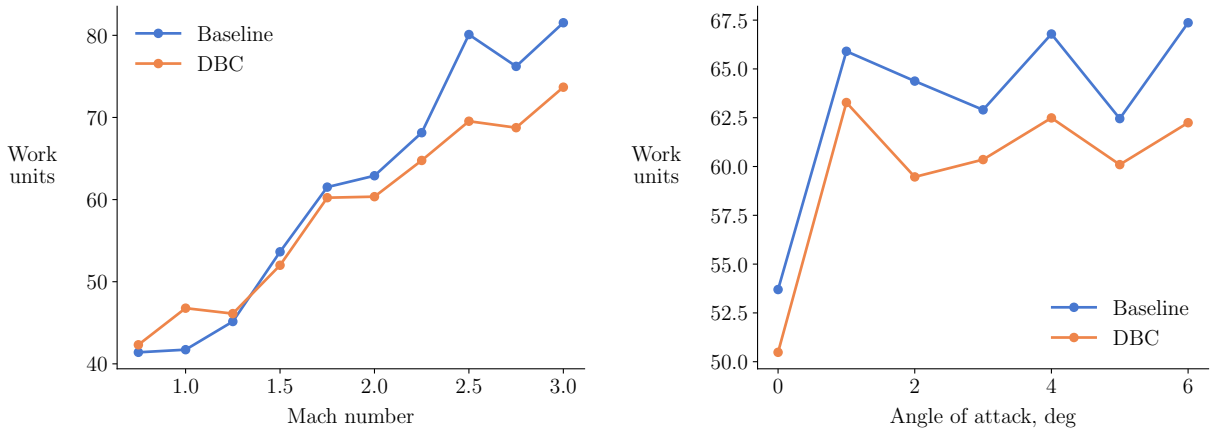
5.2 NACA 0012

We first compare the baseline solver and DBC for the NACA 0012 case at Mach numbers from 0.75 to 3.0 at a 3 deg angle of attack (Fig. 15a). DBC is between 3% and 13% faster than baseline for Mach numbers of 2.0 and above. Similarly, DBC is between 4% and 8% faster than baseline at Mach 2.0 and angles of attack from 0 to 6 deg (Fig. 15b). DBC increases the number of near unit steps, which improves convergence (Fig. 16). The converged lift and drag values with DBC match the baseline values to 8 significant digits.

5.3 Supersonic transport configuration

The advantages of DBC are more obvious when applied to a more complex case. We now consider a supersonic transport (SST) configuration with a cranked arrow wing, T-tail, and canard (Fig. 17). We consider flow at Mach numbers of 1.8 and 0.95, which represent realistic supersonic and transonic cruise conditions for the next generation of SSTs [1]. Both cases have a 4 deg angle of attack. DBC converges both cases, whereas the baseline solver does not converge either (Fig. 18). We also compare DBC to the baseline solver with κ_2 increased from the default value of 0.25 to the initial DBC value of $0.25 + 0.2M_\infty$. For Mach 1.8, increasing κ_2 converges faster than DBC. However, this solution is also less accurate because of the increased dissipation near shocks. For Mach 0.95, the solver still stalls with the higher κ_2, albeit after about 3 more orders of convergence.

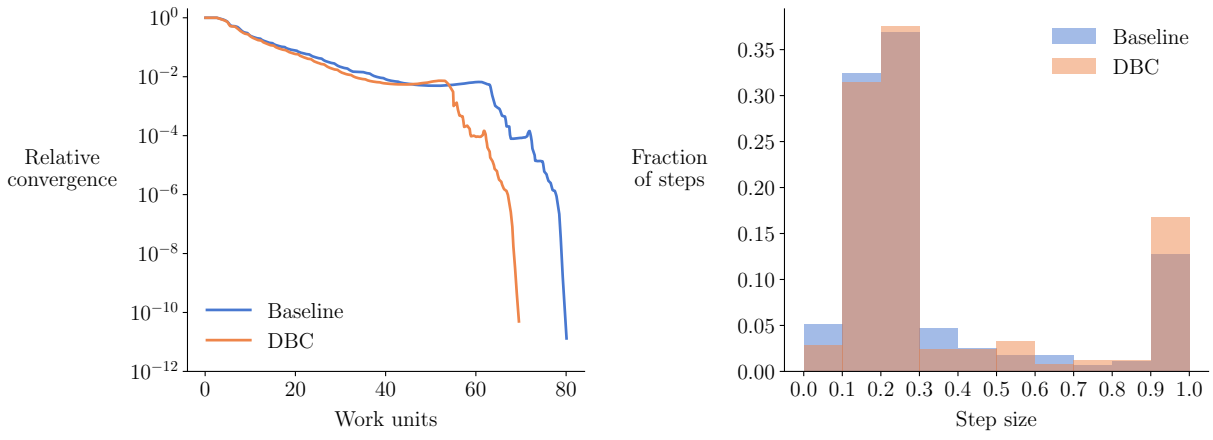
In some cases, robustness is more valuable than faster convergence. The SST mesh used here is the same as the one used by Seraj and Martins [40] for aerodynamic shape optimization studies. Reliably



(a) Varying Mach at $\alpha = 3°$

(b) Varying α at Mach 2

Figure 15: DBC offers slight speedups for the NACA 0012 case



(a)

(b)

Figure 16: DBC improves convergence by increasing step sizes (NACA 0012 at Mach 2.5)

converging the flow at different design points is essential for a well-behaved optimization [41]. DBC offers more robustness than the baseline solver for complex geometries at high Mach numbers, making it more suitable for many-query scenarios such as parameter sweeps or design optimization.

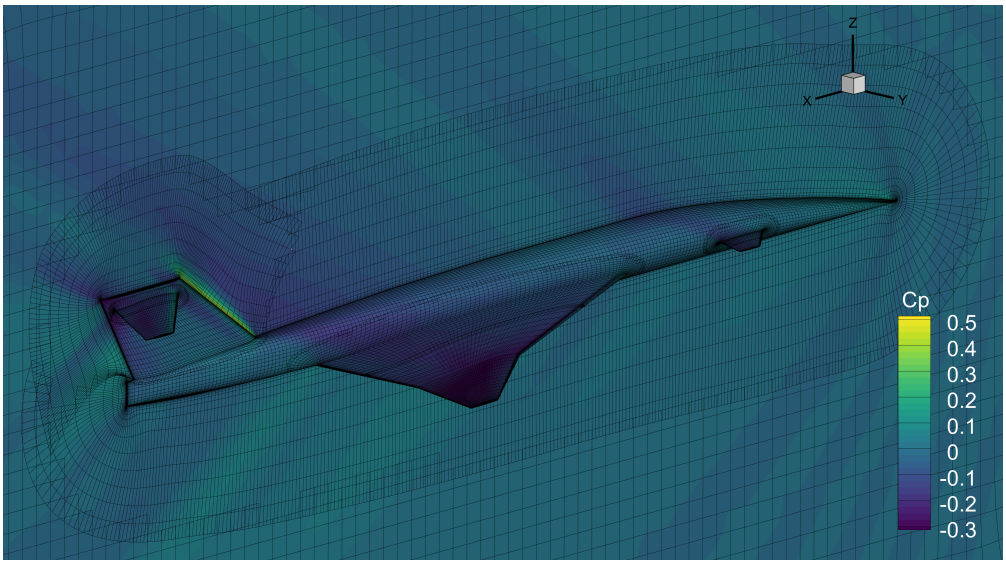
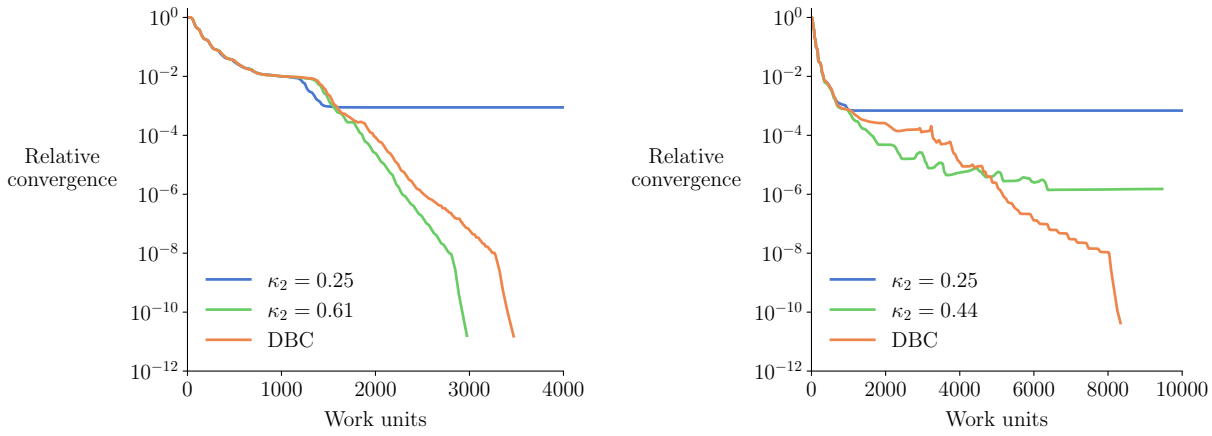


Figure 17: SST overset mesh with pressure contours at Mach 1.8



(a) Mach 1.8

(b) Mach 0.95

Figure 18: DBC converges the SST cases without sacrificing accuracy

6 Conclusions

Using compressible flow solvers for low Mach number flows typically results in poor accuracy and speed. In addition, high Mach number flows can be difficult to converge reliably without increasing the amount of dissipation in the spatial discretization. We present three contributions to address these challenges.

First, we propose a simple modification to the Jameson–Schmidt–Turkel scheme to make the artificial dissipation appropriate for low Mach number flows. We show that scaling down the acoustic contribution of the spectral radius improves accuracy and reduces sensitivity to mesh size. We recommend using a scaling

factor equal to the freestream Mach number. Second, we demonstrate the effectiveness of a characteristic time-stepping method for approximate Newton–Krylov solvers. This approach reduces the stiffness of the linear system at low Mach numbers by modifying the time-step matrix. We show that the artificial dissipation must be scaled properly to achieve speedups with characteristic time-stepping. In addition, cases with small regions of stagnation point flow do not benefit from characteristic time-stepping. Third, we present a dissipation-based continuation method for flows with shocks. The continuation approach introduces additional second-order dissipation near shocks during the initial stages of the solution and smoothly reduces the dissipation as the solution converges. This approach is faster and more robust than the baseline solver, particularly for more complex geometries. These contributions enable more accurate and efficient flow solvers across low and high Mach numbers, making CFD more dependable for applications involving a wide range of flow conditions, such as aircraft design.

Acknowledgments

This work was supported by NASA's Commercial Supersonic Technology project. S. Seraj was also supported by the Natural Sciences and Engineering Research Council of Canada (funding reference number PGSD3-545678-2020). Computational resources were provided by the NASA High-End Computing Program through the NASA Advanced Supercomputing Division at Ames Research Center.

References

[1] John Morgenstern, Michael Buonanno, Jixian Yao, Mugam Murugappan, Umesh Paliath, Lawrence Cheung, Ivan Malcevic, Kishore Ramakrishnan, Nikolai Pastouchenko, Trevor Wood, Steve Martens, Phil Viars, Trevor Tersmette, Jason Lee, Ron Simmons, David Plybon, Juan Alonso, Francisco Palacios, Trent Lukaczyk, and Gerald Carrier. Advanced concept studies for supersonic commercial transports entering service in the 2018-2020 period phase 2. Technical Report NASA/CR—2015-218719, NASA, Glenn Research Center, Cleveland, OH, 2015.

[2] Nicolas P. Bons and Joaquim R. R. A. Martins. Aerostructural design exploration of a wing in transonic flow. *Aerospace*, 7(8):118, August 2020. doi:10.3390/aerospace7080118.

[3] Vincent Gleize and Michel Costes. Low-Mach-number preconditioning applied to turbulent helicopter fuselage flowfield computation. *AIAA Journal*, 41(4):653–662, 2003. doi:10.2514/2.1995.

[4] Y. Colin, H. Deniau, and J.-F. Boussuge. A robust low speed preconditioning formulation for viscous flow computations. *Computers & Fluids*, 47(1):1–15, 2011. doi:10.1016/j.compfluid.2011.01.015.

[5] Eli Turkel. Preconditioned methods for solving the incompressible and low speed compressible equations. *Journal of Computational Physics*, 72(2):277–298, 1987. doi:10.1016/0021-9991(87)90084-2.

[6] Eli Turkel. Preconditioning techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 31(1):385–416, 1999. doi:10.1146/annurev.fluid.31.1.385.

[7] Jonathan M. Weiss and Wayne A. Smith. Preconditioning applied to variable and constant density flows. *AIAA Journal*, 33(11):2050–2057, 1995. doi:10.2514/3.12946.

[8] M. Nemec and D. W. Zingg. Aerodynamic computations using the convective-upstream split-pressure scheme with local preconditioning. *AIAA Journal*, 38(3):402–410, 2000. doi:10.2514/2.998.

[9] Bram van Leer, Wen-Tzong Lee, and Philip Roe. Characteristic time-stepping or local preconditioning of the Euler equations. In *10th Computational Fluid Dynamics Conference*, June 1979. doi:10.2514/6.1991-1552.

[10] Dohyung Lee. The design of local Navier–Stokes preconditioning for compressible flow. *Journal of Computational Physics*, 144(2):460–483, 1998. doi:10.1006/jcph.1998.5994.

[11] D. A. Knoll, P. R. McHugh, and D. E. Keyes. Newton–Krylov methods for low-Mach-number compressible combustion. *AIAA Journal*, 34(5):961–967, 1996. doi:10.2514/3.13174.

[12] Brian Weston, Robert Nourgaliev, Jean-Pierre Delplanque, and Andrew T. Barker. Preconditioning a Newton-Krylov solver for all-speed melt pool flow physics. *Journal of Computational Physics*, 397: 108847, 2019. doi:10.1016/j.jcp.2019.07.045.

[13] Ivan Mary, Pierre Sagaut, and Michel Deville. An algorithm for low Mach number unsteady flows. *Computers & Fluids*, 29(2):119–147, 2000. doi:10.1016/S0045-7930(99)00007-9.

[14] D. K. Kaushik, D. E. Keyes, and B. F. Smith. Newton-Krylov-Schwarz methods for aerodynamics problems: Compressible and incompressible flows on unstructured grids. In *11th International Conference on Domain Decomposition Methods*, 1998.

[15] F. Olawsky, F. Infed, and M. Auweter-Kurtz. Preconditioned newton method for computing supersonic and hypersonic nonequilibrium flows. *Journal of Spacecraft and Rockets*, 41(6):907–914, 2004. doi:10.2514/1.4010.

[16] Robert T. Biedron, Jan-Reneé Carlson, Joseph M. Derlaga, Peter A. Gnoffo, Dana P. Hammond, Kevin E. Jacobson, William T. Jones, Bil Kleb, Elizabeth M. Lee-Rausch, Eric J. Nielsen, Michael A. Park, Christopher L. Rumsey, James L. Thomas, Kyle B. Thompson, Aaron C. Walden, Li Wang, and William A. Wood. FUN3D manual: 13.7. Technical Report NASA/TM–20205010139, NASA Langley Research Center, Hampton, Virginia, November 2020.

[17] Jakob Öhrman. *Evaluation of a CFD method for estimating aerodynamic loads on external stores on JAS 39 Gripen*. PhD thesis, Umeå University, 2011.

[18] Charles A. Mader, Gaetan K. W. Kenway, Anil Yildirim, and Joaquim R. R. A. Martins. ADflow: An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization. *Journal of Aerospace Information Systems*, 17(9):508–527, September 2020. doi:10.2514/1.I010796.

[19] Philippe Spalart and Steven Allmaras. A One-Equation Turbulence Model for Aerodynamic Flows. *La Recherche Aerospatiale*, 1:5–21, 1994.

[20] A. Jameson, W. Schmidt, and E. Turkel. Numerical solution of the Euler equations by finite volume methods using Runge–Kutta time stepping schemes. In *14th Fluid and Plasma Dynamics Conference*, number AIAA 1981-1259, 1981. doi:10.2514/6.1981-1259.

[21] Antony Jameson. Origins and further development of the Jameson–Schmidt–Turkel scheme. *AIAA Journal*, 55(5):1487–1510, 2017. doi:10.2514/1.J055493.

[22] Luigi Martinelli. *Calculations of Viscous Flows with a Multigrid Method*. PhD thesis, Princeton University, 1987.

[23] Anil Yildirim, Gaetan K. W. Kenway, Charles A. Mader, and Joaquim R. R. A. Martins. A Jacobian-free approximate Newton–Krylov startup strategy for RANS simulations. *Journal of Computational Physics*, 397:108741, November 2019. ISSN 0021-9991. doi:10.1016/j.jcp.2019.06.018.

[24] Z.J. Wang, Krzysztof Fidkowski, Rémi Abgrall, Francesco Bassi, Doru Caraeni, Andrew Cary, Herman Deconinck, Ralf Hartmann, Koen Hillewaert, H.T. Huynh, Norbert Kroll, Georg May, Per-Olof Persson, Bram van Leer, and Miguel Visbal. High-order CFD methods: current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013. doi:10.1002/fld.3767.

[25] R. C. Swanson and Eli Turkel. On central-difference and upwind schemes. *Journal of Computational Physics*, 101(2):292–306, August 1992. doi:10.1016/0021-9991(92)90007-L.

[26] A. Jameson. Analysis and design of numerical schemes for gas dynamics 1—artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Dynamics*, 4:171–218, 1995.

[27] Bertil Gustafsson. Unsymmetric hyperbolic systems and the Euler equations at low Mach numbers. *Journal of Scientific Computing*, 2(2):123–136, 1987. doi:10.1007/BF01061482.

[28] Felix Rieper. A low-Mach number fix for Roe's approximate Riemann solver. *Journal of Computational Physics*, 230(13):5263–5287, 2011. doi:10.1016/j.jcp.2011.03.025.

[29] Mark Drela. XFOIL: An analysis and design system for low reynolds number airfoils. In Thomas J. Mueller, editor, *Low Reynolds Number Aerodynamics*, pages 1–12, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg. doi:10.1007/978-3-642-84010-4_1.

[30] Hervé Guillard and Cécile Viozat. On the behaviour of upwind schemes in the low Mach number limit. *Computers & Fluids*, 28(1):63–86, 1999. doi:10.1016/j.camwa.2018.02.028.

[31] Shu-sheng Chen, Chao Yan, and Xing-hao Xiang. Effective low-Mach number improvement for upwind schemes. *Computers & Mathematics with Applications*, 75(10):3737–3755, 2018. doi:10.1016/j.camwa.2018.02.028.

[32] D. W. Zingg. Viscous airfoil computations using Richardson extrapolation. In *10th Computational Fluid Dynamics Conference*, 1991. doi:10.2514/6.1991-1559.

[33] P. J. Roache. Perspective: A method for uniform reporting of grid refinement studies. *Journal of Fluids Engineering*, 116(3):405–413, 1994. doi:10.1115/1.2910291.

[34] Margarida Moragues Ginard, Mariano Vázquez, and Guillaume Houzeaux. Local preconditioning and variational multiscale stabilization for Euler compressible steady flow. *Computer Methods in Applied Mechanics and Engineering*, 305:468–500, 2016. doi:10.1016/j.cma.2016.02.027.

[35] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986. doi:10.1137/0907058.

[36] M. Ameen Jarrah and Holt Ashley. Impact of flow unsteadiness on maneuvers and loads of agile aircraft. In *30th Structures, Structural Dynamics and Materials Conference*, 1989. doi:10.2514/6.1989-1282.

[37] Sabet Seraj and Joaquim R. R. A. Martins. Predicting the high-angle-of-attack characteristics of a delta wing at low speed. *Journal of Aircraft*, 2022. doi:10.2514/1.C036618. (In press).

[38] Jason Hicken and David Zingg. Globalization Strategies for Inexact-Newton Solvers. In *19th AIAA Computational Fluid Dynamics*, Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, June 2009. doi:10.2514/6.2009-4139.

[39] Jason Hicken, Howard Buckley, Michal Osusky, and David Zingg. Dissipation-based continuation: a globalization for inexact-Newton solvers. In *20th AIAA Computational Fluid Dynamics Conference*, Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, June 2011. doi:10.2514/6.2011-3237.

[40] Sabet Seraj and Joaquim R. R. A. Martins. Aerodynamic shape optimization of a supersonic transport considering low-speed stability. In *AIAA SciTech Forum*, January 2022. doi:10.2514/6.2022-2177.

[41] Joaquim R. R. A. Martins. Aerodynamic design optimization: Challenges and perspectives. *Computers & Fluids*, 239:105391, 2022. doi:10.1016/j.compfluid.2022.105391.