Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012

ICCFD7-1801

An efficient Newton-Krylov-Schur parallel solution algorithm for the steady and unsteady Navier-Stokes equations

Michal Osusky, Pieter D. Boom, David C. Del Rey Fernández, and David W. Zingg
Corresponding author: michal@oddjob.utias.utoronto.ca

University of Toronto Institute for Aerospace Studies,
4925 Dufferin Street, Toronto, Ontario, Canada, M3H 5T6

Abstract: We present a parallel Newton-Krylov-Schur flow solution algorithm for the three-dimensional Navier-Stokes equations for both steady and unsteady flows. The algorithm employs second- and fourth-order summation-by-parts operators on multi-block structured grids with simultaneous approximation terms used to enforce block interface coupling and boundary conditions. The discrete equations are solved iteratively with an inexact-Newton method, while the linear system at each Newton-iteration is solved using the flexible generalized minimal residual Krylov subspace iterative method with the approximate-Schur parallel preconditioner. Time-accurate solutions are evolved in time using explicit-first-stage singly-diagonally-implicit Runge-Kutta methods. The algorithm is demonstrated through the solution of the steady transonic flow over the NASA Common Research Model wing-body configuration in a range of angles of attack where substantial flow separation occurs. Several parallel scaling studies highlight the excellent scaling characteristics of the algorithm on cases with up to 6656 processors, and grids with over 150 million nodes. Finally, the algorithm accurately captures the temporal evolution of the Taylor-Green vortex flow, highlighting the advantages of high-order spatial and temporal discretization. The algorithm presented is an efficient option for a wide range of flow problems encompassing the steady and unsteady Reynolds-averaged Navier-Stokes equations as well as large-eddy and direct numerical simulations of turbulent flows.

Keywords: Numerical Algorithms, Computational Fluid Dynamics, Newton-Krylov, Approximate-Schur Preconditioner, Steady Flows, Unsteady Flows, Parallel Computations, SBP-SAT Discretization.

1 Introduction

With recent advances in computer architectures, parallel computing, and numerical methods, large-scale CFD simulations are becoming more suitable for use in a practical setting. However, accurate simulations of flows of interest necessitate the solution of very large problems, with the results from the AIAA Drag Prediction Workshop [1] indicating that grids with over $O(10^8)$ grid nodes are required for grid-converged lift and drag values for flows over a wing-body configuration. In order to make efficient use of computational resources, algorithms that scale well with thousands of processors are required. At the same time, high-order methods present an avenue for reducing the cost of simulations. Despite being more computationally expensive per node or per time step, they can achieve a specific level of accuracy on significantly coarser grids.

In the computation of turbulent flows over complex geometries, numerical approaches span the use of structured or unstructured grids, finite-volume, finite-element, or finite-difference approximations, explicit or implicit solution strategies, and a wide range of linear solvers and preconditioners. A few examples of popular Reynolds-averaged Navier-Stokes (RANS) solvers are OVERFLOW [2], FUN3D [3], Flo3xx [4], and NSU3D [5]. This paper presents an efficient parallel three-dimensional multi-block structured solver

for turbulent flows over aerodynamic geometries, extending previous work [6, 7, 8] on an efficient parallel Newton-Krylov flow solver for the Euler equations and the Navier-Stokes equations in the laminar flow regime. The combination of techniques employed in the current solver also has the benefit of lending itself to a unified approach for performing both steady and implicit unsteady computations.

In order to accommodate complex three-dimensional shapes, the multi-block approach, which breaks the computational domain into several subdomains, is used. This approach has the added benefit of being easily utilized in a parallel solution algorithm, since each block can be assigned to an individual process. Simultaneous approximation terms (SATs) are used to impose boundary conditions, as well as inter-block solution coupling, through a penalty method approach. SATs were originally introduced to treat boundary conditions in an accurate and time-stable manner [9], and later extended to deal with block interfaces [10, 11, 12]. Svärd *et al.* [13, 14] and Nordström *et al.* [15] have shown the application of SATs for the Navier-Stokes equations to unsteady problems, as well as some steady model problems. This approach has several advantages over more traditional approaches. It eliminates the need for mesh continuity across block interfaces, reduces the communication for parallel algorithms, especially when extended to higher-order discretizations, and ensures linear time stability when coupled with summation-by-parts (SBP) operators. SBP operators allow for the construction of finite-difference approximations to derivatives, while also presenting a systematic means of deriving higher-order operators with a stable and suitably high-order boundary treatment. However, the SBP-SAT approach has received limited use in computational aerodynamics applications since SATs present a difficulty in that they can necessitate the use of small time steps with explicit solvers [16]. Hence, the combination of SATs with a parallel Newton-Krylov solver has the potential to be an efficient approach.

Parallel preconditioning is a critical component of a scalable Newton-Krylov algorithm. Hicken *et al.* [17] have shown that the approximate-Schur preconditioner scales well to at least 1000 processors when inviscid and laminar flows are considered, with similar performance reported for turbulent flows in [18]. One of the objectives of this paper is to demonstrate the applicability of a spatial discretization based on the SBP-SAT approach with a parallel Newton-Krylov-Schur algorithm to the Reynolds-averaged Navier-Stokes equations coupled with the Spalart-Allmaras one-equation turbulence model [19], resulting in an efficient parallel solver for turbulent flows. The efficient Newton-Krylov-Schur algorithm used for steady computations can also be readily extended to the solution of unsteady flows, with the solution of each time step serving as an excellent initial guess for the solution of the subsequent time step, greatly improving the convergence of the nonlinear residual. In this way, a unified approach can be readily applied to the solution of both steady and unsteady flows, with the time-accurate algorithm leveraging the solution strategy developed for steady flows.

Although more expensive per node or per time step, higher-order methods possess a higher convergence rate. In simulations where a high level of accuracy is required, higher-order methods can become more efficient than their second-order counterparts. Additionally, CFD applications with complex geometries often result in highly stiff problems. These can be solved by explicit algorithms, but are limited by the size of the time step needed for stability and will often result in prohibitively expensive computations. On the other hand, one can generate implicit methods that are stable regardless of step size and can be used to solve these stiff problems. Higher-order implicit time-marching methods benefit from both of these advantages and present a promising approach for obtaining time-accurate flow solutions in an efficient manner.

The paper is divided into the following sections. Section 2 presents a brief overview of the governing equations, while Sections 3 and 4 present the spatial and temporal discretizations used, respectively. Section 5 provides details of the Newton-Krylov-Schur method and its application to solving the large nonlinear system resulting from the discretization of the Navier-Stokes equations for both steady and unsteady flows. Section 6 presents results obtained with the current algorithm for steady and unsteady flow solutions, including steady transonic flow solutions around the NASA Common Research Model (CRM) geometry, parallel scaling performance characteristics of the current algorithm, and the Taylor-Green vortex unsteady flow simulation. Conclusions are given in Section 7.

2 Governing Equations

The current algorithm numerically solves the three-dimensional Navier-Stokes equations, with turbulent effects modeled by use of the standard form of the Spalart-Allmaras one-equation turbulence model. Applying the curvilinear coordinate transformation $(x, y, z) \rightarrow (\xi, \eta, \zeta)$ allows the application of finite-difference meth-

ods in a computational space where the grid is uniform. All steady flows are assumed to be fully turbulent, and no explicit trip terms are used in the turbulence model.

3 Spatial Discretization

The spatial discretization of the Navier-Stokes equations and the turbulence model is obtained by the use of Summation-By-Parts (SBP) operators, while inter-block coupling and boundary conditions are enforced by the use of Simultaneous Approximation Terms (SATs). This section presents the SBP operators for first and second derivatives and their application to the governing equations, as well as the various SATs required. The SBP-SAT discretization, if implemented in a dual-consistent manner, can also lead to superconvergent functional estimates [20, 21].

The numerical dissipation employs either the scalar dissipation model developed by Jameson *et al.* [22] and later refined by Pulliam [23], or the matrix dissipation model of Swanson and Turkel [24]. With the second-order spatial discretization, the numerical dissipation consists of second- and fourth-difference dissipation operators, whose magnitudes are controlled by κ_2 and κ_4 coefficients, respectively, typically set to 2.0 and 0.04 for transonic flows. For subsonic flows, κ_2 is set to 0. With the higher-order spatial operator, the order of the dissipation model is increased in order to preserve the overall order of the spatial discretization.

Grid metrics, which result from the coordinate transformation, are computed in a manner that matches the spatial finite-difference operator, with second-order metrics used with the second-order operators and fourth-order metrics used with the fourth-order operators.

The ultimate goal of the current research program is the construction of an algorithm for efficient aerodynamic optimization of aircraft. The most computationally expensive operation of the optimization procedure is the flow solution. One means of making the flow solver more efficient is to construct a higher-order (HO) discretization such that, relative to a second-order discretization, equivalent accuracy can be achieved with a coarser mesh.

In this section we present a general framework for discretizing the compressible Navier-Stokes equations with SBP operators that have orders of accuracy from 2 to 6. The turbulence model has yet to be discretized using HO operators and so the remainder of the paper concentrates on second-order SBP operators, with the exception of some results in Section 6. The premise is to give a general outline of the proposed discretization, while a more detailed presentation of the higher-order operators is provided in [25].

3.1 Summation-by-parts operators

SBP operators allow for the construction of finite-difference (FD) approximations to derivatives. Additionally, they present a systematic means of deriving HO FD operators with a stable and suitably high-order boundary treatment. As a result of only requiring C^0 continuity between block interfaces, the SBP-SAT discretization gives rise to multi-block schemes that possess less communication overhead than typical schemes using halo nodes. Furthermore, the relaxed geometric requirements at interfaces make grid generation and domain decomposition substantially easier. When dealing with the current governing equations, approximations to both the first and second derivatives are required.

The SBP operators for the first derivative were originally derived by Kreiss and Scherer [26], subsequently extended by Strand [27], and applied by various authors (see [6, 7, 15, 28, 29]). SBP operators are centered difference schemes that do not include boundary conditions; in our case these are enforced using SATs. They are constructed so that the discrete energy-method can be used to make time stability statements about a discretization and have been shown to be time-stable for the linearized Navier-Stokes equations [28]. However, for curvilinear coordinates, time-stability can only be guaranteed for SBP operators constructed with a diagonal norm, and so the discussion is limited to diagonal norm SBP operators.

3.1.1 SBP operator for first derivative

In this section we briefly present the relevant SBP operators for the first derivative. For the first derivative, the SBP property is mimetic of $\int_a^b \mathcal{Q}\partial_x \mathcal{Q} dx$, also known as integration by parts, leading to the following definition:

SBP Diagonal Norm First Derivative The matrix $D_1 \in R^{(N+1)\times(N+1)}$ is an SBP operator for the first derivative, if it approximates the first derivative, is of form $D_1 = H^{-1}\Theta$, where $H \in R^{(N+1)\times(N+1)}$ is a positive-definite diagonal matrix, called the norm, and Θ has property $\Theta + \Theta^T = \text{diag}(-1, 0, ..., 0, 1)$.

A globally second-order accurate operator for a first derivative is given by

$$D_1 = H^{-1}\Theta, \tag{1}$$

where

$$H = h\begin{bmatrix} \frac{1}{2} & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \frac{1}{2} \end{bmatrix}, \quad \Theta = \frac{1}{2}\begin{bmatrix} -1 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 1 \end{bmatrix},$$

and h takes on the value of the spatial difference in the pertinent coordinate direction, either $\Delta\xi$, $\Delta\eta$, or $\Delta\zeta$. In the context of the uniform computational grid, h has a value of 1 for all three coordinate directions.

Application to Navier-Stokes equations

The D_1 operator is used to obtain a finite difference approximation of the inviscid fluxes for the entire computational domain. For example, the inviscid flux in the ξ-direction can be taken as

$$\partial_\xi \hat{\mathbf{E}} \approx D_{1\xi}\hat{\mathbf{E}}. \tag{2}$$

For the second-order operator, this results in the following stencils in different parts of the domain:

$$\text{low side:} \quad \hat{\mathbf{E}}_2 - \hat{\mathbf{E}}_1, \qquad \text{interior:} \quad \tfrac{1}{2}\left(\hat{\mathbf{E}}_{i+1} - \hat{\mathbf{E}}_{i-1}\right), \qquad \text{high side:} \quad \hat{\mathbf{E}}_N - \hat{\mathbf{E}}_{N-1},$$

where N is the number of nodes in the ξ-direction. This is identical to the typical centered finite difference approximation, with first-order treatment at boundaries.

The D_1 operator is also used in the discretization of the cross-derivative viscous terms, which have the form

$$\partial_\xi(\beta\partial_\eta\alpha), \tag{3}$$

where β is a spatially variable coefficient and α can be a flow quantity such as the x-component of velocity, u. Using (1), the cross-derivative can be approximated as

$$D_{1\xi}\boldsymbol{\beta}D_{1\eta}\boldsymbol{\alpha}, \tag{4}$$

resulting in the following interior discretization (at node (j, k, m)):

$$\frac{1}{2}\beta_{j+1,k,m}\left(\frac{\alpha_{j+1,k+1,m} - \alpha_{j+1,k-1,m}}{2}\right) - \frac{1}{2}\beta_{j-1,k,m}\left(\frac{\alpha_{j-1,k+1,m} - \alpha_{j-1,k-1,m}}{2}\right). \tag{5}$$

Application to Spalart-Allmaras turbulence model

The advective terms that appear in the turbulence model consist of first derivatives of the turbulence variable, $\tilde{\nu}$, multiplied by velocities. An example of this is the term associated with the spatial derivative in the ξ-direction, given by

$$U\partial_\xi\tilde{\nu}, \tag{6}$$

where U is the contravariant velocity given by $\xi_x u + \xi_y v + \xi_z w$.

The authors of the model suggest the use of an upwinding strategy when discretizing this term, which is the approach taken here. However, in the context of SBP operators, we have made use of the connection between upwinding and artificial dissipation, namely that an upwinded operator can be equated to a centered

difference operator added to a dissipation operator. The derivative can be taken as

$$U\partial_\xi \tilde{\nu} \approx \mathbf{U} D_1 \tilde{\boldsymbol{\nu}} + \frac{1}{2}|\mathbf{U}| H^{-1} D_d^T D_d \tilde{\boldsymbol{\nu}} \tag{7}$$

where $\tilde{\boldsymbol{\nu}}$ represents a vector containing the turbulence quantity in the domain, and

$$\mathbf{U} = \mathrm{diag}\left(U_1, U_2, ..., U_N\right), \quad |\mathbf{U}| = \mathrm{diag}\left(|U_1|, |U_2|, ..., |U_N|\right), \quad D_d = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & 0 \end{bmatrix}.$$

The above SBP discretization provides a clear approach to dealing with block boundaries. For completeness, the following shows the resulting discretization in different parts of the domain:

$$\begin{array}{ll} \text{low side:} & \left(U_1 - |U_1|\right)\left(\tilde{\nu}_2 - \tilde{\nu}_1\right), \\ \text{interior:} & \frac{1}{2}U_i\left(\tilde{\nu}_{i+1} - \tilde{\nu}_{i-1}\right) - \frac{1}{2}|U_i|\left(\tilde{\nu}_{i+1} - 2\tilde{\nu}_i + \tilde{\nu}_{i-1}\right), \\ \text{high side:} & \left(U_N + |U_N|\right)\left(\tilde{\nu}_N - \tilde{\nu}_{N-1}\right). \end{array}$$

The first derivative operator of (1) is also used for the derivatives present in the vorticity term, S.

3.1.2 SBP operator for second derivative

The compressible Navier-Stokes equations require a discrete approximation to derivatives of the form $\partial_x(\beta\partial_x\alpha)$, where β are spatially varying coefficients. The simplest means of discretizing these terms is to apply the first derivative twice. Alternatively, one can construct a discrete approximation that has the same stencil width as the first derivative, called a compact-stencil operator. The application of the first derivative twice has several disadvantages compared to compact-stencil operators: larger bandwidth, loss of one order of accuracy, higher global error, and less dissipation of high wavenumber modes. Given these shortcomings, our approach is to derive compact SBP operators for the second derivative with variable coefficients. These have been recently derived for up to 6^{th} interior order by Mattsson [30]. In a companion paper [25] we present a novel framework for deriving SBP operators for the second derivative with variable coefficients that allow for derivation of up to 8^{th} order interior accuracy, allow for solution of the resultant nonlinear system of equations, substantially reduce the number of free parameters used in the derivation, and can be systematically constructed and optimized.

For the second derivative with variable coefficients, the SBP property is mimetic of $\int_a^\psi \mathcal{Q}\partial_x(\beta\partial_x(\mathcal{Q}))dx$, where β are the variable coefficients, leading to the following definition:

SBP Second Derivative The matrix $D_2(\beta) \in R^{(N+1)\times(N+1)}$ is an SBP operator for the second derivative, with variable coefficients $\beta > 0$, if it approximates the second derivative and is of the form, $D_2(\beta) = H^{-1}\left\{-M + EBD_b\right\}$, where H is a diagonal positive-definite matrix, called the norm, $E = \mathrm{Diag}(-1, 0, ..., 0, 1)$, $B = \mathrm{diag}(\beta_0, \beta_1, ..., \beta_{N-1}, \beta_N)$, D_b is an approximation to the first derivative at the boundaries, $M = D_1^T H B D_1 + R$, and M and R are positive-semi-definite (PSD) and symmetric.

In order to show that the proposed SBP-SAT discretization is time-stable for the linearized Navier-Stokes equations, H in the above definition must be the same norm as used with the first derivative, thus the formulation is said to be compatible with the first derivative; see Mattsson [31] for more information.

The proposed SBP operators for the second derivative with variable coefficients are $2p$ accurate on the interior and have p accurate boundary closures; nonetheless, Mattsson and Nordström [31] have proven them to be $p+2$ globally accurate.

For the second-order operator one obtains:

$$D_2^{(2,1)}(\beta) = H^{-1}\left\{-\left(D_1^{(2,1)}\right)^T H B D_1^{(2,1)} - \frac{1}{4h}\left(\tilde{D}_2^{(2,1)}\right)^T C_2 B \tilde{D}_2^{(2,1)} + EBD_1^{(2,2)}\right\}, \tag{8}$$

where

$$\tilde{D}_2^{(2,1)} = \begin{bmatrix} 1 & -2 & 1 & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & 1 \\ & & & & 1 & -2 & 1 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 0 \end{bmatrix},$$

$$EBD_1^{(2,2)} = \frac{1}{h} \begin{bmatrix} \frac{3\beta_1}{2} & -2\beta_1 & \frac{\beta_1}{2} & & \\ 0 & 0 & 0 & & \\ & \ddots & \ddots & \ddots & \\ & & 0 & 0 & 0 \\ & & \frac{\beta_{N+1}}{2} & -2\beta_{N+1} & \frac{3\beta_{N+1}}{2} \end{bmatrix},$$

and $D_1^{(2,1)}$ is defined in (1). The notation $D^{(i,b)}$ provides the order of the operator, both internally (i) and at boundaries (b). The tilde symbol signifies an undivided difference operator.

Application to governing equations

Both the viscous terms of the Navier-Stokes equations and the diffusive terms of the turbulence model contain double-derivatives which can be approximated as

$$\partial_\xi \left(\beta \partial_\xi \alpha\right) \approx D_2(\beta)\boldsymbol{\alpha}. \tag{9}$$

For an internal node, this will result in the narrow stencil used by Pulliam [23] (with k and m subscripts suppressed):

$$\frac{1}{2}(\beta_{j+1} + \beta_j)(\alpha_{j+1} - \alpha_j) - \frac{1}{2}(\beta_j + \beta_{j-1})(\alpha_j - \alpha_{j-1}). \tag{10}$$

At the block boundaries, where one-sided differences are employed, the discretization takes on the form

$$\begin{array}{ll} \text{low side:} & -\beta_j \left[2(\alpha_{j+1} - \alpha_j) - (\alpha_{j+2} - \alpha_{j+1})\right] + \beta_{j+1}(\alpha_{j+1} - \alpha_j), \\ \text{high side:} & \beta_j \left[2(\alpha_j - \alpha_{j-1}) - (\alpha_{j-1} - \alpha_{j-2})\right] - \beta_{j-1}(\alpha_j - \alpha_{j-1}). \end{array} \tag{11}$$

3.2 Simultaneous Approximation Terms

The use of SBP operators ties in closely to the application of SAT penalties at block boundaries, be they interfaces or domain boundaries. SATs are used to preserve inter-block continuity, or enforce specific boundary conditions. The purpose of this section is not to derive the forms of the various SATs used, but rather to present the implementation used in the present algorithm. See references [6, 13, 14, 15] for an analysis and derivation of the SAT terms applied to the Navier-Stokes equations. All SAT terms that follow are shown in the form in which they would be added to the right-hand-side of the governing equations.

3.2.1 SATs for Navier-Stokes equations

The form of the inviscid, or Euler, portion of the SATs on the low side of a block is

$$\text{SAT}_{\text{inv}} = -H_b^{-1} J^{-1} A_\xi^+ \left(\mathbf{Q} - \mathbf{Q}_{\text{external}}\right), \tag{12}$$

where H_b is the boundary node element of the diagonal norm matrix H,

$$A_\xi^+ = \frac{A_\xi + |A_\xi|}{2}, \quad A_\xi = \frac{\partial \hat{\mathbf{E}}}{\partial \hat{\mathbf{Q}}},$$

and $\mathbf{Q}$ are the flow variables on the boundary node in the current block. $|A_\xi|$ denotes $X^{-1}\left|\Lambda\right| X$, where X is the right eigenmatrix of A_ξ, and Λ contains the eigenvalues along its diagonal. At a high-side boundary A_ξ^-

is used to capture the incoming characteristics and the sign of the penalty is reversed. When dealing with boundaries normal to the other two coordinate directions, A_ξ is replaced by either A_η or A_ζ. The variable $\mathbf{Q}_{\text{external}}$ takes on the target values to which the local values of $\mathbf{Q}$ are being forced. When dealing with a block interface, these are the flow variable values on a coincident node in a neighbouring block, or, when dealing with a far-field boundary, they can be the free-stream flow variable values. A number of different boundary conditions, such as a slip-wall or symmetry plane, can be enforced using this approach for the Euler equations. In each case, the SAT works on a principle very similar to characteristic boundary conditions.

The basis of the viscous SATs is presented by Nordström *et al.* [15] and is summarized below, with special attention being paid to each type of block boundary.

The first type of viscous SAT deals with differences in viscous fluxes. In the ξ-direction, this term has the form

$$\text{SAT}_{\text{visc\_flux}} = \frac{H_b^{-1}\sigma^V}{Re}\left(\hat{\mathbf{E}}_\mathbf{v} - \mathbf{g}_\mathbf{v}\right), \tag{13}$$

where $\hat{\mathbf{E}}_\mathbf{v}$ is the local viscous flux, and $\mathbf{g}_\mathbf{v}$ is the target value of the viscous flux. Additionally, $\sigma^V = 1$ at a low-side boundary, and -1 at a high-side boundary. At a far-field boundary, which is supposed to force the solution towards free-stream conditions, $\mathbf{g}_\mathbf{v} = 0$. Interface SATs also make use of (13), where $\mathbf{g}_\mathbf{v}$ is equal to $\hat{\mathbf{E}}_{\mathbf{v2}}$, the viscous flux on the coincident node in the adjoining block.

A no-slip adiabatic wall boundary condition is enforced with the use of a different type of term, which is again added on top of the Euler SAT. The form of the viscous portion of the no-slip wall SAT for a boundary at the low or high side of a block in the ξ-direction, is

$$\text{SAT}_{\text{visc\_wall,1}} = \frac{H_b^{-1}\sigma^W}{Re} I\left(\mathbf{Q} - \mathbf{Q}_\mathbf{w}\right), \tag{14}$$

where

$$\sigma^W \leq -\frac{\xi_x^2 + \xi_y^2 + \xi_z^2}{J}\frac{\mu}{2\rho}\max\left(\frac{\gamma}{\mathbf{Pr}}, \frac{5}{3}\right), \mathbf{Q}_\mathbf{w} = \left[\rho_1, 0, 0, 0, \frac{\rho_1 T_2}{\gamma(\gamma-1)}\right]^T.$$

σ^W is calculated based on local values, while $\mathbf{Q}_\mathbf{w}$ is constructed in order to enforce an adiabatic no-slip wall boundary condition. The three momentum components are forced toward zero, thus satisfying the no-slip condition, while no condition is enforced on density, since the local value of density, ρ_1, will cancel out in the penalty term. The energy equation has a penalty term applied to it based on the value of the temperature of one node above the boundary, T_2. This approach will result in a zero temperature gradient at the solid boundary, along with a no-slip velocity condition. The use of T_2 to enforce the adiabatic condition relies on the assumption that the grid is perpendicular to the surface of the wing, which may not always be true. The form of the SAT presented in (14) can also be readily used to enforce an isothermal boundary condition. This can be achieved by replacing the T_2 term in $\mathbf{Q}_\mathbf{w}$ with the desired wall temperature, T_w, as described in [14]. It should also be noted that unlike a more traditional method of applying the adiabatic no-slip surface condition, the penalty approach presented here does not apply any constraint on the momentum equation. This is due to the fact that the Navier-Stokes equations are solved on all nodes, including the boundaries, whereas more traditional approaches provide the solution on boundaries explicitly.

An alternate approach to dealing with the adiabatic condition involves a combination of previously discussed penalty terms. The surface penalty in (14) can be modified to only enforce the no-slip condition, while the viscous flux penalty in (13) can be modified to enforce the zero-temperature gradient necessary for the adiabatic condition. The overall form of this SAT is

$$\text{SAT}_{\text{visc\_wall,2}} = H_b^{-1}\left[\frac{\sigma^W}{Re} I\left(\mathbf{Q} - \mathbf{Q}_{\mathbf{w2}}\right) + \frac{\sigma^V}{Re}\left(\hat{\mathbf{E}}_\mathbf{v} - \hat{\mathbf{E}}_{\mathbf{v},\mathbf{w}}\right)\right], \tag{15}$$

where

$$\mathbf{Q}_{\mathbf{w2}} = \left[\rho_1, 0, 0, 0, e_1\right]^T,$$

and $\hat{\mathbf{E}}_{\mathbf{v},\mathbf{w}}$ is identical to $\hat{\mathbf{E}}_\mathbf{v}$, except that the temperature derivative terms normal to the wall are set to zero. The local values of density and energy are given by ρ_1 and e_1, respectively, and the coefficients σ^W and σ^V retain their previously defined values. In this way, the first part of the SAT enforces only the no-slip

condition, while the second part enforces the adiabatic condition. Since this approach uses the gradients as they appear in the viscous stresses, it makes no assumptions about the grid (whether it is perpendicular to the surface), and enforces a more general condition of $\frac{\partial T}{\partial n} = 0$.

Block interfaces are treated in a similar way, but with unique viscous SATs for penalizing differences in conservative variable values. The form used is:

$$\mathrm{SAT}_{\mathrm{visc\_vars}} = -\frac{H_b^{-1}\sigma^{V_2}}{JRe} B_{\mathrm{int},\xi} \left(\mathbf{Q} - \mathbf{Q_2}\right), \tag{16}$$

where

$$\sigma^{V_2} \leq 0.5$$

for stability, and $\mathbf{Q_2}$ is the vector of conservative flow variables on the coincident node in the adjoining block. The B_{int} matrix is related to the viscous Jacobian, and is derived based on Nordström *et al.* [15]. Refer to the Appendix for the complete form.

In order to reduce the size of the computational domain, symmetry boundaries can be imposed. SATs are again used to impose this boundary condition by using (12) to impose a purely tangential flow ($\mathbf{Q}_{\mathrm{external}}$ is constructed in such a way as to force the normal velocity component to zero). In addition, (14) is used to enforce a zero normal gradient in all conservative variables.

The following is used to enforce the inviscid SAT on an outflow boundary in a viscous flow:

$$\mathrm{SAT}_{\mathrm{inv\_outflow}} = \frac{H_b^{-1}\sigma^{I}}{J} A_\xi^- \left(\mathbf{Q}_{j_{max}} - \mathbf{Q}_{j_{max}-1}\right), \tag{17}$$

in which the boundary is assumed to be on the high side of a block in the ξ-direction. The modification is appropriate in dealing with the viscous wake region. The advantage of this approach is that it requires minimal modification to the existing Euler SAT term, which uses the free-stream flow conditions instead of $\mathbf{Q}_{j_{max}-1}$. An alternate approach to dealing with the outflow condition is presented by Svärd *et al.* [13].

3.2.2 SATs for turbulence model

The SAT for the advection portion of the turbulence model needs to account for the flow direction in much the same way as the Euler equation SATs. This can be achieved using the following form of the SAT:

$$\mathrm{SAT}_{\mathrm{adv}} = H_b^{-1}\sigma_{\mathrm{a}} \left(\tilde{\nu}_{\mathrm{local}} - \tilde{\nu}_{\mathrm{target}}\right), \tag{18}$$

where $\tilde{\nu}_{\mathrm{local}}$ is the local value of the turbulence variable and $\tilde{\nu}_{\mathrm{target}}$ is the target value of the turbulence variable, which can either be specified by a boundary condition or, in the case of a block interface, the corresponding value on an adjoining block. The SAT parameter σ_{a} is constructed so that it accounts for the direction of information propagation in the flow:

$$\sigma_{\mathrm{a}} = -\frac{1}{2}\left[\max\left(|U|, \phi\right) + \delta_{\mathrm{a}} U\right], \tag{19}$$

where δ_{a} is +1 on the low side of a block, and -1 on the high side of a block. On an interface all flow related information in σ_{a}, such as the contravariant velocity U, is based on an average velocity between the coincident interface nodes, while at a domain boundary, it is constructed based on local information only. Finally, ϕ is a limiting factor introduced to prevent the SAT from completely disappearing in regions where the value of U goes to zero, such as near a solid surface. Following the work done on the Euler equation SATs, the value of ϕ was chosen to be

$$\phi = V_l \left(|U| + a\sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2}\right), \tag{20}$$

where $V_l = 0.025$, and a is the speed of sound. The quantity appearing in the brackets above is the spectral radius of the inviscid flux Jacobian.

As with the SATs used for the viscous portion of the Navier-Stokes equations, the SATs for the diffusive portion of the turbulence model consist of two parts, one dealing with the difference in the turbulent quantity,

the other dealing with the difference in the turbulent quantity gradient.

The diffusive SAT dealing with the difference in gradients of the turbulence variable has the general form

$$\mathrm{SAT}_{\mathrm{diff\_flux}} = H_b^{-1}\sigma_{\mathrm{df}}\left(g_{\mathrm{local}} - g_{\mathrm{target}}\right), \tag{21}$$

where σ_{df} is +1 on the low side of a block and -1 on the high side of a block. Additionally, the local and target gradients, denoted by g, have the form

$$g = \frac{1}{\sigma_{\mathrm{t}}\mathrm{Re}}\left(\nu + \tilde{\nu}\right)\left(\xi_x^2 + \xi_y^2 + \xi_z^2\right)\delta_\xi\tilde{\nu}, \tag{22}$$

where $\delta_\xi\tilde{\nu}$ is a one-sided first derivative consistent with the definition of the second derivative SBP operator at block boundaries, specified in the $D_1^{(2,2)}$ matrix of (8). The parameter σ_{t} is defined as part of the turbulence model with a value of 2/3. This SAT is applied at the farfield boundary, with the target gradient set to 0, or at block interfaces, with the target gradient calculated based on values at the interface of the adjoining block.

The diffusive SAT that deals with the difference in flow variables has a form analogous to the viscous SAT presented by Nordström *et al.* [15] for the Navier-Stokes equations,

$$\mathrm{SAT}_{\mathrm{diff\_vars}} = -H_b^{-1}\frac{1}{4\sigma_{\mathrm{t}}\mathrm{Re}}\sigma_{\mathrm{dv}}\left(\tilde{\nu}_{\mathrm{local}} - \tilde{\nu}_{\mathrm{target}}\right), \tag{23}$$

where

$$\sigma_{\mathrm{dv}} = \left(\nu + \tilde{\nu}\right)\left(\xi_x^2 + \xi_y^2 + \xi_z^2\right). \tag{24}$$

As with the advective SAT, the value of σ_{dv} is based on a state average when dealing with an interface, or simply the local state when at a domain boundary. Grid metrics are always taken from the local block information. This SAT is applied at block interfaces, wall boundaries (where the target value is 0), and symmetry planes (where the target value is taken from one node inside the boundary).

While the production and destruction terms act as source terms, therefore not necessitating the application of the SBP-SAT approach due to the absence of spatial derivatives, we have found it necessary to add a source term for nodes located directly on the surface of the aerodynamic body. The production and destruction terms have no physical meaning for these nodes, as they are undefined due to a division by a zero off-wall distance. However, a lack of any source term for the surface nodes leads to a significant difference in the residual between the surface nodes and the nodes directly above the surface. This difference often results in large, destabilizing updates to the turbulence variable, often causing the code to diverge.

A destruction source term is added to all nodes with a zero off-wall distance in order to stabilize the solution in the early stages of convergence. It is calculated using a value of $d = d_{\mathrm{min}}/2$, where d_{min} is the smallest non-zero off-wall distance in the entire computational domain. The use of this extra source penalty for the surface nodes does not have a significant impact on the converged solution, as it forces $\tilde{\nu}$ towards 0.

The farfield condition used with the turbulence model sets the target farfield value of $\tilde{\nu}$ to 3.0, as suggested by Spalart and Rumsey [32]. The target surface value of $\tilde{\nu}$ is set to 0.0. Furthermore, the turbulence quantity is initialized to the farfield value at the beginning of the flow solution process.

4 Temporal Discretization

General s-stage Runge-Kutta methods are described by:

$$\begin{aligned} y^{(n)} &= y^{(n-1)} + h\textstyle\sum_{j=1}^{s} b_j F(Y_j, t^{(n-1)} + c_i h), \\ Y_i &= y^{(n-1)} + h\textstyle\sum_{j=1}^{s} A_{ij} F(Y_j, t^{(n-1)} + c_i h) \quad \text{for } i = 1, \ldots, s, \end{aligned} \tag{25}$$

where Y_i are the individual stage values, $y^{(n)}$ the solution at time step n, $h = t^{(n)} - t^{(n-1)}$ is the step size, and A_{ij}, b_j and c_i are the coefficients of the given method, often presented in a Butcher tableau:

$$\begin{array}{c|c} c_i & A_{ij} \\ \hline & b_j \end{array}.$$

4.1 Explicit-first-stage and stiff-accuracy

Often the order of the internal stages in a Runge-Kutta method is lower than the global order predicted by classical order theory. Asymptotically, global order convergence is always guaranteed and is also practically realized for relatively non-stiff problems; however, when implicit Runge-Kutta methods are applied to very stiff problems with finite step sizes, the local error of these internal stages can dominate. This is known as order reduction.

In CFD applications, order reduction is not a threat for inviscid or laminar problems; however, order reduction can manifest in URANS simulations [33, 34]. Forcing the first stage of explicit-first-stage singly-diagonally-implicit Runge-Kutta (ESDIRK) methods to be explicit, allows the internal stages to have order two. The local error can be further reduced by enforcing stiff-accuracy, namely $c_s = 1$ and therefore $b_j = A_{sj}$. These conditions imply that the minimum convergence of an ESDIRK method for a stiff ODE will be at least third order. The conditions for stiff-accuracy also mean that the explicit stage needs only to be computed once.

4.2 Singly-diagonally-implicit methods and stability

Methods in this class of time-integrators are unconditionally stable (A-stable) and provide complete damping of modes at infinity (L-stable). This is particularly advantageous when the governing equations are stiff, as is often the case in CFD simulations, especially when geometries are complex. The size of the time steps is, therefore, only limited by accuracy and not stability. Explicit methods, which are conditionally stable, can be used, but the required number of time steps to maintain stability increases rapidly with stiffness. This increase in the number of times steps outweighs the cost of solving an implicit non-linear system.

Fully implicit Runge-Kutta methods can be generated which have order $2s$, where s again is the number of stages. This is very attractive for lowering the local truncation error and for increasing convergence with step size. However, fully-implicit RK methods require an implicit solution to a system of size sn, where n is the number of unknowns. For large systems of equations, which are common in CFD, this can be very expensive in terms of both CPU time and memory. In contrast, diagonally-implicit methods only require the solution to s systems of size n.

Letting the diagonal entries of A_{ij} be constant, except A_{11}, which is zero, means that the temporal component of the Jacobian (36) for the implicit stages is constant. This can be exploited during the solution process to reduce the computational costs. More information can be found in Section 5.2.2.

4.3 Runge-Kutta methods and order of accuracy

It is well known that A-stable implicit Linear Multistep Methods (LMMs) are limited to second-order [35]. However, this restriction does not apply to Runge-Kutta methods; arbitrarily high-order methods can be generated. High-order methods are desired since they have the potential be be more efficient, especially for simulations which require a high level of accuracy.

4.4 ESDIRK methods and local truncation error

Incorporating these ideas, the Butcher tableau of an ESDIRK method is of the form:

$$\begin{array}{c|ccccc} 0 & 0 & 0 & 0 & \dots & 0 \\ 2\lambda & \lambda & \lambda & 0 & \dots & 0 \\ c_3 & a_{31} & a_{32} & \lambda & \dots & 0 \\ \vdots & & \vdots & & \ddots & \vdots \\ 1 & a_{s1} & a_{s2} & a_{s3} & \dots & \lambda \\ \hline & a_{s1} & a_{s2} & a_{s3} & \dots & \lambda \end{array}.$$

Table 1: List of time-marching methods and associated characteristics, $z = \lambda h$

Method	Global Order	External Steps	Total Stages	Implicit Stages	Stability	$\lvert LTE\rvert$
BDF2	2	2	1	1	L-Stable	$\approx 0.33z^3$
BDF2OPT(0.5) [36]	2	2	1	1	L-Stable	$\approx 0.16z^3$
ESDIRK2/TRBDF2	2	1	3	2	L-Stable	$\approx 0.04z^3$
BDF3	3	3	1	1	L(86.03°)-Stable	$0.25z^4$
ESDIRK3	3	1	4	3	L-Stable	$\approx 0.0259z^4$
RK4	4	1	4	0	Conditional	$\approx 0.0083z^5$
BDF4	4	4	1	1	L(73.35°)-Stable	$0.2z^5$
MEBDF4[37]	4	3	3	3	L-Stable	$\approx 0.0892z^5$
SDIRK4	4	1	3	3	L-Stable	$\approx 0.1644z^5$
ESDIRK4	4	1	6	5	L-Stable	$\approx 0.0008z^5$

From these coefficients, the local truncation error can be evaluated. First, the stability polynomial is defined as:

$$\phi(z) = 1 + zb^T(\mathbb{I} - zA)^{-1}\mathbf{1}, \tag{26}$$

where $\mathbb{I}$ is the identity matrix, $\mathbf{1}$ is a column vector of ones, and $z = \lambda h$, where λ represents the eigenvalue of the test function $y' = \lambda y$. The stability polynomial approximates e^z, the exact solution of the test equation, up to the order of the Runge-Kutta method, p. Written as a Taylor series, the difference between the stability polynomial and the exact solution is:

$$\phi(z) - e^z = \left(1 + z + \frac{1}{2}z^2 + \ldots + \frac{1}{p!}z^p + \mathcal{O}(p+1)\right) - \left(1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \ldots\right) \tag{27}$$

$$= C_1 z^{p+1} + C_2 z^{p+2} + C_3 z^{p+3} + \ldots \tag{28}$$

The local truncation error, is then defined as $C_1 z^{p+1}$. A final advantage of ESDIRK methods is the relatively small local truncation error coefficients ($\lvert LTE\rvert$), as can be seen in Table 1, which compares some common time integration methods. Even taking into account the increased number of implicit stages, it is clear that ESDIRK methods of a given order can be very efficient.

5 Solution Methodology

5.1 Steady-state solutions

Applying the SBP-SAT discretization described in the previous section to the steady Navier-Stokes equations and the Spalart-Allmaras one-equation turbulence model results in a large system of nonlinear equations:

$$\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}) = 0, \tag{29}$$

where $\boldsymbol{\mathcal{Q}}$ represents the complete solution vector. When time-marched with the implicit Euler time-marching method and a local time linearization, this results in a large system of linear equations of the form [38]:

$$\left(\frac{\mathcal{I}}{\Delta t} + \mathcal{A}^{(n)}\right)\Delta\boldsymbol{\mathcal{Q}}^{(n)} = -\boldsymbol{\mathcal{R}}^{(n)}, \tag{30}$$

where n is the outer (nonlinear) iteration index, Δt is the time step, $\mathcal{I}$ is the identity matrix, $\boldsymbol{\mathcal{R}}^{(n)} = \boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}^{(n)})$, $\Delta\boldsymbol{\mathcal{Q}}^{(n)} = \boldsymbol{\mathcal{Q}}^{(n+1)} - \boldsymbol{\mathcal{Q}}^{(n)}$, and

$$\mathcal{A}^{(n)} = \frac{\partial\boldsymbol{\mathcal{R}}^{(n)}}{\partial\boldsymbol{\mathcal{Q}}^{(n)}}$$

is the Jacobian.

In the infinite time step limit, the above describes Newton's method and will converge quadratically if a suitable initial iterate, $\boldsymbol{\mathcal{Q}}^{(0)}$, is known. This initial iterate must be sufficiently close to the solution of (29).

Since it is unlikely that any initial guess made for a steady-state solution will satisfy this requirement, the present algorithm makes use of a start-up phase whose purpose it is to find a suitable initial iterate. The following sections describe each of the phases as they apply to the solution of the Navier-Stokes equations. Both phases result in a large set of linear equations at each outer iteration, which are solved to a specified tolerance using the preconditioned Krylov iterative solver GMRES.

5.1.1 Approximate-Newton phase

The approximate-Newton method makes use of implicit Euler time-stepping to find a suitable initial iterate for Newton's method. Since we are not interested in a time-accurate solution, some useful modifications can be made. These include a first-order Jacobian matrix and a spatially varying time step.

A first-order Jacobian matrix, $\mathcal{A}_1$, has been shown to be an effective replacement for the true Jacobian, $\mathcal{A}$, during the start-up phase [39, 40, 41]. A number of approximations are made when creating the first-order approximation to the Jacobian. When dealing with the inviscid terms, the fourth-difference dissipation coefficient, κ_4, is combined with the second-difference dissipation coefficient, κ_2, to form a modified second-difference dissipation coefficient, $\tilde{\kappa}_2$, such that

$$\tilde{\kappa}_2 = \kappa_2 + \sigma\kappa_4,$$

where σ is a lumping factor. A value of $\sigma = 8$ has been shown to work well for the Navier-Stokes solutions with scalar dissipation [42]. Current work with matrix dissipation uses $\sigma = 12$. The modified fourth-difference dissipation coefficient, $\tilde{\kappa}_4$, is set to zero. Applying this lumping approach reduces the number of matrix entries for the inviscid terms, reducing the memory requirements for the code.

The viscous terms, however, still possess a relatively large stencil. To mitigate this, the cross-derivative terms that appear in the viscous stresses are dropped when constructing the first-order Jacobian. This approach reduces the stencil of all interior nodes to nearest neighbors only, matching the stencil size of the inviscid terms, which is substantially smaller than that of the full flow Jacobian. The linearization of the viscous flux SATs for the Navier-Stokes equations is also modified to ignore the tangential derivatives, which are analogous to the cross-derivatives. Additionally, the viscosity term appearing in the viscous fluxes is treated as a constant when forming the approximate Jacobian.

No approximations are made to the discretization of the turbulence model when constructing the Jacobian entries that arise due to the solution of this extra equation, since all cross-derivatives were dropped during the coordinate transformation.

The implicit Euler method requires a time step whose inverse is added to the diagonal elements of $\mathcal{A}_1$. A spatially varying time step has been shown to improve the convergence rates of Newton-Krylov algorithms, leading to the use of the following value:

$$\Delta t^{(n)}_{j,k,m} = \frac{J_{j,k,m}\Delta t^{(n)}_{\text{ref}}}{1 + \sqrt[3]{J_{j,k,m}}}, \tag{31}$$

where (j, k, m) denote the indices of the node to which this time step is being applied. Since the solver uses the unscaled flow variables $\mathbf{Q}$, instead of the transformed variables $\hat{\mathbf{Q}}$, the J term that results from the coordinate transformation is lumped into the numerator of (31). The reference time step is

$$\Delta t^{(n)}_{\text{ref}} = a(b)^n,$$

where typical values used for turbulent flow solutions are $a = 0.001$ and $b = 1.3$.

Once formed, the first-order Jacobian is factored using block incomplete lower-upper factorization (BILU) with fill level p in order to construct the preconditioner used throughout the solution process. This is a computationally expensive task, especially in the approximate-Newton phase, which requires many outer iterations. Previous work [6, 43] has shown that lagging the update of the preconditioner (freezing it for a number of iterations) during the start-up phase can positively impact the efficiency of the flow solver. A typical fill level for the factorization is 2.

Effective preconditioning is critical to an efficient parallel linear solver. Two approaches to parallel preconditioning, namely additive-Schwarz [44] and approximate-Schur [45], have been previously investigated

in the context of a parallel Newton-Krylov flow solver for the Euler [6, 17] and Navier-Stokes [17] equations, with a thorough description of their application to the current linear system provided in the references. The approximate-Schur parallel preconditioner is used in the current work, making use of the interface nodes in constructing the global Schur complement.

An important part of using a start-up phase is knowing when a suitable iterate has been found to initiate the inexact-Newton phase. For this purpose, the relative drop in the residual is used:

$$R_d^{(n)} \equiv \frac{||\boldsymbol{\mathcal{R}}^{(n)}||_2}{||\boldsymbol{\mathcal{R}}^{(0)}||_2}. \tag{32}$$

For turbulent flows, once this value reaches 0.0001, i.e. the residual has dropped by 4 orders of magnitude in the approximate-Newton phase, the algorithm switches to the inexact-Newton method. This initial drop is larger than the one required for inviscid or laminar solutions for two reasons. First, the turbulence quantity fluctuates substantially more than the mean-flow quantities during the start-up phase, necessitating a longer start-up than flow solutions dealing with inviscid or laminar flows. Second, due to the use of grids with much finer spacing near the surface of the aerodynamic shape, the initial residual, $\boldsymbol{\mathcal{R}}^{(0)}$, begins with a much larger value, but drops by one to two orders of magnitude very quickly before settling into a convergence pattern similar to that observed with inviscid or laminar solves. Hence, the relative residual drop threshold, R_d, is adjusted to compensate for these differences. This parameter may need to be adjusted slightly depending on the complexity of the flow being solved.

5.1.2 Inexact-Newton phase

The inexact-Newton phase uses a different scheme for the reference time step, designed to ramp the time step toward infinity more rapidly than in the approximate-Newton phase. This eventually eliminates the inverse time term from the left-hand-side of the discretized Navier-Stokes equations. The present work involves the use of a scheme developed by Mulder and van Leer [46], by which a new reference time step is calculated and used in (31):

$$\Delta t_{\text{ref}}^{(n)} = \max\left[\alpha\left(R_d^{(n)}\right)^{-\beta}, \Delta t_{\text{ref}}^{(n-1)}\right],$$

where $\beta \in [1.5, 2.0]$ and α is calculated as

$$\alpha = a(b)^{n_{\text{Newt}}}\left(R_d^{(n_{\text{Newt}})}\right)^{\beta},$$

and n_{Newt} is the first iteration of the inexact-Newton phase.

In contrast with the approximate-Newton phase, the inexact-Newton phase uses the full second-order accurate Jacobian. However, since we use a Krylov subspace method, we do not need to form the full Jacobian matrix, $\mathcal{A}$, explicitly. Instead, only Jacobian-vector products are required, which can be approximated using a first-order forward difference:

$$\mathcal{A}^{(n)}\mathbf{v} \approx \frac{\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}^{(n)} + \epsilon\mathbf{v}) - \boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}^{(n)})}{\epsilon}.$$

The parameter ϵ is determined from

$$\epsilon = \sqrt{\frac{N_u\delta}{\mathbf{v}^T\mathbf{v}}},$$

where N_u is the number of unknowns, and $\delta = 10^{-12}$. The approximate Jacobian, $\mathcal{A}_1$, is still used for preconditioning the system.

Finally, neither the approximate-Newton nor the inexact-Newton phase solves its respective linear system exactly. Instead, the following inequality is used to govern how far the system is solved:

$$||\boldsymbol{\mathcal{R}}^{(n)} + \mathcal{A}^{(n)}\Delta\boldsymbol{\mathcal{Q}}^{(n)}||_2 \leq \eta_n||\boldsymbol{\mathcal{R}}^{(n)}||_2,$$

where the forcing parameter η_n is specified. If it is too small, the linear system will be over-solved and will take too much time, but if it is too large, non-linear convergence will suffer. For the present work, a value

of 0.05 is used for the approximate-Newton phase, while 0.01 is used for the inexact-Newton phase.

5.1.3 Special Considerations for Turbulence Model

The addition of the turbulence model to the linear system of (30) presents some unique challenges, as the scaling of the linear system can be adversely affected, resulting in unpredictable behavior of the linear solver. The improper scaling arises from several factors. First, the turbulence model does not contain the inherent geometric scaling present in the mean flow equations (division by J). Second, the turbulence quantity can be as large as 1000 or higher in the converged solution, while the nondimensionalized mean flow quantities rarely exceed 2. Finally, the terms that result from the linearization of the turbulence model with respect to the mean flow variables add large off-diagonal values to the Jacobian. Hence, a more sophisticated scaling approach has been implemented to account for these discrepancies, based on the work done by Chisholm and Zingg [47], in order to obtain an efficient and accurate solution of the linear system. The row, or equation, scaling of the mean flow equations is achieved by multiplying the equations by a factor that includes the metric Jacobian, removing the inherent geometric scaling, while the turbulence model is scaled by 10^{-3}. This value accounts for the maximum turbulence value that is likely to be encountered in the flow solve, effectively normalizing the turbulence equation by that quantity. In order to normalize the flow variable values, the turbulence variable quantity is also multiplied by 10^{-3}. Hence, instead of solving the system presented in (30), the solution algorithm tackles a scaled system of the form:

$$\left[S_a S_r \left(\frac{\mathcal{I}}{\Delta t} + \mathcal{A}^{(n)}\right) S_c\right] \left(S_c^{-1} \Delta \boldsymbol{\mathcal{Q}}^{(n)}\right) = -S_a S_r \boldsymbol{\mathcal{R}}^{(n)}, \tag{33}$$

where S_r and S_c are the row and column scaling matrices, respectively. S_a is an auto-scaling matrix used to bring the magnitudes of the individual equation components within an order of magnitude, further improving the scaling of the linear system. In the current implementation, these matrices are defined as

$$S_r = \mathrm{diag}\left(S_{r1}, S_{r2}, ..., S_{rN}\right), \quad S_c = \mathrm{diag}\left(S_{c1}, S_{c2}, ..., S_{cN}\right),$$

where

$$S_{ri} = \begin{bmatrix} J_i^{2/3} & & & & & & \\ & J_i^{2/3} & & & & & \\ & & J_i^{2/3} & & & & \\ & & & J_i^{2/3} & & & \\ & & & & J_i^{2/3} & & \\ & & & & & 10^{-3} J_i^{-1/3} \end{bmatrix}, \quad S_{ci} = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 10^3 \end{bmatrix},$$

and J_i is the value of the metric Jacobian at the i^{th} node in the computational domain. The values in the auto-scaling matrix are calculated based on the equation-wise residual L_2-norms of the partially scaled system $S_r \boldsymbol{\mathcal{R}}^{(n)}$, and are identical for each node in the domain. Instead, they scale the individual component equations by different amounts. Any residual values required for the time step calculation make use of the partially scaled residual $S_r \boldsymbol{\mathcal{R}}^{(n)}$.

During the convergence to steady state, it is not atypical to encounter negative values of $\tilde{\nu}$ in the flowfield. These values are nonphysical and merely a result of the occurrence of large transients in the solution, especially during the early stages of convergence or after the switch from the approximate-Newton phase to the inexact-Newton phase. However, it is important to address these negative values, since they can destabilize the solution process. The approach taken is to trim any negative $\tilde{\nu}$ values to a very small positive quantity. In particular, any negative turbulence quantities that are encountered on a solid surface are trimmed to $10^{-14}\mu/\rho$, while all other locations are trimmed to $10^{-3}\mu/\rho$. The local μ/ρ term is introduced such that advective and diffusive fluxes do not vanish completely from regions where several adjacent nodes are trimmed during the same iteration.

Additional trimming is used when dealing with the value of vorticity, S. In order to avoid numerical problems, this value is not allowed to fall below 8.5×10^{-10}. Finally, the vorticity-like term, $\tilde{S}$, cannot

be allowed to reach zero or become negative, which would have a destabilizing effect on the values of the production and destruction terms. We have found that preventing this value from becoming smaller than $10^{-5}M$ works well, where M is the farfield Mach number.

5.2 Unsteady solutions

Using an ESDIRK scheme with s stages, the fully discretized Navier-Stokes equations form a system of non-linear equations:

$$\hat{\boldsymbol{\mathcal{R}}}_k^{(n)}(\boldsymbol{\mathcal{Q}}_k^{(n)},\ldots,\boldsymbol{\mathcal{Q}}_1^{(n)},\boldsymbol{\mathcal{Q}}^{(n-1)}) = \frac{\boldsymbol{\mathcal{Q}}_k^{(n)}-\boldsymbol{\mathcal{Q}}^{(n-1)}}{\lambda\Delta t} + \frac{1}{\lambda}\sum_{j=1}^{s} A_{kj}\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}_j^{(n)}) = 0, \quad k=2,\ldots,s, \tag{34}$$

where $\hat{\boldsymbol{\mathcal{R}}}$ defines the unsteady residual, and $\boldsymbol{\mathcal{R}}$ is the spatial residual (29) defined in Section 5. Each stage is treated as a steady problem in pseudo time, and the unsteady residual equations are solved with the inexact-Newton method with iteration counter p:

$$\hat{\mathcal{A}}_k^{(p)}\Delta\boldsymbol{\mathcal{Q}}_k^{(p)} = -\hat{\boldsymbol{\mathcal{R}}}^{(p,n)}(\boldsymbol{\mathcal{Q}}_k^{(p)},\boldsymbol{\mathcal{Q}}_{k-1}^{(n)}\ldots,\boldsymbol{\mathcal{Q}}_1^{(n)},\boldsymbol{\mathcal{Q}}^{(n-1)}), \quad k=2,\ldots,s, \tag{35}$$

where $\Delta\boldsymbol{\mathcal{Q}}_k^{(p)} = \boldsymbol{\mathcal{Q}}_k^{(p+1)} - \boldsymbol{\mathcal{Q}}_k^{(p)}$ and the Jacobian becomes:

$$\hat{\mathcal{A}}_k^{(p)} = \frac{1}{\lambda\Delta t}\mathcal{I} + \frac{\partial\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}_k^{(p)})}{\partial\boldsymbol{\mathcal{Q}}_k^{(p)}}, \quad k=2,\ldots,s. \tag{36}$$

No approximate-Newton phase is needed.

5.2.1 Polynomial extrapolation

The performance of Newton's method can be improved by providing better initial iterates. Previous solution information can be used to generate low-order inexpensive approximations of the solution at the next time step.

Consider a sequence of solution values $u^{(n-1)},\ldots,u^{(n-k)}$ at $t^{(n-1)},\ldots,t^{(n-k)}$. These times do not have to be equally spaced or monotonic. The solution $u^{(n)}$ at $t^{(n)}$ is then approximated by:

$$u^{(n)} = \sum_{i=1}^{k} l_{n-i}(t^{(n-i)})u^{(n-i)}, \tag{37}$$

where

$$l_{n-i}(t^{(n-i)}) = \prod_{j=1,j\neq i}^{k+1}\left(\frac{t^{(n)}-t^{(n-j)}}{t^{(n-i)}-t^{(n-j)}}\right). \tag{38}$$

Increasing the number of past solutions increases the accuracy of the approximation. In this work, three past solutions are used as a balance between accuracy and memory usage.

5.2.2 Delayed preconditioner updates

The temporal component of the Jacobian in (36) is constant and is often significantly larger than the change in the spatial Jacobian over a stage or an entire time step. Therefore, it is possible to freeze the preconditioner over a stage or time step without a significant impact on the convergence of the system. Current results were obtained by freezing the preconditioner over each time step, resulting in a significant reduction in CPU time.

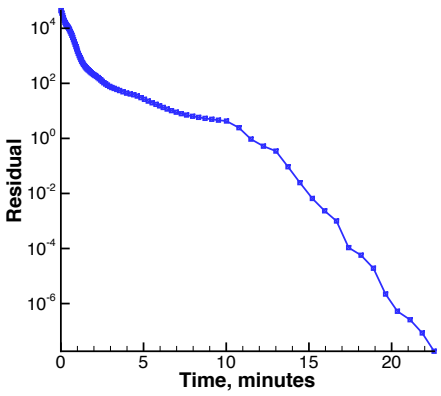


Figure 1: Convergence history for transonic solution at $\alpha = 2.75°$ (with 704 processors)

5.2.3 Termination of non-linear iterations

The temporal integration has a certain level of truncation error. The convergence of the residual equations can, therefore, be terminated when the residual is less than this error. This reduces computational cost and is done without any loss in global accuracy. In this work, termination is based on a preset reduction from the initial residual value. The necessary relative tolerance is fairly step size independent since a reduction in step size will result in a better initial iterate from polynomial extrapolation and therefore a lower initial residual. A typical relative tolerance used in the current work is 10^{-6}.

6 Results

This section presents the application of the current algorithm to range of steady and unsteady flow simulations. For steady simulations, the solution of transonic flow around the NASA CRM wing-body geometry [48] will be shown, along with a parallel scaling study making use of both the ONERA M6 wing and CRM wing-body geometries. The Taylor-Green vortex flow [49] is used to highlight the unsteady solution capabilities of the algorithm.

6.1 Transonic flow around CRM wing-body geometry

The first case concerns the solution of a transonic flow around the CRM wing-body geometry. The O-O topology structured multi-block grid, obtained from the organizers of the 5th Drag Prediction Workshop (DPW5), contains 5.97 million nodes, with an off-wall spacing of 5.3×10^{-6} chord units. The original grid has been subdivided into 704 blocks to leverage the parallel capabilities of the current algorithm. It has also been scaled down from dimensional units, with mean aerodynamic chord (MAC) of 275.80 inches, to have an MAC of 1.0.

To investigate the range of angles of attack during which buffet onset should occur, the flow conditions used in this case are

$$\text{Ma} = 0.85,\ \text{Re} = 5.00 \times 10^6,\ \alpha = 2.00° \text{ to } 4.00°.$$

Solutions were computed with the scalar dissipation model for angles of attack up to 3.15°. A representative residual convergence plot is shown in Figure 1. The nonlinear residual is reduced by 12 orders of magnitude, and both solution phases of the steady flow solution algorithm are clearly visible. Figure 2 shows the streamlines above the wing for solutions at all angles of attack.

At 3.25° and above, the steady solution algorithm fails to converge, with stalled nonlinear convergence. Unsteady flow features were assumed to be the cause of the convergence difficulty. Hence, time-accurate simulations were undertaken. The unsteady solution algorithm was used to run solutions for all angles above 3.25°. After advancing all solutions 160 nondimensional time units, no oscillations were observed in the

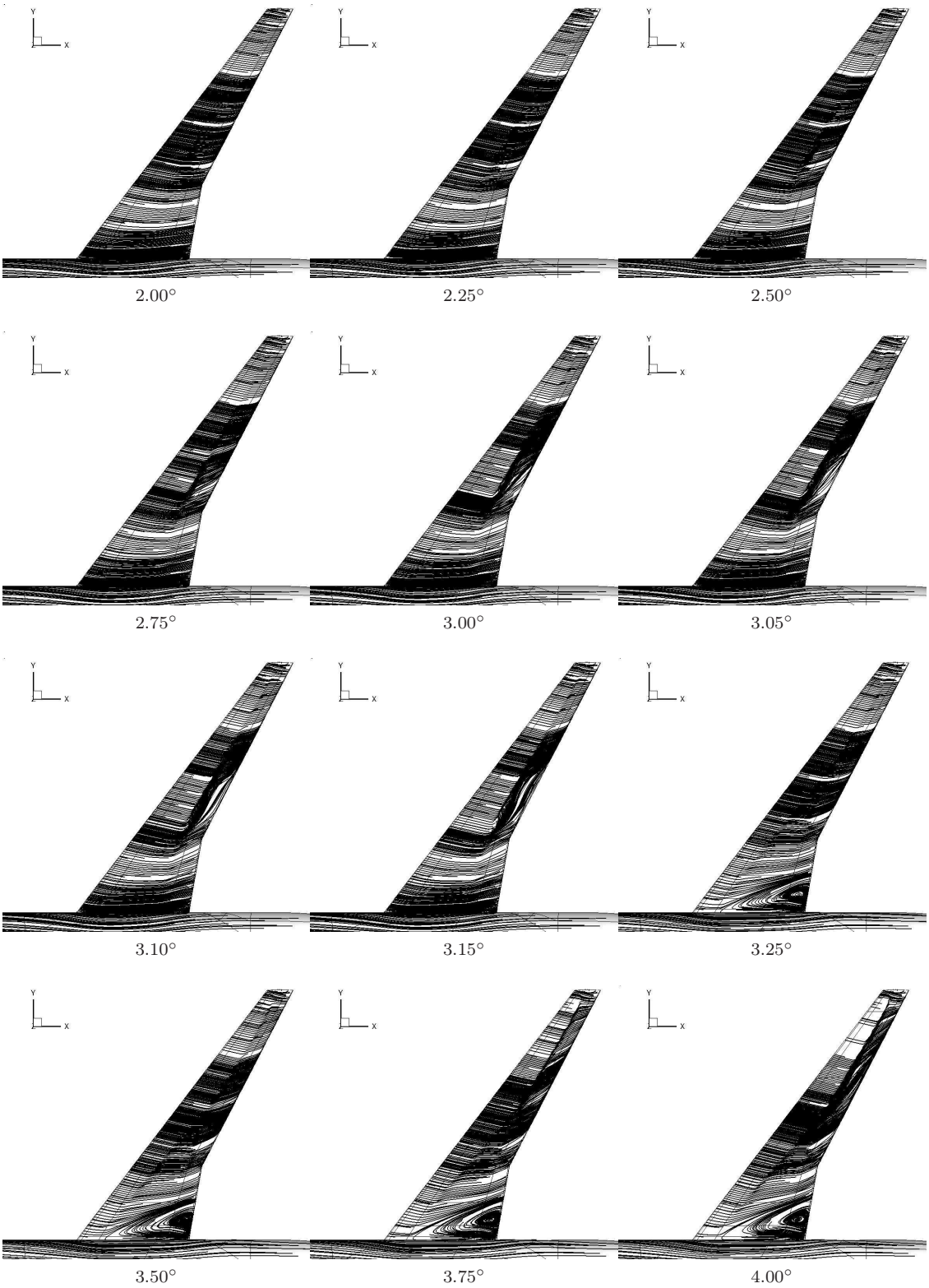
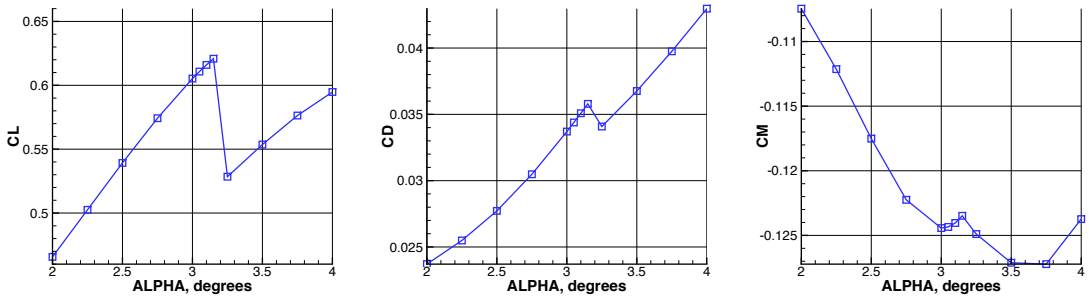


Figure 2: Transonic CRM flow solutions at various angles of attack



Lift Drag Moment

Figure 3: CRM force coefficients

force coefficients, and the solution was converging to a steady flow. Using the final unsteady solution as a starting point, the steady solution algorithm was used to fully converge the solutions. Subsequently, further simulations with substantially reduced time-step ramping and reductions based on residual behavior allowed the steady-state solution algorithm to convergence without any use of the unsteady algorithm. At all angles above 3.25°, a substantially different flow pattern is observed, as can be seen in Figure 2. A substantial recirculation bubble is present, originating at the wing-body junction. It is not presently known whether the large separation region represents a physically accurate phenomenon or is an artifact associated with the interplay of the grid resolution in the area and the turbulence model.

Figure 3 presents the lift, drag, and moment coefficients for all angles of attack. All coefficient values exhibit a discontinuity at an angle of attack of 3.25° due to the presence of the large recirculation bubble.

6.2 Parallel scaling study

In order to evaluate the parallel performance of the algorithm, a parallel scaling study was conducted for the solution of a steady subsonic flow around the ONERA M6 wing and a steady transonic flow around the CRM wing-body geometry. The subsonic flow conditions are

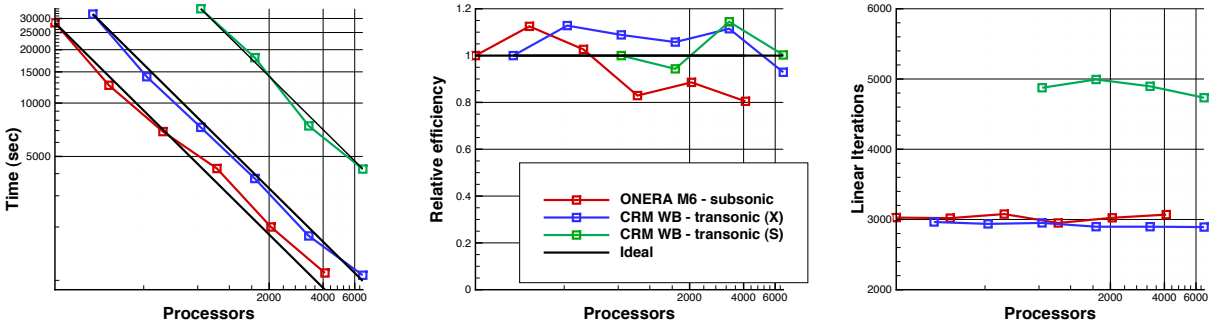
$$\text{Ma} = 0.30,\ \text{Re} = 7.48 \times 10^6,\ \alpha = 2.0°,$$

while transonic flow conditions are

$$\text{Ma} = 0.85,\ \text{Re} = 5.00 \times 10^6,\ \text{and}\ \text{C}_\text{L} = 0.500.$$

The ONERA M6 grid consists of a C-H topology mesh comprising 8192 blocks, with a total of 40 million nodes. The off-wall spacing is 8×10^{-7} root chord units. The CRM wing-body case was run on two grids, with each grid consisting of a 6656 block O-O topology mesh obtained from the "X" and "S" grid levels used in DPW5. The "X" grid contains 48 million nodes and an off-wall spacing of 1.83×10^{-6} MAC units, while the "S" grid contains 154 million nodes and an off-wall spacing of 1.29×10^{-6} MAC units. The subsonic flow solution was computed with scalar artificial dissipation, converging the residual by 12 orders of magnitude, while the transonic solutions made use of the matrix dissipation model, converging the residual by 10 orders of magnitude. All meshes are perfectly load-balanced, with an equal number on nodes in each block.

The results for all cases, presented in Figure 4, show that the code exhibits excellent parallel scaling characteristics up to 6656 processors. The performance of the code is measured by relative efficiency, which is based on the lowest possible number of processors that each case can be computed with. Due to memory requirements, the ONERA M6 case can be run with a minimum of 128 processors, while the two CRM cases require a minimum of 208 and 832 processors, respectively. In the range of processors considered, the relative efficiency does not drop below 80%. In fact, many processor counts exhibit super-linear scaling, due partly to the changing form the the preconditioner, with different numbers of interface nodes contributing to the global Schur complement, and partly to the manner in which the parallel computing hardware manages parallel communication with varying numbers of processors. Additionally, the nearly constant number of linear iterations required to converge the solutions at different processor numbers highlights the effectiveness of the approximate-Schur preconditioner even when large numbers of processors (>6000) are used.



Time | Relative parallel efficiency | Linear iterations

Figure 4: Parallel scaling performance of code

6.3 Taylor-Green vortex flow

The final case is the Taylor-Green vortex flow. It was originally developed to study vortex stretching, the lengthening of vortices which creates small eddies from larger ones. This process is believed to be an important mechanism in the turbulent energy cascade [49]. The flow is initialized with a smooth, uni-modal velocity field. As the solution develops, smaller and smaller modes are generated, eventually mimicking homogeneous non-isotropic turbulence. Finally, the turbulence decays as the smallest modes are dissipated due to viscous effects. The initial conditions are:

$$\begin{aligned} u &= M_\circ \sin(x)\cos(y)\cos(z), \\ v &= -M_\circ \cos(x)\sin(y)\cos(z), \\ w &= 0, \\ p &= p_\circ + \tfrac{\rho_\circ M_\circ^2}{16}\left(\cos(2x)+\cos(2y)\right), \\ \rho &= p/p_\circ, \end{aligned}$$

where $\rho_\circ = 1$ and $p_\circ = \frac{1}{\gamma}$. To minimize the effects of compressibility and to be consistent with the AIAA's 1st International Workshop on High-Order CFD Methods, the free-stream Mach number is set to 0.1. The Reynolds number is 1600, which corresponds to a peak Taylor microscale Reynolds number of about 22, and the Prandtl number is 0.71. The convective time unit is defined as $[t_c] = \frac{1}{M_\circ}[t]$, and the simulation is advanced to $t_{final} = 20t_c$. The simulation domain is a periodic box, $-\pi \leq x, y, z \leq \pi$.

6.3.1 Basic Definitions

In this study, kinetic energy is defined as:

$$E_k = \frac{1}{2V}\int_V \rho \mathbf{v}\cdot\mathbf{v} dV,$$

where V is the volume, $\mathbf{v}$ is the velocity vector and ρ is the density. Dissipation rate is then $e = -\frac{dE_k}{dt}$, and enstrophy is defined as,

$$\epsilon = \frac{1}{2V}\int_V \rho\boldsymbol{\omega}\cdot\boldsymbol{\omega} dV,$$

6.3.2 Grid convergence studies

A grid convergence study was conducted for second- and fourth-order spatial discretizations on four successively finer grids: 32^3, 64^3, 128^3 and 256^3 nodes. Each grid was decomposed into blocks of 32^3 nodes with a one-to-one distribution of blocks to processors. The workload per processor is therefore constant. The solutions were advanced with the fourth-order ESDIRK method and a constant maximum CFL number,

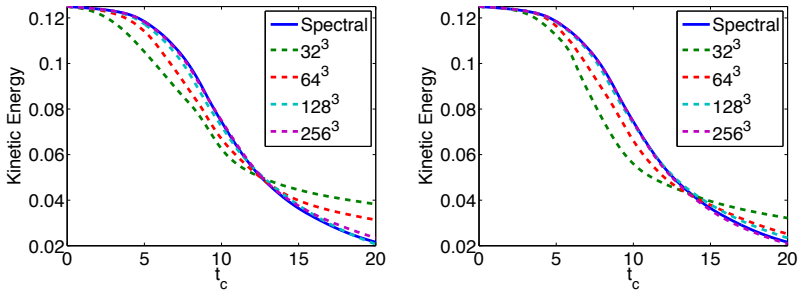


Figure 5: Taylor-Green flow: temporal evolution of kinetic energy.

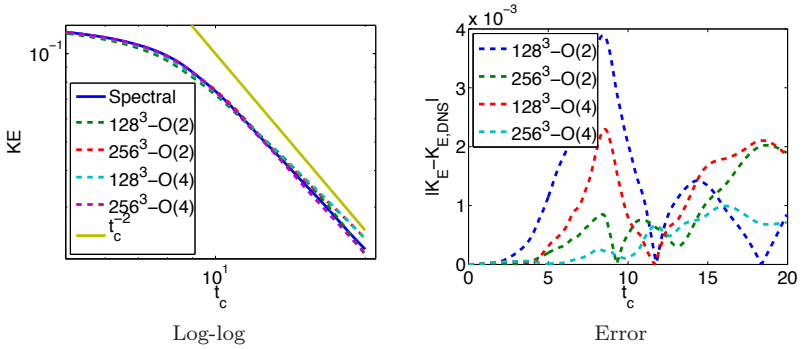


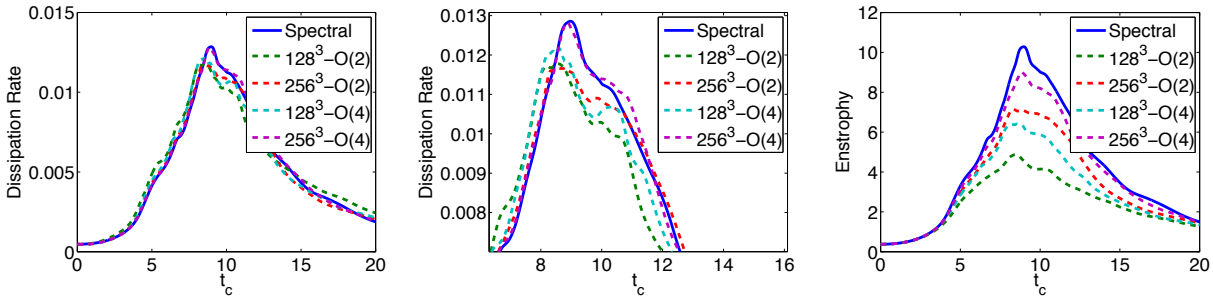
Figure 6: Taylor-Green flow: temporal evolution of kinetic energy

~ 31 for second-order and ~ 50 for the fourth-order simulation. The step size is constant for grids of the same size, therefore the difference in CFL number is due to the difference in the maximum value of the modified wave number between the second and fourth-order discretizations. The fourth-order discretization makes use of the fourth-order first-derivative operator twice to construct the second derivative, resulting in a non-compact-stencil formulation.

Figure 5 displays the evolution of kinetic energy between the second- and fourth-order simulations and compares them with the 512^3 mode dealiased spectral direct numerical simulation (DNS) of van Rees *et al.* [50]. In both cases, the coarsest simulations are not able to accurately capture the decay of kinetic energy. The higher-frequency modes cannot be represented on these grids and are, therefore, damped by the dissipation model. As a consequence, less energy is transferred to the higher frequency modes, and this is believed to be the cause of the lower dissipation rate and higher kinetic energy present at the end of the simulation. The fourth-order simulations do not dissipate quite as early as the second-order simulations and the 64^3 simulation is able to recover a more accurate dissipation rate in the latter half of the simulation. However, there is still a noticeable deviation from the DNS results.

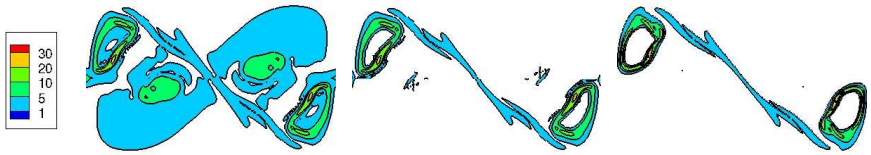
The finer simulations more accurately capture the decay of kinetic energy. These simulations are isolated in log-log form in Figure 6 along with the error with respect to the DNS result in [50]. The coarser second-order solution still dissipates too early and only the finest fourth-order simulation lies on top of van Rees' DNS result. The fine second-order and coarser fourth-order results are comparable, accurately capturing the initial dissipation, but under-predicting the final dissipation rate.

These conclusions are further supported by considering the evolution of dissipation rate directly, as seen in Figure 7. The coarser fourth-order simulation marginally over-predicts the dissipation rate just before the peak, but also has a slightly higher and more pronounced peak than the finer second-order result. The main difference is that the coarse fourth-order simulation costs about 6.3 times less that the fine second-order simulation in terms of CPU time. Again, only the fine fourth-order solution approximates the reference solution well. The relatively high accuracy of the evolution of kinetic energy is somewhat surprising considering the large deviation in normalized enstrophy, also shown in Figure 7. Enstrophy is an



Dissipation rate | Dissipation rate (zoomed) | Enstrophy

Figure 7: Taylor-Green flow: temporal evolution of normalized enstrophy and dissipation rate of kinetic energy



$\frac{1}{M_\circ}|\omega|$ | $\frac{1}{M_\circ}|\omega_x|$ | $\frac{1}{M_\circ}|\omega|$ - van Rees *et al.*

Figure 8: Taylor-Green flow (256^3 grid): vorticity magnitude and x-component of vorticity of the present study compared with vorticity norm from van Rees *et al.* [50]

indication of the resolving power of the discretization and the results suggest that even the finest simulation is under-resolved.

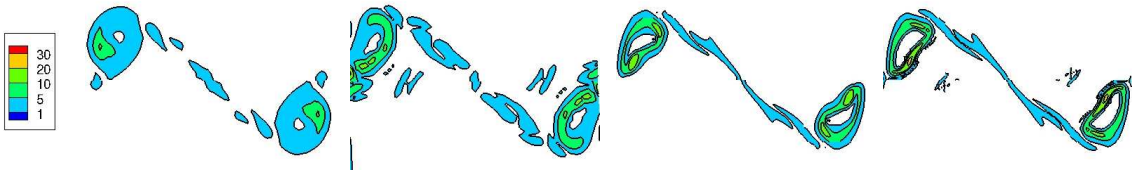
Finally, contours of the vorticity norm at one of the periodic faces, $x = \pi$, are shown in Figure 8 for the fourth-order result obtained on the finest grid. The structures presented by van Rees *et al.* [50] are recovered; however, extra structures are visible in the present simulations. These structures are fairly large, but are formed by the lowest vorticity contour lines. If only the x-component of the vorticity is shown, the structures then match very well. The difference is likely due to the difference in grid resolution in the present simulation. Figure 9 shows contours of only the x-component of vorticity for the other high-resolution simulations. The circular high-vorticity structure of the second-order 128^3 result is weak and spread out. By increasing the order, the vorticity becomes much stronger and more annular in structure, however, there are some erroneous artifacts. The finest second-order solution is similar to the complementary fourth-order solution. However, the regions of high vorticity extend further towards the center of the circular structure.

6.3.3 Temporal convergence studies

The temporal accuracy and efficiency of the second and fourth-order ESDIRK methods were evaluated in a temporal convergence study with time steps $\Delta = 0.005, \ldots, 0.8$; this corresponds to maximum CFL $\approx 5, \ldots, 808$. Simulations were computed on a 128^3 grid with the fourth-order non-compact-stencil spatial discretization. The reference solution was obtained with the classical fourth-order Runge-Kutta (RK4) method and a time step of 0.003125, corresponding to CFL ≈ 0.3. The error is computed as the root-mean-square of the difference in kinetic energy:

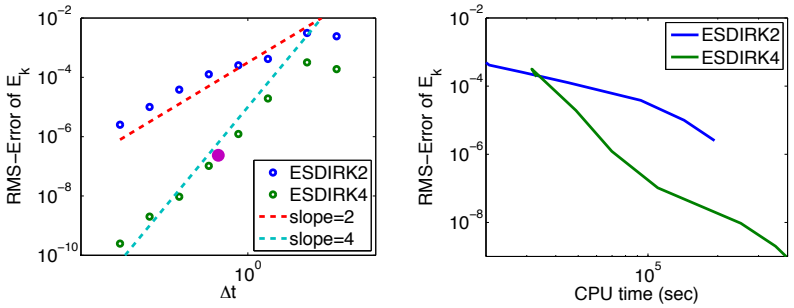
$$\text{RMS-error} = \sqrt{\frac{\sum_{i=0}^{\#\text{ time steps}} (E_{k,i} - E_{k,i,ref})^2}{\#\text{ time steps}}}.$$

The temporal convergence and efficiency of the ESDIRK methods are shown in Figure 10, along with an estimation of the temporal error found in the spatial convergence study. The design order of each method is recovered. The main result is the efficiency of the methods: the CPU time required to obtain a preset



128^3 grid - Order 2 128^3 grid - Order 4 256^3 grid - Order 2 256^3 grid - Order 4

Figure 9: Taylor-Green flow: x-component of vorticity



Convergence Efficiency

Figure 10: Taylor-Green flow: convergence and efficiency of second- and fourth-order ESDIRK methods. The estimate of the temporal error found in the spatial convergence study is shown by (•)

level of error. ESDIRK2 is efficient only for simulations requiring a minimum level of temporal accuracy. As the required accuracy is lowered, ESDIRK4 quickly and decidedly becomes more efficient. Furthermore, accurate simulations were obtained at CFL values well beyond the stable region of explicit methods, like RK4, and in less CPU time, confirming the value of implicit methods for such problems.

7 Conclusions and Future Work

The parallel Newton-Krylov-Schur algorithm with the SBP-SAT spatial discretization was shown to provide accurate and efficient flow solutions to the three-dimensional Navier-Stokes equations and the one-equation Spalart-Allmaras turbulence model. Both steady and unsteady flow solutions were considered, with time marching carried out with ESDIRK methods.

The solution of transonic flows around the NASA Common Research Model wing-body configuration over a wide range of angles of attack highlights the capability of the algorithm to converge for complex flows with substantial separation. Parallel scaling studies on the same geometry, as well as the ONERA M6 wing, show excellent algorithm scaling characteristics with up to 6656 processors.

Taylor-Green vortex results highlight the advantages of high-order spatial and temporal discretization; the algorithm accurately captures the evolution of the large vortical structures and integrated quantities.

Future work will extend the higher-order spatial discretization to the turbulence model, allowing higher-order solutions for the steady and unsteady RANS equations.

Acknowledgments

The authors gratefully acknowledge financial assistance from the Natural Sciences and Engineering Research Council (NSERC), the Ontario Graduate Scholarship program, the Canada Research Chairs program, Bombardier Aerospace, and the University of Toronto.

Computations were performed on the GPC supercomputer at the SciNet HPC Consortium and the Guillimin supercomputer of the CLUMEQ consortium, both part of Compute Canada.

Appendix

The following are the complete forms of the conservative formulation of the $B_{\text{int},j}$ matrices, where $j = \xi, \eta, \zeta$.

$$B_{\text{int},j} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -a_1 u - a_2 v - a_3 w & a_1 & a_2 & a_3 & 0 \\ -a_2 u - a_4 v - a_5 w & a_2 & a_4 & a_5 & 0 \\ -a_3 u - a_5 v - a_6 w & a_3 & a_5 & a_6 & 0 \\ b_{51} & b_{52} & b_{53} & b_{54} & a_7 \end{bmatrix},$$

where, for the ξ-direction,

$$\begin{aligned} a_1 &= t_1(4/3\xi_x^2 + \xi_y^2 + \xi_z^2), & a_2 &= t_1 1/3 \xi_x \xi_y, \\ a_3 &= t_1 1/3 \xi_x \xi_z, & a_4 &= t_1(\xi_x^2 + 4/3\xi_y^2 + \xi_z^2), \\ a_5 &= t_1 1/3 \xi_y \xi_z, & a_6 &= t_1(\xi_x^2 + \xi_y^2 + 4/3\xi_z^2), \\ a_7 &= t_2\gamma(\xi_x^2 + \xi_y^2 + \xi_z^2), & & \\ b_{52} &= -a_7 u + a_1 u + a_2 v + a_3 w, & b_{53} &= -a_7 u + a_2 u + a_4 v + a_5 w, \\ b_{54} &= -a_7 u + a_3 u + a_5 v + a_6 w, & & \\ t_1 &= \rho^{-1}(\mu + \mu_t), & t_2 &= \rho^{-1}(\mu/Pr + \mu_t/Pr_t), \end{aligned}$$

and

$$b_{51} = a_7\left(-e/\rho + (u^2 + v^2 + w^2)\right) - a_1 u^2 - a_4 v^2 - a_6 w^2 - 2(a_2 uv + a_3 uw + a_5 vw).$$

References

[1] Mavriplis, D. J., J. C. Vassberg, E. N. Tinoco, M. Mani, O. P. Brodersen, B. Eisfeld, R. A. Wahls, J. H. Morrison, T. Zickuhr, D. Levy, and M. Murayama. Grid quality and resolution issues from the drag prediction workshop series. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–2008–0930. Reno, Nevada, January 2008.

[2] Jespersen, D., T. Pulliam, and P. Buning. Recent enhacement to OVERFLOW (Navier-Stokes code). In *35th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–97–0644. Reno, Nevada, January 1997.

[3] Nielsen, E. J., R. W. Walters, W. K. Anderson, and D. E. Keyes. Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code. In *12th AIAA Computational Fluid Dynamics Conference*. San Diego, California, United States, 1995. AIAA–95–1733.

[4] May, G. and A. Jameson. Unstructured algorithms for inviscid and viscous flows embedded in a unified solver architecture: Flo3xx. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–2005–0318. Reno, Nevada, January 2005.

[5] Mavriplis, D. J. Grid resolution of a drag prediction workshop using the nsu3d unstructured mesh solver. In *17th AIAA Computational Fluid Dynamics Conference*, AIAA–2005–4729. Toronto, Canada, June 2005.

[6] Hicken, J. E. and D. W. Zingg. A parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms. *AIAA Journal*, 46(11):2773–2786, November 2008.

[7] Dias, S. C. and D. W. Zingg. A high-order parallel Newton-Krylov flow solver for the Euler equations. In *19th AIAA Computational Fluid Dynamics Conference*, AIAA–2009–3657. San Antonio, Texas, United States, June 2009.

[8] Osusky, M., J. E. Hicken, and D. W. Zingg. A parallel Newton-Krylov-Schur flow solver for the Navier-Stokes equations using the SBP-SAT approach. In *48th AIAA Aerospace Sciences Meeting and Aerospace Exposition*, AIAA–2010–116. Orlando, Florida, United States, January 2010.

[9] Carpenter, M. H., D. Gottlieb, and S. Abarbanel. Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: methodology and application to high-order compact schemes. *Journal of Computational Physics*, 111(2):220–236, 1994.

[10] Hesthaven, J. S. A stable penalty method for the compressible Navier-Stokes equations: III. multidimensional domain decomposition schemes. *SIAM Journal on Scientific Computing*, 20(1):62–93, 1998.

[11] Carpenter, M. H., J. Nordström, and D. Gottlieb. A stable and conservative interface treatment of arbitrary spatial accuracy. *Journal of Computational Physics*, 148(2):341–365, 1999.

[12] Nordström, J. and M. H. Carpenter. High-order finite difference methods, multidimensional linear problems, and curvilinear coordinates. *Journal of Computational Physics*, 173(1):149–174, 2001.

[13] Svärd, M., M. H. Carpenter, and J. Nordström. A stable high-order finite difference scheme for the compressible Navier-Stokes equations, far-field boundary conditions. *Journal of Computational Physics*, 225(1):1020–1038, July 2007.

[14] Svärd, M. and J. Nordström. A stable high-order finite difference scheme for the compressible Navier-Stokes equations, no-slip wall boundary conditions. *Journal of Computational Physics*, 227(10):4805–4824, May 2008.

[15] Nordström, J., J. Gong, E. van der Weide, and M. Svärd. A stable and conservative high order multi-block method for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 228(24):9020–9035, 2009.

[16] Nordström, J. and M. H. Carpenter. Boundary and interface conditions for high-order finite-difference methods applied to the Euler and Navier-Stokes equations. *Journal of Computational Physics*, 148(2):621–645, 1999.

[17] Hicken, J. E., M. Osusky, and D. W. Zingg. Comparison of parallel preconditioners for a Newton-Krylov flow solver. In *6th International Conference on Computational Fluid Dynamics*. St. Petersburg, Russia, July 2010.

[18] Osusky, M. and D. W. Zingg. A parallel Newton-Krylov-Schur flow solver for the Reynolds-Averaged Navier-Stokes equations. In *50th AIAA Aerospace Sciences Meeting and Aerospace Exposition*, AIAA–2012–0442. Nashville, Tennessee, United States, January 2012.

[19] Spalart, P. R. and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. In *30th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–92–0439. Reno, Nevada, United States, January 1992.

[20] Hicken, J. E. and D. W. Zingg. The role of dual consistency in functional accuracy: error estimation and superconvergence. In *20th AIAA Computational Fluid Dynamics Conference*, AIAA–2011–3855. Honolulu, Hawaii, United States, June 2011.

[21] Hicken, J. E. and D. W. Zingg. Superconvergent functional estimates from summation-by-parts finite-difference discretizations. *SIAM Journal on Scientific Computing*, 33(2):893–922, 2011.

[22] Jameson, A., 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*, AIAA–81–1259. Palo Alto, California, United States, 1981.

[23] Pulliam, T. H. Efficient solution methods for the Navier-Stokes equations. Technical report, Lecture Notes for the von Kármán Inst. for Fluid Dynamics Lecture Series: Numerical Techniques for Viscous Flow Computation in Turbomachinery Bladings, Rhode-Saint-Genèse, Belgium, January 1986.

[24] Swanson, R. C. and E. Turkel. On central-difference and upwind schemes. *Journal of Computational Physics*, 101(2):292–306, 1992.

[25] Del Rey Fernández, D. C. and D. W. Zingg. High-order compact-stencil summation-by-parts operators for the second derivative with variable coefficients. In *7th International Conference on Computational Fluid Dynamics*, ICCFD7–2803. Big Island, Hawaii, USA, July 2012.

[26] Kreiss, H.-O. and G. Scherer. Finite element and finite difference methods for hyperbolic partial differential equations. In C. de Boor, ed., *Mathematical Aspects of Finite Elements in Partial Differential Equations*. Mathematics Research Center, the University of Wisconsin, Academic Press, 1974.

[27] Strand, B. Summation by parts for finite difference approximations for d/dx. *Journal of Computational Physics*, 110(1):47–67, 1994.

[28] Mattsson, K., M. Svärd, and M. Shoeybi. Stable and accurate schemes for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 227(4):2293–2316, 2008.

[29] Diener, P., E. N. Dorband, E. Schnetter, and M. Tiglio. Optimized high-order derivative and dissipation operators satisfying summation by parts, and application in three-dimensional multi-block evolutions. *Journal of Scientific Computing*, 32(1):109–145, 2007.

[30] Mattsson, K. Summation by parts operators for finite-difference approximations of second derivatives

with variable coefficients. *Journal of Scientific Computing*, 51(3):650–682, 2012.

[31] Mattsson, K. and J. Nordström. Summation by parts operators for finite-difference approximations of second derivatives. *Journal of Computational Physics*, 199(2):503–540, 2004.

[32] Spalart, P. R. and C. L. Rumsey. Effective inflow conditions for turbulence models in aerodynamic calculations. *AIAA Journal*, 45(10):2544–2553, 2007.

[33] Carpenter, M. H., C. A. Kennedy, H. Bijl, S. A. Viken, and V. N. Vatsa. Fourth-order runge-kutta schemes for fluid mechanics applications. *Journal of Scientific Computing*, 25(1/2):157–194, November 2005.

[34] Bijl, H., M. H. Carpenter, V. N. Vasta, and C. A. Kennedy. Implicit time integration schemes for the unsteady compressible Navier-Stokes equations: Laminar flow. *Journal of Computational Physics*, 179:313–329, 2002.

[35] Dahlquist, G. G. A special stability problem for linear multistep methods. *BIT Numerical Mathematics*, 3(1):27–43, March 1963.

[36] Vatsa, V., M. Carpenter, and D. Lockard. Re-evaluation of an optimized second order backward difference (bdf2opt) scheme for unsteady flow applications. In *48th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–2010–0122. Orlando, Florida, United States, January 2010.

[37] Cash, J. R. The integration of stiff initial value problems in odes using modified extended backward differentiation formulae. *Journal of Computers & Mathematics with Applications*, 9(5):645–657, 1983.

[38] Lomax, H., T. H. Pulliam, and D. W. Zingg. *Fundamentals of Computational Fluid Dynamics*. Springer–Verlag, Berlin, Germany, 2001.

[39] Nichols, J. and D. W. Zingg. A three-dimensional multi-block Newton-Krylov flow solver for the Euler equations. In *17th AIAA Computational Fluid Dynamics Conference*, AIAA–2005–5230. Toronto, Canada, June 2005.

[40] Pueyo, A. and D. W. Zingg. Efficient Newton-Krylov solver for aerodynamic computations. *AIAA Journal*, 36(11):1991–1997, November 1998.

[41] Blanco, M. and D. W. Zingg. Fast Newton-Krylov method for unstructured grids. *AIAA Journal*, 36(4):607–612, April 1998.

[42] Kam, D. C. W. *A three-dimensional Newton-Krylov Navier-Stokes flow solver using a one-equation turbulence model*. Master's thesis, University of Toronto, Toronto, Ontario, Canada, 2007.

[43] Kim, D. B. and P. D. Orkwis. Jacobian update strategies for quadratic and near-quadratic convergence of Newton and Newton-like implicit schemes. In *31st AIAA Aerospace Sciences Meeting and Exhibit*, AIAA–93–0878. Reno, Nevada, 1993.

[44] Keyes, D. E. Aerodynamic applications of Newton-Krylov-Schwarz solvers. In *Proceedings of the 14th International Conference on Numerical Methods in Fluid Dynamics*, 1–20. Springer, New York, 1995.

[45] Saad, Y. and M. Sosonkina. Distributed Schur complement techniques for general sparse linear systems. *SIAM Journal of Scientific Computing*, 21(4):1337–1357, 1999.

[46] Mulder, W. A. and B. van Leer. Experiments with implicit upwind methods for the Euler equations. *Journal of Computational Physics*, 59(2):232–246, 1985.

[47] Chisholm, T. T. and D. W. Zingg. A Jacobian-free Newton-Krylov algorithm for compressible turbulent fluid flows. *Journal of Computational Physics*, 228(9):3490–3507, 2009.

[48] Vassberg, J. C., M. A. DeHann, S. M. Rivers, and R. A. Wahls. Development of a Common Research Model for applied CFD validation studies. In *26th AIAA Applied Aerodynamics Conference*, AIAA–2008–6919. Honolulu, Hawaii, United States, August 2008.

[49] Taylor, G. I. and A. E. Green. Mechanism of the production of small eddies from large ones. *Proceedings of the Royal Society of London. Series A*, 158(895):499–521, February 1937.

[50] van Rees, W. M., A. Leonard, D. I. Pullin, and P. Koumoutsakos. A comparison of vortex and pseudo-spectral methods for the simulation of periodic vortical flows at high reynolds numbers. *Journal of Computational Physics*, 230(8):2794–2805, April 2011.