2013-2870

21st AIAA Computational Fluid Dynamics Conference, June 25, 2013, San Diego, CA

Design of a Variational Multiscale Method for Turbulent Compressible Flows

Laslo T. Diosady* and Scott M. Murman†

NASA Ames Research Center, Moffett Field, CA, USA

A spectral-element framework is presented for the simulation of subsonic compressible high-Reynolds-number flows. The focus of the work is maximizing the efficiency of the computational schemes to enable unsteady simulations with a large number of spatial and temporal degrees of freedom. A collocation scheme is combined with optimized computational kernels to provide a residual evaluation with computational cost independent of order of accuracy up to 16th order. The optimized residual routines are used to develop a low-memory implicit scheme based on a matrix-free Newton-Krylov method. A preconditioner based on the finite-difference diagonalized ADI scheme is developed which maintains the low memory of the matrix-free implicit solver, while providing improved convergence properties. Emphasis on low memory usage throughout the solver development is leveraged to implement a coupled space-time DG solver which may offer further efficiency gains through adaptivity in both space and time.

I. Introduction

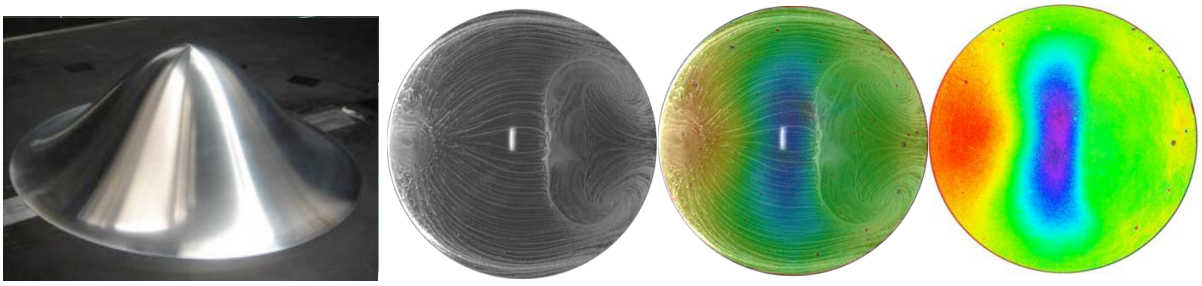
The Variational Multiscale Method (VMM) is a reformulation of the classical Large-eddy Simulation (LES), where the filtering operation, used to explicitly separate resolved and unresolved scales, is replaced by a finite-element projection operator.[1] As projection and differentiation commute, VMM does not suffer from commutation errors present in classical LES. VMM subgrid-scale models can be limited to the finest computed modes, so that no modeling error is introduced into the resolved coarser modes.[2] Combined with adaptivity, this provides a robust *h-p* spectral-element framework for viscous simulations which has shown promise for predicting attached wall-bounded flows.[1,3–8]

The ultimate objective of this work is to apply the VMM to compressible, high-Reynolds number separated flows where Direct Numerical Simulation (DNS) is impractical, and engineering Reynolds-Averaged Navier-Stokes (RANS) turbulence models struggle to accurately capture the important flow features. Specifically, our initial target is unsteady simulation of the Fundamental Aeronautics Investigation of The Hill (FAITH) experiment,[9] shown in Fig. 1. The FAITH experiment includes a number of quantitative and qualitative measurement techniques to characterize the separated flow over an axisymmetric hill at $Re_h = 500$k, where h is the height of the hill. VMM simulations will complement this experimental dataset with detailed statistical measurements of the turbulence correlations and budget terms necessary to develop and validate novel RANS models capable of predicting separated flows.

Simulation of the FAITH experiment with a wall-resolved VMM poses a significant computational challenge due to the high Reynolds number. To estimate the total cost, we consider a computational domain including the majority of the wind-tunnel test section, with the wall-normal direction resolved to within one friction length scale at the wall, i.e. $\Delta y^+|_w = 1$. Using the VMM, we estimate a requirement for the average streamwise and spanwise resolution of $\Delta x^+ = \Delta z^+ \approx 20$ respectively. To accurately propagate the resolved turbulent fluctuations an appropriate unsteady time scale is required, which corresponds roughly to a CFL=1 based on the r.m.s. mean velocity. With this temporal resolution, we bookkeep a simulation time of 10 flow-through periods, which is a minimum to capture statistical measurements. From this, we estimate a spatial computational mesh requiring on the order of 10^9 degrees of freedom (DOF), and a simulation time requiring 10^9 timesteps, for a total of 10^{18} space-time DOF.

*Postdoctoral Fellow, Oak Ridge Associated Universities, laslo.diosady@nasa.gov

†Scott.M.Murman@nasa.gov



(a) FAITH Model (b) Oil Flow and Pressure-sensitive Paint Visualization

Figure 1. Fundamental Aeronautics Investigation of The Hill (FAITH) experiment.[9]

Given these computational cost estimates, the efficiency of a VMM solver is critical to enabling FAITH simulations. After surveying the open-source community, NASA, and NASA's collaborative partnerships, none of the available options for an existing spectral-element framework were deemed sufficient, or could be easily modified, to provide the necessary efficiency and accuracy. From this, a decision was made to develop a VMM spectral-element solver specifically for low-speed, high-Reynolds number flows over geometrically simple configurations. The goal of this paper is to describe the design of this framework in terms of algorithm choice, data layout in memory, and tightly-coupled optimizations for the target computer hardware. Specifically, we seek to minimize three weaknesses of high-order finite-element formulations - the high mathematical operation count per element, the restrictive stability constraint for explicit methods, and the large memory required for efficient implicit methods. The current work focuses solely on these numerical challenges, with closure modeling and accurate reproduction of the relevant physics deferred to future discussions.

The paper begins with an introduction to the nomenclature and basic spectral-element formulation. From this, algorithm and computational optimizations are discussed, along with representative computational timings. A matrix-free implicit method is described, along with a novel method for incorporating a diagonalized Alternating-Direction-Implicit (ADI) preconditioner. Lastly, a space-time formulation is presented which effectively leverages spectral elements in space and time to reduce the computational cost. Two model problems, convection of an isentropic vortex and the Taylor-Green vortex evolution, are used throughout to provide computational examples. The paper concludes with a summary of the major points and some topics for future work.

II. Spatial Discretization

The compressible Navier-Stokes equations are written in conservative form as:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{F}^I - \boldsymbol{F}^V) = 0 \tag{1}$$

The conservative state vector is $\boldsymbol{u} = [\rho, \rho\mathbf{v}, \rho E]$, where ρ is the density, $\mathbf{v}$ the velocity, and E the total energy. The inviscid and viscous fluxes are given, respectively, by:

$$\boldsymbol{F}^I \cdot \boldsymbol{n} = \begin{bmatrix} \rho v_n \\ \rho \mathbf{v} v_n + p\boldsymbol{n} \\ \rho H v_n \end{bmatrix}, \qquad v_n = \mathbf{v} \cdot \boldsymbol{n}, \qquad \text{and} \qquad \boldsymbol{F}^V = \begin{bmatrix} 0 \\ \boldsymbol{\tau} \\ \mathbf{v} \cdot \boldsymbol{\tau} + \kappa_T \nabla T \end{bmatrix}, \tag{2}$$

where p is the static pressure, $H = E + \frac{p}{\rho}$ is the total enthalpy, $\boldsymbol{\tau}$ is the shear stress tensor, κ_T is the thermal conductivity, $T = p/\rho R$ is the temperature, and R is the gas constant. The pressure is given by:

$$p = (\gamma - 1)\left(\rho E - \tfrac{1}{2}\rho \mathbf{v} \cdot \mathbf{v}\right), \tag{3}$$

where γ is the specific heat ratio. The viscous stress tensor, $\boldsymbol{\tau}$, is given by:

$$\boldsymbol{\tau} = \mu\left(\nabla \mathbf{v} + \nabla \mathbf{v}^T\right) - \lambda(\nabla \cdot \mathbf{v})I \tag{4}$$

where μ is the viscosity, $\lambda = \frac{2}{3}\mu$ is the bulk viscosity and I is the identity matrix.

In this work, we use a Galerkin finite-element method, which extends to arbitrary order of accuracy. Higher-order methods show potential for simulations requiring high spatial and temporal resolution, allowing for solutions with fewer degrees of freedom and lower computational cost than traditional second order CFD methods.[10] As this work is focused on subsonic flows without shocks, the exact solution is in C^∞, and thus we do not expect the convergence rate of our higher order scheme to be limited by solution irregularity. Higher-order finite element methods are particularly attractive due to the possibility of using local h- and p-adaptation. Galerkin finite-element methods have an increased number of operations, relative to high-order finite-difference methods, due to the increased coupling between degrees of freedom. High-order finite-difference schemes couple degrees of freedom along lines in each coordinate direction, so that the cost of a residual evaluation scales as $N \times d$ for a fixed number of degrees of freedom, where $N = p + 1$ is the order of accuracy and d the spatial dimension. A general Galerkin method couples all degrees of freedom on an element, leading to a cost that scales as N^d. This scaling suggests that in three dimensions, a general Galerkin formulation is penalized in terms of operation count relative to finite-difference methods beyond third order. In order to attain similar efficiency within a Galerkin framework, we use tensor product bases and a sum factorization approach, such that differentiation and integration reduce to a sequence of one-dimensional operations. This results in a cost which also scales as $N \times d$.

The domain, Ω, is partitioned into non-overlapping hexahedral elements κ. We define a finite-element space $\mathcal{V}_h$ consisting of piece-wise polynomial functions on each element κ:

$$\mathcal{V}_h = \{\boldsymbol{w}, \boldsymbol{w}|_\kappa \in [\mathcal{P}(\kappa)]^m\} \tag{5}$$

where $m = 5$ is the number of flow equations. Choosing $\mathcal{V}_h$ to be C^0-continuous across elements leads to a continuous Galerkin (CG) discretization, while allowing $\mathcal{V}_h$ to be discontinuous across elements leads to a discontinuous Galerkin (DG) discretization. We have developed a generic finite-element framework using both CG and DG formulations, however, the results presented in this work focus on DG to clarify the discussion.

We seek a solution $\boldsymbol{u} \in \mathcal{V}_h$ which satisfies the weak form of Eq. (1):

$$\sum_\kappa \left\{ \int_\kappa \left(\boldsymbol{w} \frac{\partial \boldsymbol{u}}{\partial t} - \nabla \boldsymbol{w} \cdot (\boldsymbol{F}^I - \boldsymbol{F}^V) \right) + \int_{\partial\kappa} \boldsymbol{w} (\hat{\boldsymbol{F}}^I - \hat{\boldsymbol{F}}^V) \cdot \boldsymbol{n} \right\} = 0 \qquad \forall \boldsymbol{w} \in \mathcal{V}_h. \tag{6}$$

Here $\hat{\boldsymbol{F}}^I$ and $\hat{\boldsymbol{F}}^V$ denote numerical fluxes which are functions of the state and its gradient on both sides of a face shared by two elements, or state and boundary data for a face on the domain boundary. The inviscid flux is computed using the Roe flux,[11] while the viscous flux is computed using the method of Bassi and Rebay.[12] In a CG method, the surface integrals corresponding to faces on the interior of the domain cancel to zero, leaving only surface integrals on the boundary of the domain.

Using a tensor product bases on each element, the solution $\boldsymbol{u}$ is given by a product of Lagrange polynomials:

$$\boldsymbol{u}(\mathbf{x}(\xi)) = \sum_{i,j,k} \boldsymbol{U}_{ijk} \Phi_{ijk} \qquad \Phi_{ijk} = \phi_i(\xi_1)\phi_j(\xi_2)\phi_k(\xi_3) \tag{7}$$

where $\mathbf{x}(\xi)$ defines a mapping from the reference cube, $\xi \subset [-1, 1]^3$, to physical space. ϕ_i is a one-dimensional Lagrange basis defined at Gauss-Legendre (GL) or Gauss-Legendre-Lobatto (GLL) points, while $\boldsymbol{U}_{ijk}$ is the corresponding nodal value of the solution. The integrals in Eq. (6) are evaluated using numerical quadrature. For volume integrals:

$$\int_\kappa \left(\boldsymbol{w} \frac{\partial \boldsymbol{u}}{\partial t} - \nabla \boldsymbol{w} \cdot (\boldsymbol{F}^I - \boldsymbol{F}^V) \right) \simeq \sum_{p,q,r} \left\{ \left(\boldsymbol{w} \frac{\partial \boldsymbol{u}}{\partial t} - \nabla_\xi \boldsymbol{w} \cdot (\tilde{\boldsymbol{F}}^I - \tilde{\boldsymbol{F}}^V) \right) |J| \right\}_{\xi_p \xi_q \xi_r} w_p w_q w_r \tag{8}$$

where ξ_p, ξ_q, ξ_r are one-dimensional GL or GLL quadrature points, and w_p, w_q and w_r are the associated quadrature weights. J denotes the Jacobian of the mapping from element reference space to physical space, ∇_ξ denotes the gradient with respect to the local coordinate ξ, while $\tilde{\boldsymbol{F}}^I = J^{-1}\boldsymbol{F}^I$ and $\tilde{\boldsymbol{F}}^V = J^{-1}\boldsymbol{F}^V$ are the fluxes mapped to the local element coordinate system. Similarly, surface integrals in Eq. (6) are evaluated as:

$$\int_{\partial\kappa} \boldsymbol{w} (\boldsymbol{F}^I - \boldsymbol{F}^V) \cdot \boldsymbol{n} \simeq \sum_{p,q} \left\{ \boldsymbol{w} (\boldsymbol{F}^I - \boldsymbol{F}^V) \cdot \boldsymbol{n} \right\}_{\xi_p \xi_q} w_p w_q \tag{9}$$

Both volume and surface integrals (Eqs. (8) and (9)) are performed as a sequence of three steps described below:

1. Evaluation of the state and gradient at the quadrature points.
2. Evaluation of the fluxes at the quadrature points.
3. Multiplication of the fluxes with the gradient of the basis functions.

The sum-factorization approach[13] is used in steps 1 and 3. The state at a particular quadrature point is given by:

$$\hat{\boldsymbol{U}}_{pqr} \equiv \boldsymbol{u}(\xi_p, \xi_q, \xi_r) = \sum_i \sum_j \sum_k \boldsymbol{U}_{ijk} \phi_i|_{\xi_p} \phi_j|_{\xi_q} \phi_k|_{\xi_q} \tag{10}$$

The state evaluated at all quadrature points within an element is computed as:

$$\hat{\boldsymbol{U}} = (B_3 \otimes B_2 \otimes B_1)\boldsymbol{U} = (B_3 \otimes I \otimes I)\{(I \otimes B_2 \otimes I)\{(I \otimes I \otimes B_1)\boldsymbol{U}\}\} \tag{11}$$

where $\hat{\boldsymbol{U}}$ is vector of states at all quadrature points in an element, $(B_3 \otimes B_2 \otimes B_1)$ is the matrix formed as the tensor product of $N_q \times N$ matrices B_1, B_2 and B_3, where N_q is the number of 1-dimensional quadrature points, while N is the number of 1-dimensional basis functions. The entries of B_1, B_2 and B_3 are:

$$[B_1]_{pi} = \phi_i|_{\xi_p} \qquad [B_2]_{qj} = \phi_j|_{\xi_q} \qquad [B_3]_{rk} = \phi_k|_{\xi_r} \tag{12}$$

In Eq. (11), we show that the tensor product structure the multiplication of $(B_3 \otimes B_2 \otimes B_1)$, is performed as a sequence of small matrix multiplications corresponding to each coordinate direction. The derivative of the state with respect to ξ_1 is computed at the quadrature points as:

$$\frac{\partial \hat{\boldsymbol{U}}}{\partial \xi_1} = (B_3 \otimes B_2 \otimes D_1)\boldsymbol{U} = (B_3 \otimes I \otimes I)\{(I \otimes B_2 \otimes I)\{(I \otimes I \otimes D_1)\boldsymbol{U}\}\} \tag{13}$$

where the entries of the matrix D_1 are given by:

$$[D_1]_{pi} = \frac{\partial \phi_i}{\partial \xi}|_{\xi_p} \tag{14}$$

Derivatives in the ξ_2 and ξ_3 directions are computed in a similar manner.

The use of the sum factorization approach leads to an operation count which scales as $N \times d$ for fixed number of degrees of freedom. However, in order to further reduce the cost of our finite-element discretization we use a collocation approach to compute the integrals involved in the residual. When using a collocation approach, the solution points are used as quadrature points reducing the cost of the state and residual evaluations since operations involving the matrices B_1, B_2 and B_3 are eliminated (B_1, B_2 and B_3 reduce to the identity matrix, and hence multiplication by these terms is not necessary). The state is stored at the collocation points and thus is directly available, while the derivatives are computed using terms of the form:

$$\frac{\partial \hat{\boldsymbol{U}}}{\partial \xi_1} = (I \otimes I \otimes D_1)\boldsymbol{U} \tag{15}$$

The third step of the residual evaluation, requiring the weighting of the flux with the gradient of the basis functions, is computed in a similar manner. Namely, using collocation, we compute volume terms of the form:

$$\boldsymbol{R}_\kappa = -(I \otimes I \otimes D_1)^T \hat{\boldsymbol{F}}_1 - (I \otimes D_2 \otimes I)^T \hat{\boldsymbol{F}}_2 - (D_3 \otimes I \otimes I)^T \hat{\boldsymbol{F}}_3 \tag{16}$$

where $\hat{\boldsymbol{F}}_1$ is the vector of fluxes scaled by the Jacobian determinant and quadrature weights at each point:

$$\left[\hat{\boldsymbol{F}}_1\right]_{pqr} = \left(\tilde{\boldsymbol{F}}_1 |J|\right)_{\xi_p \xi_q \xi_r} w_p w_q w_r \tag{17}$$

while $\hat{\boldsymbol{F}}_2$ and $\hat{\boldsymbol{F}}_3$ have similar forms.

With the use of collocation and tensor products, the amount of computational work is reduced to a minimum, however we still have a high operation count per degree of freedom. We minimize this cost through code optimization. Recently, increases in computational performance have been largely achieved through increased parallelization, both through the use of systems with increasing number of processors, as well as on-processor parallelization through threading and vectorization. High-order finite-element methods are well suited to take advantage of parallel computing. The finite-element method provides a straight-forward domain-decomposition requiring only data on element interfaces to couple elements (i.e. no neighboring ghost cells). Thus, we take advantage of multi-processor parallelism with MPI-based message passing, where the small amount of communication overhead is masked by overlapping communication and computation. Additionally, operations local to an element are designed to align data for vectorization.

The finite-element framework developed here is designed for use with the computing resources at NASA Ames Research Center. In particular, we look to use the Intel Sandy Bridge and Intel Xeon Phi micro-processor architectures, as well as their future descendants, which achieve high-performance through single-instruction/multiple-data (SIMD) vector operations. The Sandy Bridge architecture uses the Advanced Vector Extensions (AVX) instruction set with a SIMD register of 256 bits (4 doubles), while the Xeon Phi has a SIMD register of 512 bits (8 doubles) allowing 4 or 8 double operations to be performed in parallel. In order to take full advantage of the available computing resources, it is essential to use the vectorizing capabilities of these micro-architectures by careful alignment of data and unit-stride memory operations.

In order to achieve high computational efficiency, we have optimized the numerical kernels involved in the computation of the residual and reuse these kernels throughout our finite-element software. In particular, we use a set of optimized matrix-matrix multiplication routines for computing the sum-factorization terms in Eqs. (11), (14) and (16), while the evaluation of fluxes at collocation points is performed using data-aligned, unit-stride vector functions.

The use of the sum-factorization approach and optimized routines allow for residual evaluations with a computational cost that is nearly independent of N for $N = [2, 16]$. The CPU time for a residual evaluation, normalized by the number of degrees of freedom for our DG discretization is shown in Fig. 2. The CPU time is obtained by performing a simulation of the Taylor-Green vortex problem described in Section III.B on a Sandy Bridge node of the Pleiades supercomputer at NASA Ames Research Center. Each Sandy Bridge node consists of 2 eight-core Intel Xeon E5-2670 processors with a clock speed of 2.6Ghz and 2GB per core memory. The simulations were performed with approximately 30000 degrees of freedom per core. Through optimized linear algebra kernels, the increase in operation count with N is offset by more effective use of computing resources. For $N = [4, 16]$, the CPU time for a single residual evaluation is approximately $0.75\mu s/DOF$ which is comparable to the cost for a residual evaluation using the OVERFLOW finite-difference code on the same architecture. For reference, the TAU benchmark for this architecture runs in 7.6s.[10]

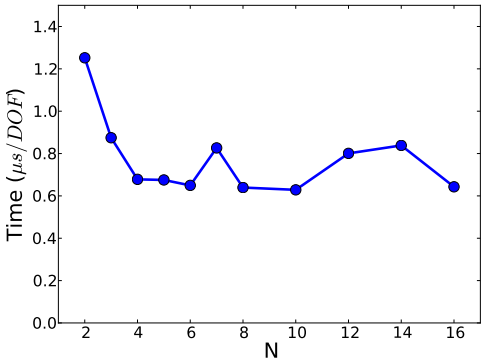


Figure 2. CPU time for a single residual evaluation per degree of freedom on an Intel Sandy Bridge node

III. Model Problems

Two model problems are used throughout to examine the behavior of the spectral-element framework. These are the inviscid convection of a two-dimensional isentropic vortex, and viscous simulation of the

Taylor-Green vortex evolution. These problems are representative of the physics without containing solid boundaries, which would necessarily require sub-grid modeling for any appreciable Reynolds number.

III.A. Convection of an Isentropic Vortex

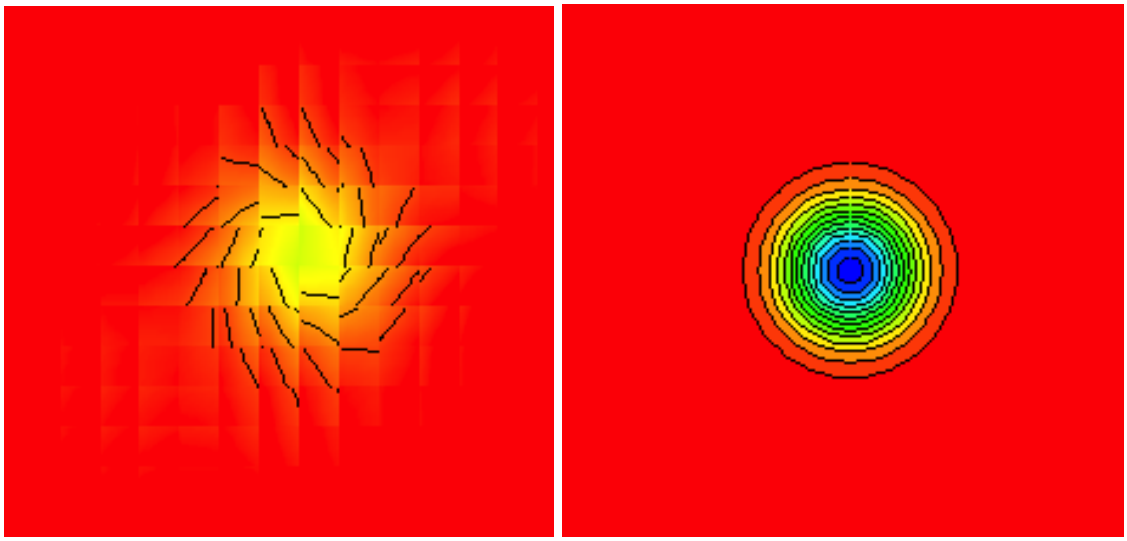
The inviscid isentropic vortex convection problem is initialized with a perturbation about a uniform flow given by:

$$\delta u = -U_\infty \beta \frac{y - y_c}{R} e^{-\frac{r^2}{2}} \tag{18}$$

$$\delta v = U_\infty \beta \frac{x - x_c}{R} e^{-\frac{r^2}{2}} \tag{19}$$

$$\delta \left(\frac{P}{\rho} \right) = \frac{1}{2} \frac{\gamma}{\gamma - 1} U_\infty^2 \beta^2 e^{-r^2} \tag{20}$$

where U_∞ is the convecting speed of the vortex, $\beta = 1/5$ is the vortex strength, $R = 0.05$ the characteristic radius, and (x_c, y_c) the vortex center. The convection velocity is angled 30° from the x-direction so that the flow does not align with the mesh. The exact solution to this flow is a pure convection of the vortex at speed U_∞. Figure 3 shows contours of the density after 5 periods for the simulation of the isentropic vortex at a Mach number of 0.5 using $N = 2$ and $N = 16$ using 64 degrees of freedom in x- and y- directions. Figure 3 demonstrates the benefits of higher-order methods as the 16th-order method is able to propagate the vortex through 5 periods nearly exactly. The quantitative estimate of formal accuracy will be described in Sec. V.A.



(a) $N = 2$ (b) $N = 16$

Figure 3. Density contours for isentropic vortex convection problem after 5 periods using $N = 2$ and $N = 16$ with 64 DOF in each coordinate direction.

III.B. Taylor-Green Vortex

The Taylor-Green vortex flow is simulated using the compressible Navier-Stokes equations at $M_0 = 0.1$. The flow is solved on an isotropic domain which spans $[0, 2\pi L]$ in each coordinate direction. The initial conditions are given by:

$$u = V_0 \sin(x/L) \cos(y/L) \cos(z/L) \tag{21}$$

$$v = -V_0 \cos(x/L) \sin(y/L) \cos(z/L) \tag{22}$$

$$w = 0 \tag{23}$$

$$p = \rho_0 V_0^2 \left[\frac{1}{\gamma M_0^2} + \frac{1}{16} \left(\cos(2x) + \sin(2y) \right) \left(\cos(2z) + 2 \right) \right] \tag{24}$$

where u, v and w are the components of the velocity in the x, y and z-directions, p is the pressure and ρ is the density. The flow is initialized to be isothermal ($\frac{p}{\rho} = \frac{p_0}{\rho_0} = RT_0$). Figure 4 shows the temporal evolution of the kinetic energy dissipation rate $-\frac{dE_k}{dt}$ for the Taylor-Green vortex problem computed at $M_0 = 0.1$ and $Re = \frac{\rho_0 V_0 L}{\mu} = 1600$ using 256 degrees of freedom in each coordinate direction. Figure 4 also shows results using an incompressible spectral code with 512 degrees of freedom in each coordinate direction.[14] Figure 5 show the corresponding iso-contours of vorticity at the point of peak dissipation using 2nd- and 16th-order schemes. The higher-order scheme is less dissipative than the 2nd-order scheme and is able to more accurately match the spectral data.

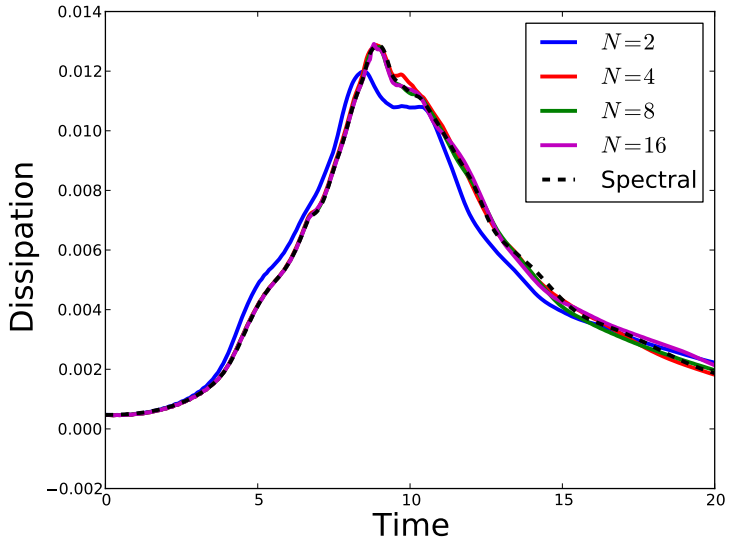
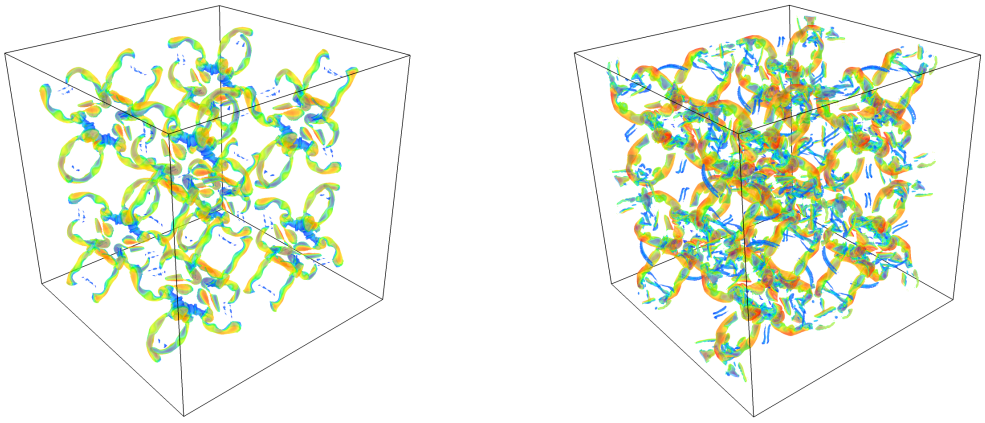


Figure 4. Taylor-Green vortex problem at $M = 0.1$, $Re = 1600$ computed using 256^3 degrees of freedom



(a) $N = 2$ (b) $N = 16$

Figure 5. Iso-contours of vorticity at peak dissipation for the Taylor-Green vortex evolution at $M = 0.1$, $Re = 1600$ computed using 256^3 degrees of freedom.

IV. Polynomial De-aliasing

A VMM or LES method resolves only large scale motion in a flow to reduce the computational cost relative to DNS, with the unresolved finest scales modeled. While the coarsest scales are accurately resolved, the finest representable scales in the flow are under-resolved. These under-resolved flows cause numerical

stability problems, especially for high-order numerical methods, and require some form of stabilization. There are many techniques for stabilization within a CG or DG spectral-element framework.[15–18] Here we examine polynomial de-aliasing within an element[18] (often referred to as "over-integration") within a DG framework to demonstrate the issues involved and performance. Section VI discusses some more general issues for combining stabilization and physical modeling in a VMM for future work.

The use of numerical quadrature implies inexact projection of the nonlinear products in the flux terms onto a finite-element space. This inexact projection causes aliasing errors and may ultimately lead to instability. Similar to de-aliasing techniques in spectral methods, projecting the state variables to a higher polynomial space to evaluate the nonlinear products, then projecting the result back to the original polynomial order, can remove these aliasing errors.

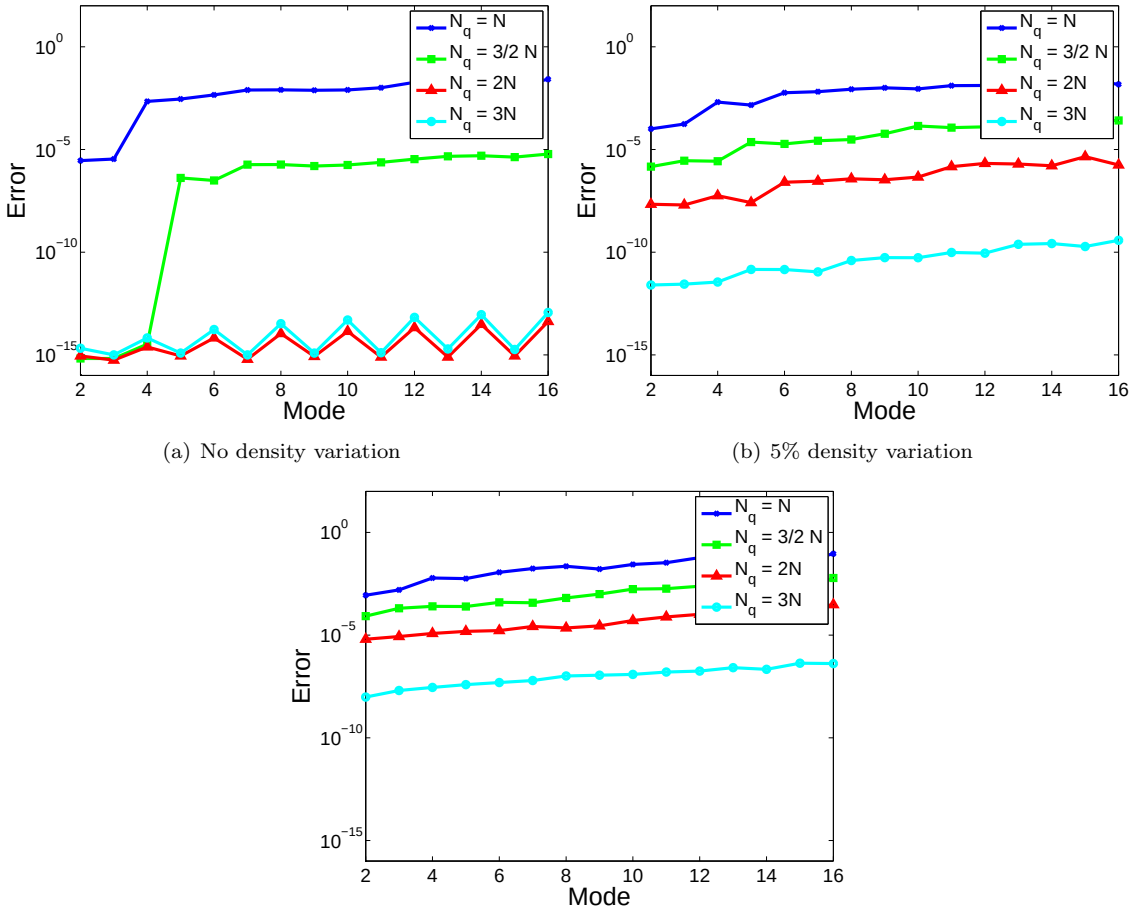
For incompressible Navier-Stokes simulations, the nonlinear convection terms are quadratic in the state, implying that for a solution with $N = p+1$, exact projection of the nonlinear terms may be performed using a Gaussian quadrature rule with $3/2N$ points.[19] In the case of the compressible Navier-Stokes equations, the nonlinearity in the inviscid flux is a rational function of the conservative state and Gaussian quadrature is unable to exactly compute the projection of the Navier-Stokes equations onto the polynomial space.[a] It has been suggested that since the inviscid flux in the compressible Navier-Stokes equations is a cubic function of the primitive variables, using a quadrature rule with $2N$ points is sufficient,[18,20] while others recommend a $3/2N$ quadrature.[21] Clearly these polynomial de-aliasing methods are somewhat *ad-hoc* for compressible flows.

In order to understand the behavior of polynomial de-aliasing for subsonic compressible flows, we consider the errors in integration introduced through the use of quadrature. As a model problem, we consider a one-dimensional periodic domain using a single element with $N = 16$. We evaluate the spatial residual using increasing quadrature rules. The residual is computed about a flow with mean Mach number of 0.2 with 20% fluctuations in velocity and pressure. We consider flows with density fluctuations of 0, 5% and 20% about the mean. Figure 6 plots the maximum error in the residual for the energy equation for each mode using a Legendre polynomial basis, averaged over 100 random initial states. For flows with zero density fluctuation, using the $3/2N$ quadrature points allows for exact integration of the lowest quartile of modes, while Gaussian quadrature rules using $2N$ or more points are exact for all modes. When density perturbations exist in the flow, then exact projection of the flux onto the polynomial space is not possible and an error is introduced for all modes. However, for density fluctuations on the order of 5% the use of $2N$ points is able to reduce this error by nearly 4 orders of magnitude. As the magnitude of the density fluctuations increases, additional quadrature is required to accurately integrate the rational functions. This accuracy (or lack thereof) appears to correlate with numerical stability, even in the presence of upwinding in the DG method. In particular, we have found that with increasing Mach and Reynolds number, polynomial de-aliasing with $2N$ or $3N$ points is not always sufficient.

We demonstrate the aliasing issue by performing a simulation of the Taylor-Green vortex evolution at a $M = 0.1$ and $Re = 1600$ using 64 DOF in each coordinate direction. In this computation it is known that 64 DOF is under-resolved, however increasing resolution by only two or four times with a suitable high-order scheme leads to relatively good predictions, so this is representative of an under-resolved LES simulation. In Figure 7 we show the evolution of the kinetic energy dissipation rate using both collocation, and polynomial de-aliasing with $2N$ points. Using collocation, for 2nd- and 4th- order the DG method provides sufficient numerical dissipation to maintain stability, however the higher-order simulations become unstable. Using polynomial de-aliasing with $2N$ quadrature points all of the simulations now remain stable, consistent with the experience in the literature.[20]

Increasing the Reynolds number to $Re = 16$k further reduces the available resolution, and numerical stability becomes difficult to maintain with either scheme. Figure 7 presents the kinetic energy dissipation rate using both collocation and polynomial de-aliasing with $2N$ points for this higher Reynolds number. While polynomial de-aliasing allows the simulation to progress further in time than using collocation, the 8th- and 16th-order simulations eventually become unstable. Increasing the de-aliasing to $3N$ points is still insufficient to maintain stability for this test case. Given these results for a relatively benign problem, stabilization is a concern for our more complex target flow which has higher speed, higher Reynolds number, and larger perturbations. This is an area of planned future investigation.

[a] Maintaining standard conservative form for the compressible equations, it is possible to formulate the Navier-Stokes flux terms purely as products of the primitive variables. In this situation multiplication by the inverse of the mass matrix leads to rational functions.

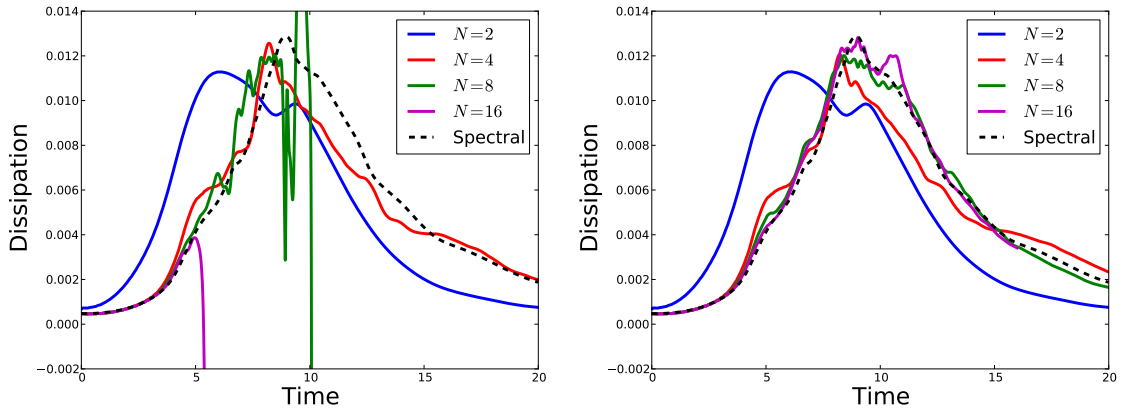


(a) No density variation

(b) 5% density variation

(c) 20% density variation

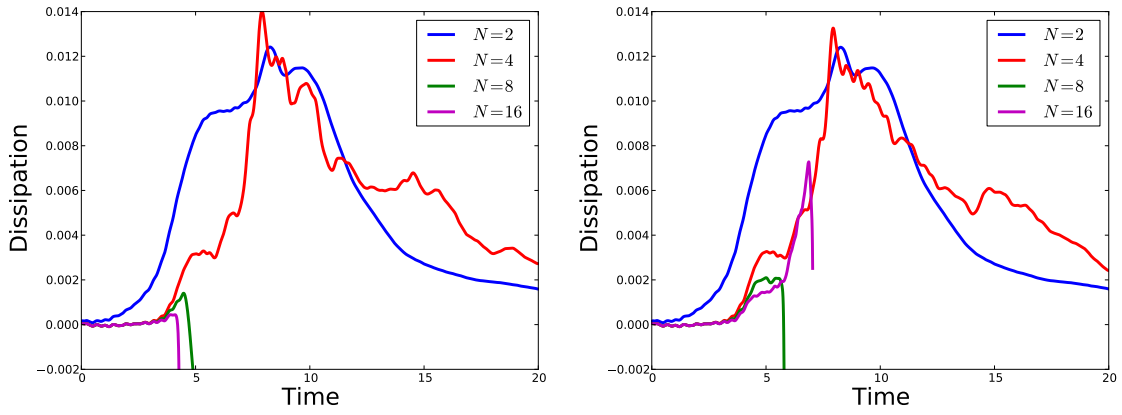
Figure 6. Errors in evaluation of the residual using numerical quadrature as a function of N with 20% variation in velocity and pressure and density variation of a) 0, b) 5% c) 20%



(a) Collocation

(b) Polynomial de-aliasing $2N$

Figure 7. Taylor-Green vortex problem at $M = 0.1$, $Re = 1600$ computed using meshes with 64^3 degrees of freedom using a) collocation, and b) polynomial de-aliasing with $2N$ points



(a) Collocation

(b) Polynomial de-aliasing $2N$

Figure 8. Taylor-Green vortex problem at $M = 0.2$, $Re = 16000$ computed using meshes with 64^3 degrees of freedom using a) collocation, and b) polynomial de-aliasing with $2N$ points

The use of polynomial de-aliasing with $2N$ quadrature points in each coordinate direction increases the computational cost of a residual evaluation by roughly 10 times relative to collocation. For explicit time-stepping schemes the simulation cost is driven entirely by the cost of residual evaluations and hence the use of polynomial de-aliasing implies an order of magnitude increase in total computational cost. However, as we'll see in the next section, the use of explicit schemes with our spectral-element framework is impractical for these high-Reynolds-number wall-bounded flows. For implicit time-stepping schemes, the cost of the simulation is primarily driven by the cost of the linear solver for a Newton method. As such, the increased cost of evaluating the target residual using a $2N$ quadrature rule is amortized over the linear solve, where approximate methods may be employed. By switching to collocation in the linear solve we can effectively provide both stability and efficiency while sacrificing very little in terms of convergence rate.

V. Temporal Discretization

V.A. Explicit

To introduce the notation and demonstrate the high-order performance, we consider an explicit time-stepping scheme. This scheme also forms the basis of the preconditioner for the implicit scheme discussed in the following section. Using the method of lines, the semi-discrete form of Eq. (6), is written as:

$$\mathbb{M}\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{R}(\boldsymbol{U}) \quad = \quad 0 \tag{25}$$

where $\boldsymbol{U}$ is the vector of spatial degrees of freedom, $\boldsymbol{R}(\boldsymbol{U})$ is the spatial residual and $\mathbb{M}$ is the mass matrix:

$$\mathbb{M}_{ij} \quad = \quad \sum_{\kappa} \int_{\kappa} \Phi_i \Phi_j. \tag{26}$$

Equation (25) is a coupled system of ODEs which is solved numerically using an explicit time-stepping scheme. Each stage of an explicit scheme requires inversion of the mass matrix. In general, a CG discretization has a mass-matrix which is globally coupled, while a DG formulation will have an element-wise block-diagonal mass matrix. In either case, the cost of the storing, factoring and inverting the mass matrix is prohibitive, especially at high-order. However, using our collocation approach, the mass-matrix reduces to an easily invertible diagonal matrix for either CG or DG formulations.

The use of GLL collocation to compute the mass matrix is inexact even in the case of elements with constant Jacobians, as the GLL quadrature with N points is exact only for polynomials up to order $2N-3$. In a CG formulation a GLL basis ensures that degrees of freedom are associated with edges, faces and nodes, while use of a GL basis necessarily couples all degrees of freedom in the entire domain to ensure continuity.

On the other hand in a DG formulation, where the degrees of freedom are associated with elements, the use of a GL collocated basis provides increased accuracy with minimal additional cost.

In order to demonstrate the numerical efficiency of different formulations we compute the isentropic convection of a two-dimensional vortex described in Section III.A at a Mach number of 0.5. Figure 9 shows the L_2 error after 10 flow through periods. We use the DG collocation method with a classical 4th-order explicit Runge-Kutta time-stepping scheme. For this problem the temporal error is orders of magnitude lower than the spatial error, and the error is determined by the spatial error alone. As shown in Fig. 9, the use of GL quadrature is more accurate than GLL quadrature for a fixed number of degrees of freedom. The mass lumping with GLL quadrature reduces the order of accuracy of the scheme, consistent with the observation in [22], where the error is shown to act as a filter. As such, the effect of mass lumping is more significant at low order. We note that the reduced accuracy of using the GLL formulation is due solely to the mass-lumping and not to the reduced quadrature in the integration of the spatial part of the residual. This was verified by using more accurate quadrature for the spatial residual term, which gave results identical to those using collocation.

As we are primarily concerned with efficiency, it is necessary to consider any additional cost of using GL quadrature versus GLL quadrature. For a single residual evaluation, the cost of GL quadrature is slightly more than GLL quadrature, due to evaluation of state or basis functions on element faces which requires interpolation using GL points, while are directly available using GLL points. However, for a viscous simulation interpolation of gradients is required (for either GLL or GL quadrature), and the additional cost of interpolating the state at the same time is minimal (on the order of 1-2 % of a residual evaluation). Thus, in terms of a single residual evaluation we view the use of the GL quadrature in a DG discretization as removing the unnecessary filtering of the highest modes in a colocation scheme, at essentially no additional cost.

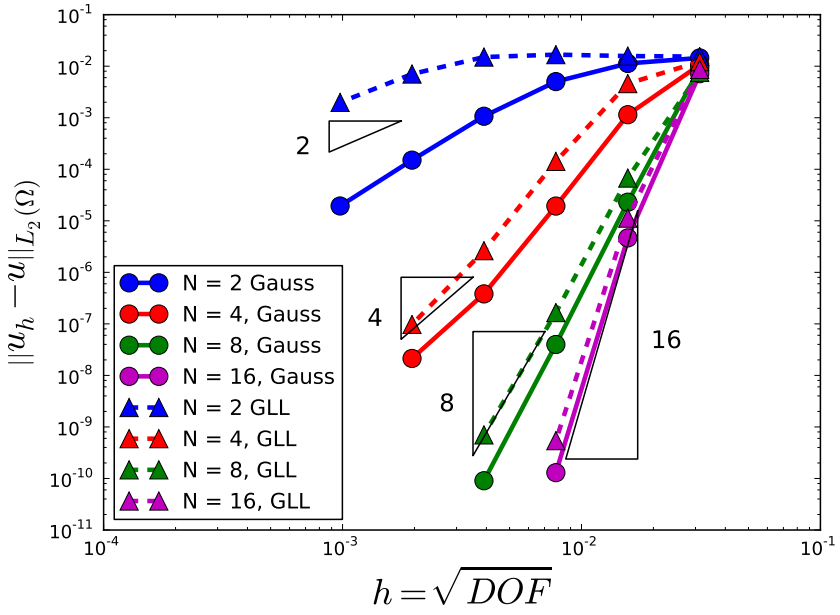


Figure 9. Isentropic vortex convection problem, Error vs. h using GL and GLL collocation

Explicit methods have a maximum time-step which is limited by the CFL condition. For high-Reynolds-number wall-bounded flows the numerical stiffness is greatly increased by the resolution required to accurately compute gradients near the wall. While we are interested in computing unsteady flows, and thus our time step is also limited by the desired temporal resolution of the physics of interest, the restriction due to stability may be much more severe, particularly with increasing order of accuracy.

In order to demonstrate the stiffness associated with higher-order methods, we evaluate the maximum stable CFL number using the classical 4th-order explicit Runge-Kutta scheme for two model problems. While we are interested in solving viscous flows, the inviscid problem is representative of regions in the flow field far from the wall where viscous effect do not play a significant role. The viscous simulation, performed using a mesh size with cell Reynolds number, $Re_\Delta \equiv \frac{ch}{\nu} = O(1)$, is representative of near wall regions of the flow field where fine mesh resolution is required to resolve steep gradients in the flow field.

Figure 10 presents the maximum stable CFL number versus solution order N for the sample test problems. We compute the CFL number based on the acoustic speed $CFL = \frac{c\Delta t}{h}$, where Δt is the time step, $h = \sqrt{DOF}$ is the resolution length scale and c the free-stream speed of sound. For the inviscid isentropic convection problem, the maximum allowable time step scales as $1/N$ for a fixed number of degrees of freedom. When the cell Reynolds number is $O(1)$, the maximum stable time step scales as $1/N^2$ for a fixed number of degrees of freedom. The scaling of the maximum allowable time step implies an increase in the number of time steps (and hence cost) which scales linearly for inviscid problems and quadratically for viscous problems as we increase the spatial resolution via increasing p. From this we see that our total computational cost estimate outlined in the introduction increases by N^2 with an explicit method, as our time-step is limited by our most restrictive stability limit, making the simulations prohibitive for higher-order methods and negating their advantage.

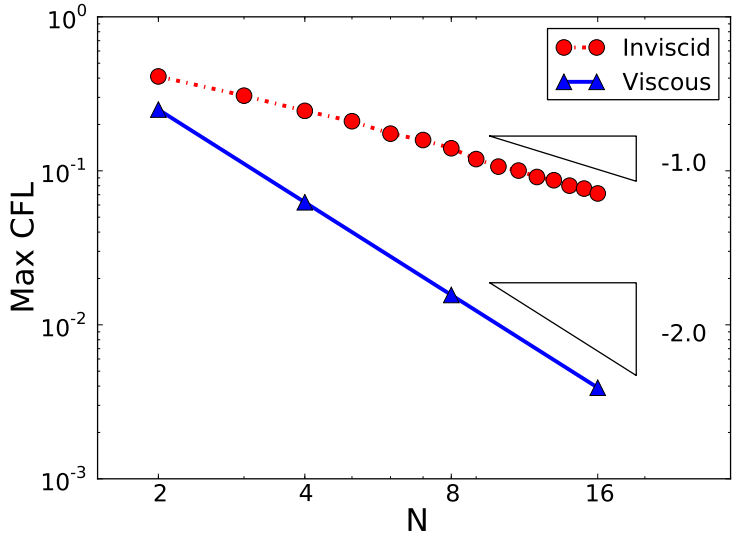


Figure 10. Maximum CFL vs. solution order for the inviscid isentropic vortex convection problem and the viscous Taylor-Green vortex problem at $Re_\Delta = O(1)$.

V.B. Matrix-free Implicit

Implicit methods have the possibility for greater efficiency than explicit methods for higher-order methods, but require the solution of a globally coupled nonlinear problem at each iteration. In order for implicit methods to be competitive, the additional cost of solving this globally coupled problem must be offset by a sufficient decrease in the computational work relative to explicit methods. The memory required for the linearization of the residual grows rapidly with solution order N. Table 1 gives estimates of the memory required to store the full Jacobian matrix for the FAITH problem described in Section I. For reference, the state vector at a single time-step requires 40Gb of storage. Clearly, for the class of problems we are considering, the cost of storing this Jacobian matrix is prohibitive.

We choose a scheme with memory requirements similar to an explicit scheme, i.e. we trade the memory

N	Total Memory (Tb)
2	11
4	89
8	720
16	5700

Table 1. Estimated memory usage for storing the Jacobian for the FAITH simulation using 10^9 degrees of freedom.

for increased operations which can be hidden via optimization and algorithm choice. As such, we consider Jacobian-free Newton-Krylov methods[23] for the solution of the nonlinear problem arising at each iteration,

$$\boldsymbol{R}^*(\boldsymbol{U}^{n+1}) \quad = \quad 0 \tag{27}$$

where $\boldsymbol{R}^*(\boldsymbol{U})$ is the unsteady residual. In a Newton-Krylov method, Eq. (27) is solved using a Newton iteration, where each Newton step requires the solution of a linear system:

$$\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}} \Delta \boldsymbol{U}^k \quad = \quad -\boldsymbol{R}^*(\boldsymbol{U}^k) \tag{28}$$

The linear system, Eq. (28), is solved using a Krylov method. In this work, we use a preconditioned Generalized Minimal Residual (GMRES) algorithm as our Krylov method.[24] In a Krylov method, the matrix $\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}}$ is not required; we only require the application of $\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}}$ to a vector $\boldsymbol{V}$, i.e. we need to compute the Fréchet derivative of $\boldsymbol{R}^*$ in the GMRES search direction.

In this work we consider two approaches for computing the Fréchet derivative. The first approach, is to approximate the linearization using finite differences:

$$\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}} \boldsymbol{V} \simeq \frac{\boldsymbol{R}^*(\boldsymbol{U} + \epsilon \boldsymbol{V}) - \boldsymbol{R}^*(\boldsymbol{U})}{\epsilon} \tag{29}$$

with suitable choice of step size ϵ.[25] This approach allows the computation of a Fréchet derivative for the cost of a single residual evaluation. A particular advantage of this approach is that the approximate Fréchet derivative is available once the residual has been computed, and requires no linearization of residual terms. This is particularly advantageous for turbulence models, as hand linearization of various modeling terms is not required. On the other hand, the finite-difference computation of the Fréchet derivative is sensitive to the step length ϵ and inaccuracies in the Fréchet derivative degrade the convergence of the Newton solver.[23]

An alternative approach is to compute the exact Fréchet derivative. The exact Fréchet derivative corresponds to the linearized form of Eq. (6). As with the residual, the linearized forms of Eqs. (8) and (9) are computed as a sequence of three steps:

1. Evaluation of the state, gradient, linearized state and linearized gradient ($\boldsymbol{U}$, $\nabla \boldsymbol{U}$, $\boldsymbol{V}$ and $\nabla \boldsymbol{V}$) at the quadrature point.

2. Evaluation of the linearized fluxes ($\boldsymbol{F}_{\text{Lin}} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}} \boldsymbol{V} + \frac{\partial \boldsymbol{F}}{\partial \nabla \boldsymbol{U}} \nabla \boldsymbol{V}$) at the quadrature points.

3. Multiplication of the linearized fluxes with the gradient of the basis functions.

This approach is more expensive than the finite-difference approach, however, the additional cost eliminates the dependence on ϵ, and is consistent with the solution of adjoint (dual) problems. In computing the exact Fréchet derivative, we reuse the optimized kernels described in Section II for computing the sum factorization, though we require the additional hand linearization of the fluxes. In our implementation the cost of the exact Fréchet derivative is approximately 50% more than a residual evaluation, though some of this cost is offset by improved convergence.

We demonstrate the benefits of an implicit method by simulating the Taylor-Green vortex evolution at low Reynolds number($Re = 16$) using 8th- and 16th-order spatial discretizations and 64 DOF in each coordinate direction. We use a 4th-order, 5-stage, diagonally-implicit Runge-Kutta (DIRK) scheme as our time-integration scheme. At each stage, the Newton-Krylov algorithm is used to reduce the unsteady residual by 10 orders of magnitude, where at each Newton step the GMRES method is run until the linear residual

has been reduced by 6 orders of magnitude or a maximum of 20 Krylov vectors are used. We use the mass-matrix as the preconditioner to GMRES.

The cost of the implicit (DIRK) scheme relative to the 4-stage explicit Runge-Kutta method is presented in Fig. 11. We plot the cost as a function of time-step relative the explicit method and as a function of the CFL number. For this viscous test case, with $Re_\Delta = O(1)$, the maximum stable CFL for the explicit method scales as $1/N^2$. The implicit method allows us to run at larger CFL number as the time-step is not restricted due to stability, however, we limit the CFL to be $O(1)$ in order to maintain temporal accuracy of the simulation. The cost reported is the number of residual and Fréchet derivative evaluations used for the implicit scheme, divided by the number of residual evaluations required for the explicit scheme. At a CFL number which corresponds to the largest stable timestep for the explicit scheme, the implicit and explicit schemes require a comparable amount of work. As expected, by increasing the timestep the cost of the implicit scheme decreases relative to the explicit scheme. At $CFL \approx 1$, the implicit method is more efficient than the explicit method, requiring 15-25% of the residual evaluations for the explicit scheme even though we are still only using the mass-matrix as a preconditioner. Increasing the CFL number much beyond $CFL \approx 1$ causes the GMRES algorithm to stall and a stronger preconditioner is required.

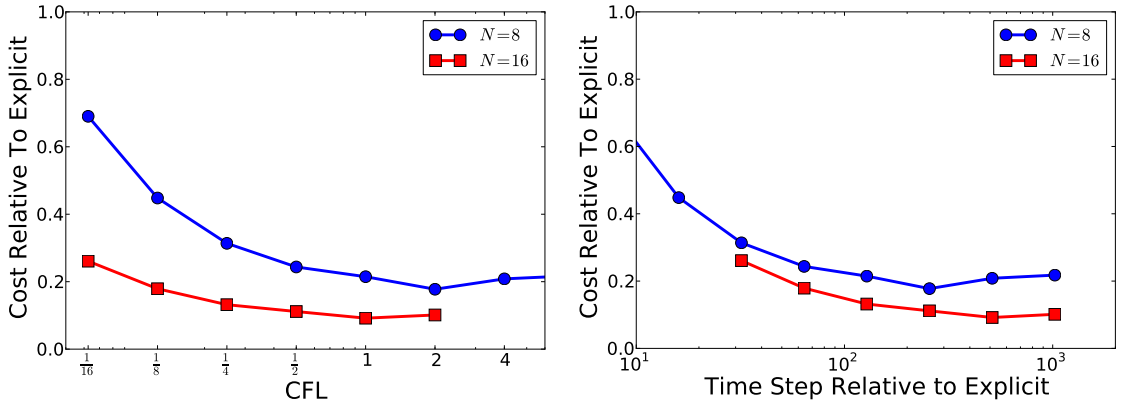


Figure 11. Computational cost of using diagonally-implicit Runge-Kutta method relative to explicit Runge-Kutta method for the Taylor-Green vortex problem ($M = 0.1$, $Re = 16$) using $N = 8$ and $N = 16$ with $h = 1/\sqrt{DOF} = 1/64$.

V.C. Preconditioning

Preconditioning is necessary for stiff problems in order of obtain good performance in a Newton-Krylov scheme.[23] An effective preconditioner approximates the inverse of the Jacobian matrix and has the effect of clustering the eigenvalues of the preconditioned system, allowing for rapid convergence of the Krylov scheme.[23] We require preconditioning methods with minimal memory overhead as storage of a linearization is prohibitive. To this end, we consider an element-wise block-Jacobi preconditioner where the elemental blocks are solved approximately using an alternating-direction-implicit (ADI) scheme.[26] We extend this to a diagonalized ADI scheme, which has previously been used in finite-difference simulations,[27] but to our knowledge not in finite/spectral-element methods.

We derive the diagonalized ADI scheme by considering a constant coefficient problem:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{A}\boldsymbol{u}) = 0 \tag{30}$$

Applying a DG discretization with upwind flux, we have;

$$\sum_\kappa \left\{ \int_\kappa \left\{ \boldsymbol{w} \frac{\partial \boldsymbol{u}}{\partial t} - \nabla \boldsymbol{w} \cdot \boldsymbol{A}\boldsymbol{u} \right\} + \int_{\partial\kappa} \boldsymbol{w} \widehat{\boldsymbol{A}_n \boldsymbol{u}} \right\} = 0 \tag{31}$$

where $\widehat{\boldsymbol{A}_n \boldsymbol{u}}$ is the upwind flux evaluated as:

$$\widehat{\boldsymbol{A}_n \boldsymbol{u}} = \tfrac{1}{2}\boldsymbol{A}_n(\boldsymbol{u}^+ + \boldsymbol{u}^-) + \tfrac{1}{2}|\boldsymbol{A}_n|(\boldsymbol{u}^+ - \boldsymbol{u}^-) = \boldsymbol{A}_n^+ \boldsymbol{u}^+ + \boldsymbol{A}_n^- \boldsymbol{u}^- \tag{32}$$

where $\boldsymbol{A}_n = \boldsymbol{A} \cdot \boldsymbol{n}$, while $\boldsymbol{u}_+$ and $\boldsymbol{u}_-$ denote, respectively, trace values taken from inside or outside the element κ. Applying a backward Euler time integration scheme, the contribution to Eq. (31) due to $\boldsymbol{w}$ and $\boldsymbol{u}$ restricted to an element κ is given by:

$$\boldsymbol{r}^*_\kappa(\boldsymbol{w}|_\kappa, \boldsymbol{u}|_\kappa) \equiv \int_\kappa \left\{ \boldsymbol{w} \frac{1}{\Delta t} \boldsymbol{u}^{n+1} - \nabla \boldsymbol{w} \cdot \boldsymbol{A} \boldsymbol{u}^{n+1} \right\} + \int_{\partial\kappa} \boldsymbol{w} \boldsymbol{A}^+_n \boldsymbol{u}^{n+1} \tag{33}$$

In Eq. (33), $\boldsymbol{r}^*_\kappa(\boldsymbol{w}|_\kappa, \boldsymbol{u}|_\kappa)$ corresponds to an element-wise block diagonal entry of the Jacobian $\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}}$. Using a tensor product basis in two-dimensions we write $\boldsymbol{u} = \boldsymbol{U}_{ij}\phi_i(\xi_1)\phi_j(\xi_2)$ and $\boldsymbol{w} = \phi_m(\xi_1)\phi_n(\xi_2)$. Integrating in the reference element, the corresponding discrete form is given by:

$$\begin{aligned}
\frac{\partial \boldsymbol{R}^*_{mn}}{\partial \boldsymbol{U}_{ij}} &= \int_{\xi_1}\int_{\xi_2} \left\{ \phi_m\phi_n \frac{1}{\Delta t}\phi_i\phi_j - \frac{\partial \phi_m}{\partial \xi_1}\phi_n \tilde{\boldsymbol{A}}_1 \phi_i\phi_j - \phi_m \frac{\partial \phi_n}{\partial \xi_2} \tilde{\boldsymbol{A}}_2 \phi_i\phi_j \right\} |J| \\
&\quad + \int_{\xi_2} \left\{ \phi_m\phi_n \tilde{\boldsymbol{A}}_1^- \phi_i\phi_j |J| \right\}_{\xi_1=-1} + \int_{\xi_1} \left\{ \phi_m\phi_n \tilde{\boldsymbol{A}}_2^- \phi_i\phi_j |J| \right\}_{\xi_1=-1} \\
&\quad + \int_{\xi_2} \left\{ \phi_m\phi_n \tilde{\boldsymbol{A}}_1^+ \phi_i\phi_j |J| \right\}_{\xi_1=1} + \int_{\xi_1} \left\{ \phi_m\phi_n \tilde{\boldsymbol{A}}_2^+ \phi_i\phi_j |J| \right\}_{\xi_1=1}
\end{aligned} \tag{34}$$

where $\tilde{\boldsymbol{A}} = J^{-1}\boldsymbol{A}$, $\tilde{\boldsymbol{A}}_1^{\pm} = \frac{1}{2}(\tilde{\boldsymbol{A}}_1 \pm |\tilde{\boldsymbol{A}}_1|)$ and $\tilde{\boldsymbol{A}}_2^{\pm} = \frac{1}{2}(\tilde{\boldsymbol{A}}_2 \pm |\tilde{\boldsymbol{A}}_2|)$. Equation (34) is written in matrix form as:

$$\begin{aligned}
\frac{1}{|J|}\frac{\partial \boldsymbol{R}^*_\kappa}{\partial \boldsymbol{U}_\kappa} &= \left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{D}_1^I \otimes \tilde{\boldsymbol{A}}_1 \otimes \mathbb{M}_2 \right) - \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2^I \right) \\
&\quad + \left(\mathbb{D}_1^{B-} \otimes \tilde{\boldsymbol{A}}_1^- \otimes \mathbb{M}_2 \right) + \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2^- \otimes \mathbb{D}_2^{B-} \right) + \left(\mathbb{D}_1^{B+} \otimes \tilde{\boldsymbol{A}}_1^+ \otimes \mathbb{M}_2 \right) + \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2^+ \otimes \mathbb{D}_2^{B+} \right)
\end{aligned} \tag{35}$$

where

$$\mathbb{M}_1 = \int_{\xi_1} \phi_m\phi_i \qquad \mathbb{D}_1^I = \int_{\xi_1} \frac{\partial \phi_m}{\partial \xi_1}\phi_i \qquad \mathbb{D}_1^{B\pm} = \{\phi_m\phi_i\}_{\xi_1=\pm 1} \tag{36}$$

$$\mathbb{M}_2 = \int_{\xi_2} \phi_n\phi_j \qquad \mathbb{D}_2^I = \int_{\xi_2} \frac{\partial \phi_n}{\partial \xi_2}\phi_j \qquad \mathbb{D}_2^{B\pm} = \{\phi_n\phi_j\}_{\xi_2=\pm 1} \tag{37}$$

In Eq. (35), we have assumed that the Jacobian of the mapping from element reference space to physical space is constant. The non-constant Jacobian case is addressed below. In order to clarify the derivation, we define operators $\mathbb{D}_1$ and $\mathbb{D}_2$ which correspond to the action of interior and boundary convection operators in the ξ_1 and ξ_2 directions, respectively, such that we may rewrite Eq. (35) as:

$$\frac{1}{|J|}\frac{\partial \boldsymbol{R}^*_\kappa}{\partial \boldsymbol{U}_\kappa} \equiv \left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{D}_1 \otimes \tilde{\boldsymbol{A}}_1 \otimes \mathbb{M}_2 \right) - \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2 \right) \tag{38}$$

$$\begin{aligned}
&= \left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{D}_1 \otimes \tilde{\boldsymbol{A}}_1 \otimes \mathbb{M}_2 \right) \right] \left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right)^{-1} \left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2 \right) \right] \\
&\quad - \left(\mathbb{D}_1 \otimes \tilde{\boldsymbol{A}}_1 \otimes \mathbb{M}_2 \right) \left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right)^{-1} \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2 \right)
\end{aligned} \tag{39}$$

Following [26] the second term in Eq. (39) is omitted resulting in the approximate inverse:

$$\begin{aligned}
|J| \left(\frac{\partial \boldsymbol{R}^*_\kappa}{\partial \boldsymbol{U}_\kappa} \right)^{-1} &\simeq \left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{D}_1 \otimes \tilde{\boldsymbol{A}}_1 \otimes \mathbb{M}_2 \right) \right]^{-1} \left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) \times \\
&\qquad \left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\mathbb{M}_1 \otimes \tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2 \right) \right]^{-1}
\end{aligned} \tag{40}$$

$$\begin{aligned}
&= \left\{ \left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t} \right) - \left(\mathbb{D}_1 \otimes \tilde{\boldsymbol{A}}_1 \right) \right] \otimes I \right\}^{-1} \left(I \otimes \frac{I}{\Delta t} \otimes I \right) \times \\
&\qquad \left\{ I \otimes \left[\left(\frac{I}{\Delta t} \otimes \mathbb{M}_2 \right) - \left(\tilde{\boldsymbol{A}}_2 \otimes \mathbb{D}_2 \right) \right] \right\}^{-1}
\end{aligned} \tag{41}$$

Here we have factored terms in Eq. (41) to highlight that the inverses correspond to the solution of one-dimensional problems in either ξ_1 or ξ_2 directions. The application of the ADI scheme can be computed using the following sequence of steps:

1. Solve N independent one-dimensional problems in the ξ_1 direction.

2. Scale by the time-step.

3. Solve N independent one-dimensional problems in the ξ_2 direction.

The one-dimensional problems in steps 1 and 3 have m coupled flow equations. Following Pulliam and Chaussee[27] a diagonal form of Eq. (41) is obtained using an eigenvalue decomposition of $\tilde{\boldsymbol{A}}$. We write Eq. (41) as:

$$\begin{aligned}
|J|\left(\frac{\partial \boldsymbol{R}^*_\kappa}{\partial \boldsymbol{U}_\kappa}\right)^{-1} &\simeq \left\{\left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t}\right) - \left(\mathbb{D}_1 \otimes \boldsymbol{T}_1^{-1}\boldsymbol{\Lambda}_1\boldsymbol{T}_1\right)\right] \otimes I\right\}^{-1} \left(I \otimes \frac{I}{\Delta t} \otimes I\right) \times \\
&\qquad \left\{I \otimes \left[\left(\frac{I}{\Delta t} \otimes \mathbb{M}_2\right) - \left(\boldsymbol{T}_2^{-1}\boldsymbol{\Lambda}_2\boldsymbol{T}_2 \otimes \mathbb{D}_2\right)\right]\right\}^{-1} \\
&= (I \otimes \boldsymbol{T}_1 \otimes I)\left\{\left[\left(\mathbb{M}_1 \otimes \frac{I}{\Delta t}\right) - \left(\mathbb{D}_1 \otimes \boldsymbol{\Lambda}_1\right)\right] \otimes I\right\}^{-1} \left(I \otimes \frac{\boldsymbol{T}_1^{-1}\boldsymbol{T}_2}{\Delta t} \otimes I\right) \times \\
&\qquad \left\{I \otimes \left[\left(\frac{I}{\Delta t} \otimes I\right) - \left(\boldsymbol{\Lambda}_2 \otimes \mathbb{D}_2\right)\right]\right\}^{-1} (I \otimes \boldsymbol{T}_2^{-1} \otimes I) \qquad (42)
\end{aligned}$$

where $\boldsymbol{\Lambda}_1$ and $\boldsymbol{\Lambda}_2$ are eigenvalues of $\tilde{\boldsymbol{A}}_1$ and $\tilde{\boldsymbol{A}}_2$ with corresponding eigenvector $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$. The diagonalized ADI scheme, corresponding to the action of the approximate inverse in Eq. (42), is summarized in the following steps:

1. Transform to characteristic variables in the ξ_1 direction at each collocation point.

2. Solve $N \times m$ independent scalar one-dimensional advection problems in ξ_1 direction.

3. Transform to characteristic variables in the ξ_2 direction and scale by the time-step.

4. Solve $N \times m$ scalar one-dimensional advection problems in ξ_2 direction.

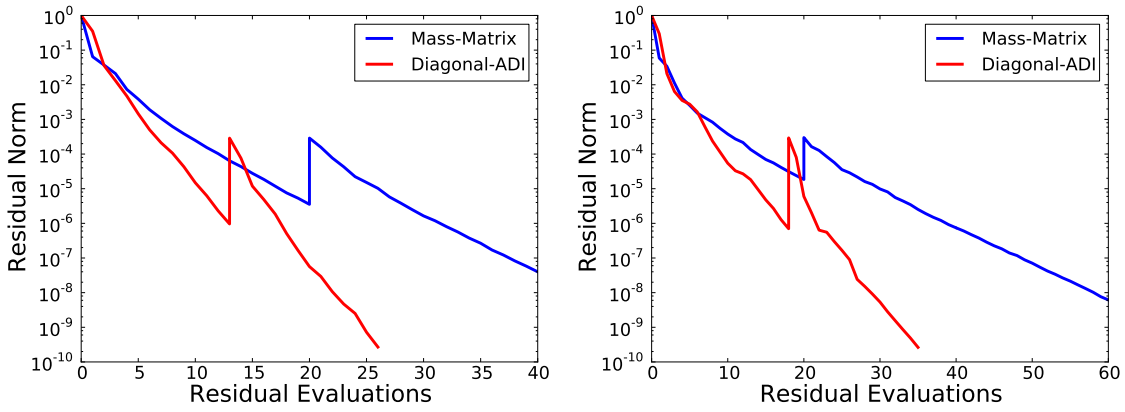
5. Transform back to original variables

We note that in the diagonalized ADI scheme, we solve one-dimensional scalar advection problems, as opposed to a system of equations, which results in a savings without significant loss of convergence rate. Specifically, the diagonalized scheme requires forming and factoring 3 $N \times N$ systems corresponding to 3 distinct eigenvalues as opposed to a $5N \times 5N$ system for the full ADI scheme. This reduces the cost by approximately a factor of 40.

In case of variable coefficients (or variable Jacobian) the linearization, $\frac{\partial \boldsymbol{R}^*}{\partial \boldsymbol{U}}$, can not be expressed simply as the sum of tensor product of matrices. However, we may still apply the steps of the ADI or diagonalized ADI algorithms, recognizing that the one-dimensional problems now have variable coefficients. We note that while for constant coefficient problems the ADI scheme and diagonalized ADI scheme are mathematically equivalent, in the variable coefficient case the diagonalized scheme introduces an additional $O(\Delta t)$ error.[27]

Applying the diagonalized ADI scheme for the solution of the isentropic vortex convection problem at a Mach number of 0.5 with $h = \sqrt{DOF} = 1/64$, Figure 12 shows the convergence history for a typical stage of the implicit scheme at $CFL = 1.0$ for 8th- and 16th-order solutions. The linear residual is plotted as a function of the number of residual or Fréchet derivative evaluations using both mass-matrix and diagonalized ADI preconditioning. The diagonalized ADI scheme reduces the number of residual evaluations required by about a factor of two for this problem. As the cost of applying the diagonalized ADI scheme is the same as a single residual evaluation, the total CPU time for the diagonalized ADI scheme is the same as using mass-matrix preconditioning. However, using the diagonalized ADI scheme has the benefit that fewer Krylov vectors are required.

The diagonalized ADI scheme is an effective low-memory preconditioner for convection dominated problems. In order to develop an efficient preconditioner for viscous dominated regions we intend to leverage our spectral-element framework and pursue spectral/p-multigrid methods. Spectral/p-multigrid methods have been widely used for CG spectral-element discretizations where theoretical and numerical results gives p-independent convergence rates for elliptic problems.[28,29] While no corresponding convergence theory exists for DG discretizations, spectral/p- multigrid approaches have also been widely applied to DG discretizations.[30–33]



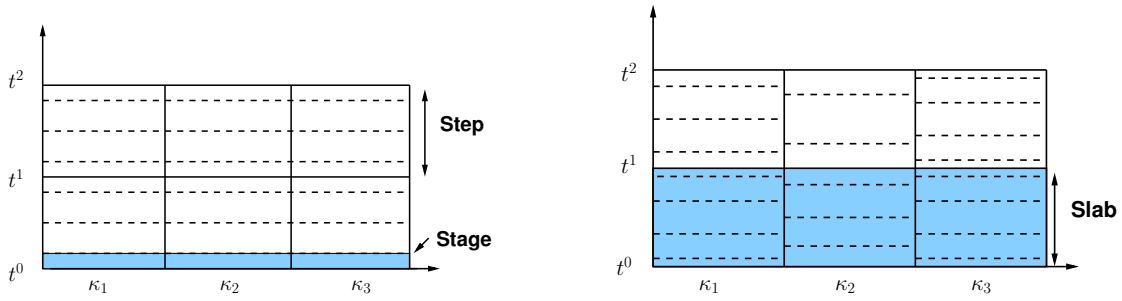
(a) $N = 8$ (b) $N = 16$

Figure 12. Residual convergence history for a single Newton-Step using diagonally-implicit Runge-Kutta scheme for the isentropic vortex convection problem with $h = 1/\sqrt{DOF} = 1/64$.

V.D. Space-Time formulation

The implicit method with preconditioning provides an efficient scheme for higher-order unsteady problems. We can further improve this scheme by the use of a space-time formulation to integrate forward in time. The use of a space-time formulation enables us to adapt locally in both the spatial and temporal directions, similar to AMR methods with sub-cycling. Here the temporal spectral order (and hence the number of DOF) is controlled, leading to a significant reduction in cost as the increased resolution in time is only applied where necessary.

The space-time formulation is contrasted with the implicit Runge-Kutta scheme graphically in Fig. 13. In an implicit RK scheme each stage is solved sequentially, requiring at each stage an implicit solve corresponding to a problem which is globally coupled in space at a single point in time. In a space-time formulation the solution is computed at the temporal collocation points. However, unlike the implicit scheme all temporal collocation points within a time-slab are solved concurrently in a single globally coupled problem. Figure 13 also depicts space-time adaptation, where different temporal solution orders are used in different elements, potentially reducing the total cost.



(a) Implicit Runge-Kutta (b) Space-Time

Figure 13. Graphical representation of implicit and space-time formulations. The implicit scheme requires a globally coupled solve at each stages, while the space-time formulation requires a globally coupled solve over the entire space-time slab.

Space-time formulations are generally considered too expensive for practical engineering simulations, as they require the solution of a globally coupled system of equations for each time-slab. As we have noted, the cost of storing the linearization for a single step of an implicit scheme is prohibitively expensive. Using a space-time formulation this storage cost is scaled by the order of the basis used in the temporal direction.

In the previous section, we demonstrate an implicit scheme with similar storage requirement as an explicit method using a matrix-free approach and diagonalized ADI preconditioner. Leveraging this full matrix-free approach allows us to afford the storage and efficient solution of the space-time formulation.

Define a time interval (time-slab) $I^n = [t^n, t^{n+1}]$. A space-time finite element space $\mathcal{V}_h$ consisting of piece-wise polynomial functions in both space and time on each element κ is given by:

$$\mathcal{V}_h = \{\boldsymbol{w}, \boldsymbol{w}|_\kappa \in [\mathcal{P}(\kappa \times I)]^m\} \tag{43}$$

We seek a solution $\boldsymbol{u} \in \mathcal{V}_h$ satisfying the weak form of Eq. (1):

$$\begin{aligned}\sum_\kappa \Bigg\{ \int_I \int_\kappa \left(-\frac{\partial \boldsymbol{w}}{\partial t}\boldsymbol{u} - \nabla \boldsymbol{w} \cdot (\boldsymbol{F}^I - \boldsymbol{F}^V) \right) + \int_I \int_{\partial\kappa} \boldsymbol{w}(\hat{\boldsymbol{F}}^I - \hat{\boldsymbol{F}}^V) \cdot \boldsymbol{n} \\ + \int_\kappa \boldsymbol{w}(t_-^{n+1})\boldsymbol{u}(t_-^{n+1}) - \boldsymbol{w}(t_+^n)\boldsymbol{u}(t_-^n) \Bigg\} &= 0 \qquad \forall \boldsymbol{w} \in \mathcal{V}_h. \end{aligned} \tag{44}$$

As with our spatial discretization, we use a tensor product basis:

$$\boldsymbol{u}(x(\xi), t(\tau)) = \sum_{i,j,k,l} U_{ijkl}\Phi_{ijkl} \qquad \Phi_{ijkl} = \phi_i(\xi_1)\phi_j(\xi_2)\phi_k(\xi_3)\phi_l(\tau) \tag{45}$$

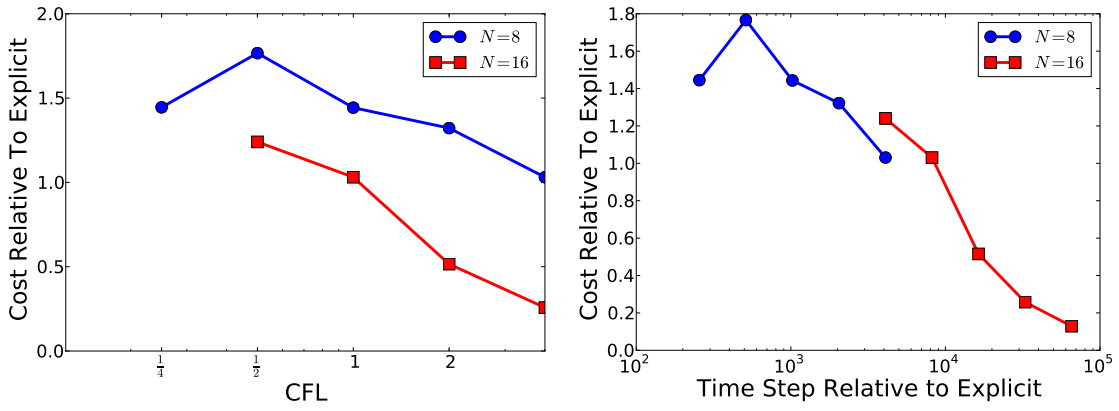
where $t(\tau)$ defines a mapping from the reference interval $\tau \in [-1, 1]$ to the interval $I = [t^n, t^{n+1}]$. The integrals in the space-time domain are computed using collocation and the sum-factorization approach, leading to a residual evaluation cost which scales as $N \times (d+1)$. Equation (44) gives a system of nonlinear equations of the form $\boldsymbol{R}^*(\boldsymbol{U}) = 0$, which must solved for each time slab. For a given spatial discretization, the size of system which must be solved in each time slab is now N times larger than that for the implicit scheme, as the space-time system involves all temporal collocation points in a space-time slab. We note that this results in an increase relative to the implicit scheme in the memory required for the storage of the Krylov vectors in GMRES. In the current implementation we can easily absorb this cost.

The efficiency of our space-time formulation hinges on the ability of the Newton-Krylov scheme to rapidly converge the nonlinear problem in each time slab. We demonstrate the feasibility of space-time approach using a simple mass-matrix preconditioning for the viscous Taylor-Green problem at $Re_\Delta = O(1)$. We solve the viscous Taylor-Green vortex problem at $M = 0.1$ and $Re = 16$ using 8th- and 16th-order spatial discretizations using 64 degrees of freedom in each coordinate direction. We use 8th- and 16th-order solutions respectively in the temporal direction. For each time-slab, the Newton-Krylov algorithm is used to reduce the unsteady residual by 10 orders of magnitude. For each Newton iteration we use the mass-matrix preconditioned GMRES method with 20 Krylov vectors.

Figure 14 shows the cost of the Taylor-Green simulation relative to the explicit scheme as a function of the space-time CFL number $CFL = \frac{c\Delta t}{Nh}$ and the relative number of explicit time steps. In Fig. 14 the cost of a space-time residual evaluation is N times that of a single spatial residual. As the space-time formulation does not have a time-step limited by stability constraints, we are able to use much larger time-steps than the explicit scheme. At $CFL = O(1)$ the space-time formulation using a simple mass-matrix preconditioner has a cost similar to the explicit scheme. For $CFL > 4$ the GMRES algorithm stalls and improved preconditioning may be required. We emphasize that our space-time formulation has similar memory requirements as an explicit scheme, and using only a very simple mass-matrix preconditioning has equivalent computational cost for viscous dominated problems. We anticipate that for wall bounded flows with improved preconditioning the space-time formulation will provide significant efficiency gain relative the explicit scheme.

VI. Summary and Future Work

This paper summarizes a CG and DG spectral-element framework for the simulation of compressible high-Reynolds-number flows in the absence of discontinuities. The focus of the work is maximizing the efficiency of the computational schemes in terms of memory and cpu time to enable unsteady simulations with a large number of space-time DOF. A collocation scheme combined with optimized computational kernels for the Intel SIMD extensions for x86_64 micro-architectures provides a residual evaluation with computational cost independent of order of accuracy through 16th order, and competitive with existing low-order production computing codes. This optimized residual framework is leveraged to provide an efficient



(a) (b)

Figure 14. Computational cost of using space-time formulation relative to explicit Runge-Kutta method for the Taylor-Green vortex problem ($M = 0.1$, $Re = 16$) using $N = 8$ and $N = 16$ with $h = 1/\sqrt{DOF} = 1/64$.

implicit scheme based on a matrix-free Newton-Krylov nonlinear solver. A novel preconditioner based on the finite-difference diagonalized ADI scheme was developed which maintains the low memory of the matrix-free implicit solver, while still providing improved convergence properties for practical problems. Finally, the emphasis on low memory usage throughout the solver development was leveraged to implement a coupled space-time DG solver to efficiently provide spectral-element $h-p$ adaptive capability in both space and time.

While there still remain some open items within the current documented capability, such as extending the ADI preconditioning to space-time, the core functionality is designed and implemented. The next step is to apply the new capability to benchmark turbulence problems, such as the Taylor-Green evolution, isotropic turbulence and decay, and the wall-bounded planar channel flow, before progressing to more complex separated flows. Further work is aimed at improving the existing capability further, primarily through the planned use of a spectral-multigrid solver, and development of an appropriate closure model. We view the closure modeling as coupled to the choice of stabilization strategy as in a variational multiscale framework both numerical stabilization and the subgrid scale model approximate the effect of the unresolved modes on the resolved modes.[1,34] Thus, we will consider alternative stabilization strategies such as Streamline-Upwind/Petrov-Galerkin (SUPG),[35,36] Galerkin-Least-Square (GLS)[37] or multiscale[38] methods within our finite-element framework.

Acknowledgments

Laslo Diosady was funded by the NASA Fundamental Aeronautics program Revolutionary Computational Aerosciences project through an appointment to the NASA Postdoctoral Program at the Ames Research Center, administered by Oak Ridge Associated Universities.

References

[1]Hughes, T. J. R., Feijoo, G. R., Mazzei, L., and Quincy, J.-B., "The variational multiscale method - a paradigm for computational mechanics," *Comput. Methods Appl. Math.*, Vol. 166, 1998, pp. 3–24.

[2]Collis, S. S., "Monitoring unresolved scales in multiscale turbulence modeling," *Phys. Fluids*, Vol. 13, 2001, pp. 1800–1806.

[3]Hughes, T. J. R., "Multiscale Phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods," *Comput. Methods Appl. Math.*, Vol. 127, 1995, pp. 387–401.

[4]Hughes, T. J., Mazzei, L., and Jansen, K. E., "Large Eddy Simulation and the variational multiscale method," *Computing and Visualization in Science*, Vol. 3, 2000, pp. 47–59.

[5]Koobus, B. and Farhat, C., "A variational multiscale method for the large eddy simulation of compressible turbulent flows on unstructured meshes – application to vortex shedding," *Comput. Methods Appl. Mech. Engrg*, Vol. 193, 2004, pp. 1367–1383.

[6]Farhat, C., Rajasekharan, J., and Koobus, B., "A dynamic variational multiscale method for large eddy simulations on unstructured meshes," *Computer Methods in Applied Mechanics and Engineering*, Vol. 195, No. 1316, 2006, pp. 1667 – 1691.

[7]Oberai, A. and Hughes, T., "The Variational Multiscale Formulation of LES: Channel Flow at $Re_\tau = 590$," AIAA 2002-1056, 2002.

[8]Ramakrishnan, S. and Collis, S. S., "The Local Variational Multi-Scale Method for Turbulence Simulation," Sandia Report SAND2005-2733, 2005.

[9]Bell, J. H., Heineck, J. T., Zilliac, G., and Mehta, R. D., "Surface and Flow Field Measurements on the FAITH Hill Model," AIAA Paper 2012-0704, 2012.

[10]Wang, Z., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H., Kroll, N., May, G., Persson, P.-O., van Leer, B., and Visbal, M., "High-Order CFD Methods: Current Status and Perspective," *International Journal for Numerical Methods in Fluids*, Vol. 00, 2012, pp. 1–42.

[11]Roe, P. L., "Approximate Riemann solvers, parameter vectors, and difference schemes," *J. Comput. Phys.*, Vol. 43, No. 2, 1981, pp. 357–372.

[12]Bassi, F. and Rebay, S., "GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations," *Discontinuous Galerkin Methods: Theory, Computation and Applications*, edited by K. Cockburn and Shu, Springer, Berlin, 2000, pp. 197–208.

[13]Vos, P., Sherwin, S., and Kirby, R., "From h to p Efficiently: Implementing finite and spectral/hp element discretizations to achieve optimal performance at low and high order approximations." *J. Comput. Phys.*, Vol. 229, No. 13, 2010, pp. 5161–5181.

[14]van Ress, W., Leonard, A., Pullin, D., and Koumoutsakos, P., "A comparison of vortex and pseudo-spectral methods for the simulation of periodic vortical flows at high Reynolds number," *J. Comput. Phys.*, Vol. 230, 2011, pp. 2794–2805.

[15]Karamanos, G.-S. and Karniadakis, G., "A Spectral Vanishing Viscosity Method for Large-Eddy Simulations," *J. Comput. Phys.*, Vol. 163, 2000, pp. 22–50.

[16]Kirby, R. and Sherwin, S., "Stabilisation of spectral/hp element methods through spectral vanishing viscosity: Application to fluid mechanics modelling," *Comput. Methods Appl. Mech. Engrg*, Vol. 195, 2006, pp. 3128–3144.

[17]Honein, A. E. and Moin, P., "Higher entropy conservation and numerical stability of compressible turbulence simulations," *J. Comput. Phys.*, Vol. 201, 2004, pp. 531–545.

[18]Kirby, R. M. and Karniadakis, G. E., "De-aliasing on non-uniform grids: algorithms and applications," *J. Comput. Phys.*, Vol. 191, 2003, pp. 249–264.

[19]Orszag, S. A., "On the Elimination of Aliasing in Finite-Difference Schemes by Filtering High-Wavenumber Components." *Journal of Atmospheric Sciences*, Vol. 28, Sept. 1971, pp. 1074–1074.

[20]Gassner, G. and Beck, A. D., "On the accuracy of high-order discretizations for underresolved turbulence simulations," *Theor. Comput. Fluid Dyn.*, Vol. 33, 2011, pp. 2560–2579.

[21]Persson, P.-O., "A sparse and high-order accurate line-based discontinuous Galerkin method for unstructured meshes," *J. Comput. Phys.*, Vol. 233, No. 0, 2013, pp. 414 – 429.

[22]Gassner, G. and Kopriva, D. A., "A comparison of the dispersion and dissipation errors of Gauss and Gauss-Lobatto discontinuous Galerkin spectral element methods," *SIAM J. Sci. Comput.*, Vol. 33, 2011, pp. 2560–2579.

[23]Knoll, D. A. and Keyes, D. E., "Jacobian-free Newton-Krylov methods: a survey of approaches and applications," *J. Comput. Phys.*, Vol. 193, No. 1, 2004, pp. 357–397.

[24]Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM J. Sci. Comput.*, Vol. 7, No. 3, 1986, pp. 856–869.

[25]Saad, Y., *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 1996.

[26]Beam, R. and Warming, R., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *AIAA Journal*, Vol. 16, No. 4, 1978, pp. 393 – 402.

[27]Pulliam, T. and Chaussee, D., "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *J. Comput. Phys.*, Vol. 39, 1981, pp. 347–363.

[28]Ronquist, E. M. and Patera, A. T., "Spectral Element Multigrid. I. Formulation and Numerical Results," *Journal of Scientific Computing*, Vol. 2, 1987, pp. 389–406.

[29]Maday, Y. and Munoz, R., "Spectral Element Multigrid. II. Theoretical Justification," *Journal of Scientific Computing*, Vol. 3, 1988, pp. 323–353.

[30]Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., "p-Multigrid solution of high-order discontiunous Galerkin discretizations of the compressible Navier-Stokes equations," *J. Comput. Phys.*, Vol. 207, No. 1, 2005, pp. 92–113.

[31]Persson, P.-O. and Peraire, J., "An efficient low memory implicit DG algorithm for time dependent problems," AIAA 2006-0113, 2006.

[32]Diosady, L. T. and Darmofal, D. L., "Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations," *J. Comput. Phys.*, Vol. 228, 2009, pp. 3917–3935.

[33]van der Vegt, J. and Rhebergen, S., "hp-Multigrid as Smoother algorithm for higher order discontinuous Galerkin discretizations of advection dominated flows: Part I. Multilevel Analysis," *J. Comput. Phys.*, Vol. 231, 2012, pp. 7537–7564.

[34]Bazilevs, Y., Calo, V., Cottrell, J., Hughes, T., Reali, A., and Scovazzi, G., "Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows," *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, No. 14, 2007, pp. 173 – 201.

[35]Hughes, T. J. R. and Tezduyar, T. E., "Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations," *Comput. Methods Appl. Math.*, Vol. 45, 1984, pp. 217–284.

[36]Hughes, T. J. R., Franca, L., and Mallet, M., "A new finite element formulation for computational fluid dynamics: I Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics," *Comput. Methods Appl. Math.*, Vol. 54, 1986, pp. 223–234.

[37]Hughes, T. J. R., Franca, L. P., and Hulbert, G. M., "A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations," *Comput. Methods Appl. Math.*, Vol. 73, 1989, pp. 173–189.

[38]Franca, L. P., Frey, S. L., and Hughes, T. J. R., "Stabilized finite element methods: I. Application to the advective-diffusive model," *Comput. Methods Appl. Math.*, Vol. 95, 1992, pp. 253–276.