Effect of Flux Function Order and Working Precision in Spectral Element Methods

Will Trojak
Texas A&M University
20 PUBLICATIONS **29** CITATIONS
SEE PROFILE

Ashley Scillitoe
Alan Turing Institute
18 PUBLICATIONS **55** CITATIONS
SEE PROFILE

Rob Watson
Queen's University Belfast
38 PUBLICATIONS **122** CITATIONS
SEE PROFILE

Effect of Flux Function Order and Working Precision in Spectral Element Methods

Will Trojak[*]

Department of Engineering, University of Cambridge, Cambridge, UK, CB2 1PZ

A. Scillitoe[†]

The Alan Turing Institute, Kings Cross, London, UK, NW1 2DB

Rob Watson[‡]

School of Mechanical and Aerospace Engineering, Queen's University Belfast, Belfast, UK, BT9 5AH

We investigate the effect of aliasing when applied to the storage of variables, and their reconstruction for the solution of conservation equations. In particular, we investigate the effect on the error caused by storing primitives versus conserved variables for the Navier–Stokes equations. It was found that storing the conserved variables introduces less dissipation and that the dissipation caused by constructing the conversed variables from the primitives grows factorially with order. Hence, this problem becomes increasingly important with the continuing move towards higher orders. Furthermore, the method of gradient calculation is investigated, as applied to the viscous fluxes in the Navier–Stokes equations. It was found that in most cases the difference was small, and that the product rule applied to the gradients of the conserved variables should be used due to a lower operation count. Finally, working precision is investigated and found to have a minimal impact on free-stream-turbulence-like flows when the compressible equations are solved, except at low Mach numbers.

I. Introduction

Over the course of the last three decades Large Eddy Simulation (LES) has become increasingly used for the exploration of flow physics. Looking forward to how CFD will be used tomorrow, NASA CFD Vision 2030[1] predicts that hybrid RANS/LES and wall-modelled LES will become increasingly used in aerospace design. These methods are likely to prevail until sufficient technological developments allow for wall-resolved LES to become a feasible part of the design process. One effect of this continuing shift from low fidelity modelling to high fidelity simulation is that the gap is bridged, in part, by adapting existing RANS tools for LES. For example, ANSYS Fluent began as a tool for solving the RANS equations, but has increasingly developed LES capability.[2] While, more recently, some tools have been developed from the outset to be spectral or capable of high order, for example Nektar++,[3] PyFR,[4] etc.

Through this journey from predominantly low order RANS to LES and then onto high-order several dogmas have developed that have continued into recent methods. Chief among these is the form of the variables stored when solving a conservation equation. In the context of fluid mechanics, this commonly comes down to whether the primitive or conservative variables are stored. To the knowledge of the authors, the only justification given for this choice one way or the other comes from Fluent's documentation, where the given arguments are 'it is a natural choice when solving incompressible flows' and 'to obtain more accurate velocity and temperature gradients in viscous fluxes, and pressure gradients in inviscid fluxes'. Yet, which variables are stored is an important feature of a scheme, as it will impact how the flux terms are constructed and their relative order. Challenging this received wisdom forms the primary question to be answered. In particular: is the error introduced through the construction of the terms required in fluids dynamics sufficient for one storage method to be favourable?

In fluid dynamics, it is common to require second derivatives to simulate viscous effects. In turn, this requires gradients of the primitives. Several methods are actively used to calculate these terms, the Senga2 code[5] uses

[*] PhD Candidate, AIAA Student Member, wt247@cam.ac.uk

[†] Research Associate in Computational Fluid Dynamics, AIAA Member

[‡] Lecturer, AIAA Member

the product rule on the gradient of the conserved variables, whereas several industrial codes directly calculate the gradient of the primitives. Due to the differences in the two methods, the viscous fluxes will have different orders. We wish to understand if these differences are significant, and, secondly, if one method is more efficient.

A related topic that has seen much work is relating to spectral aliasing in LES and the mechanisms by which it is brought about. For example, see Moser et al.,[6] Blaisdell et al.,[7] and Kravchenko et al.[8] These works focus on the aliasing brought about in spectral methods by the differentiation stage of a non-linear flux. It was broadly concluded that the skew-symmetric form can reduce the effect of aliasing, without resorting to active de-aliasing techniques. But, depending on the flow conditions, occasionally the skew-symmetric aliasing error can be large. A more recent study, by Winters et al.,[9] compared two splitting techniques when applied to the DG method. These investigations are mentioned as this work was found to be the most similar in nature to the problem confronted here. However, throughout the work on the skew-symmetric form little attention has been paid to which variables are stored, yet the same spectral analysis would indicate it could have a noticeable impact.

The second dogma that we wish to investigate is whether it is valid that variables should be stored in double precision, or if single precision is actually adequate. This links to the question of which variables are stored, as different operation counts will cause the effect of working precision to be different.

In most typical calculations the norm is to use double precision throughout, however, as the size of problems to be tackled grows so too does the memory usage. Therefore, it would be beneficial for both reducing memory overhead and increasing computational speed if single precision were used. Further to this, some hardware — notably many GPUs — include only comparatively few double precision arithmetic units. Therefore, the increase in computational speed can be as much as 32 times by moving move from 32 to 64 bit precision. Some investigation into this question has been performed, notably by Homann et al.,[10] on the DNS of incompressible homogeneous turbulence using a variable precision incompressible pseudo-spectral scheme. They observed little to no difference when the precision was changed, but this may have been due to the explicit enforcing of incompressibility. Another investigation into precision was presented in the review paper by Bailey,[11] which spanned several regimes of flow physics. This investigation, however, was in the opposite direction, looking at the effect of using 128-bit precision. It was found that it could be important and concluded that better support of adaptive precision should be made by software and hardware. Therefore, we wish to investigate if the same insensitivity to working precision is true for high order polynomial based methods, or as Bailey[11] found it could, in fact, be important.

The remainder of this paper is structured around a high-order numerical method introduced in section II. The theory of polynomial aliasing and the effect of the order is presented in section III. We then go on to set out in section IV the variable forms and conversion methods that will be investigated. Then, in sections V & VI, numerical experiments with Euler's equations and the Navier-Stokes equations are performed respectively. This is followed by the variation of the working precision when applied to the Navier-Stokes equations. Finally, the conclusions are presented in section VII.

II. High-Order Flux Reconstruction

This paper aims to investigate how the aforementioned dogmas are affected in the context of a spectral element method. To provide a flexible framework for performing simulation at various orders of accuracy, we will make use of a particular spectral element type method called Flux Reconstruction (FR).[12,13] This section aims to introduce the methodology behind FR, helping to inform the later investigation into aliasing. FR is broadly based on the techniques used in Nodal Discontinuous-Galerkin,[14] and, as such, we begin by subdividing the domain $\mathbf{\Omega}$ into n sub-domains.

$$\mathbf{\Omega}=\bigcup_{n=1}^{N}\mathbf{\Omega}_n, \quad \text{and} \quad \mathbf{\Omega}_i\cap\mathbf{\Omega}_j=\emptyset\forall i\neq j \tag{1}$$

If we then focus on the method as applied to 1D conservation equations, we can define a spatial transformation from the physical sub-domain $\mathbf{\Omega}_n\in[x_n,x_{n+1}]$ to a reference domain $\hat{\mathbf{\Omega}}\in[-1,1]$. This can be achieved via the mapping $\Gamma_n:x\rightarrow\xi$, where x is a variable in $\mathbf{\Omega}_n$ and ξ is in $\hat{\mathbf{\Omega}}$. Γ_n is then defined as:

$$\xi=\Gamma_n(x)=2\left(\frac{x-x_n}{x_{n+1}-x_n}\right)-1 \tag{2}$$

If we proceed to solve the 1D first order conservation equation, then:

$$\frac{\partial u}{\partial t}+\frac{\partial f}{\partial x}=0 \tag{3}$$

where u is the conserved variable and the flux is $f=f(u)$. Within each sub-domain, we use the data stored at a series of points to form a local polynomial of u and f:

$$\hat{u}^{\delta}(\xi)=\sum_{i=0}^{p}\hat{u}_i^{\delta}l_i(\xi) \tag{4}$$

$$\hat{f}^{\delta D}(\xi)=\sum_{i=0}^{p}\hat{f}_i^{\delta}l_i(\xi) \tag{5}$$

where p is the order, the superscript delta symbolises that the polynomials are local to one element, and the hat marks that the variable has been transformed from the physical to reference domain. In the case of the flux, there is also a D to symbolise it is currently only a fit based on the data and hence not strictly continuous. Here, the polynomial basis, $l_i(\xi)$, is the Lagrange basis, defined as:

$$l_i(\xi)=\prod_{\substack{j=0\\ j\neq i}}^{p}\frac{\xi-\xi_i}{\xi_j-\xi_i}. \tag{6}$$

Now the approximation in Eq. (4) can be used to extrapolate to the edges of the element at $\xi=\pm1$. This data can be combined with the edge values of the surrounding elements to calculate a common value at each element interface. This is key in enabling the solution between elements to be made continuous. There are several methods of finding a common value, for example, central differencing can be used, but at the expense of needing smoothing. Alternatively, a method such as a Riemann solver can be used that accounts for the upwind direction in hyperbolic equations[15] and consequently adds some stabilising dissipation.

With left and right common interface values calculated, defined as $\hat{f}_L^{\delta I}$ and $\hat{f}_R^{\delta I}$, the common value then needs to be propagated into the element to form a continuous solution. This is achieved via correction functions, h_L and h_R, that have the following properties:

$$h_L(-1)=h_R(1)=1 \tag{7}$$

$$h_R(-1)=h_L(1)=0. \tag{8}$$

There are several families of correction function, with it being known that the choice can have a large impact on the behaviour of the method.[16–19] The correction to the flux term is calculated as:

$$\hat{f}^{\delta C}=(\hat{f}_L^{\delta I}-\hat{f}_L^{\delta D})h_L+(\hat{f}_R^{\delta I}-\hat{f}_R^{\delta D})h_R \tag{9}$$

then formulating the corrected flux gradient:

$$\frac{\partial\hat{f}^{\delta}}{\partial\xi}=\frac{\partial\hat{f}^{\delta D}}{\partial\xi}+(\hat{f}_L^{\delta I}-\hat{f}_L^{\delta D})\frac{\mathrm{d}h_L}{\mathrm{d}\xi}+(\hat{f}_R^{\delta I}-\hat{f}_R^{\delta D})\frac{\mathrm{d}h_R}{\mathrm{d}\xi} \tag{10}$$

lastly by using the transformed equation:

$$\frac{\partial\hat{u}^{\delta}}{\partial t}+\frac{\partial\hat{f}^{\delta}}{\partial\xi}=0 \tag{11}$$

with Eq. (10), a suitable temporal integration method may be used to advance the solution in time.

One advantage of FR is that by varying the number of points and hence the order of the interpolation in Eq. (4 & 5), the order accuracy of the scheme may be changed with relative ease. Hence, this will easily allow for the effects of aliasing under investigation here to monitored at different orders, with the aim of drawing further conclusions for the development of high-order methods.

III. Discrete Error Mechanisms

In this section we wish to study the mechanism by which errors can enter approximate solutions. In particular, we do this from a polynomial point of view, which is most applicable to an approximation in a finite element framework. Let us start by studying a simple generalised conservative equation:

$$\frac{\partial u}{\partial t}+\frac{\partial f}{\partial x}=0 \tag{12}$$

In this case, we will solve on the periodic domain $[-1,1]$, for simplicity. Then the effect of solving this numerically is that we have some finite basis. Let us then set that the solution, u, may be constructed as some p^{th} order polynomial. In this case, we will use the Legendre basis:

$$u=\sum_{i=0}^{p}\tilde{u}_i\psi_i(x) \tag{13}$$

where ψ_m is the m^{th} order Legendre polynomial of the first kind. If the flux function is then $f=f(u^n)$ for $n\in\mathbb{N}$, then for the flux we get:

$$f=\sum_{i=0}^{np}\tilde{f}_i\psi_i(x) \tag{14}$$

However, as previously stated, the functional space of the numerical solver is limited to be p^{th} order. We wish to project the high order flux on to this space using an ℓ_2 projection, which is commonly used in Galerkin methods.[14] Starting by defining the projected flux as:

$$f^P=\sum_{i=0}^{p}\tilde{f}_i^P\psi_i(x) \tag{15}$$

we wish to minimise:

$$\int_{-1}^{1}\left(f-f^P\right)^2\mathrm{d}x \tag{16}$$

By using the modal presentation and the orthogonality of Legendre polynomial the optimal projection is then just the truncation of Eq. (14) to p^{th} order. We can now define the truncation error as:

$$e_T=f-f^P=\sum_{i=p+1}^{np}\tilde{f}_i\psi_i(x) \tag{17}$$

This is not the complete picture however, as in nodal methods such as FR, a Galerkin projection is not typically employed. Instead, Lagrange polynomials are used to form a polynomial approximation from point values. This may be presented in 1D as:

$$f^\delta=\sum_{i=0}^{p}f(x_i)l_i(x) \tag{18a}$$

$$l_i(x)=\prod_{\substack{j=0\\ j\neq i}}^{p}\frac{x-x_j}{x_i-x_j} \tag{18b}$$

Due to this representation f^δ and f^P are not strictly equal. We may then define a projection error as:

$$e_P=f^P-f^\delta=\sum_{i=0}^{p}\tilde{f}_i\psi_i-\sum_{i=0}^{p}f(x_i)l_i(x) \tag{19}$$

and hence:

$$f^\delta=f-e_T-e_P \tag{20}$$

Of the two components of the error, the truncation error is unavoidable due to the need to use a finite basis. The projection error term on the other hand is more problematic as it is this term that introduces polynomial aliasing into the solution. Furthermore, due to the dependence of e_P on the point location it is difficult to gain insight into general trends. Through Taylor's theorem though we can look at the effect of both e_T and e_P. Let us begin by defining the interpolation remainder as:

$$\mathcal{R}_pf=f-\mathcal{L}_pf=f-f^\delta \tag{21}$$

where $\mathcal{L}_p$ is a p^{th} order linear interpolation operator. Using Taylor's theorem it can then be stated that:[20]

$$\mathcal{R}_pf(\xi)=\frac{f^{(p+1)}(\epsilon)}{(p+1)!}\prod_{i=0}^{p}(\xi-\xi_i) \tag{22}$$

where we restrict $\xi \in [-1,1]$, ξ_i are the interpolation points, and $\epsilon \in [-1,1]$ is dependent on ξ.

As an example let us modify the Burger's equation by squaring the conserved variable, which leads to:

$$\frac{\partial u^2}{\partial t} + \frac{\partial u^4}{\partial \xi} = 0 \tag{23}$$

This gives the opportunity for information to be stored in two ways comparable to methods used for Euler's equations — namely, storing u or u^2 and forming the flux term by either squaring u^2 or raising u to the power of four. This gives two possible flux polynomials when transformed into the computational domain:

$$\hat{v}^2(\xi) = (\hat{u}^2)^2 = \hat{f}_2 = \sum_{i=0}^{2p} \tilde{f}_{2,i}\psi_i \tag{24}$$

$$\hat{v}^4(\xi) = \hat{u}^4 = \hat{f}_4 = \sum_{i=0}^{4p} \tilde{f}_{4,i}\psi_i \tag{25}$$

where we have defined v to be the variable that is stored. To understand how errors may then enter the solution we wish to understand the scaling of the remainder of the flux interpolation to a finite polynomial space of order p. The maximal norm can then be used to give an estimate as:

$$\|\mathcal{R}_p f\|_\infty \leqslant \frac{1}{(p+1)!}\|q_{p+1}\|_\infty \|f^{(p+1)}\|_\infty. \tag{26}$$

Here q_{p+1} is defined as:

$$q_{p+1} = (\xi - \xi_0)(\xi - \xi_1)\dots(\xi - \xi_p) \tag{27}$$

with ξ_i being the points at which the value of f is stored. Taking the domain to be $[-1,1]$ therefore $\|q_{p+1}\|_\infty \leqslant 2$. We now use Eqs. (24 & 25) to refine the remainder estimates, which requires a bounding value of $\|f^{(p+1)}\|_\infty$. Firstly, it is known that the maximum absolute value of a Legendre polynomial is at $\xi = \pm 1$ ($\xi \in [-1,1]$) and, due to the recursive definition of Legendre polynomials, the maximum value of the derivative is at $\xi = \pm 1$. If the value of a differentiated Legendre polynomial at ±1 is:

$$\frac{\mathrm{d}^m \psi_n(\pm 1)}{\mathrm{d}\xi^m} = \frac{(\pm 1)^{n-m}(n+m)!}{2^m m!(n-m)!} \tag{28}$$

A consequence is that, for a given set of differentiated Legendre polynomials, $\{\psi_n', \psi_n'', \dots, \psi_n^{(m)}\}$, the maximum value in this set is the edge value of the m^{th} derivative. Hence, a bound can be placed on $\|f^{(p+1)}\|_\infty$ using Eq. (28) and the maximum Legendre mode coefficient as:

$$\|f_2^{(p+1)}\|_\infty \leqslant \left[\frac{2(3p+1)!}{2^{p+1}(p+1)!(p-1)!}\right] \max_{i\in\{0\dots 2p\}} |\tilde{f}_{2,i}| \tag{29}$$

$$\|f_4^{(p+1)}\|_\infty \leqslant \left[\frac{2(5p+1)!}{2^{p+1}(p+1)!(3p-1)!}\right] \max_{i\in\{0\dots 4p\}} |\tilde{f}_{4,i}| \tag{30}$$

Hence, the interpolation remainder may be bounded as:

$$\|\mathcal{R}_p f_2\|_\infty \leqslant 4\left[\frac{(3p+1)!}{2^{p+1}(p-1)![(p+1)!]^2}\right] \max_{i\in\{0\dots 2p\}} |\tilde{f}_{2,i}| \tag{31}$$

$$\|\mathcal{R}_p f_4\|_\infty \leqslant 4\left[\frac{(5p+1)!}{2^{p+1}(3p-1)![(p+1)!]^2}\right] \max_{i\in\{0\dots 4p\}} |\tilde{f}_{4,i}| \tag{32}$$

It can be seen by inspection that:

$$\frac{(3p+1)!}{(p-1)!} \leqslant \frac{(5p+1)!}{(3p-1)!}, \quad \forall p \in \mathbb{N} \tag{33}$$

From this, there are two conclusions that can be drawn. Firstly, the interpolation remainder of f_4 will always be bigger than f_2. Secondly, the difference between the remainders will grow factorially fast as the order is increased. Therefore, higher order methods will be far more affected by this mechanism of error introduction.

By reconsidering Eq. (23) it is apparent that we are actually concerned with approximating the derivative of the flux. To calculate this the remainder may be differentiated to:

$$\mathcal{R}'_p f = \frac{\mathrm{d}f}{\mathrm{d}x} - \frac{\mathrm{d}}{\mathrm{d}x}\mathcal{L}_p f = \frac{\mathrm{d}}{\mathrm{d}x}\mathcal{R}_p f \tag{34}$$

Following this through to differentiate the Taylor theorem result of Eq. (22) we get:

$$\mathcal{R}'_p f(\xi) = \frac{f^{(p+1)}(\epsilon)}{(p+1)!}\frac{\mathrm{d}}{\mathrm{d}\xi}\prod_{i=0}^{p}(\xi-\xi_i), \quad \xi\in[-1{,}1] \tag{35}$$

Again finding the maximal norm:

$$\|\mathcal{R}'_p f\|_\infty = \frac{p}{(p+1)!}\max_{i\in\{0..p\}}(\|q_p^i\|_\infty)\|f^{(p+1)}\|_\infty \tag{36}$$

defining q_p^i as:

$$q_p^i = \prod_{j=0,j\neq i}^{p}(\xi-\xi_j) \tag{37}$$

If we then apply the results of Eqs. (31 & 32), it is demonstrated that the primary difference in the gradient remainder is a factor of p.

Also, within the FR algorithm, we require the interpolated values of the flux at the interfaces. This can be more straightforwardly calculated, *i.e.* $\mathcal{R}_p f(\pm 1)$.

$$\mathcal{R}_p f(\pm 1) = \frac{f^{(p+1)}(\epsilon)}{(p+1)!}\prod_{i=0}^{p}(\pm 1-\xi_i) \tag{38}$$

We will not consider the infinity norm in this case, as Eq. (38) gives sufficient details. The main feature of note is that the behaviour of this remainder is primarily influenced by the interpolation point locations. For example, if $\xi_0=-1$ and $\xi_p=1$, as in a Gauss–Lobatto quadrature, the aliasing error introduced through interpolation to the edges would be zero. However, for other reasons explored by Castonguay[21] and Roe[22] this is problematic in higher dimensions. By comparing the remainder due to differentiation and interface interpolation, it can be seen that the differentiation gives a remainder that is approximately p times bigger. This indicates that the error that will dominate is due to the differentiation of polynomials experiencing aliasing.

IV. Primitive and Conserved Variables

IV.A. Euler's Equations

We will begin by considering the 1D Euler's equations in the conservative form.

$$\frac{\partial \mathbf{Q}_c}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{Q}_c)}{\partial x} = 0 \tag{39}$$

for

$$\mathbf{Q}_c = \begin{bmatrix}\rho\\ \rho u\\ E\end{bmatrix}, \quad \text{and} \quad \mathbf{f}(\mathbf{Q}_c) = \begin{bmatrix}\rho u\\ \rho u^2+p\\ u(E+p)\end{bmatrix}. \tag{40}$$

The concern is what information should be stored between time steps, while still solving this equation, and considering the previous section what will the effect be on the flux function order.

IV.A.1. Conserved Variable Computation

In an implementation where the conserved variables are not stored directly, if the conserved form of Euler's equations is to be solved, then the conserved variables must be computed at some stage.

$$\mathbf{Q}_p \rightarrow \mathbf{Q}_c \tag{41a}$$

$$\begin{bmatrix}\rho\\ u\\ p\end{bmatrix} \rightarrow \begin{bmatrix}\rho\\ \rho u\\ \frac{p}{\gamma-1}+\frac{1}{2}\rho(u^2)\end{bmatrix} = \mathcal{O}\begin{bmatrix}\xi^p\\ \xi^{2p}\\ \xi^{3p}\end{bmatrix} \tag{41b}$$

This transformation is shown in Eq. (41). It should be clear that if $\mathbf{Q}_p$ is represented by a polynomial of order p, then the terms ρu, ρv, and ρw will be polynomials of order $2p$, while $u(E+p)$ will be of order $3p$. Therefore, the energy equation will be most impacted by truncation and aliasing and, furthermore, the spatial variation in $\mathbf{Q}_p$ will not have to be significant before truncation and aliasing occurs due to the high degree of the energy equation here.

IV.A.2. Inviscid Flux Computation

In most implementations seen by the authors, when the primitives are stored they are also subsequently used to form the flux, as opposed to using $\mathbf{Q}_c$. Therefore, the order of the flux formed from the primitives is:

$$\mathbf{Q}_p \rightarrow \mathbf{f} \tag{42a}$$

$$\begin{bmatrix}\rho\\ u\\ p\end{bmatrix} \rightarrow \begin{bmatrix}\rho u\\ \rho u^2+p\\ u(\frac{\gamma p}{\gamma-1}+\frac{1}{2}\rho(u^2))\end{bmatrix} = \mathcal{O}\begin{bmatrix}\xi^{2p}\\ \xi^{3p}\\ \xi^{4p}\end{bmatrix} \tag{42b}$$

If the conserved variables are used instead, we obtain:

$$\mathbf{Q}_c \rightarrow \mathbf{f} \tag{43a}$$

$$\begin{bmatrix}\rho\\ \rho u\\ E\end{bmatrix} \rightarrow \begin{bmatrix}(\rho u)\\ \frac{(3-\gamma)(\rho u)^2}{2\rho}+(\gamma-1)E\\ \frac{(\rho u)}{\rho}\left(\gamma E-\frac{1}{2}(\gamma-1)\frac{(\rho u)^2}{\rho}\right)\end{bmatrix} = \mathcal{O}\begin{bmatrix}\xi^p\\ \xi^{2p}/\xi^p\\ \xi^{3p}/\xi^{2p}\end{bmatrix} \tag{43b}$$

Here we have been somewhat careless with notation. It is intended for $\mathcal{O}(\xi^{2p}/\xi^p)$ to mean a $2p^{\text{th}}$ order polynomial divided by a p^{th} order polynomial. If the polynomial $1/\mathcal{O}(\xi^p)$ is then expanded about zero to form a series of monomials, the series is $\mathcal{O}(\xi^\infty)$. From this, we can see that in Eq. (43) we have avoided the ξ^{4p} term, but at the expense of dividing by ρ. This raises the question of whether this formulation is more accurate — specifically, is the convergence of the $1/\rho$ series sufficiently fast to reduce aliasing? It may be possible to extend the analysis of section III to include reciprocals using the work of Leslie,[23] however so far it has not been possible to provide a bound. Therefore, this effect will be investigated numerically. Importantly, though, this storage method avoids the error introduced through the conversion in Eq. (41).

Another option that will be explored is storing the conserved variables but with energy substituted for pressure, $\mathbf{Q}_{c+p}$. The reason being that in industrial codes pressure is used frequently and this option would reduce the work involved in converting an implementation. Hence, the conversion from $\mathbf{Q}_{c+p}$ to the flux, $\mathbf{f}$, is:

$$\mathbf{Q}_{c+p} \rightarrow \mathbf{f} \tag{44a}$$

$$\begin{bmatrix}\rho\\ \rho u\\ p\end{bmatrix} \rightarrow \begin{bmatrix}(\rho u)\\ \frac{(\rho u)^2}{\rho}+p\\ \frac{(\rho u)}{\rho}\left(\frac{\gamma p}{\gamma-1}+\frac{1}{2}\frac{(\rho u)^2}{\rho}\right)\end{bmatrix} = \mathcal{O}\begin{bmatrix}\xi^p\\ \xi^{2p}/\xi^p\\ \xi^{3p}/\xi^{2p}\end{bmatrix} \tag{44b}$$

This method will also require a conversion step to retrieve the conserved variables if Eqs. (39 & 40) are to be solved. This will then introduce aliasing of order:

$$\mathbf{Q}_{c+p} \rightarrow \mathbf{Q}_c \tag{45a}$$

$$\begin{bmatrix}\rho\\ \rho u\\ p\end{bmatrix} \rightarrow \begin{bmatrix}(\rho)\\ (\rho u)\\ \frac{p}{\gamma-1}+\frac{1}{2}\frac{(\rho u)^2}{\rho}\end{bmatrix} = \mathcal{O}\begin{bmatrix}\xi^p\\ \xi^p\\ \xi^{2p}/\xi^p\end{bmatrix} \tag{45b}$$

This method has the potential to reduce the aliasing in forming the conserved variables and flux, as there is no longer the ξ^{4p} that is present in Eq. (42). However, this is again dependent on the nature of $1/\rho$.

IV.B. Navier–Stokes Equations

To confront more complex problems of fluid dynamical relevance, it is essential to consider the Navier–Stokes equations, written in the 3D conservative form as:

$$\frac{\partial \mathbf{Q}_c}{\partial t}+\nabla\cdot\mathbf{F}(\mathbf{Q}_c,\nabla\mathbf{Q}_c)=0 \tag{46}$$

where

$$\nabla\cdot\mathbf{F}=(\mathbf{f}^{\mathrm{inv}}-\mathbf{f}^{\mathrm{vis}})_x+(\mathbf{g}^{\mathrm{inv}}-\mathbf{g}^{\mathrm{vis}})_y+(\mathbf{h}^{\mathrm{inv}}-\mathbf{h}^{\mathrm{vis}})_z \tag{47}$$

If we take the bulk viscosity, μ_b, to be zero, then $\mathbf{f}^{\mathrm{vis}}$ can be defined as:

$$\mu\begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx}+v\tau_{xy}+w\tau_{xz}+\frac{\kappa}{\mu}T_x \end{bmatrix}=\mu\begin{bmatrix} 0 \\ \frac{4}{3}u_x-\frac{2}{3}(v_y+w_z) \\ u_y+v_x \\ w_x+u_z \\ u(\frac{4}{3}u_x-\frac{2}{3}(v_y+w_z))+v(u_y+v_x)+w(w_x+u_z)+\frac{\kappa}{\mu}T_x \end{bmatrix} \tag{48}$$

with $\mathbf{g}^{\mathrm{vis}}$ and $\mathbf{h}^{\mathrm{vis}}$ similarly defined.

The importance of considering this equation is that — due to phenomena such as the energy cascade — in a method which does not suffer from polynomial aliasing, aliasing will arise in LES anyway, due to the partial resolution of vortical motions. Hence, for turbulent flows, any difference is likely to be more marked as truncation and polynomial aliasing tends to amplify the numerical aliasing.

Clearly for the case when primitive variables are stored, the gradients of the primitive can be directly calculated and used to form the viscous flux. However, when the conserved variables are stored there are two options available to form the gradients needed here: convert the conserved variables to the primitives and calculate the gradients needed directly:

$$\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}\rightarrow\begin{bmatrix} \rho \\ u \\ v \\ w \\ p \end{bmatrix}\rightarrow\begin{bmatrix} \rho_x & \dots \\ u_x & \dots \\ v_x & \dots \\ w_x & \dots \\ \frac{c_v}{\gamma-1}(\rho^{-1}p_x-\rho^{-2}p\rho_x) & \dots \end{bmatrix}; \tag{49}$$

Or calculate the gradient of the conserved variables and use the product rule to convert them to what is needed:

$$\begin{bmatrix} \rho_x & \rho_y & \rho_z \\ (\rho u)_x & (\rho u)_y & (\rho u)_z \\ (\rho v)_x & (\rho v)_y & (\rho v)_z \\ (\rho w)_x & (\rho w)_y & (\rho w)_z \\ E_x & E_y & E_z \end{bmatrix}\rightarrow\begin{bmatrix} \rho_x & \rho_y & \rho_z \\ u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \\ T_x & T_y & T_z \end{bmatrix}. \tag{50}$$

These two options can be more simply written as:

$$\mathbf{Q}_c\rightarrow\quad\mathbf{Q}_p\rightarrow\nabla\mathbf{Q}_p \tag{51}$$

$$\mathbf{Q}_c\rightarrow\nabla\mathbf{Q}_c\rightarrow\nabla\mathbf{Q}_p \tag{52}$$

where $\nabla\mathbf{Q}$ is the gradient of $\mathbf{Q}$. Here the final row of $\nabla\mathbf{Q}$ is the gradient of temperature, ∇T, which for convenience is incorporated into the calculation of the viscous flux.

The method for calculating the required gradients from the product rule, applied to the conserved variable gradient formulation, is:

$$\frac{1}{\rho}\begin{bmatrix} \rho\rho_x & \cdots \\ \big((\rho u)_x - \rho^{-1}(\rho u)\rho_x\big) & \cdots \\ \big((\rho v)_x - \rho^{-1}(\rho v)\rho_x\big) & \cdots \\ \big((\rho w)_x - \rho^{-1}(\rho w)\rho_x\big) & \cdots \\ \big(E_x - \rho^{-1}E\rho_x\big) - \big((\rho u)u_x + (\rho v)v_x + (\rho w)w_x\big) & \cdots \end{bmatrix} = \begin{bmatrix} \rho_x & \rho_y & \rho_z \\ u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \\ T_x & T_y & T_z \end{bmatrix} \tag{53}$$

The polynomial orders of this step are then:

$$\begin{gathered} \frac{1}{\rho}\begin{bmatrix} \rho\rho_x \\ \big((\rho u)_x - \rho^{-1}(\rho u)\rho_x\big) \\ \big((\rho v)_x - \rho^{-1}(\rho v)\rho_x\big) \\ \big((\rho w)_x - \rho^{-1}(\rho w)\rho_x\big) \\ \big(E_x - \rho^{-1}E\rho_x\big) - \big((\rho u)u_x + (\rho v)v_x + (\rho w)w_x\big) \end{bmatrix} = \\ \mathcal{O}\begin{bmatrix} \xi^{p-1}(\eta\zeta)^p \\ \xi^{p-1}(\eta\zeta)^p/(\xi\eta\zeta)^p + \xi^{2p-1}(\eta\zeta)^{2p}/(\xi\eta\zeta)^{2p} \\ \xi^{p-1}(\eta\zeta)^p/(\xi\eta\zeta)^p + \xi^{2p-1}(\eta\zeta)^{2p}/(\xi\eta\zeta)^{2p} \\ \xi^{p-1}(\eta\zeta)^p/(\xi\eta\zeta)^p + \xi^{2p-1}(\eta\zeta)^{2p}/(\xi\eta\zeta)^{2p} \\ \xi^{p-1}(\eta\zeta)^p/(\xi\eta\zeta)^p + \xi^{2p-1}(\eta\zeta)^{2p}/(\xi\eta\zeta)^{2p} + \xi^{2p-1}(\eta\zeta)^{2p}/(\xi\eta\zeta)^{p} \end{bmatrix} \end{gathered} \tag{54}$$

Again, it should be clear that the momentum and energy (rows 2-5) terms experience the most aliasing, although it is not clear what effect that the division will have on aliasing. However, it is likely that the decay rate of the infinite quotient series will be fast in most cases. This will be dependent though on the nature of the function ρ, especially when higher Mach number flows are considered with large spatial variations in the density field.

Table 1: Variable storage schemes to be compared

Type	Primary Storage	Stress Tensor Calculation
A	$\mathbf{Q}_p$	$\mathbf{Q}_p \rightarrow \nabla\mathbf{Q}_p$
B	$\mathbf{Q}_c$	$\mathbf{Q}_c \rightarrow \mathbf{Q}_p \rightarrow \nabla\mathbf{Q}_p$
C	$\mathbf{Q}_c$	$\mathbf{Q}_p \rightarrow \nabla\mathbf{Q}_c \rightarrow \nabla\mathbf{Q}_p$
D	$\mathbf{Q}_{c+p}$	$\mathbf{Q}_{c+p} \rightarrow \mathbf{Q}_p \rightarrow \nabla\mathbf{Q}_p$

Table 1 summaries the methods of variable storage and gradient calculation that will be investigated for the Navier–Stokes equations and Euler's equations (where applicable).

At this point, we wish to link the ideas presented in Section III with the methods of this section. It should be clear that in order for this form of error to be incorporated into the solution, at some stage interpolation or polynomial fitting must be used within the calculation. For FR, this arises when the gradient is calculated or the edge points are interpolated from the points inside the element. However, if only the nodal values are used, as is the case in second-order Finite Volume (FV) methods, then there is no mechanism by which this error mechanism can affect the solution. Take the example of converting primitive variables to conservative variables, and back again:

$$\mathbf{Q}_p \rightarrow \mathbf{Q}_c \rightarrow \mathbf{Q}'_p. \tag{55}$$

It should be apparent that beyond any rounding error introduced in the floating-point arithmetic, $\mathbf{Q}_p = \mathbf{Q}'_p$. Therefore, the means of variable storage will not affect FV but will affect any method that in some way interpolates or fits a polynomial. To test this claim a second order FV method was subjected to some investigations presented later and the difference was found to be of the order of machine accuracy.

V. Isentropic Convecting Vortex

To evaluate the impact of the changes suggested in Section IV we will begin by studying the effect on the error and total kinetic energy on the isentropic convecting vortex (ICV).[24] The ICV is of interest as it is an analytical solution to Euler's equations and hence allows for the error at a given time to be calculated. One problem that we are confronted with when using high order, the ICV, and a periodic domain, is that the solution is only guaranteed to be C^0 continuous. This can be understood by considering the initial condition:

$$\rho = \left(1 - \frac{(\gamma-1)\beta^2}{8\gamma\pi^2}\exp(1-r^2)\right)^{\frac{1}{\gamma-1}} \tag{56a}$$

$$u = u_0 + \frac{\beta}{2\pi}(y_0 - y)\exp\left(\frac{1-r^2}{2}\right) \tag{56b}$$

$$v = v_0 + \frac{\beta}{2\pi}(x - x_0)\exp\left(\frac{1-r^2}{2}\right) \tag{56c}$$

$$w = 0 \tag{56d}$$

$$p = \left(1 - \frac{(\gamma-1)\beta^2}{8\gamma\pi^2}\exp(1-r^2)\right)^{\frac{\gamma}{\gamma-1}} \tag{56e}$$

$$r^2 = (x - x_0)^2 + (y - y_0)^2 \tag{56f}$$

where u_0 and v_0 are the advective velocities and β is the vortex strength (typically $\beta = 5$ is used). To set the initial values for $\mathbf{Q}_p$ etc. the nodal values of the initial condition are used, as opposed to a Galerkin projection. From Eq. (56), it can be seen that as the distance r is increased the vortex slowly decays and, on a finite but periodic domain, this will lead to discontinuities in the gradient. This is a point that will be of importance later when reviewing results and was explored by Spiegel et al.[25] where some interesting plots of shear near the boundaries are presented.

The metrics that we will use to review the accuracy are the point averaged absolute error in the density:

$$e(t) = \frac{1}{N_p}\sum_{i=1}^{N_p}|\rho_i - \rho(\mathbf{x}_i,t)|_2 \tag{57}$$

and the total kinetic energy:

$$E_k(t) = \frac{1}{2|\mathbf{\Omega}|}\int_{\mathbf{\Omega}}\rho\mathbf{V}\cdot\mathbf{V}\mathrm{d}\mathbf{x} \tag{58}$$

where $|\mathbf{\Omega}|$ is the domain volume.

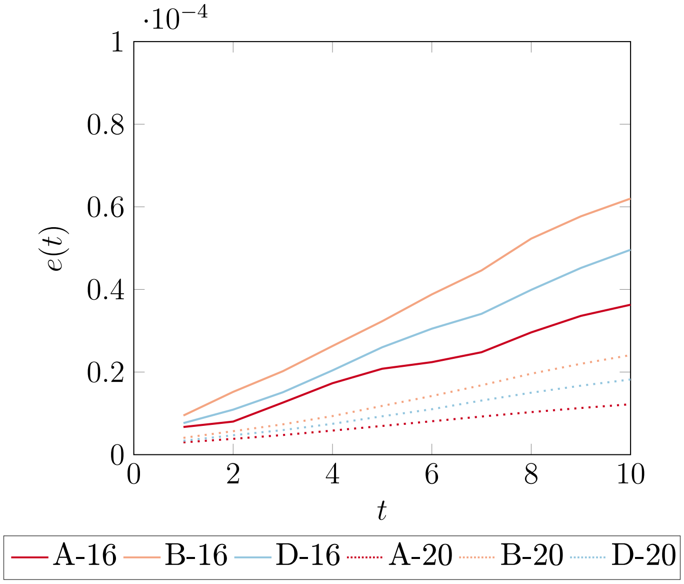
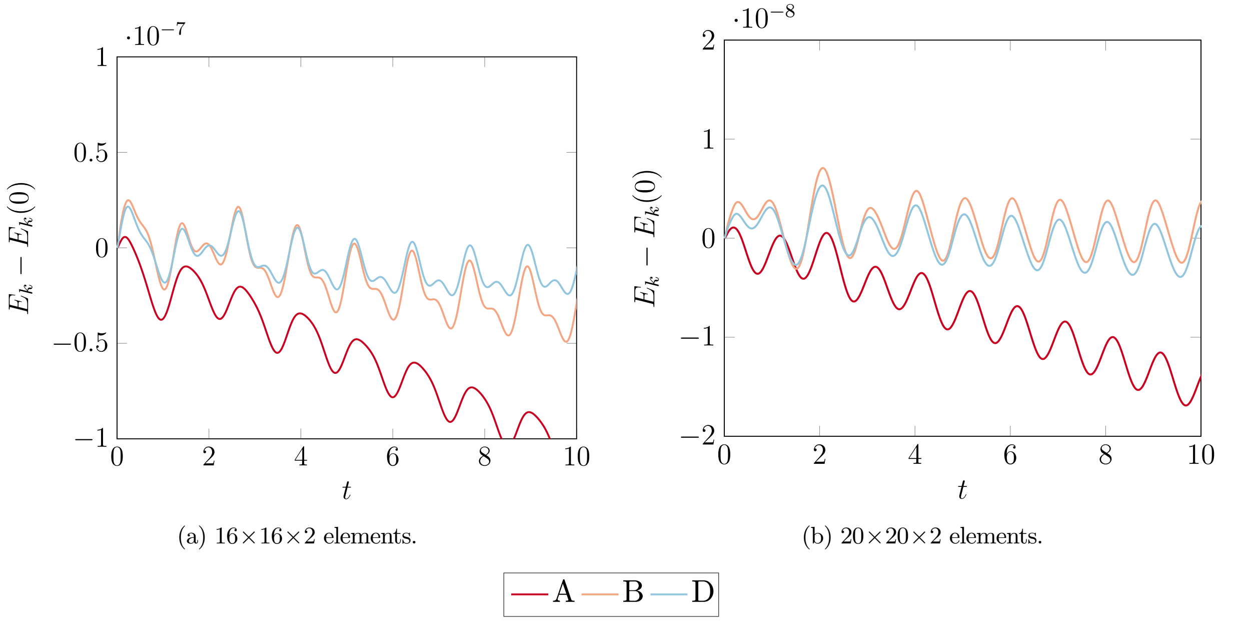


Figure 1: Variation of error in ICV density with time for FR, $p = 4$, using methods A, B and D on $16\times16\times2$ and $20\times20\times2$ element grids.

We begin by investigating the effect of storing the primitive variables (A), conservative variables (B), and conservative variables with energy substituted for pressure (D) on the error. This is shown in Fig. 1. Clearly, method A has the lowest levels of error followed by D then B and this ordering does not change as the grid is refined. This result makes clear that storing the pressure instead of energy can make a marked difference to the scheme. In schemes (B) and (D), as the stored variable-to-flux conversation is the same, the origin of the difference can be found to be from the conversion from stored variable to conserved variable that occurs, Eq. (45).

The result of the error increasing for (B) and (D) compared to (A) may be thought to be contrary to the expected outcome. However, to understand what is going on let us now consider how the kinetic energy changes with time.



(a) 16×16×2 elements.

(b) 20×20×2 elements.

Figure 2: Variation in total kinetic energy of the ICV, FR $p{=}4$, for two grid resolutions. using methods A, B, and D.

Figure 2 shows how the kinetic energy in the domain changes with time. For both grid resolutions, the rate of kinetic energy dissipation of A is higher than B and D, while B and D are similar. B and D show far less dissipation with the dissipation of method B being less than D in the higher resolution case, Fig. 2b. The mechanism responsible for this behaviour is that the reduction in the flux function order and the change to the stored to conservative conversions, reduces to the error introduced via truncation and projection. This in turn increases the resolution of the schemes (B) and (D) as higher wavenumbers can be resolved without causing excess error. To understand how this is reconciled with the somewhat contradictory results of Fig. 1 consider the ICV definition. The initial conditions defined earlier are formally C^0 continuous on a finite domain. Therefore, the lower dissipation exhibited by methods B and D leads to the errors introduced via the discontinuities in the gradient not being as damped as in the case of A. Hence, the error grows faster while also showing less dissipation.

VI. Taylor–Green Vortex

VI.A. Effect of Flux Function Order

If the various forms of variable storage are now applied to the full Navier–Stokes equations for a flow with turbulence, we can investigate in a more practical sense what the effect may be. The flow of choice for this is the canonical Taylor–Green vortex,[26,27] where the exact flow field used is defined in.[28–30] This case is chosen as not only is it a case for the Navier–Stokes equations, but it exhibits transition from an inviscid regime to a fully turbulent flow, via the mechanism of vortex stretching and shearing. This is key, as not only is it more representative of real engineering flows, but transition to turbulence will introduce the energy cascade to the flow and hence induce aliasing. We will go onto use this particular flow throughout this work and therefore some time will be devoted here to an explanation of its set-up and behaviour.

The initial condition is taken to be:

$$u = U_0 \sin\left(\frac{x}{L}\right)\cos\left(\frac{y}{L}\right)\cos\left(\frac{z}{L}\right) \tag{59a}$$

$$v = -U_0 \cos\left(\frac{x}{L}\right)\sin\left(\frac{y}{L}\right)\cos\left(\frac{z}{L}\right) \tag{59b}$$

$$w = 0 \tag{59c}$$

$$p = p_0 + \frac{\rho_0 U_0^2}{16}\left(\cos\left(\frac{2x}{L}\right)+\cos\left(\frac{2y}{L}\right)\right)\left(\cos\left(\frac{2z}{L}\right)+2\right) \tag{59d}$$

$$\rho = \frac{p}{RT_0} \tag{59e}$$

where we define the case by the non-dimensional parameters as:

$$R_e = \frac{\rho_0 U_0 L}{\mu}, \qquad P_r = 0.71 = \frac{\mu\gamma R}{\kappa(\gamma-1)}, \qquad M_a = \frac{U_0}{\sqrt{\gamma R T_0}} \tag{60}$$

with the free variables set as:

$$U_0 = 1, \qquad \rho_0 = 1, \qquad p_0 = 100, \qquad R = 1, \qquad \gamma = 1.4, \qquad L = 1 \tag{61}$$

This is then solved on a domain $\mathbf{\Omega} \in [-\pi, \pi]^3$ with periodic boundary conditions. Again the initial condition is set using the nodal values. From the above definitions, varying R_e and M_a allow for a series of different flow regimes to be explored, however first and foremost this case is intended to be incompressible case. Hence, if we will begin by using a Mach number that is typical in this case, M_a of approximately 0.08. Furthermore, it is known that for $R_e > \sim 500$ the turbulence exhibited is isotropic and homogeneous.[27]

The metrics that we will use to study the behaviour of the numerical method applied to the TGV are the rate of kinetic energy dissipation and enstrophy dissipation:

$$\epsilon_1 = -\frac{\mathrm{d}E_k}{\mathrm{d}t} = -\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{2\rho_0 U_0^2 |\mathbf{\Omega}|}\int_{\mathbf{\Omega}} \rho \mathbf{V}\cdot\mathbf{V}\mathrm{d}\mathbf{x}\right) \tag{62}$$

$$\epsilon_2 = \frac{\mu}{\rho_0^2 U_0^2 |\mathbf{\Omega}|}\int_{\mathbf{\Omega}} \rho(\boldsymbol{\omega}\cdot\boldsymbol{\omega})\mathrm{d}\mathbf{x} \tag{63}$$

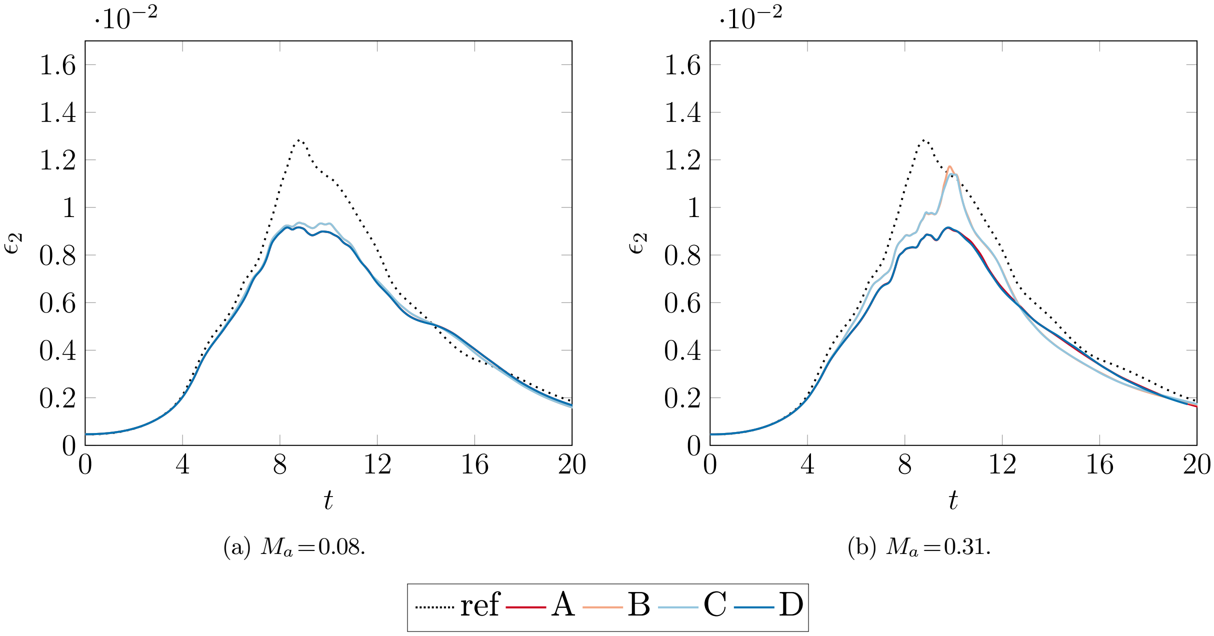
where $\boldsymbol{\omega}$ is the vector is vorticity, μ is the shear viscosity, and where ϵ_1 & ϵ_2 have been normalised. We will also make use of the enstrophy error term:

$$\mathcal{E} = \frac{\epsilon_2 - \epsilon_{2,\mathrm{ref}}}{\epsilon_{2,\mathrm{ref}}} \tag{64}$$

Flux function order is the primary focus of this work, and, as such, we wish to investigate if the different methods of variable storage impact the accuracy of the solution. As a result, there are two things which will be varied, the first of which is the Reynolds number. Three cases are investigated, with $R_e = 400$, 1600, and 3000, with DNS data (ref) available from Brachet et al.[27] Variation of the Reynolds numbers over this range will trigger a variety of different physics in this case, as will be discussed later. The second variable we propose changing is the Mach number, where values of $M_a = 0.08$ and 0.3 will be tested. The effect of compressibility on the TGV was investigated by Peng et al.[31] at various Mach numbers between 0.5 and 2, with 0.5 not being found to exhibit shocklets. Therefore, testing at $M_a = 0.3$ will test the introduction of errors due to larger spatial variations in ρ, but without introducing issues relating to shock capturing.

For the majority of the investigation, a 3D Navier–Stokes FR scheme will be used. The grid topology used will be hexahedral, constructed using a tensor product of the 1D FR scheme. More details on this construction of FR can be found in[4,21,32] including the extension to diffusion equations. The method of calculating the inviscid common interface flux chosen is Rusanov flux with Davis wave speeds.[33,34] The viscous common interface flux is found using Bassi and Rebay's BR1 scheme.[35,36] At the present we are not concerned with the associated effects of correction functions choice and, because of this, an FR correction function that recovers Nodal DG will be used.[12,14]

Let us first consider the TGV case when $R_e = 1600$ at Mach numbers 0.08 and 0.31. We will look to compare the four methods presented in Section IV for FR with $p = 4$ on a mesh with 80^3 degrees of freedom, the results of which are shown in Fig. 3. It is seen that when $M_a = 0.08$, Fig. 3a, there is a small improvement when storing



(a) $M_a = 0.08$.

(b) $M_a = 0.31$.

Figure 3: Enstrophy of the Taylor–Green Vortex with $R_e = 1600$, $p = 4$ and 80^3 degrees of freedom for storage methods A-D.

data as conserved variables over primitive variables. The results of the conservative variables with pressure instead of energy can be seen to be almost identical to the primitive variable results. It can also be noted that the largest difference is seen around the time of peak dissipation and not in the region $4 < t < 7$. This seems to indicate that the effect of changing the method of variable storage is to reduce the dissipation at the smallest scales, as the lower flux order again increases resolution. It is apparent that it does not introduce extra sources of dispersion which would cause over dissipation around $4 < t < 7$ when small scales begin to enter the flow.

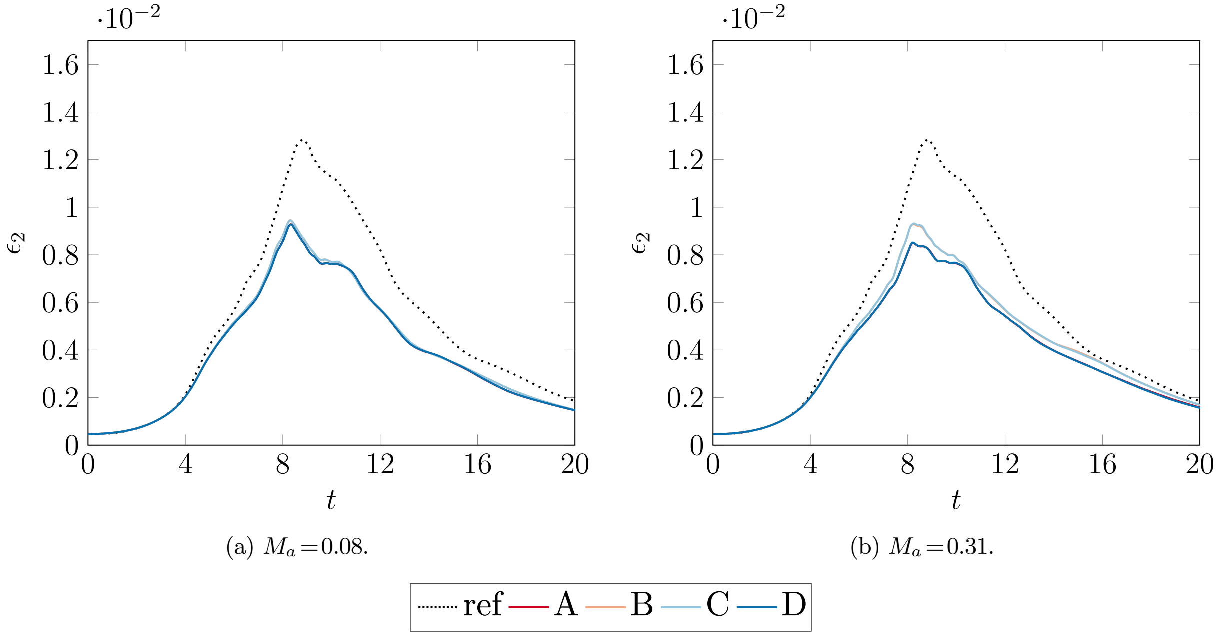
Moving onto the case when $M_a = 0.31$, the higher Mach number will introduce larger spatial variation in the density as the flow becomes more compressible. From the analysis in Section IV it is clear that the division by density with large spatial variations could lead to higher levels of error, predominantly through the truncation mechanism.

The enstrophy for the $M_a = 0.31$ case is shown in Fig. 3b and clearly shows a far larger change between the full conservative and the primitive methods. Again the cases of primitive and partial conservative with pressure are similar — this suggests that the improvement is largely originating from the change in the handling of the energy equation. Clearly the largest difference between the method happens for $8 < t < 12$. From DeBonis et al.,[28] we see that around this time energy has moved to the higher frequencies, but the $-5/3$ power law has not yet been established. This means that at this time the spatial variation of the variables is large. Therefore, at higher Mach numbers the error in schemes B and C has two competing components. The reduction due to the use of the momentum terms in the flux, and a potential increase due to spatial variation in the density.

At the higher Mach number, there is a noticeable difference in Fig. 3b between the fully conservative approaches with the gradient calculated from the converted primitives, and the gradient calculated from the application of the product rule. It is hard to attribute this difference to a particular aspect, this will be explored further at lower Reynolds numbers.

In Section III we showed analytically the dependency of interpolation rounding error on order and that it increases factorially as the polynomial order is increased. To investigate the effect of order we consider the case of $R_e = 1600$ run at $p = 3$ for the same number of degrees of freedom. The results of this are shown in Fig. 4. By comparison of Fig. 4a & 4b, we can see that there is still a larger difference between the methods in the high Mach number case than at low Mach number. However, when comparing Fig. 4 & 3, the difference between methods is markedly smaller at lower order. This evidence is in agreement with the earlier analytical predictions: as we move to a higher order, this mechanism of error becomes increasingly important.

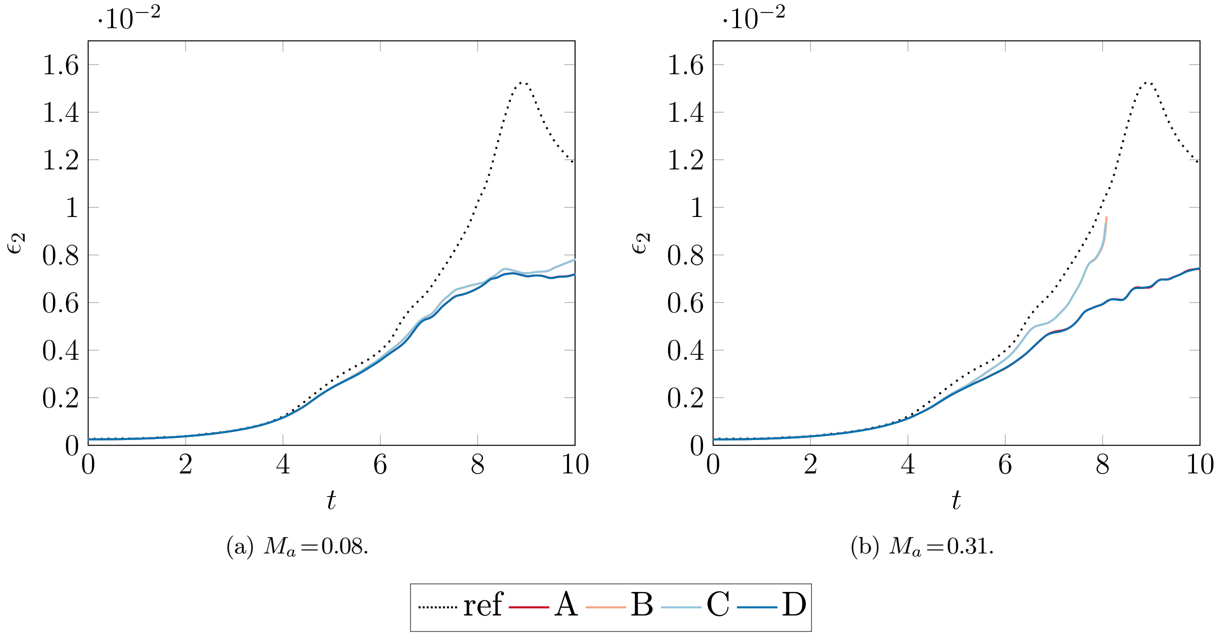
We will now explore the effect of increasing the Reynolds numbers for the same grid resolution. In particular, we choose $R_e = 3000$, which was explored with DNS by Brachet et al.[27] and with DG by Chapelier et al.[29] The results of the application of $p = 4$ FR with the various methods of storage are presented in Fig. 5. Firstly studying



(a) $M_a = 0.08$.

(b) $M_a = 0.31$.

Figure 4: Enstrophy of the Taylor–Green Vortex with $R_e = 1600$, $p = 3$ and 80^3 degrees of freedom.



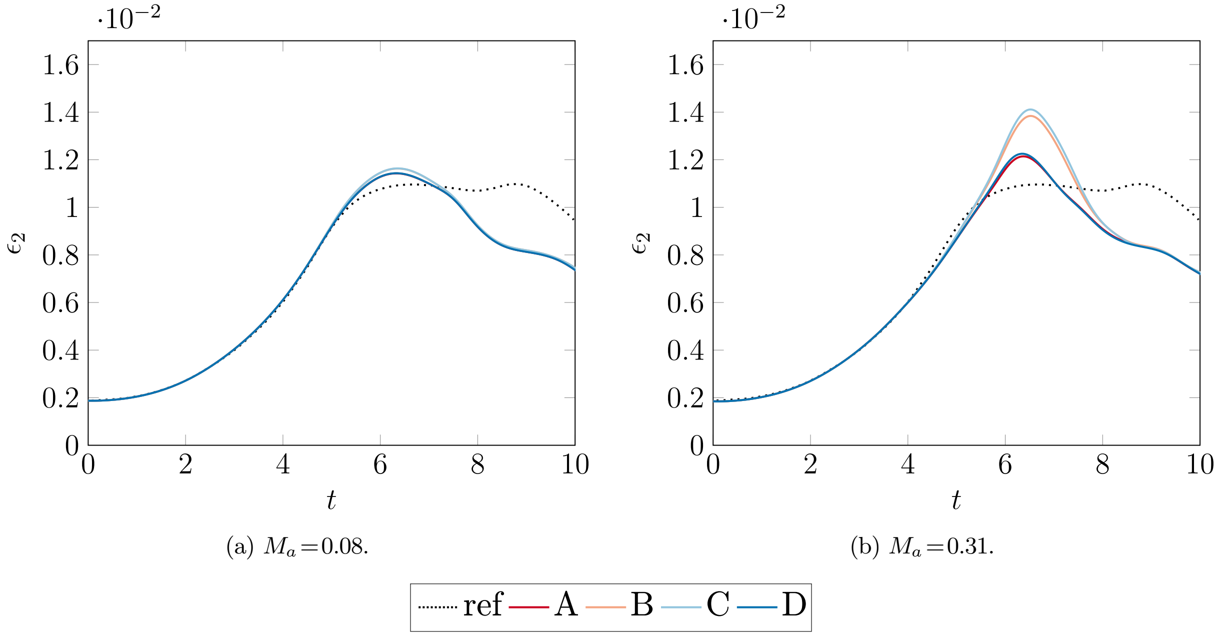
(a) $M_a = 0.08$.

(b) $M_a = 0.31$.

Figure 5: Enstrophy of the Taylor–Green Vortex with $R_e = 3000$, $p = 4$ and 80^3 degrees of freedom.

the $M_a = 0.08$ case, there is again a noticeable difference between the conservative and primitive enstrophy. This can be attributed to the decrease in numerical/aliasing based dissipation, due to the absence of over-dissipation when small scales begin to be generated and the increase in dissipation at the expected peak. This suggests that the small scales are being preserved for longer thus enabling their increased contribution to physical dissipation.

When the Mach number is increased to $M_a = 0.31$ we initially see a larger difference between the formulations, followed by the solution diverging. A similar divergence was observed by Chapelier et al.[29] when using DG on an under-resolved mesh, although they were solving the explicitly filtered LES equations. They attributed the divergence to insufficient numerical dissipation to stabilise the under-resolved grid. From the other results presented here, it has been shown that storing the conserved variables leads to reduction in dissipation at high wavenumbers due to the lower order of the flux function. Also, it is well-known that DG, particularly for non-linear problems, will require stabilisation through a de-aliasing method.[14] This appears to be the problem now confronted by methods B and C, and mitigations for FR have been investigated by Spiegel et al.[37] and for NDG by Hesthaven et al.[14]



(a) $M_a = 0.08$.

(b) $M_a = 0.31$.

Figure 6: Enstrophy of the Taylor–Green Vortex with $R_e = 400$, $p = 4$ and 40^3 degrees of freedom.

Finally for the storage methods, differences will be compared between methods B and C when applied to a case with larger magnitude viscous terms. For this we will consider a case at $R_e = 400$. Initially a grid with 80^3 degrees of freedom was used, however this resolution was found to give approximately DNS results. Because of this there was little to no content in the flow that was effected by truncation of polynomial aliasing and hence the degrees of freedom was reduced to 40^3. This give a cell Reynolds number of $R_{e,\text{cell}} = 50$ at $p = 4$, which is more indicative of LES. The results, shown in Fig. 6, indicate only a small change between the methods B and C. Due to overshoot at both Mach numbers, it is hard to discern if one method is better than another. However, as we are about to explore, there may be a computation saving of one method over another.

Table 2: Computation time comparison for one full RK44 explicit time step on a 8^3, p=4, mesh.

Type	Computation time (ms)	Speed-Up relative to A
A	6.423	1.000
B	5.832	1.101
C	5.634	1.140
D	6.757	0.951

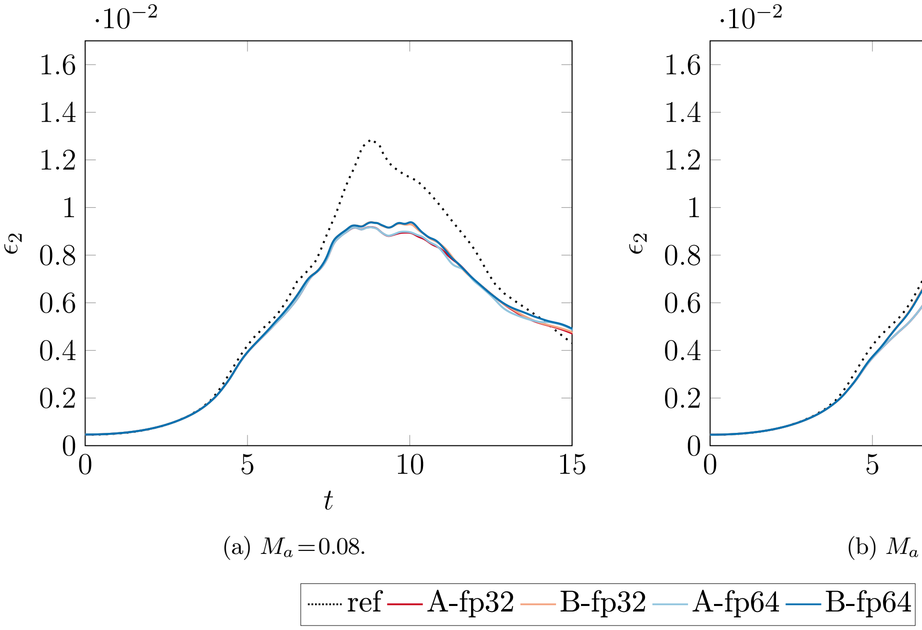
The different methods outlined in Section IV require differing numbers of floating-point operations, as some conversion steps or different numbers of multiplications are required to build quantities such as the flux terms. Therefore, we wish to understand what the impact on computational performance is, and to this end we profile the implementation. The implementation of FR used is an in house FR solver called Forflux, written in Fortran with CUDA Fortran and cuBLAS — both version 9.1 — for GPU acceleration. The implementation has been constructed such that all the data required for the FR algorithm is resident in the on-board GPU memory, hence the CPU plays little to no role in the computation. However, the CPU is required for inter-block communication, but the current implementation does not have support for MPI and so is only suitable for small cases.

The case profiled is a TGV, $p = 4$, with 8^3 elements run on a Titan Xp. Using the profiler, `pgprof`, the runtime for one complete explicit time step was found and is shown in Table 2.

It is clear that the continual conversion to or from the primitive variables has a noticeable impact on the computational time. In this case, the method that required the fewest number of conversions, method C (conservative variables using the product rule to calculate the gradient of the primitives), was the fastest. Method C gave a 12.3% reduction in computational time, which, all other things being equal, makes this a reasonable optimisation strategy to consider. Method D on the other hand, (conservative variables with E swapped for p), was slower as there are even more conversions required than in the base primitive method.

VI.B. Effect of Working Precision

Finally, we perform a brief numerical investigation on the impact of varying the working precision of the calculation when applied to turbulent and transitional flows. For this, we will limit our comparison to methods A and B, as it has previously been shown that the largest difference was between these two methods. The results of tests are shown in Fig. 7, where 32-bit floating-point (fp32) and 64-bit floating-point (fp64) precisions were used. Initially, the preprocessor ran to the same arthritic accuracy as the solver, but inaccuracies caused a loss of preservation, as such the preprocessor is run in fp64 while the working precision of the solver is varied.



(a) $M_a = 0.08$.

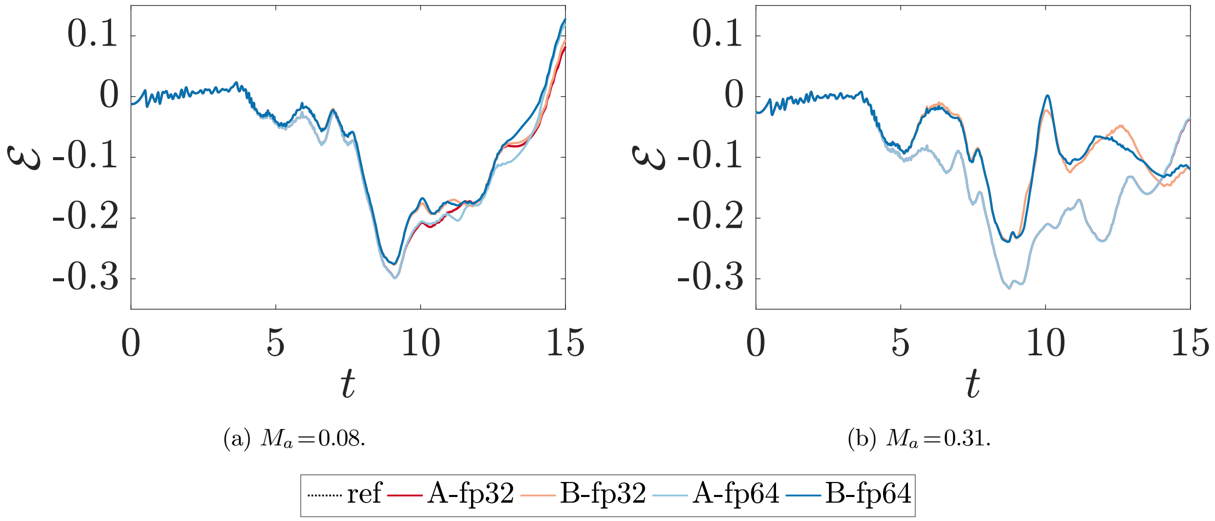
(b) $M_a = 0.31$.

Figure 7: Enstrophy of the Taylor–Green Vortex with $R_e = 1600$, $p = 4$ and 80^3 degrees of freedom for storage methods A and B in 32 (fp32) and 64 (fp64) bit precision.

The difference between precisions is made clearer in Fig. 8. It is apparent that the difference is most visible for $t > 10$. By this time the flow field has transitioned and is dominated by small scale vortex interaction. However, it is not solely due to high frequencies in the solution that the effect is more pronounced, but also due to the decay. As time proceeds, the range of solution reduces. For example the absolute velocity range goes from [0,1] to [0,0.52] from $t = 0$ to $t = 20$. Therefore, as the error in terms like the Jacobian remain constant, the precision error will have a greater impact when the solution range is smaller.

Figure 8 goes on to show that larger differences due to precision are experienced at lower Mach number. It is believed that this is because at lower Mach number the scheme is more sensitive to numerical aliasing occurring in the interpolation. As the Mach number is increased and the physics begins to exhibit non-constant ρ, the small floating-point errors in variables are more compatible with the physics and hence its effect appears to be lessened.

These results show that low Mach compressible flow are only negligibly affected by working precision when considering 32 and 64 bit precisions. However, in the calculation of global statistics care must be taken to the relative error of single precision compared to double. For example single precision has an epsilon of order 10^{-7} compared to 10^{-16}. This is the largest value where $(1 \pm \epsilon) = 1$ is true. Therefore, in calculating globally averaged properties care has to be taken as the accumulator can saturate. To mitigate this here, it was sufficient to have an accumulator over solution points in each cell and then a final accumulator over all cells in the domain.



(a) $M_a = 0.08$.

(b) $M_a = 0.31$.

Figure 8: Enstrophy error of the Taylor–Green Vortex with $R_e = 1600$, $p = 4$ and 80^3 degrees of freedom for storage methods A and B in 32 (fp32) and 64 (fp64) bit precision.

VII. Conclusions

Through this work, we have sought to challenge two received wisdoms in CFD. The first of these was the use of primitive variables to construct flux functions. We compared the use of primitives with conserved variables and showed numerically that a noticeable difference can be seen in methods that use reconstruction. Analysis using Taylor's theorem showed that this is primarily due to the flux function order — and as order increases, the difference found will become increasingly important.

An investigation into the effects of these choices on the method for constructing the viscous flux was also performed. When reconstructing from the conserved variables there are two potential methods, where either primitives are used as an intermediary or the product rule is used. Analysis showed a clear difference between the approaches, however in numerical investigations, these differences were limited due to the typically small magnitude of the viscous terms.

A final investigation was performed that aimed to challenge the dogma that double precision is important in CFD calculations. Numerical investigations on transitional flows showed that differences between single and double precision were negligible. However, differences become more noticeable over long-time integration, which can be attributed to two things. Firstly, that in single-precision error accumulation will become apparent more quickly under explicit temporal integration due to the larger relative error. Secondly, in this case, the range of the variable fields reduces with time due to dissipation, and so as time proceeds the absolute error from early on will get relatively larger. These points also highlight some key considerations when reducing the precision, due to the increased relative error of lower precision care has to be taken in accumulation. For example, when calculating globally averaged statistics or in statistics calculated through many operations.

Acknowledgements

The support of the Engineering and Physical Sciences Research Council of the United Kingdom is gratefully acknowledged under the award reference 1750012. The authors would also like to thank Nvidia for the GPU Seeding Grant received.

References

[1] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. Mavriplis, CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences, Tech. Rep. March, NASA (2014). `arXiv:arXiv:1011.1669v3`, `doi:10.1017/CBO9781107415324.004`.
URL `http://ntrs.nasa.gov/search.jsp?R=20140003093`

[2] M. A. Wahab, A Breif History of the ANSIS Package, in: Mechanics of Adhesives in Composite and Metal Joints: Finite

Element Analysis with ANSYS, 1st Edition, DEStech Publications, Lancaster, Pa, 2014, Ch. 3, pp. 59–60.

[3] G. E. Karniadakis, S. J. Sherwin, Spectral/hp Element Methods for Computational Fluid Dynamics, 1st Edition, Oxford University Press, Oxford, 2013.

[4] F. D. Witherden, A. M. Farrington, P. E. Vincent, PyFR: An Open Source Framework for Solving Advection-Diffusion Type Problems on Streaming Architectures Using the Flux Reconstruction Approach, Computer Physics Communications 185 (11) (2014) 3028–3040. arXiv:1312.1638, doi:10.1016/j.cpc.2014.07.011.
URL http://dx.doi.org/10.1016/j.cpc.2014.07.011

[5] R. S. Cant, SENGA2 User Guide, Tech. Rep. CUED/ATHERMO/TR67, Universit of Cambridge (2012).

[6] R. D. Moser, P. Moin, A. Leonard, A Spectral Numerical Method for the Navier–Stokes Equations with Applications to Taylor–Couette Flow, Journal of Computational Physics 52 (3) (1983) 524–544.

[7] G. A. Blaisdell, E. T. Spyropoulos, J. H. Qin, The Effect of the Formulation of Nonlinear Terms on Aliasing Errors in Spectral Methods, Applied Numerical Mathematics 21 (1996) 207–219.

[8] A. G. Kravchenko, P. Moin, On the Effect of Numerical Errors in Large Eddy Simulations of Turbulent Flows, Journal of Computational Physics 131 (2) (1997) 310–322. doi:10.1006/jcph.1996.5597.
URL http://linkinghub.elsevier.com/retrieve/pii/S0021999196955977

[9] A. R. Winters, R. C. Moura, G. Mengaldo, G. J. Gassner, S. Walch, J. Peiro, S. J. Sherwin, A Comparative Study on Polynomial Dealiasing and Split Form Discontinuous Galerkin Schemes for Under-Resolved Turbulence Computations, Journal of Computational Physics 372 (2018) 1–21. arXiv:1711.10180, doi:10.1016/j.jcp.2018.06.016.
URL https://doi.org/10.1016/j.jcp.2018.06.016

[10] H. Homann, J. Dreher, R. Grauer, Impact of the Floating-Point Precision and Interpolation Scheme on the Results of DNS of Turbulence by Pseudo-Spectral Codes, Computer Physics Communications 177 (2007) 560–565. doi:10.1016/j.cpc.2007.05.019.

[11] D. H. Bailey, High-Precision Floating-Point Arithmetic in Scientific Computing, Computing in Science and Engineering 7 (3) (2005) 54–61.

[12] H. T. Huynh, A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods, in: 18th AIAA Computational Fluid Dynamics Conference, Vol. 2007-4079, 2007, pp. 1–42. doi:10.2514/6.2007-4079.
URL http://arc.aiaa.org/doi/pdf/10.2514/6.2007-4079

[13] P. E. Vincent, P. Castonguay, A. Jameson, A New Class of High-Order Energy Stable Flux Reconstruction Schemes, Journal of Scientific Computing 47 (1) (2010) 50–72. doi:10.1007/s10915-010-9420-z.
URL http://link.springer.com/10.1007/s10915-010-9420-z

[14] J. S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, 1st Edition, Vol. 54 of Texts in Applied Mathematics, Springer New York, New York, NY, 2008. doi:10.1007/978-0-387-72067-8.
URL http://link.springer.com/10.1007/978-0-387-72067-8

[15] E. F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics - A Practical Introduction, 3rd Edition, Springer-Verlag Berlin Heidelberg, Dordrecht Berlin Heidelberg London New York, 2009.
URL http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Riemann+Solvers+and+Numerical+Methods+for+Fluid+Dynamics{#}1{%}5Cnhttp://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Riemann+solvers+and+numerical+methods+for+fluid+dynamics.+1999{%}230

[16] P. E. Vincent, P. Castonguay, A. Jameson, Insights From von Neumann Analysis of High-Order Flux Reconstruction Schemes, Journal of Computational Physics 230 (22) (2011) 8134–8154. doi:10.1016/j.jcp.2011.07.013.
URL http://dx.doi.org/10.1016/j.jcp.2011.07.013

[17] W. Trojak, Generalised Sobolev Stable Flux Reconstruction, Arxiv 1804 (04714) (2018) 1–19. arXiv:1804.04714.
URL http://arxiv.org/abs/1804.04714

[18] W. Trojak, Generalised Lebesgue Stable Flux Reconstruction, Arxiv 1805 (12481v2) (2018) 1–15. arXiv:1805.12481.
URL http://arxiv.org/abs/1805.12481

[19] W. Trojak, F. D. Witherden, A New Family of Weighted One-Parameter Flux Reconstruction Schemes (2018) 1–37arXiv:arXiv:1809.07846v1.

[20] R. Kress, Numerical Analysis, 1st Edition, Vol. 181 of Graduate Texts in Mathematics, Springer New York, New York, NY, 1998. doi:10.1007/978-1-4612-0599-9.
URL http://link.springer.com/10.1007/978-1-4612-0599-9

[21] P. Castonguay, P. E. Vincent, A. Jameson, Application of High-Order Energy Stable Flux Reconstruction Schemes to the Euler Equations, 49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 686 (January). doi:10.1007/s10915-011-9505-3.

[22] P. L. Roe, Is Discontinuous Reconstruction Really a Good Idea?, Journal of Scientific Computing 73 (2-3) (2017) 1094–1114. doi:10.1007/s10915-017-0555-z.

[23] R. A. Leslie, How Not to Repeatedly Differentiate a Reciprocal, The American Mathematical Monthly 98 (8) (1991) 732–735.

[24] C. W. Shu, Essentially Non-oscillatory and Weighted Essentially Non-oscillatory Schemes for Hyperbolic Conservation Laws Chi-Wang, in: A. Quarteroni, A. Dold, F. Takens, B. Teisser (Eds.), Advanced Numerical Approximation of Non-Linear Hyperbolic Equations, 1st Edition, Springer-Verlag, Berlin Heidelberg, 1997, Ch. 4, pp. 327–432. doi:10.1007/BFb0096351.

[25] S. C. Spiegel, H. T. Huynh, J. R. DeBonis, A Survey of the Isentropic Euler Vortex Problem using High-Order Methods, in: 22nd AIAA Computational Fluid Dynamics Conference, no. June, 2015, pp. 1–21. doi:10.2514/6.2015-2444.
URL http://arc.aiaa.org/doi/10.2514/6.2015-2444

[26] G. I. Taylor, A. E. Green, Mechanism of the Production of Small Eddies from Large Ones, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 158 (895) (1937) 499–521. arXiv:arXiv:1205.0516v2, doi:10.1098/rspa.1937.0036.
URL http://rspa.royalsocietypublishing.org/cgi/doi/10.1098/rspa.1937.0036

[27] M. E. Brachet, D. L. Merion, S. A. Orszag, B. G. Nickel, R. H. Morf, U. Frisch, Small-Scale Structure of the Taylor-Green Vortex, Journal of Fluid Mechanics 130 (1983) 411–452.

[28] J. R. DeBonis, Solutions of the Taylor-Green Vortex Problem Using High-Resolution Explicit Finite Difference Methods, 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition (February 2013). doi:10.2514/6.2013-382.
URL http://arc.aiaa.org/doi/10.2514/6.2013-382

[29] J.-B. Chapelier, M. De La Llave Plata, F. Renac, Inviscid and Viscous Simulations of the Taylor-Green Vortex Flow Using a Modal Discontinuous Galerkin Approach, in: 42nd AIAA Fluid Dynamics Conference and Exhibit, no. June, 2012, pp. 1–17. doi:10.2514/6.2012-3073.
URL http://arc.aiaa.org/doi/10.2514/6.2012-3073

[30] C. W. Shu, W. S. Don, D. Gottlieb, O. Schilling, L. Jameson, Numerical Convergence Study of Nearly Incompressible, Inviscid Taylor-Green Vortex Flow, Journal of Scientific Computing 24 (1) (2005) 569–595. doi:10.1007/s10915-004-5407-y.

[31] N. Peng, Y. Yang, Effects of the Mach Number on the Evolution of Vortex-Surface Fields in Compressible Taylor-Green Flows, Physical Review Fluids 3 (1) (2018) 1–21. doi:10.1103/PhysRevFluids.3.013401.

[32] D. M. Williams, A. Jameson, Energy Stable Flux Reconstruction Schemes for Advection-Diffusion Problems on Tetrahedra, Journal of Scientific Computing 59 (3) (2014) 721–759. doi:10.1007/s10915-013-9780-2.
URL http://dx.doi.org/10.1016/j.jcp.2013.05.007

[33] V. Rusanov, The Calculation of the Interaction of Non-Stationary Shock Waves with Barriers, Zh. Vychisl. Mat. Mat. Fiz. 1 (2) (1961) 267–279. arXiv:arXiv:1011.1669v3, doi:10.18287/0134-2452-2015-39-4-453-458.

[34] S. Davis, Simplified Second-order Godunov-type Methods, SIAM Journal on Scientific and Statistical Computing 9 (3) (1988) 445–473.

[35] F. Bassi, S. Rebay, A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible NavierStokes Equations, Journal of Computational Physics 131 (2) (1997) 267–279. doi:10.1006/jcph.1996.5572.
URL http://linkinghub.elsevier.com/retrieve/pii/S0021999196955722

[36] F. Bassi, S. Rebay, An Implicit High-Order Discountinuous Galerkin Method for the Steady State Compressible Navier-Stokes Equations, in: Computational Fluid Dynamics '98, John Wilery & Sons Ltd., Athens,Greece, 1998, pp. 1226–1233.

[37] S. C. Spiegel, H. T. Huynh, J. R. DeBonis, De-Aliasing through Over-Integration Applied to the Flux Reconstruction and Discontinuous Galerkin Methods, in: 22nd AIAA Computational Fluid Dynamics Conference, AIAA AVIATION Forum, (AIAA 2015-2744), Dallas (TX), 2015, pp. 1–22. doi:10.2514/6.2015-2744.

A. Nomenclature

Roman

A	scheme storing $\mathbf{Q}_p$ and $\nabla\mathbf{Q}_p$
B	scheme storing $\mathbf{Q}_c$ and $\nabla\mathbf{Q}_p$
C	scheme storing $\mathbf{Q}_c$ and $\nabla\mathbf{Q}_c$
D	scheme storing $\mathbf{Q}_{c+p}$ and $\nabla\mathbf{Q}_p$
e_a	aliasing error term
E_k	volume averaged kinetic energy
$\mathbf{f}^{\text{inv}}$,...	inviscid flux vector in x,...
$\mathbf{f}^{\text{vis}}$,...	viscous flux vector in x,...
h_L & h_R	left and right correction function
l_i	i^{th} Lagrange basis polynomial
M_a	Mach number
P_r	Prandlt number
q_p	non-normalised p^{th} order Lagrange basis
q_p^i	non-normalised $p-1^{\text{th}}$ order Lagrange basis formed from q_p excluding i^{th} term
$\mathbf{Q}_c$	conserved variables
$\mathbf{Q}_{c+p}$	conserved variables with E exchanged for p
$\mathbf{Q}_p$	primitive variables
$\nabla\mathbf{Q}_c$	gradient of conserved variables
$\nabla\mathbf{Q}_p$	gradient of primitive variables
R_e	Reynolds number
T	temperature
∇T	gradient of temperature
$\mathbf{V}$	vector of velocity components

Greek

β	Icentropic Convecting Vortex spread rate
$\Gamma_n(x)$	projection operator from real to reference domain, $\Gamma_n : \boldsymbol{\Omega}_n \mapsto \hat{\boldsymbol{\Omega}}$
ϵ_1	global averaged kinetic energy based dissipation, $-\mathrm{d}E_k/\mathrm{d}t$
ϵ_2	global enstrophy based dissipation
μ	dynamic viscosity
ξ	spatial variable in reference domain
τ_{xx},...	viscous stress tensor
ψ_i	i^{th} order Legendre polynomial of the first kind
$\boldsymbol{\omega}$	vorticity, $\nabla\times\mathbf{V}$
$\boldsymbol{\Omega}$	spatial domain
$\boldsymbol{\Omega}_n$	n^{th} spatial sub-domain
$\hat{\boldsymbol{\Omega}}$	reference domain

Superscript

$\bullet^{\delta}$	approximation of variable in sub-domain
$\bullet^{\delta C}$	correction to approximation of variable in sub-domain
$\bullet^{\delta D}$	discontinuous approximation of variable in sub-domain
$\bullet^{\delta I}$	common interface values based on approximation of variable in sub-domain
$\hat{\bullet}$	variable transformed into reference domain
$\tilde{\bullet}$	variable transformed into polynomial space
$\bullet^{(n)}$	n^{th} derivative of variable

Subscript

$\bullet_L$	variable at left interface
$\bullet_R$	variable at right interface
$\bullet_x$	differentiation of variable with respect to x

Other Symbols

$\mathcal{L}_n$	n^{th} order interpolation operator
$\mathbb{N}$	set of natural numbers, i.e positive non-zero integers
$\mathcal{O}$	big O notation of leading order in limiting behaviour
$\mathbb{R}$	set of real numbers
$\mathcal{R}_n$	n^{th} order interpolation remainder operator, i.e. $f - \mathcal{L}_n f$