A high-order discontinuous Galerkin Method using a mixture of Gauss-Legendre and Gauss-Lobatto quadratures for improved efficiency

Stéphanie Chaillat, Régis Cottereau, Ruben Sevilla

A high-order discontinuous Galerkin Method using a mixture of Gauss-Legendre and Gauss-Lobatto quadratures for improved efficiency

S. Chaillat[1], R. Cottereau[2], and R. Sevilla[3]

[1] Laboratoire POEMS, CNRS-ENSTA-INRIA, IP Paris, France.
[2] Aix-Marseille University, CNRS, Centrale Marseille, LMA UMR 7031, Marseille, France.
[3] Zienkiewicz Centre for Computational Engineering, Faculty of Science and Engineering, Swansea University, Bay Campus, Fabian Way, Crymlyn Burrows, SA1 8EN, Swansea, Wales, UK.

In discontinuous Galerkin spectral element methods (DGSEM), the two most common approaches to numerically integrate the terms of the weak form are either using Gauss-Legendre or Gauss-Lobatto quadratures. The former yields more accurate results but at a higher computational cost, so that a priori it is not clear whether one approach is more efficient that the other. In this paper, it is shown (theoretically for a particular case and numerically for the general case) that using Gauss-Lobatto quadrature for the convection matrix actually introduces a negligible error. In contrast, using Gauss-Lobatto quadratures for the evaluation of the jump term in the element faces introduces a sizeable error. This leads to the proposal of a new DG approach, where the convection matrix is evaluated using Gauss-Lobatto quadratures, whereas the face mass matrices are integrated using Gauss-Legendre quadratures. For elements with constant Jacobian and constant coefficients, a formal proof shows that no numerical integration error is actually introduced in the evaluation of the residual, even though both the mass and the convection matrices are not computed exactly with Gauss-Lobatto quadratures. For elements with non-constant Jacobian and/or non-constant coefficients, the impact of numerical integration error on the overall error is evaluated through a series of numerical tests, showing that this is also negligible. In addition, the computational cost associated to the matrix-vector products required to evaluate the residual is evaluated precisely for the different cases considered. The proposed approach is particularly attractive in the most general case, since the use of Gauss-Lobatto quadratures significantly speeds-up the evaluation of the residual.

Keywords discontinuous Galerkin method, spectral element method, high-order methods, wave propagation

1 Introduction

High-order discontinuous Galerkin (DG) methods (Bey et al. 1996; Hu et al. 1999; Warburton et al. 1999; Rasetarinera et al. 2000; Sherwin 2000; Houston et al. 2002; Ainsworth 2004a; Cockburn et al. 2005) have proven very efficient to approximate the solution of wave propagation over large complex media. They come in different flavours, depending in particular on the choice of quadrature for integration, Gauss-Legendre (GLegQ) or Gauss-Lobatto (GLobQ), on whether the interpolation and integrations points are the same or not, on whether the weak formulation is integrated by parts once or twice, and on the choice of flux that weakly enforces the continuity between elements.

When considering collocation approaches, using the nodes of either the GLegQ or the GLobQ, the mass matrix becomes diagonal and very efficient strategies arise in combination with time marching algorithms and fast matrix-vector products in the evaluation of the residual (Deville et al. 2002) (see also Section 3.2). At first, using the GLegQ seems like a better option in terms of accuracy because that quadrature can integrate exactly higher order polynomials. However, the fluxes then need to be interpolated on the faces (Canuto et al. 2007), which induces an additional cost at each time step to compute the residual (evaluated at 15% of the total simulation cost in Kopriva et al. (2010)). These competing effects make it difficult to know which method is better in terms of accuracy and cost, although most authors (Castel et al. 2009; Kopriva et al. 2010; Gassner et al. 2011) seem to (slightly) prefer GLegQ over GLobQ, in particular with arguments of stability (Castel et al. 2009; Gassner et al. 2011).

Another flavour of the high-order DG method is the so-called nodal version (Giraldo et al. 2002; Hesthaven et al. 2008b; Giraldo et al. 2008), where the interpolation points are the nodes of the GLobQ, but integration follows the GLegQ. In that version, a good accuracy is expected because integrals are accurately evaluated, but the cost of computing the residual at each time step is potentially high because the mass matrix is not diagonal. Here again, the combination of these advantages and defects makes it very difficult to draw a definite conclusion on the interest of this version in terms of accuracy and cost. In this paper, a comparison and an analysis is provided of the nodal and the GLobQ versions discussed above, where the GLobQ is used for both integration and interpolation. These two versions will be referred as the DG method with GLegQ and GLobQ respectively because the only difference lies in the quadrature formula used for numerical integration. The interpolation polynomials are the same in both approaches and defined using the nodes of the GLobQ.

The first objective of this paper consists of evaluating precisely the complexity of both DG methods in terms of operation count required to perform the matrix-vector products required when employing an explicit time marching scheme. Additionally, accuracy of both methods will be evaluated by using a series of numerical simulations of increasing complexity. The second objective of the paper is to propose a novel version of the high-order DG method, bridging the gap between the DG methods with GLegQ and GLobQ, where the convection part of the residual is integrated with the GLobQ, while the face jump term is evaluated with the GLegQ. It will be shown that this simple change allows to improve greatly the accuracy of the method, while retaining its numerical efficiency. In particular, we show analytically that in the case of a constant Jacobian, the use of GLobQ to integrate the convection part of the residual induces absolutely no error with respect to the GLegQ. In the non-constant Jacobian case, a series of numerical examples illustrate that this property is approximately conserved.

The remainder of the paper is organised as follows. Section 2 introduces the classical DG formulation for a linear conservation law, its computational complexity in the most general case for tensor-product elements and a brief discussion on the different choices to perform the numerical integration. Note that the discussion throughout the paper is limited to tensorial elements (quadrilaterals and hexaedra). Section 3 presents alternative formulations of the residual for the general case and for the particular case when constant coefficients and meshes with affine elements (i.e., constant Jacobian of the isoparametric mapping) are considered. It also provides the formulation of the residual for the DG method with GLobQ. In all cases the cost required to compute the matrix-vector products required during the explicit time marching process is detailed. In Section 4, the new DG method with mixed quadratures is proposed. First, a formal proof is provided showing that, under certain hypotheses, the contribution to the residual given by the convection term is identical if GLegQ or GLobQ are used. This motivates the introduction of the new method that bridges the gap between the DG methods with GLegQ and GLobQ. Section 5 shows a series of numerical examples to evaluate the accuracy of the DG approaches considered in this work. The examples involve problems with constant and non-constant velocity fields both in two and three dimensions. In addition, the use of Cartesian and non-Cartesian grids is also studied, to evaluate the influence of having elements with non-constant Jacobian of the isoparametric mapping. The discussion also considers the computational cost of the DG approaches considered. Finally, Section 6 presents the conclusions of the work.

2 DG formulation for conservation laws

This section introduces the DG formulation of a scalar conservation law and the spatial and temporal discretisations. The particular choice considered here of tensor product elements is recalled and the cost of evaluating the residual of the semi-discrete system is presented. Finally the different choices for the numerical integration of the terms appearing in the weak form are briefly discussed.

2.1 Model problem

Let us consider the general linear conservation law in a open bounded domain $\Omega \subset \mathbb{R}^{\mathrm{n_{sd}}}$ with boundary $\partial\Omega$, where $\mathrm{n_{sd}}$ denotes the number of spatial dimensions,

$$\frac{\partial u}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{f}(\boldsymbol{x}, u) = 0, \qquad \text{on } \Omega \times (0, T]. \tag{1}$$

where $u(\boldsymbol{x}, t)$ is the conserved variable, $\boldsymbol{f}(\boldsymbol{x}, u)$ is the hyperbolic flux vector and T is the final time.

It is often convenient to write the hyperbolic flux vector as

$$\boldsymbol{f}(\boldsymbol{x}, u) = \boldsymbol{a}(\boldsymbol{x})u, \tag{2}$$

where the velocity field vector $\boldsymbol{a} = (a_1, \dots, a_{\mathrm{n_{sd}}}) \in [\mathcal{L}^\infty(\Omega)]^{\mathrm{n_{sd}}}$ is assumed to be divergence free.

The boundary of the domain is assumed to be partitioned as $\partial\Omega = \overline{\Gamma}_{\mathrm{in}} \cup \overline{\Gamma}_{\mathrm{out}}$ where Γ_{in} and Γ_{out} are the inflow and outflow parts of the boundary respectively, defined as

$$\Gamma_{\mathrm{in}} = \{\boldsymbol{x} \in \partial\Omega \mid a_n(\boldsymbol{x}) < 0\}, \qquad \Gamma_{\mathrm{out}} = \{\boldsymbol{x} \in \partial\Omega \mid a_n(\boldsymbol{x}) > 0\}, \tag{3}$$

where $a_n(\boldsymbol{x}) = \boldsymbol{a}(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x})$ is the normal component of the velocity vector and $\boldsymbol{n}(\boldsymbol{x})$ is the outward unit normal vector at $\boldsymbol{x} \in \partial\Omega$.

As usual for hyperbolic problems, boundary conditions can only be imposed on the inflow part of the boundary (LeVeque 2002). Here a Dirichlet boundary condition is considered, namely

$$u(\boldsymbol{x}, t) = u_D(\boldsymbol{x}, t) \qquad \text{on } \Gamma_{\mathrm{in}} \times (0, T]. \tag{4}$$

Finally, the strong form of the problem must be completed with an initial condition

$$u(\boldsymbol{x}, t_0) = u_0(\boldsymbol{x}) \qquad \text{in } \Omega. \tag{5}$$

2.2 DG weak formulation

The domain is partitioned in $\mathrm{n_{el}}$ disjoint elements Ω_e such that

$$\overline{\Omega} = \bigcup_{e=1}^{\mathrm{n_{el}}} \overline{\Omega}_e. \tag{6}$$

The weak formulation for a generic element Ω_e is obtained after multiplying Eq. (1) by a test function, v and integrating in Ω_e

$$\int_{\Omega_e} v \frac{\partial u}{\partial t} d\Omega - \int_{\Omega_e} \boldsymbol{\nabla} v \cdot \boldsymbol{f}(\boldsymbol{x}, u) d\Omega + \int_{\partial\Omega_e} v \hat{f}_n(\boldsymbol{x}, u, u^{\mathrm{out}}) d\Gamma = 0, \tag{7}$$

where the integration by parts has been already performed on the divergence term and the boundary term features the numerical normal flux, $\hat{f}_n$.

A natural choice to define the numerical flux, for the linear hyperbolic equation considered here, is to employ a flux splitting technique (Hesthaven et al. 2008a). First, the physical normal flux $f_n(\boldsymbol{x}, u) = \boldsymbol{f}(\boldsymbol{x}, u) \cdot \boldsymbol{n}(\boldsymbol{x})$ is split as

$$f_n(\boldsymbol{x}, u) = f_n^-(\boldsymbol{x}, u) + f_n^+(\boldsymbol{x}, u), \tag{8}$$

where

$$f_n^-(\boldsymbol{x}, u) = a_n^-(\boldsymbol{x})u, \qquad f_n^+(\boldsymbol{x}, u) = a_n^+(\boldsymbol{x})u, \tag{9}$$

are the inflow and outflow normal fluxes, respectively, and

$$a_n^+(\boldsymbol{x}) := \frac{1}{2}\Big(a_n(\boldsymbol{x}) + |a_n(\boldsymbol{x})|\Big), \qquad a_n^-(\boldsymbol{x}) := \frac{1}{2}\Big(a_n(\boldsymbol{x}) - |a_n(\boldsymbol{x})|\Big). \tag{10}$$

The numerical normal flux, evaluated in terms of the trace of the solution on element Ω_e and the trace of the solution u^{out} on the neighbouring element, is then defined as

$$\hat{f}_n(\boldsymbol{x}, u, u^{\text{out}}) = f_n^+(\boldsymbol{x}, u) + f_n^-(\boldsymbol{x}, u^{\text{out}}) = a_n^+(\boldsymbol{x})u + a_n^-(\boldsymbol{x})u^{\text{out}}. \tag{11}$$

The DG weak formulation of Eq. (1) with flux splitting can be written as

$$\int_{\Omega_e} v\frac{\partial u}{\partial t} d\Omega + \int_{\Omega_e} v\left(\boldsymbol{a}(\boldsymbol{x}) \cdot \boldsymbol{\nabla} u\right) d\Omega + \int_{\partial\Omega_e} v a_n^-(\boldsymbol{x}) [\![u]\!] d\Gamma = 0. \tag{12}$$

where a second integration by parts has been performed after introducing the numerical normal flux and the operator $[\![u]\!] := u^{\text{out}} - u$ denotes the jump of the trace of the solution on the element boundary.

2.3 Spatial and temporal discretisations

The semi-discrete formulation is obtained by approximating the solution u as

$$u(\xi, t) \simeq u^h(\xi, t) = \sum_{J=1}^{\mathrm{n_{en}}} u_J(t) N_J(\xi) \tag{13}$$

on a reference element, with local coordinates $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_{\mathrm{n_{sd}}})$. Here $u_J(t)$ denotes the value of the solution at node $\boldsymbol{x}_J$ at time t, N_J is the shape function associated with the J-th node and $\mathrm{n_{en}}$ is the total number of nodes on the element. The shape functions considered here are standard Lagrange polynomials of order p, in each direction, defined on the nodes of the GLobQ.

The isoparametric mapping is employed to link the reference element $\hat{\Omega}$ and a physical element Ω_e, namely

$$\begin{aligned} \boldsymbol{\varphi} \ : \hat{\Omega} \subset \mathbb{R}^{\mathrm{n_{sd}}} &\longrightarrow \Omega_e \subset \mathbb{R}^{\mathrm{n_{sd}}} \\ \boldsymbol{\xi} &\longmapsto \boldsymbol{\varphi}(\boldsymbol{\xi}) := \sum_{J=1}^{\mathrm{n_{en}}} \boldsymbol{x}_J N_J(\boldsymbol{\xi}), \end{aligned} \tag{14}$$

where $\{\boldsymbol{x}_J\}_{J=1,\ldots,\mathrm{n_{en}}}$ are the nodal coordinates of the physical element Ω_e (Zienkiewicz et al. 2000).

Similarly, an isoparametric mapping is used to link the reference face $\hat{\Gamma}$ and a physical face $\Gamma_e^f \subset \partial\Omega_e$

$$\begin{aligned} \boldsymbol{\psi} \ : \hat{\Gamma} \subset \mathbb{R}^{\mathrm{n_{sd}}-1} &\longrightarrow \Gamma_e^f \subset \mathbb{R}^{\mathrm{n_{sd}}} \\ \boldsymbol{\eta} &\longmapsto \boldsymbol{\psi}(\boldsymbol{\eta}) := \sum_{J=1}^{\mathrm{n_{fn}}} \boldsymbol{x}_J^f N_J^f(\boldsymbol{\eta}), \end{aligned} \tag{15}$$

where $\{\boldsymbol{x}_J^f\}_{J=1,\ldots,\mathrm{n_{fn}}}$ are the nodal coordinates of the subset of nodes that belong to the physical face Γ_e^f and N_J^f is the shape function associated to J-th face node, defined in local coordinates $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_{\mathrm{n_{sd}}-1})$.

Introducing the approximation of the solution in the weak formulation of Eq. (12) and selecting the space of the weighting functions to be the same as the space of the interpolation functions, leads to the following system of ordinary differential equations (ODEs)

$$\mathbf{M}\frac{d\mathbf{U}}{dt} + \mathbf{C}\mathbf{U} + \sum_{f=1}^{\mathrm{n_{fa}^e}} \mathbf{M}^f [\![\mathbf{U}^f]\!] = \mathbf{0}, \tag{16}$$

where $\mathrm{n_{fa}^e}$ is the number of faces of the element Ω_e. It is worth noting that $\mathbf{U}^f$ is obtained by restricting $\mathbf{U}$ to the degrees of freedom associated to the nodes of the faces of Ω_e.

The elemental mass and convection matrices are given, respectively, by

$$\mathbf{M}_{IJ} = \int_{\hat{\Omega}} N_I N_J |\mathbf{J}| d\Omega, \qquad \mathbf{C}_{IJ} = \sum_{\ell=1}^{\mathrm{n_{sd}}} \int_{\hat{\Omega}} N_I G_\ell \frac{\partial N_J}{\partial \xi_\ell} d\Omega, \tag{17}$$

for $I, J = 1, \dots, \mathrm{n_{en}}$, where

$$G_\ell(\xi) = |\mathbf{J}(\xi)| \sum_{k=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\xi)\right) J^{-1}_{kl}(\xi) = \sum_{k=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\xi)\right) H_{kl}(\xi), \tag{18}$$

for $\ell = 1, \dots, \mathrm{n_{sd}}$, $\mathbf{J} = \partial\boldsymbol{\varphi}/\partial\boldsymbol{\xi}$ denotes the Jacobian of the isoparametric mapping and $\mathbf{H} = \mathrm{adj}(\mathbf{J})$ is the adjoint of $\mathbf{J}$.

Similarly, the face mass matrix is given by

$$\mathbf{M}^f_{IJ} = \begin{cases} \displaystyle\int_{\hat{\Gamma}} a_n^- N^f_I N^f_J \|\mathbf{J}^f\| d\Gamma, & \text{if } \boldsymbol{x}_I \in \Gamma^f_e \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

for $I = 1, \dots, \mathrm{n_{en}}$ and $J = 1, \dots, \mathrm{n_{fn}}$, where $\mathbf{J}^f$ is the Jacobian of the restriction of the isoparametric mapping to an element face. In two dimensions $\mathbf{J}^f = \partial\boldsymbol{\psi}/\partial\eta_1$ and in three dimensions $\mathbf{J}^f = \partial\boldsymbol{\psi}/\partial\eta_1 \times \partial\boldsymbol{\psi}/\partial\eta_2$.

Remark 1 The size of the mass and convection matrices is $\mathrm{n_{en}} \times \mathrm{n_{en}}$, whereas the face mass matrix $\mathbf{M}^f$ is defined as a rectangular matrix of size $\mathrm{n_{en}} \times \mathrm{n_{fn}}$ only to ensure consistency in Eq. (16). Therefore, when analysing the cost of the different approaches presented in this work (in Sections 2.5, 3.1.3 and 3.2.3), non necessary operations are not accounted for.

When the system of ODEs given by Eq. (16) is advanced in time using an explicit time marching algorithm, the scheme requires the solution of a linear system of equations element by element at each time step, avoiding the assembly and solution of a large sparse linear system in each time step. Here a classical fourth order explicit Runge-Kutta scheme is considered, namely

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \frac{\Delta t}{6}\left(\mathbf{R}^{(1)} + 2\mathbf{R}^{(2)} + 2\mathbf{R}^{(3)} + \mathbf{R}^{(4)}\right). \tag{20}$$

The four stages are given by

$$\begin{aligned} \mathbf{R}^{(1)} &= \mathbf{R}(\mathbf{U}^n, t^n) \\ \mathbf{R}^{(2)} &= \mathbf{R}(\mathbf{U}^n + \mathbf{R}^{(1)}\Delta t/2, t^n + \Delta t/2) \\ \mathbf{R}^{(3)} &= \mathbf{R}(\mathbf{U}^n + \mathbf{R}^{(2)}\Delta t/2, t^n + \Delta t/2) \\ \mathbf{R}^{(4)} &= \mathbf{R}(\mathbf{U}^n + \mathbf{R}^{(3)}\Delta t, t^n + \Delta t). \end{aligned} \tag{21}$$

where $\mathbf{R}$ denotes the residual of the system of ODEs

$$\mathbf{R} := -\left\{ \boldsymbol{C}\mathbf{U} + \sum_{f=1}^{\mathrm{n^e_{fa}}} \boldsymbol{\mathcal{M}}^f [\![\mathbf{U}^f]\!] \right\}. \tag{22}$$

where $\boldsymbol{C} := \mathbf{M}^{-1}\mathbf{C}$ and $\boldsymbol{\mathcal{M}}^f := \mathbf{M}^{-1}\mathbf{M}^f$.

This time marching algorithm is known to be conditionally stable, so the time step must be selected below the stability limit (Hairer et al. 2006). In all the numerical examples the time step is chosen small enough so that the error is dominated by the spatial discretisation. However, as the stability can influence the overall complexity by changing the number of time steps necessary to reach a final time, some comments and numerical tests will be provided in Section 5.

Remark 2 Although the discussion presented in this work is based on this particular fourth order explicit Runge-Kutta scheme, the conclusions of the paper apply to any explicit time integrator.

2.4 Tensorial basis and tensor products

This work focuses on tensor-product elements, namely quadrilateral and hexahedral elements in two and three dimensions, respectively. The reference element is taken as $\hat{\Omega} = [-1,1]^{\mathrm{n_{sd}}}$ and the reference face is $\hat{\Gamma} = [-1,1]^{\mathrm{n_{sd}}-1}$. Assuming for simplicity that the interpolation polynomials have the same order p in each direction, the number of element nodes is $\mathrm{n_{en}} = (p+1)^{\mathrm{n_{sd}}}$ and the number of face nodes are $\mathrm{n_{fn}} = (p+1)^{\mathrm{n_{sd}}-1}$, for each face. Throughout the document, lowercase (both for matrices and indices) denotes one dimensional quantities whereas uppercase is used for multi-dimensional quantities.

In this scenario, the shape functions $\{N_J\}_{J=1,\dots,\mathrm{n_{en}}}$ are defined as a product of one-dimensional shape functions, denoted by $\{n_j\}_{j=1,\dots,p+1}$. Using the lexicographic order, multi-dimensional shape functions are defined as

$$N_J(\boldsymbol{\xi}) = \prod_{k=1}^{\mathrm{n_{sd}}} n_{j_k}(\xi_k), \qquad \text{with} \qquad J = 1 + \sum_{k=1}^{\mathrm{n_{sd}}} (j_k - 1)(p+1)^{k-1}. \tag{23}$$

2.5 Operations count to evaluate the residual

Once the elemental matrix $\boldsymbol{C}$, of size $\mathrm{n_{en}} \times \mathrm{n_{en}}$ is computed for an element, the evaluation of the term of the residual in Eq. (22) associated to an element requires $\mathrm{n_{op}^e} = (2\mathrm{n_{en}} - 1)\mathrm{n_{en}}$ operations to perform the matrix-vector product. The leading term in this evaluation is thus of the order of $O(2\mathrm{n_{en}^2})$.

Similarly, once the matrix $\boldsymbol{\mathcal{M}}^f$ is computed for a face, the evaluation of the term of the residual in Eq. (22) associated to an internal face reduces to the evaluation of a matrix-vector product. Since each $\boldsymbol{\mathcal{M}}^f$ is a matrix of size $\mathrm{n_{en}} \times \mathrm{n_{fn}}$, the evaluation requires a total of $\mathrm{n_{op}^f} = (2\mathrm{n_{fn}} - 1)\mathrm{n_{en}}$ operations. The leading term in this evaluation is of the order of $O(2\mathrm{n_{fn}}\mathrm{n_{en}})$.

The overall cost of evaluating the residual, once the matrices $\boldsymbol{\mathcal{M}}^f$ and $\boldsymbol{C}$ are computed, is $\mathrm{n_{op}} = \mathrm{n_{el}}(2\mathrm{n_{en}} - 1)\mathrm{n_{en}} + \mathrm{n_{fa}}(2\mathrm{n_{fn}} - 1)\mathrm{n_{en}}$, where $\mathrm{n_{el}}$ is the number of mesh elements and $\mathrm{n_{fa}}$ is the total number of mesh edges or faces in two or three dimensions respectively.

For a regular mesh of tensor-product elements, i.e. quadrilateral or hexahedral elements, in two and three dimensions respectively, the number of mesh edges/faces can be written in terms of the total number of mesh elements, namely $\mathrm{n_{fa}} = \mathrm{n_{sd}}\mathrm{n_{el}}$, by assuming that the mesh is large enough so that the number of exterior faces can be neglected. This leads to the following expression for the total number of operations, only in terms of the number of elements, $\mathrm{n_{el}}$, and the degree of approximation, p,

$$\mathrm{n_{op}} = \mathrm{n_{el}}(p+1)^{\mathrm{n_{sd}}} \left[2(p + \mathrm{n_{sd}} + 1)(p+1)^{\mathrm{n_{sd}}-1} - \mathrm{n_{sd}} - 1 \right], \tag{24}$$

with leading term $O(2\mathrm{n_{el}}p^{2\mathrm{n_{sd}}})$. As the number of elements can be factorised, in the remainder of the paper the total number of operations per element will be considered. The number of operations required to compute the matrix-vector products in Eq. (22) for a tensor-product element are given in Table 1 for future reference.

	Operations	Leading term
$\mathrm{n_{op}^e}$	$[2(p+1)^{\mathrm{n_{sd}}} - 1]\,(p+1)^{\mathrm{n_{sd}}}$	$O(2p^{2\mathrm{n_{sd}}})$
$\mathrm{n_{op}^f}$	$\left[2(p+1)^{\mathrm{n_{sd}}-1} - 1\right](p+1)^{\mathrm{n_{sd}}}$	$O(2p^{2\mathrm{n_{sd}}-1})$

Table 1: Number of operations required (per element) to compute the matrix-vector products in Eq. (22) using the GLegQ for numerical integration.

Remark 3 This approach requires to store the matrices $\boldsymbol{C}$ and $\boldsymbol{\mathcal{M}}^f$. The cost to assemble these matrices and the memory requirements are thus both of the order of $O(\mathrm{n_{el}}p^{2\mathrm{n_{sd}}})$. Depending on the available resources, it can be interesting to save these matrices but that does not change the main complexity of the algorithm.

2.6 Choices of quadrature for integration

The computation of the elemental mass and convection matrices $\mathbf{M}$ and $\mathbf{C}$ in Eq. (17) is performed using a numerical quadrature defined on the reference element $\hat{\Omega}$, with $\mathrm{n_{ip}}$ points $\{\boldsymbol{\xi}_G\}_{G=1,\dots,\mathrm{n_{ip}}}$ and weights $\{\omega_G\}_{G=1,\dots,\mathrm{n_{ip}}}$, namely

$$\begin{aligned}\mathbf{M}_{IJ} &\simeq \sum_{G=1}^{\mathrm{n_{ip}}} N_I(\boldsymbol{\xi}_G)N_J(\boldsymbol{\xi}_G)|\mathbf{J}(\boldsymbol{\xi}_G)|\omega_G,\\ \mathbf{C}_{IJ} &\simeq \sum_{G=1}^{\mathrm{n_{ip}}} N_I(\boldsymbol{\xi}_G)\omega_G \sum_{\ell=1}^{\mathrm{n_{sd}}} G_\ell(\boldsymbol{\xi}_G)\frac{\partial N_J(\boldsymbol{\xi}_G)}{\partial \xi_\ell}.\end{aligned} \tag{25}$$

Similarly the face mass matrix $\mathbf{M}^f$ in Eq. (19) is computed by using a numerical quadrature defined on the reference face $\hat{\Gamma}$, with $\mathrm{n_{ip}^f}$ points $\{\boldsymbol{\eta}_G\}_{G=1,\dots,\mathrm{n_{ip}^f}}$ and weights $\{\varpi_G\}_{G=1,\dots,\mathrm{n_{ip}^f}}$, namely

$$\mathbf{M}^f_{IJ} \simeq \begin{cases} \displaystyle\sum_{G=1}^{\mathrm{n_{ip}^f}} a_n^-(\boldsymbol{\psi}(\boldsymbol{\eta}_G))N_I(\boldsymbol{\eta}_G)N_J(\boldsymbol{\eta}_G)\|\mathbf{J}^f(\boldsymbol{\eta}_G)\|\varpi_G & \text{if } \boldsymbol{x}_I \in \Gamma_e^f\\ 0 & \text{otherwise,}\end{cases} \tag{26}$$

In this work two numerical quadratures, namely Gauss-Legendre (GLegQ) and Gauss-Lobatto (GLobQ) quadratures, are considered and compared. For integration over the interval $[-1, 1]$, GLegQ are based on the roots of the Legendre polynomials, of order $p+1$, and they integrate exactly polynomials of order $2p+1$. In contrast, GLobQ are based on the extremes of the segment and the roots of the derivative of the Legendre polynomials, of order $p-1$, (for a total of $p+1$ nodes) and integrates exactly polynomials of order $2p-1$.

In general, i.e. even when the Jacobian is not constant, it is possible to select the number of integration points to integrate exactly the terms of the elemental mass matrix. For an approximation with polynomials of degree p, the entries of the mass matrix require the integration of a polynomial of degree $(\mathrm{n_{sd}}+2)p-1$, which can be computed exactly with a GLegQ with $2p-1$ points. The convection matrix can be computed exactly only if the convection velocity is a polynomial. For instance, if the convection velocity is a polynomial of degree q, the entries of the mass matrix require the integration of a polynomial of degree $(\mathrm{n_{sd}}+q+1)p-1$. Finally, the face mass matrix cannot be computed exactly in general due to the non-polynomial nature of the norm of $\mathbf{J}^f$ (Sevilla et al. 2011).

3 Alternative formulations of the residual

This section presents an efficient strategy to evaluate the residual of the ODE system of Eq. (22) for tensor-product elements. First, the particular case involving a constant velocity field and meshes with elements with constant Jacobian is considered. The more general case, where the velocity is not constant and/or the Jacobian of the isoparametric mapping is not constant is later discussed.

The main property used in the following is that the tensor product $\mathbf{C} = \mathbf{a} \otimes \mathbf{b}$ of two matrices of size $n_a \times m_a$ and $n_b \times m_b$ is a matrix of size $(n_a n_b) \times (m_a m_b)$ given by

$$C_{IJ} = a_{i_1 j_1} b_{i_2 j_2}, \qquad \text{with} \quad I = i_2 + (i_1 - 1)n_b, \quad J = j_2 + (j_1 - 1)m_b. \tag{27}$$

In addition, the derivations presented in this section use the following properties of matrix multiplication, assuming compatible sizes,

$$(\mathbf{a} \otimes \mathbf{b})(\mathbf{f} \otimes \mathbf{g}) = \mathbf{af} \otimes \mathbf{bg}, \tag{28}$$

and inversion, assuming compatible sizes and that $\mathbf{a}$ and $\mathbf{b}$ are invertible,

$$(\mathbf{a} \otimes \mathbf{b})^{-1}(\mathbf{f} \otimes \mathbf{g}) = \mathbf{a}^{-1}\mathbf{f} \otimes \mathbf{b}^{-1}\mathbf{g}. \tag{29}$$

These properties also apply to tensor products in higher dimensions (see Deville et al. 2002, for more details).

Finally, let us introduce the one dimensional mass and convection elemental matrices

$$\widehat{\mathbf{m}}_{ij} = \int_{-1}^{1} n_i(\xi) n_j(\xi) d\xi, \qquad \widehat{\mathbf{c}}_{ij} = \int_{-1}^{1} n_i(\xi) \frac{\partial n_j}{\partial \xi}(\xi) d\xi. \tag{30}$$

3.1 Case of constant Jacobian and velocity field

When the isoparametric mapping of Eq. (14) is affine, its Jacobian becomes constant for each element. If, additionally, the vector field $\boldsymbol{a}(\boldsymbol{x})$ is constant on the element, this can be used to accelerate the code substantially by adopting a quadrature-free implementation of the DG method (Atkins et al. 1998a; Sevilla et al. 2014). In this situation, the elemental matrices are computed as

$$\mathbf{M} = |\mathbf{J}|\widehat{\mathbf{M}}, \qquad \mathbf{C} = \sum_{l=1}^{\mathrm{n_{sd}}} G_\ell \widehat{\mathbf{C}}^\ell, \qquad \mathbf{M}^f = a_n^- \|\mathbf{J}^f\| \widehat{\mathbf{M}}^f, \tag{31}$$

where the functions G_ℓ, given in Eq. (18), become constant in each element. The elemental matrices

$$\widehat{\mathbf{M}}_{IJ} = \int_{\hat{\Omega}} N_I N_J d\Omega, \qquad \widehat{\mathbf{C}}^\ell_{IJ} = \int_{\hat{\Omega}} N_I \frac{\partial N_J}{\partial \xi_\ell} d\Omega, \tag{32}$$

for $I, J = 1, \ldots, \mathrm{n_{en}}$, are computed only once, in the reference element and stored. Similarly, the face matrix

$$\widehat{\mathbf{M}}^f_{IJ} = \begin{cases} \displaystyle\int_{\hat{\Gamma}} N_I^f N_J^f d\Gamma, & \text{if } \boldsymbol{x}_I \in \Gamma_e^f \\ 0 & \text{otherwise,} \end{cases} \tag{33}$$

for $I = 1, \ldots, \mathrm{n_{en}}$ and $J = 1, \ldots, \mathrm{n_{fn}}$ is computed only once, in the reference face and stored.

This strategy leads to a very efficient implementation of the DG method where, at each time step, the evaluation of the residual of the system of ODEs, given by Eq. (22), reduces to a matrix scaling and matrix-vector product operations. It is also worth noting that in this situation, it is possible to compute the integrals of all the matrices in Eq. (32) exactly by using a numerical quadrature of order $2p$, for instance using a GLegQ.

3.1.1 Residual evaluation for quadrilateral elements

Using the tensor product nature of the shape functions described in Section 2.4, the mass and convection matrices of the reference element given in Eq. (32) can be written as

$$\widehat{\mathbf{M}} = \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}}, \qquad \widehat{\mathbf{C}}^1 = \widehat{\mathbf{c}} \otimes \widehat{\mathbf{m}}, \qquad \widehat{\mathbf{C}}^2 = \widehat{\mathbf{m}} \otimes \widehat{\mathbf{c}}. \tag{34}$$

Using these tensorised expressions of the elemental mass and convection matrices, the contribution of the convection term to the residual in Eq. (22) can be evaluated as

$$\boldsymbol{C} = G_1 \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right) \otimes \mathbf{i} + G_2 \mathbf{i} \otimes \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right), \tag{35}$$

where $\mathbf{i}$ is the identity matrix of size $(p+1) \times (p+1)$.

The mass matrices corresponding to the four edges of a quadrilateral can be written as

$$\widehat{\mathbf{M}}^1 = \mathbf{e}_1 \otimes \widehat{\mathbf{m}}, \quad \widehat{\mathbf{M}}^2 = \mathbf{e}_2 \otimes \widehat{\mathbf{m}}, \quad \widehat{\mathbf{M}}^3 = \widehat{\mathbf{m}} \otimes \mathbf{e}_1, \quad \widehat{\mathbf{M}}^4 = \widehat{\mathbf{m}} \otimes \mathbf{e}_2, \tag{36}$$

where the vectors $\mathbf{e}_1 = [1, 0, \ldots, 0]^T$ and $\mathbf{e}_2 = [0, \ldots, 0, 1]^T$ are of dimension $p+1$.

Figure 1 shows the reference quadrilateral element with the assumed local numbering of the nodes for the particular case of $p = 2$. The numbering of the edges assumed is such that the first edge contains vertices 1 and 3, the second edge contains vertices 7 and 9, the third face contains vertices 1 and 7 and the fourth face contains vertices 3 and 9.

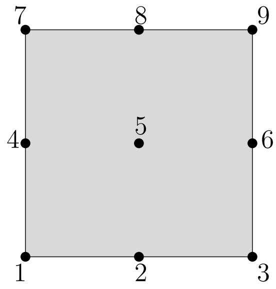


Figure 1: Reference quadrilateral element, showing the local numbering of the nodes for $p = 2$.

The contribution to the residual corresponding to the four edges of a quadrilateral can then be written in tensorised form as

$$\begin{aligned} \boldsymbol{\mathcal{M}}^1 &= a_n^- \frac{\|\mathbf{J}^1\|}{|\mathbf{J}|}(\widehat{\mathbf{m}}^{-1}\mathbf{e}_1) \otimes \mathbf{i}, \qquad & \boldsymbol{\mathcal{M}}^2 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}(\widehat{\mathbf{m}}^{-1}\mathbf{e}_2) \otimes \mathbf{i}, \\ \boldsymbol{\mathcal{M}}^3 &= a_n^- \frac{\|\mathbf{J}^3\|}{|\mathbf{J}|}\mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_1), \qquad & \boldsymbol{\mathcal{M}}^4 &= a_n^- \frac{\|\mathbf{J}^4\|}{|\mathbf{J}|}\mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_2), \end{aligned} \tag{37}$$

3.1.2 Residual evaluation for hexahedral elements

The tensorial decompositions described above generalize to hexahedral elements in 3D. The mass and convection matrices of the reference element given in Eq. (32) can be written as

$$\widehat{\mathbf{M}} = \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}}, \quad \widehat{\mathbf{C}}^1 = \widehat{\mathbf{c}} \otimes \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}}, \quad \widehat{\mathbf{C}}^2 = \widehat{\mathbf{m}} \otimes \widehat{\mathbf{c}} \otimes \widehat{\mathbf{m}}, \quad \widehat{\mathbf{C}}^3 = \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}} \otimes \widehat{\mathbf{c}}, \tag{38}$$

and the contribution of the convection term to the residual in Eq. (22) can be evaluated as

$$\boldsymbol{C} = G_1 \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right) \otimes \mathbf{i} \otimes \mathbf{i} + G_2 \mathbf{i} \otimes \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right) \otimes \mathbf{i} + G_3 \mathbf{i} \otimes \mathbf{i} \otimes \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right). \tag{39}$$

The mass matrices corresponding to the six faces of a hexahedral element can be written as

$$\begin{aligned} \widehat{\mathbf{M}}^1 &= \mathbf{e}_1 \otimes \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}}, \qquad & \widehat{\mathbf{M}}^2 &= \mathbf{e}_2 \otimes \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}}, \qquad & \widehat{\mathbf{M}}^3 &= \widehat{\mathbf{m}} \otimes \mathbf{e}_1 \otimes \widehat{\mathbf{m}} \\ \widehat{\mathbf{M}}^4 &= \widehat{\mathbf{m}} \otimes \mathbf{e}_2 \otimes \widehat{\mathbf{m}}, \qquad & \widehat{\mathbf{M}}^5 &= \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}} \otimes \mathbf{e}_1, \qquad & \widehat{\mathbf{M}}^6 &= \widehat{\mathbf{m}} \otimes \widehat{\mathbf{m}} \otimes \mathbf{e}_2, \end{aligned} \tag{40}$$

leading to the following tensorised expressions for the face contributions to the residual:

$$\begin{aligned} \boldsymbol{\mathcal{M}}^1 &= a_n^- \frac{\|\mathbf{J}^1\|}{|\mathbf{J}|}(\widehat{\mathbf{m}}^{-1}\mathbf{e}_1) \otimes \mathbf{i} \otimes \mathbf{i}, \qquad & \boldsymbol{\mathcal{M}}^2 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}(\widehat{\mathbf{m}}^{-1}\mathbf{e}_2) \otimes \mathbf{i} \otimes \mathbf{i}, \\ \boldsymbol{\mathcal{M}}^3 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}\mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_1) \otimes \mathbf{i}, \qquad & \boldsymbol{\mathcal{M}}^4 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}\mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_2) \otimes \mathbf{i}, \\ \boldsymbol{\mathcal{M}}^5 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}\mathbf{i} \otimes \mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_1), \qquad & \boldsymbol{\mathcal{M}}^6 &= a_n^- \frac{\|\mathbf{J}^2\|}{|\mathbf{J}|}\mathbf{i} \otimes \mathbf{i} \otimes (\widehat{\mathbf{m}}^{-1}\mathbf{e}_2). \end{aligned} \tag{41}$$

Figure 2 shows the reference hexahedral element with the assumed local numbering of the nodes for the particular case of $p = 2$. The numbering of the faces assumed is such that the first face contains vertices 1, 3, 9 and 7, the second face contains vertices 19, 21, 27 and 25, the third face contains vertices 7, 9, 27 and 25, the fourth face contains vertices 1, 3, 21 and 19, the fifth face contains vertices 1, 7, 25 and 19 and the sixth face contains vertices 3, 9, 27 and 21.

3.1.3 Operations count to evaluate the residual

Using the tensorisation proposed above, the cost to compute the residual term is re-evaluated. We recall that the cost to compute the action on a vector of the tensorised matrix $\mathbf{B} = \mathbf{i} \otimes \mathbf{a}$, where $\mathbf{a}$ is a matrix of size $n \times n$ and $\mathbf{i}$ is the identity matrix of size $m \times m$, is $mn(2n-1)$. We also recall that $\mathbf{i} \otimes \mathbf{i}$ is just the identity matrix of size $m^2 \times m^2$, and that the action of $\mathbf{i} \otimes \mathbf{a} \otimes \mathbf{i}$ has the same complexity as the action of $\mathbf{i} \otimes \mathbf{i} \otimes \mathbf{a}$ on that same vector.

It follows that, once the elemental matrices $G_\ell \left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right)$, of size $(p+1) \times (p+1)$, are computed for an element, the evaluation of the term of the residual in Eq. (22) associated to such an

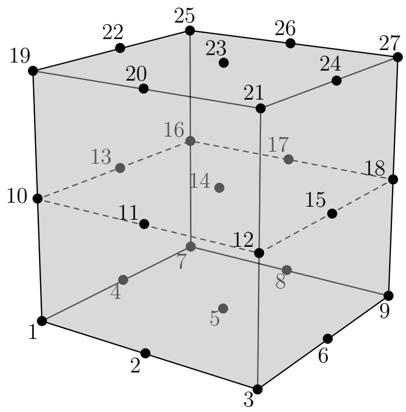


Figure 2: Reference hexahedral element, showing the local numbering of the nodes for $p = 2$.

elemental matrix requires $(p+1)^{\mathrm{n_{sd}}}[2(p+1)-1]$ operations to perform the matrix-vector product. There are $\mathrm{n_{sd}}$ such matrix-vector products to perform leading to $\mathrm{n}^{\mathrm{e}}_{\mathrm{op}} = \mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}[2(p+1)-1]$. The leading term in this evaluation is thus of the order of $O(2\mathrm{n_{sd}}p^{\mathrm{n_{sd}}+1})$.

Considering vectors now, the general cost to compute the action on a vector of the tensorised matrix $\mathbf{B} = \mathbf{i} \otimes \mathbf{w}$, where $\mathbf{w}$ is a vector of size n and $\mathbf{i}$ is still the identity matrix of size $m \times m$, is mn. The cost to evaluate the term of the residual in Eq. (22) associated to an internal face now reduces to the application of $\mathrm{n_{sd}}$ tensorised matrices per element (as explained in Section 2.5). The vector $\left(\widehat{\mathbf{m}}^{-1}\mathbf{e}_i\right)$ of size $(p+1)$ is tensorised with an identity matrix of size $(p+1)^{\mathrm{n_{sd}}-1} \times (p+1)^{\mathrm{n_{sd}}-1}$, so the cost to apply a matrix $\boldsymbol{\mathcal{M}}^i$ is thus $\mathrm{n}^{\mathrm{f}}_{\mathrm{op}} = (p+1)^{\mathrm{n_{sd}}}$.

These numbers of operations are summarized in Table 2.

	Operations	Leading term
$\mathrm{n}^{\mathrm{e}}_{\mathrm{op}}$	$\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}[2(p+1)-1]$	$O(2\mathrm{n_{sd}}p^{\mathrm{n_{sd}}+1})$
$\mathrm{n}^{\mathrm{f}}_{\mathrm{op}}$	$\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}$	$O(\mathrm{n_{sd}}p^{\mathrm{n_{sd}}})$

Table 2: Number of operations required (per element) to compute the matrix-vector products in Eq. (22) using the GLegQ for numerical integration, assuming a constant Jacobian and a constant velocity field.

Remark 4 The use of a tensorized version of the operators drastically reduces the memory requirements. It is thus necessary in this context to save only the the $\mathrm{n_{sd}}$ elemental matrices $G_\ell\left(\widehat{\mathbf{m}}^{-1}\widehat{\mathbf{c}}\right)$ of size $(p+1) \times (p+1)$ and the $\mathrm{n_{sd}}$ vectors $\left(\widehat{\mathbf{m}}^{-1}\mathbf{e}_i\right)$ of size $(p+1)$. The total memory requirements are thus of the order of $O(\mathrm{n_{e1}}p^2)$ advocating in favor of an approach where these matrices are precomputed.

3.2 General case using Gauss-Lobatto quadratures

This sections considers the general case, where the Jacobian and/or the velocity field are not constant. Despite the tensorised expressions obtained in the previous section are no longer valid in the general case, it is still possible to obtain efficient formulas for estimating the residual, but only when using the GLobQ for numerical integration. In order to simplify the comparisons in the next section, the mass, convection, and face mass matrices will be denoted by $\mathbb{M}$, $\mathbb{C}$ and $\mathbb{M}^f$, respectively, when they have been integrated using GLobQ.

Using the delta Kronecker property of the shape functions, that is $N_I(\xi_J) = \delta_{IJ}$, along with the GLobQ numerical quadrature, simplified expressions for the elemental and face matrices in Eqs. (25) and (26) can be obtained. Indeed, the elemental mass matrix can be written as

$$\mathbb{M}_{IJ} \simeq \sum_{G=1}^{\mathrm{n_{ip}}} N_I(\xi_G)N_J(\xi_G)|\mathbf{J}(\xi_G)|\omega_G = \delta_{IJ}|\mathbf{J}(\xi_I)|\omega_I, \tag{42}$$

with the trivial expression for the inverse

$$\mathbb{M}_{IJ}^{-1} = \frac{\delta_{IJ}}{|\mathbf{J}(\boldsymbol{\xi}_I)|\omega_I}. \tag{43}$$

Similarly, the elemental convection matrix can be written as

$$\mathbb{C}_{IJ} \simeq \sum_{G=1}^{\mathrm{n_{ip}}} N_I(\boldsymbol{\xi}_G)\omega_G \sum_{\ell=1}^{\mathrm{n_{sd}}} G_\ell(\boldsymbol{\xi}_G)\frac{\partial N_J(\boldsymbol{\xi}_G)}{\partial \xi_\ell} = \sum_{\ell=1}^{\mathrm{n_{sd}}} G_\ell(\boldsymbol{\xi}_I)\frac{\partial N_J}{\partial \xi_\ell}(\boldsymbol{\xi}_I)\omega_I. \tag{44}$$

Finally, the face mass matrix can be written as

$$\mathbb{M}_{IJ}^f \simeq \begin{cases} \displaystyle\sum_{G=1}^{\mathrm{n_{ip}^f}} a_n^-(\boldsymbol{\zeta}_G)N_I(\boldsymbol{\zeta}_G)N_J(\boldsymbol{\zeta}_G)\|\mathbf{J}^f(\boldsymbol{\zeta}_G)\|\varpi_G = \delta_{IJ}a_n^-(\boldsymbol{\zeta}_I)\|\mathbf{J}^f(\boldsymbol{\zeta}_I)\|\varpi_I & \text{if } \boldsymbol{x}_I \in \Gamma_e^f \\ 0 & \text{otherwise.} \end{cases} \tag{45}$$

3.2.1 Residual evaluation for quadrilateral elements

Observing in the 2D case that

$$\frac{\partial N_J}{\partial \xi_1}(\boldsymbol{\xi}_I^*) = n'_{j_1}(\xi_{i_1})n_{j_2}(\xi_{i_2}), \quad \frac{\partial N_J}{\partial \xi_2}(\boldsymbol{\xi}_I^*) = n_{j_1}(\xi_{i_1})n'_{j_2}(\xi_{i_2}), \quad \omega_I = \omega_{i_1}\omega_{i_2}, \tag{46}$$

the matrices $\mathbf{n}_{\ell j} := \omega_\ell n_j(\xi_\ell)$ and $\mathbf{d}_{\ell j} := \omega_\ell n'_j(\xi_\ell)$, of size $(p+1)\times(p+1)$ are introduced. These matrices are used to define the 2D derivative matrices, namely

$$\mathbf{D}_1 := \mathbf{d}\otimes\mathbf{n}, \quad \mathbf{D}_2 := \mathbf{n}\otimes\mathbf{d}. \tag{47}$$

Introducing the diagonal matrix $\hat{\mathbf{G}}_\ell$ (of size $\mathrm{n_{en}}\times\mathrm{n_{en}}$), whose elements are $G_\ell(\boldsymbol{\xi}_I)$, the convection matrix can then be computed as

$$\mathbb{C} = \hat{\mathbf{G}}_1(\mathbf{d}\otimes\mathbf{n}) + \hat{\mathbf{G}}_2(\mathbf{n}\otimes\mathbf{d}), \tag{48}$$

and

$$\mathbb{M}^{-1}\mathbb{C} = \mathbb{M}^{-1}\hat{\mathbf{G}}_1(\mathbf{d}\otimes\mathbf{n}) + \mathbb{M}^{-1}\hat{\mathbf{G}}_2(\mathbf{n}\otimes\mathbf{d}), \tag{49}$$

where $\mathbb{M}^{-1}$ is the diagonal matrix defined in Eq. (43). It is worth noting that each term in the sum above is the product of a diagonal matrix ($\mathbb{M}^{-1}\hat{\mathbf{G}}_i$) by a matrix that has a tensorised form.

For the evaluation of the part of the residual related to edges, the following diagonal formula is obtained

$$\mathbb{M}^{-1}\mathbb{M}_{IJ}^f \simeq \begin{cases} \delta_{IJ}a_n^-(\boldsymbol{\zeta}_I)\dfrac{\|\mathbf{J}^f(\boldsymbol{\zeta}_I)\|\varpi_I}{|\mathbf{J}(\boldsymbol{\xi}_I)|\omega_I} & \text{if } \boldsymbol{x}_I \in \Gamma_e^f \\ 0 & \text{otherwise.} \end{cases} \tag{50}$$

3.2.2 Residual evaluation for hexahedral elements

The decompositions described above generalize to 3D tensor-product elements in the following way for the convection matrix,

$$\mathbb{C} = \hat{\mathbf{G}}_1(\mathbf{d}\otimes\mathbf{n}\otimes\mathbf{n}) + \hat{\mathbf{G}}_2(\mathbf{n}\otimes\mathbf{d}\otimes\mathbf{n}) + \hat{\mathbf{G}}_3(\mathbf{n}\otimes\mathbf{n}\otimes\mathbf{d}), \tag{51}$$

and the contribution of the convection matrices to the residual of the ODE system,

$$\mathbb{M}^{-1}\mathbb{C} = \mathbb{M}^{-1}\hat{\mathbf{G}}_1(\mathbf{d}\otimes\mathbf{n}\otimes\mathbf{n}) + \mathbb{M}^{-1}\hat{\mathbf{G}}_2(\mathbf{n}\otimes\mathbf{d}\otimes\mathbf{n}) + \mathbb{M}^{-1}\hat{\mathbf{G}}_3(\mathbf{n}\otimes\mathbf{n}\otimes\mathbf{d}). \tag{52}$$

Finally, the contribution of the face mass matrices to the ODE residual is the same as in Eq. (50).

3.2.3 Operations count to evaluate the residual

To evaluate the cost of computing the action of the tensorised matrix $\mathbf{a} \otimes \mathbf{b}$ on a vector, where $\mathbf{a}$ and $\mathbf{b}$ are matrices of size $n \times n$, the decomposition $\mathbf{a} \otimes \mathbf{b} = (\mathbf{a} \otimes \mathbf{i})(\mathbf{i} \otimes \mathbf{b})$ is introduced, where $\mathbf{i}$ is the identity matrix of size $n \times n$. As previously shown, the cost of evaluating $(\mathbf{i} \otimes \mathbf{b})\mathbf{u}$, which is also the cost of $(\mathbf{a} \otimes \mathbf{i})\mathbf{u}$, is equal to $n^2(2n-1)$. The total cost to compute the action of the tensorised matrix $\mathbf{a} \otimes \mathbf{b}$ on a vector is thus $2n^2(2n-1)$. Similarly, in 3D the decomposition $\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c} = (\mathbf{a} \otimes \mathbf{i} \otimes \mathbf{i})(\mathbf{i} \otimes \mathbf{b} \otimes \mathbf{i})(\mathbf{i} \otimes \mathbf{i} \otimes \mathbf{c})$ is used to obtain that the cost to compute the action of the tensorised matrix $\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}$ on a vector is $3n^3(2n-1)$.

Using Eq. (49), the action of the matrix $\mathbb{M}^{-1}\mathbb{C}$ on a vector can be therefore decomposed into: (i) the action of $\mathrm{n_{sd}}$ tensorised matrices on that vector, with cost $\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}(2p+1)$ per matrix; and (ii) the action of $\mathrm{n_{sd}}$ diagonal matrices of size $(p+1)^{\mathrm{n_{sd}}}$, with a cost of $\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}$. The total cost to evaluate the residual part associated with the convection is therefore of order $\mathrm{n}_{\mathrm{sd}}^2(p+1)^{\mathrm{n_{sd}}}(2p+1) + \mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}$, with a leading order similar to the constant case.

The cost to evaluate the term of the residual in Eq. (22) associated to an internal face now reduces to the application of $\mathrm{n_{sd}}$ diagonal matrices of size $(p+1)^{\mathrm{n_{sd}}-1} \times (p+1)^{\mathrm{n_{sd}}-1}$, with a cost of $\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}-1}$.

These numbers of operations are summarized in Table 3.

	Operations	Leading term
$\mathrm{n}_{\mathrm{op}}^{\mathrm{e}}$	$\mathrm{n}_{\mathrm{sd}}^2(p+1)^{\mathrm{n_{sd}}}(2p+1) + \mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}}$	$\mathcal{O}(\mathrm{n}_{\mathrm{sd}}^2 p^{\mathrm{n_{sd}}+1})$
$\mathrm{n}_{\mathrm{op}}^{\mathrm{f}}$	$\mathrm{n_{sd}}(p+1)^{\mathrm{n_{sd}}-1}$	$\mathcal{O}(\mathrm{n_{sd}} p^{\mathrm{n_{sd}}-1})$

Table 3: Number of operations required (per element) to compute the matrix-vector products in Eq. (22) using the GLobQ for numerical integration.

Remark 5 Similarly to the case with a constant Jacobian and velocity the use of a tensorized version of the operators drastically reduces the memory requirements. Due to the use of Gauss-Lobatto quadrature rules, most of the matrices are diagonal ones. It is thus necessary in this context to save only the matrices $\mathbf{n}$ and $\mathbf{d}$ of size $(p+1) \times (p+1)$, the $\mathrm{n_{sd}}$ diagonal matrices $\hat{\mathbf{G}}_\ell$ of size $(p+1)^{\mathrm{n_{sd}}} \times (p+1)^{\mathrm{n_{sd}}}$, the diagonal matrices $\mathbb{M}^{-1}$ together with the diagonal matrices $\mathbb{M}^f$ of size $(p+1)^{\mathrm{n_{sd}}-1} \times (p+1)^{\mathrm{n_{sd}}-1}$. The total memory requirements are thus of the order of $\mathcal{O}(p^{\mathrm{n_{sd}}})$ advocating in favor of an approach where these matrices are precomputed.

4 A DG method with mixed quadratures

In the previous section, it was concluded that, in general, the most expensive part of the computation of the residual of Eq. (22) corresponds to the convection contribution. In this section, a formal proof is provided showing that under certain hypothesis (explained further down), the contribution to the residual given by the convection term is identical if GLegQ or GLobQ are used. This result is in principle not expected because the GLobQ does not provide an exact integration for the entries of the mass matrix. Furthermore, numerical integration with GLobQ does not provide exact integration for the convection matrix under the hypothesis considered. So even if both the mass and convection matrices are not integrated exactly, the result provided here shows that the contribution to the residual can be computed exactly.

This is the main motivation for proposing a new DG method using mixed quadratures (MixQ). The proposed approach uses a GLobQ for computing the contribution to the residual due to convection whereas a GLegQ is used to compute the face contribution. Numerical examples in Section 5 will be used to numerically demonstrate that, in the most general case, it is indeed the inaccurate integration of the face mass contribution (with the GLobQ) that influences accuracy most drastically, rather than the inaccurate integration of the convection contribution. Therefore, the proposed approach is considered attractive as it will improve the performance of the code by reducing the cost of the most expensive part of the computation, i.e., the convection part of the residual, whilst maintaining the overall accuracy of the standard DG method with GLegQ.

4.1 A surprising equality for residual evaluation

This section formally proves that, somewhat surprisingly, the evaluation of the convection part of the residual is exact (given an hypothesis that will be explained below) with GLobQ even when both the convection and mass matrices contain errors due to the numerical integration.

Let us first consider the mass and convection matrices obtained by integration with the GLobQ. Using Eqs. (43), (25) and (18), the expression for the elemental mass and convection matrices can be written as

$$\mathbb{M}^{-1}_{IJ} = \frac{\delta_{IJ}}{|\mathbf{J}(\boldsymbol{\xi}_I)|\omega_I}, \quad \mathbb{C}_{IJ} = \omega_I |\mathbf{J}(\boldsymbol{\xi}_I)| \sum_{k,\ell=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\boldsymbol{\xi}_I)\right) J^{-1}_{kl}(\boldsymbol{\xi}_I) \frac{\partial N_J(\boldsymbol{\xi}_I)}{\partial \xi_\ell}, \tag{53}$$

so that the convection part of the residual is given by

$$(\mathbb{M}^{-1}\mathbb{C})_{IJ} = \sum_{k,\ell=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\boldsymbol{\xi}_I)\right) J^{-1}_{kl}(\boldsymbol{\xi}_I) \frac{\partial N_J(\boldsymbol{\xi}_I)}{\partial \xi_\ell}. \tag{54}$$

We first note that any polynomial $P(\boldsymbol{\xi})$ over $\hat{\Omega}$, of order p in each direction, can be exactly represented on the basis of functions $N_I(\boldsymbol{\xi})$ in the following manner:

$$P(\boldsymbol{\xi}) = \sum_{I=0}^{\mathrm{n_{en}}} N_I(\boldsymbol{\xi}) \int_{\hat{\Omega}} N_I(\boldsymbol{\xi}) P(\boldsymbol{\xi}) d\Omega. \tag{55}$$

Secondly, if the order of that polynomial is less than or equal to $p-1$ in each direction, numerical integration with the GLobQ is exact (because $N_I(\boldsymbol{\xi})P(\boldsymbol{\xi})$ is then a polynomial of order less than $2p-1$), so that we further have

$$P(\boldsymbol{\xi}) = \sum_{I=0}^{\mathrm{n_{en}}} N_I(\boldsymbol{\xi}) P(\boldsymbol{\xi}_I). \tag{56}$$

Therefore, if, for any $1 \le I, J \le \mathrm{n_{en}}$, $\sum_{k,\ell=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\boldsymbol{\xi})\right) J^{-1}_{kl}(\boldsymbol{\xi}) \frac{\partial N_J}{\partial \xi_\ell}(\boldsymbol{\xi})$ is a polynomial of order less than $p-1$ in each direction,

$$\sum_{k,\ell=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\boldsymbol{\xi})\right) J^{-1}_{kl}(\boldsymbol{\xi}) \frac{\partial N_J}{\partial \xi_\ell}(\boldsymbol{\xi}) = \sum_{I=0}^{\mathrm{n_{en}}} N_I(\boldsymbol{\xi}) (\mathbb{M}^{-1}\mathbb{C})_{IJ}. \tag{57}$$

If we now consider the convection and mass matrices integrated exactly (for instance using GLegQ with enough integration points), as in Eq. (17) and (18), we obtain, using also Eq. (57),

$$\sum_{J=1}^{\mathrm{n_{en}}} \mathbf{M}_{IJ} (\mathbb{M}^{-1}\mathbb{C})_{JK} = \int_{\hat{\Omega}} N_I(\boldsymbol{\xi}) |\mathbf{J}(\boldsymbol{\xi})| \left(\sum_{J=1}^{\mathrm{n_{en}}} N_J(\boldsymbol{\xi}) (\mathbb{M}^{-1}\mathbb{C})_{JK} \right) d\Omega = \mathbf{C}_{IK}, \tag{58}$$

or, equivalently,

$$\mathbb{M}^{-1}\mathbb{C} = \mathbf{M}^{-1}\mathbf{C}. \tag{59}$$

Therefore, provided that for any $1 \le I, J \le \mathrm{n_{en}}$, $\sum_{k,\ell=1}^{\mathrm{n_{sd}}} a_k\left(\boldsymbol{\varphi}(\boldsymbol{\xi})\right) J^{-1}_{kl}(\boldsymbol{\xi}) \frac{\partial N_J}{\partial \xi_\ell}(\boldsymbol{\xi})$ is a polynomial of order less than $p-1$ in each direction, the product $\mathbb{M}^{-1}\mathbb{C}$ is evaluated exactly, even though both the matrices $\mathbb{M}$ and $\mathbb{C}$ may be wrongly integrated. Since this product is the only quantity required during the evaluation of the convective part of the residual, this means that using GLobQ essentially brings no additional error with respect to GLegQ. Of course, this says nothing of the face mass part or the evaluation of the residual.

Note that the constraint on the polynomial order is quite strong, but is at least verified for a constant velocity field and a constant Jacobian.

4.2 The DG method with mixed quadratures

Combined with the numerical tests of next section (in both cases when the polynomial hypothesis is verified or not), the previous section essentially states that integration error bears little weight in the evaluation of the convective part of the residual. Otherwise said, when aiming for a certain overall accuracy, it is less important to integrate exactly the convective part of the residual than the face mass part. This leads us to propose a novel DG method, with the following characteristics:

- Interpolation points are the nodes of the GLobQ;
- Residual is evaluated using Eq. (22) where
  - The convective part is $\boldsymbol{C}\mathbf{U} = \mathbb{M}^{-1}\mathbb{C}\mathbf{U}$, where the elements of $\mathbb{M}^{-1}\mathbb{C}$ are integrated using GLobQ, as in Eq. (54);
  - The face mass part is $\sum_{f=1}^{\mathrm{n}_{\mathrm{fa}}^{\mathrm{e}}} \boldsymbol{M}^f [\![\mathbf{U}^f]\!] = \sum_{f=1}^{\mathrm{n}_{\mathrm{fa}}^{\mathrm{e}}} \mathbf{M}^{-1}\mathbf{M}^f [\![\mathbf{U}^f]\!]$ where the elements of $\mathbf{M}^{-1}\mathbf{M}^f$ are integrated exactly (for instance using GLegQ in Eq. (25) and (26) with enough integration points).

The overall operations count for the evaluation of the residual is therefore $[2(p+1)^{\mathrm{n}_{\mathrm{sd}}-1} + \mathrm{n}_{\mathrm{sd}}^2(2p+1) + \mathrm{n}_{\mathrm{sd}} - 1](p+1)^{\mathrm{n}_{\mathrm{sd}}}$, with leading order $O(2p^{2\mathrm{n}_{\mathrm{sd}}-1})$ or $O(2\mathrm{n}_{\mathrm{sd}}^2 p^{\mathrm{n}_{\mathrm{sd}}+1})$ depending on the dimension n_{sd} and p. It is interesting to note here that except for $\mathrm{n}_{\mathrm{sd}} = 3$ and high order p, the contributions of the two terms in the residual are on the same order. In the worst case, the proposed method is therefore slightly more expensive than the full DGSEM, but next section will show that it is also much more accurate. With respect to the DG method with exact integration, the proposed method is more efficient, and the examples of next section will show that it is as accurate.

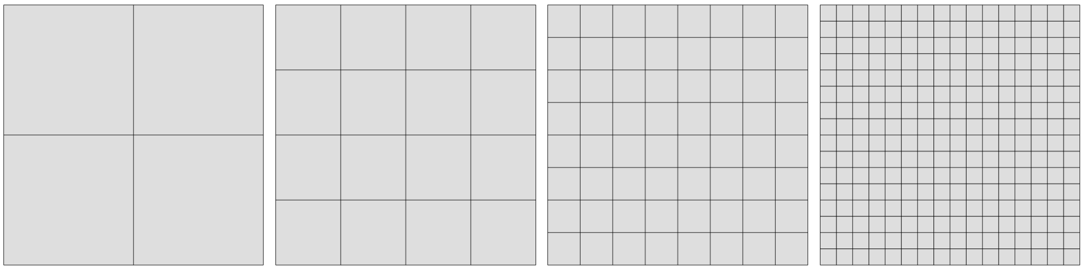
Table 4 summarises the number of operations of the DG methods with GLegQ, GLobQ and with the proposed MixQ.

Constant Jacobian and velocity				
	$\mathrm{n}_{\mathrm{sd}} = 2$		$\mathrm{n}_{\mathrm{sd}} = 3$	
	$\mathrm{n}_{\mathrm{op}}^{\mathrm{e}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{f}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{e}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{f}}$
GLegQ	$O(4p^3)$	$O(2p^2)$	$O(6p^4)$	$O(3p^3)$
GLobQ	$O(4p^3)$	$O(2p)$	$O(9p^4)$	$O(3p^2)$
MixQ	$O(4p^3)$	$O(2p^2)$	$O(9p^4)$	$O(3p^3)$
Non-constant Jacobian and/or velocity				
	$\mathrm{n}_{\mathrm{sd}} = 2$		$\mathrm{n}_{\mathrm{sd}} = 3$	
	$\mathrm{n}_{\mathrm{op}}^{\mathrm{e}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{f}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{e}}$	$\mathrm{n}_{\mathrm{op}}^{\mathrm{f}}$
GLegQ	$O(2p^4)$	$O(2p^3)$	$O(2p^6)$	$O(2p^5)$
GLobQ	$O(4p^3)$	$O(2p)$	$O(9p^4)$	$O(3p^2)$
MixQ	$O(4p^3)$	$O(2p^3)$	$O(9p^4)$	$O(2p^5)$

Table 4: Orders of magnitude of the numbers of operations required (per element) to compute the matrix-vector products to evaluate the ODE residual in Eq. (16) for the different DG approaches considered. The results, from Tables 1, 2 and 3, are particularised for two and three dimensions and displayed separately for the particular case of constant velocity and constant Jacobian alongside the general case.

5 Numerical examples

This section presents a set of numerical tests to evaluate the accuracy of the DG approaches considered in this work. The examples involve problems with constant and non-constant velocity fields both in two and three dimensions. In addition, the use of Cartesian and non-Cartesian grids is also studied, to evaluate the influence of having elements with non-constant Jacobian of the isoparametric mapping. The discussion also considers the computational cost of the DG approaches considered, as well as the impact of stability.



(a) $\mathcal{M}^1$ (b) $\mathcal{M}^2$ (c) $\mathcal{M}^3$ (d) $\mathcal{M}^4$

Figure 3: Cartesian meshes of quadrilateral elements.

p	h	$\epsilon_{\mathtt{GLegQ}} = \epsilon_{\mathtt{MixQ}}$	$r_{\mathtt{GLegQ}} = r_{\mathtt{MixQ}}$	$\epsilon_{\mathtt{GLobQ}}$	$r_{\mathtt{GLobQ}}$	$\frac{\epsilon_{\mathtt{GLobQ}}}{\epsilon_{\mathtt{GLegQ}}}$
1	0.5000	4.5×10^{-1}	–	9.7×10^{-1}	–	2.1
1	0.2500	1.4×10^{-1}	1.7	6.0×10^{-1}	0.7	4.2
1	0.1250	3.3×10^{-2}	2.1	2.3×10^{-1}	1.4	6.9
1	0.0625	7.7×10^{-3}	2.1	6.6×10^{-2}	1.8	8.6
2	0.5000	8.8×10^{-2}	–	3.1×10^{-1}	–	3.5
2	0.2500	1.0×10^{-2}	3.1	4.9×10^{-2}	2.7	5.0
2	0.1250	1.2×10^{-3}	3.0	6.2×10^{-3}	3.0	5.1
2	0.0625	1.5×10^{-4}	3.0	7.6×10^{-4}	3.0	5.1
3	0.5000	1.3×10^{-2}	–	4.9×10^{-2}	–	4.0
3	0.2500	7.9×10^{-4}	4.0	3.4×10^{-3}	3.9	4.3
3	0.1250	4.8×10^{-5}	4.0	2.2×10^{-4}	4.0	4.5
3	0.0625	3.0×10^{-6}	4.0	1.4×10^{-5}	4.0	4.5
4	0.5000	1.7×10^{-3}	–	6.5×10^{-3}	–	3.9
4	0.2500	5.1×10^{-5}	5.0	2.1×10^{-4}	4.9	4.1
4	0.1250	1.6×10^{-6}	5.0	6.8×10^{-6}	5.0	4.2
4	0.0625	5.0×10^{-8}	5.0	2.1×10^{-7}	5.0	4.2

Table 5: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ and GLobQ for the 2D problem with constant velocity using Cartesian meshes.

5.1 Two dimensional examples

The first two dimensional example considers the propagation of a sinusoidal wave with a constant velocity field in a Cartesian mesh. The domain is $\Omega = [0, 1]^2$ and the velocity field is given by $\boldsymbol{a} = [\cos(\pi/6), \sin(\pi/6)]$. The exact solution of the problem is given by

$$u(\boldsymbol{x}, t) = \sin\left(2\pi(\boldsymbol{a} \cdot \boldsymbol{x} - t)\right). \tag{60}$$

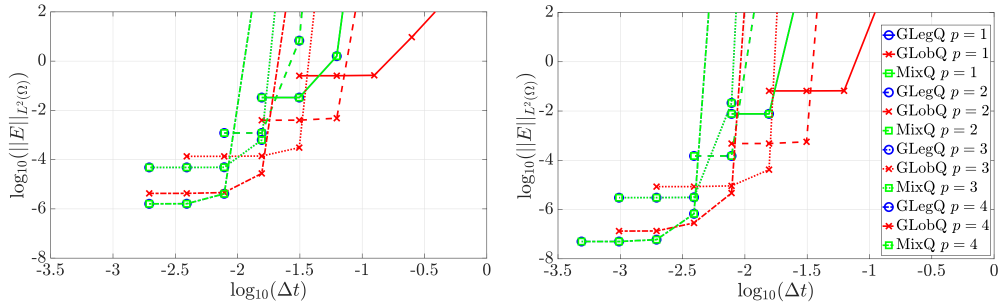
The final time for the simulations is taken as $T = 1$ and the time step is selected small enough so that the error induced by the time marching scheme is negligible when compared to the error induced by the spatial discretisation.

A set of Cartesian meshes is first utilised to test the optimal convergence properties of the DG methods with Gauss-Legendre (GLegQ) and Gauss-Lobatto (GLobQ) quadratures. As detailed in Section 4.1, the proposed approach with mixed quadratures (MixQ) leads to identical results when compared to the DG method with GLegQ.

The mesh corresponding to the i-th level of refinement is denoted by $\mathcal{M}^i$ and contains $2^i \times 2^i$ quadrilateral elements with characteristic element size $h = 2^{-i}$, as shown in figure 3.

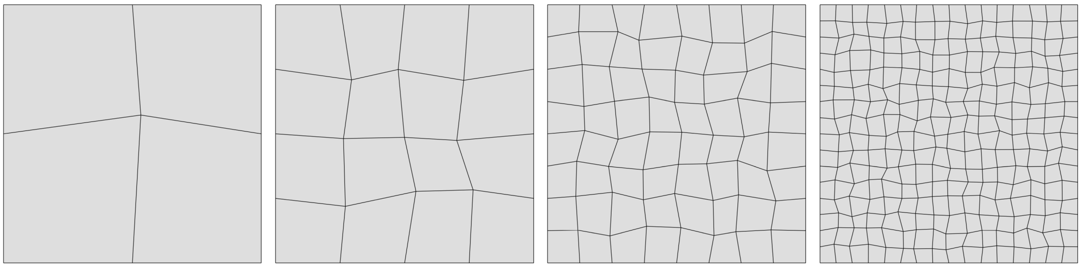
Table 5 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm and denoted by ϵ, for the DG methods with GLegQ and GLobQ for a polynomial approximation of order ranging from $p = 1$ up to $p = 4$. The table also shows the rates of convergence, denoted by r and the ratio between the error of both approaches.

The results show that both approaches provide an optimal rate of convergence, approximately $p + 1$, in the $\mathcal{L}^2(\Omega)$ norm. The results also reveal the extra accuracy provided by the DG method



(a) $\mathcal{M}^3$ **(b)** $\mathcal{M}^4$

Figure 4: Influence, in the case of constant velocity and constant jacobian, of time step on the $\mathcal{L}^2(\Omega)$-error at $T = 1$ for different polynomial orders, meshes and method.



(a) $\mathcal{M}^1$ **(b)** $\mathcal{M}^2$ **(c)** $\mathcal{M}^3$ **(d)** $\mathcal{M}^4$

Figure 5: Non-Cartesian meshes of quadrilateral elements.

with GLegQ, when compared to the results of the DG method with GLobQ on the same mesh. It is worth emphasising that the computation of the convection term in both cases is identical because the velocity is constant and the Jacobian of the isoparametric mapping is constant for the selected meshes. Therefore, the extra accuracy is due to the more accurate integration provided by the GLegQ for the contribution to the residual of the face term. For the lowest order case, the use of GLegQ provides an error nine times lower than that produced by the DG method with GLobQ. For higher orders,the use of GLegQ consistently provides an accuracy between four to five times higher than the DG method with GLobQ.

In terms of the computational cost, the DG method with GLegQ requires $O(4p^3)$ and $O(2p^2)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. The DG method with GLobQ requires $O(4p^3)$ and $O(2p)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. Therefore, it is apparent that the approach with GLegQ is beneficial as the dominant term, corresponding to the convection term, has the same order of operations.

For completeness, we propose an additional series of numerical tests to try and evaluate the influence of stability on this conclusion. In Fig. 4 is represented the influence of the time step on the error (at $T = 1$) for all three methods, different polynomial orders, and the refined meshes ($\mathcal{M}^3$ and $\mathcal{M}^4$). The unstability appears as an explosion of the error above a given critical time step. As already discussed, that time step is the same for GLegQ and MixQ. We observe that the stability time is consistently twice as large for GLobQ that GLegQ/MixQ. This means that the conclusion of the previous paragraph is somehow lessened as larger time steps can be considered (and hence twice less computational effort) with GLobQ than the other two method. However, MixQ remain the overall best method, with 2 to 4 times less error for the same computational cost as GLobQ, and the same error for a smaller computational cost as GLegQ.

To evaluate the influence of having elements with non-constant Jacobian, the same problem is now solved using the non-Cartesian meshes shown in Figure 5. The meshes have been obtained by randomly perturbing the position of the interior nodes of the Cartesian meshes of Figure 3.

Table 6 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm, for the DG methods with GLegQ, GLobQ and MixQ. The table also shows the rates of convergence and the ratio between the error of the DG methods with GLobQ and MixQ with respect to the GLegQ.

p	h	ϵ_{GLegQ}	r_{GLegQ}	ϵ_{GLobQ}	r_{GLobQ}	ϵ_{MixQ}	r_{MixQ}	$\frac{\epsilon_{\text{GLobQ}}}{\epsilon_{\text{GLegQ}}}$	$\frac{\epsilon_{\text{MixQ}}}{\epsilon_{\text{GLegQ}}}$
1	0.5737	4.7×10^{-1}	–	1.0	–	4.8×10^{-1}	–	2.1	1.0
1	0.2991	1.5×10^{-1}	1.8	6.7×10^{-1}	0.6	1.5×10^{-1}	1.8	4.4	1.0
1	0.1699	3.5×10^{-2}	2.6	2.8×10^{-1}	1.6	3.8×10^{-2}	2.4	7.9	1.1
1	0.0871	8.4×10^{-3}	2.1	7.8×10^{-2}	1.9	1.2×10^{-2}	1.8	9.3	1.4
2	0.5737	9.6×10^{-2}	–	2.6×10^{-1}	–	9.8×10^{-2}	–	2.7	1.0
2	0.2991	1.2×10^{-2}	3.1	3.9×10^{-2}	2.9	1.3×10^{-2}	3.1	3.1	1.0
2	0.1699	1.5×10^{-3}	3.7	5.3×10^{-3}	3.5	1.6×10^{-3}	3.7	3.5	1.0
2	0.0871	1.9×10^{-4}	3.1	6.2×10^{-4}	3.2	2.4×10^{-4}	2.8	3.3	1.3
3	0.5737	1.4×10^{-2}	–	4.2×10^{-2}	–	1.4×10^{-2}	–	3.0	1.0
3	0.2991	8.5×10^{-4}	4.3	2.4×10^{-3}	4.4	8.5×10^{-4}	4.3	2.9	1.0
3	0.1699	5.9×10^{-5}	4.7	1.7×10^{-4}	4.7	6.0×10^{-5}	4.7	2.9	1.0
3	0.0871	4.0×10^{-6}	4.0	1.2×10^{-5}	4.0	4.2×10^{-6}	4.0	2.9	1.1
4	0.5737	2.1×10^{-3}	–	5.3×10^{-3}	–	2.1×10^{-3}	–	2.6	1.0
4	0.2991	8.2×10^{-5}	4.9	2.1×10^{-4}	4.9	8.3×10^{-5}	5.0	2.6	1.0
4	0.1699	2.6×10^{-6}	6.1	7.2×10^{-6}	6.0	2.6×10^{-6}	6.1	2.7	1.0
4	0.0871	7.9×10^{-8}	5.2	2.2×10^{-7}	5.2	8.0×10^{-8}	5.2	2.7	1.0

Table 6: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ, GLobQ and MixQ for the 2D problem with constant velocity using non-Cartesian meshes.

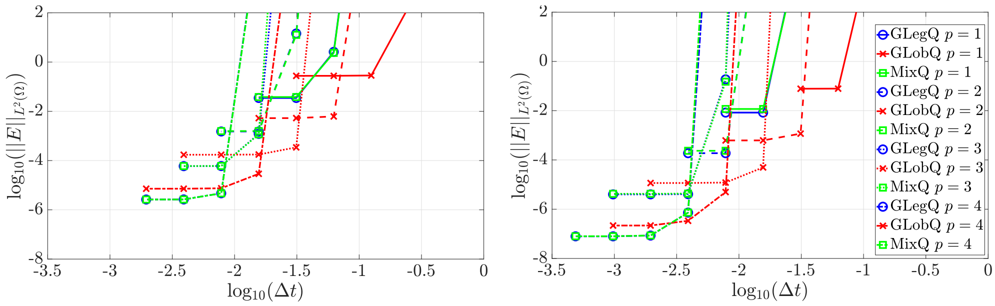
The results show the optimal rate of convergence for the three DG approaches. In terms of the accuracy, the DG method with GLegQ and the proposed DG method with MixQ provide almost identical results. This shows that the extra error present in the DG method with MixQ that is induced by the numerical integration of the part of the residual corresponding to the convection is almost negligible. In contrast when GLobQ are used for both the convection and the face integrals, the error increases by a factor between 2.7 and 3.3 for $p > 1$.

When comparing the computational costs of the three approaches, the most expensive is the DG method with GLegQ as the number of operations associated to the matrix-vector products of the convection and face contributions to the residual are $O(2p^4)$ and $O(2p^3)$, respectively. For the DG method with GLobQ the number of operations associated to the matrix-vector products of the convection and face contributions to the residual are $O(4p^3)$ and $O(2p)$, respectively. With the proposed DG method with MixQ $O(4p^3)$ and $O(2p^3)$ operations are required to compute the matrix-vector products of the convection and face contributions to the residual, respectively. In all cases the cost is dominated by the convection term, which requires $O(2p^4)$, $O(4p^3)$, and $O(4p^3)$ operations for the DG method with GLegQ, GLobQ and MixQ, respectively. In fact it is worth noting that the cost of the approach with MixQ is only slightly higher than the cost of the DG method with GLobQ, but as a result the accuracy is approximately three times higher. It is also worth noting that the proposed DG method with MixQ is more efficient than the DG method with GLegQ. In terms of accuracy both methods provide almost identical results but in terms of the cost, the proposed method has a dominant cost of $O(p^3)$ whereas the DG method with GLegQ has a dominant cost of $O(p^4)$.

As in the previous case, the conclusions are lessened when considering stability (see Fig. 6), but the MixQ remains 1.5 to 3 times more efficient than the GLobQ, and as precise as the GLeqQ for the same computational cost.

The next example considers a deformational flow, similar to the tests employed to assess the accuracy of advection schemes for climate and weather applications (Kent 2019). The velocity field is given by

$$\boldsymbol{a}(\boldsymbol{x}, t) = \cos(\pi t/T) \begin{bmatrix} 0.1 \sin^2(\pi x_1) \sin(2\pi x_2) \\ 0.1 \sin(2\pi x_1) \sin^2(\pi x_2) \end{bmatrix}^T \tag{61}$$



(a) $\mathcal{M}^3$ **(b)** $\mathcal{M}^4$

Figure 6: Influence, in the case of constant velocity and non-constant jacobian, of time step on the $\mathcal{L}^2(\Omega)$-error at $T = 1$ for different polynomial orders, meshes and method.

and the final time is $T = 1$.

The initial condition is given by $u_0(\boldsymbol{x}) = 0.5 + 0.5\sin(2\pi x_1)\sin(2\pi x_2)$ and periodic boundary conditions are considered on the boundary of the domain $\partial\Omega$. It is worth noting that the problem has no known analytical solution but the cosine term in Eq. (61) ensures that the flow reverses at time $t = T/2$ and reaches the same state as the initial condition at time $t = T$. This idea was originally proposed in (Leveque 1996) to test the accuracy of numerical schemes in problems with no known analytical solution.

Table 7 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm, for the three DG approaches considered in the previous section using Cartesian meshes.

The results show that all methods achieve the optimal rate of convergence, but it is worth noting that the DG method with GLobQ takes substantially longer to reach the asymptotic range of convergence compared to the DG methods with GLegQ and MixQ. Once the asymptotic rate of convergence is reached, the DG methods with GLegQ and MixQ provide between 2 and 2.5 times more accurate results than the DG method with GLobQ for $p > 1$. The results also show that in some cases the proposed DG method with MixQ is marginally more accurate than the DG method with GLegQ. This is attributed to the non-exact integration of the convection term, which involves a non-polynomial velocity field.

As the cost associated to the matrix-vector products when the velocity field or the Jacobian are not constant, the conclusions are very similar to the ones obtained in the previous example. The dominant cost, associated to the convection term, is $O(2p^4)$, $O(4p^3)$, and $O(4p^3)$ operations for the DG method with GLegQ, GLobQ and MixQ, respectively. Therefore, the proposed DG method with MixQ has the same dominant cost as the GLobQ with an accuracy more than two times higher. The proposed DG method with MixQ is also more efficient than the DG method with GLegQ because the accuracy is almost identical but the dominant cost is lower.

The last two dimensional example involves the solution of the deformational flow with non-constant velocity by using non-Cartesian meshes. Table 8 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm, for the three DG approaches considered.

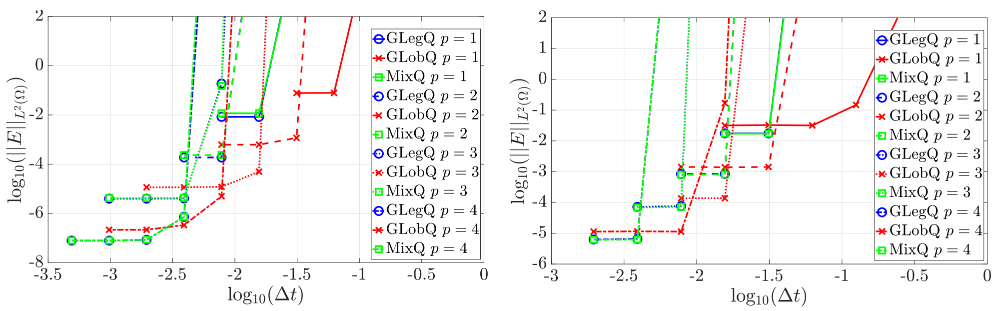
The results are both, qualitatively and quantitatively, very similar to the ones obtained in the previous example with a non-constant velocity field using Cartesian meshes. In all cases the proposed DG method with MixQ produces almost identical results when compared to the DG method with GLegQ, despite the extra accuracy of the numerical quadratures used for the convection term in the DG method with GLegQ. This denotes that even if the velocity and Jacobian are both not constant, the error induced by a numerical integration of the convection term by a Gauss-Lobatto quadrature compared to the error induced by a Gauss-Legendre quadrature is below the discretisation error.

This time however, the conclusions are not so clear when considering stability (see Fig. 7), as the higher precision of the MixQ with respect to GLobQ is somewhat compensated by the higher stability of the GlobQ, so that both methods seem equivalent in terms of efficiency in the case of non-constant velocity.

As a conclusion for this series of 2D numerical tests, the MixQ method is an attractive alternative to the DG methods with GLegQ and GLobQ. At least for constant velocity, the

p	h	$\epsilon_{\texttt{GLegQ}}$	$r_{\texttt{GLegQ}}$	$\epsilon_{\texttt{GLobQ}}$	$r_{\texttt{GLobQ}}$	$\epsilon_{\texttt{MixQ}}$	$r_{\texttt{MixQ}}$	$\frac{\epsilon_{\texttt{GLobQ}}}{\epsilon_{\texttt{GLegQ}}}$	$\frac{\epsilon_{\texttt{MixQ}}}{\epsilon_{\texttt{GLegQ}}}$
1	0.5000	3.5×10^{-1}	–	3.5×10^{-1}	–	3.5×10^{-1}	–	1.0	1.0
1	0.2500	1.5×10^{-1}	1.2	1.5×10^{-1}	1.2	1.5×10^{-1}	1.2	1.0	1.0
1	0.1250	4.4×10^{-2}	1.8	4.4×10^{-2}	1.8	4.4×10^{-2}	1.8	1.0	1.0
1	0.0625	1.2×10^{-2}	1.9	1.2×10^{-2}	1.9	1.2×10^{-2}	1.9	1.0	1.0
1	0.0313	3.0×10^{-3}	2.0	3.2×10^{-3}	1.9	3.0×10^{-3}	2.0	1.1	1.0
1	0.0156	7.7×10^{-4}	2.0	9.2×10^{-4}	1.8	7.7×10^{-4}	2.0	1.2	1.0
1	0.0078	1.9×10^{-4}	2.0	2.6×10^{-4}	1.8	1.9×10^{-4}	2.0	1.3	1.0
2	0.5000	4.0×10^{-2}	–	4.0×10^{-2}	–	4.0×10^{-2}	–	1.0	1.0
2	0.2500	1.2×10^{-2}	1.7	1.2×10^{-2}	1.7	1.2×10^{-2}	1.7	1.0	1.0
2	0.1250	1.6×10^{-3}	2.9	1.7×10^{-3}	2.9	1.6×10^{-3}	2.9	1.0	1.0
2	0.0625	2.1×10^{-4}	2.9	2.3×10^{-4}	2.8	2.1×10^{-4}	2.9	1.1	1.0
2	0.0313	2.2×10^{-5}	3.2	4.6×10^{-5}	2.3	2.2×10^{-5}	3.2	2.0	1.0
2	0.0156	2.8×10^{-6}	3.0	5.6×10^{-6}	3.0	2.7×10^{-6}	3.0	2.0	1.0
2	0.0078	3.5×10^{-7}	3.0	7.3×10^{-7}	2.9	3.4×10^{-7}	3.0	2.1	1.0
3	0.5000	2.2×10^{-2}	–	2.2×10^{-2}	–	2.2×10^{-2}	–	1.0	1.0
3	0.2500	1.0×10^{-3}	4.4	1.1×10^{-3}	4.4	1.1×10^{-3}	4.4	1.0	1.0
3	0.1250	8.5×10^{-5}	3.6	9.1×10^{-5}	3.5	8.2×10^{-5}	3.7	1.1	1.0
3	0.0625	6.4×10^{-6}	3.7	8.7×10^{-6}	3.4	6.3×10^{-6}	3.7	1.4	1.0
3	0.0313	4.0×10^{-7}	4.0	8.0×10^{-7}	3.5	4.0×10^{-7}	4.0	2.0	1.0
3	0.0156	2.3×10^{-8}	4.2	5.2×10^{-8}	3.9	2.3×10^{-8}	4.1	2.3	1.0
3	0.0078	1.2×10^{-9}	4.2	2.9×10^{-9}	4.2	1.2×10^{-9}	4.2	2.3	1.0
4	0.5000	8.8×10^{-4}	–	7.5×10^{-4}	–	7.6×10^{-4}	–	0.8	0.9
4	0.2500	9.6×10^{-5}	3.2	9.5×10^{-5}	3.0	8.9×10^{-5}	3.1	1.0	0.9
4	0.1250	3.6×10^{-6}	4.7	3.9×10^{-6}	4.6	3.2×10^{-6}	4.8	1.1	0.9
4	0.0625	1.2×10^{-7}	4.9	1.9×10^{-7}	4.3	1.1×10^{-7}	4.9	1.6	0.9
4	0.0313	3.8×10^{-9}	5.0	8.0×10^{-9}	4.6	3.3×10^{-9}	5.0	2.1	0.9
4	0.0156	1.2×10^{-10}	5.0	2.6×10^{-10}	5.0	1.0×10^{-10}	5.0	2.2	0.9
4	0.0078	3.5×10^{-12}	5.1	8.4×10^{-12}	4.9	3.3×10^{-12}	5.0	2.4	0.9

Table 7: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ, GLobQ and MixQ for the 2D problem with non-constant velocity using Cartesian meshes.



(a) $\mathcal{M}^3$ **(b)** $\mathcal{M}^4$

Figure 7: Influence, in the case of non-constant velocity and non-constant jacobian, of time step on the $\mathcal{L}^2(\Omega)$-error at $T = 1$ for different polynomial orders, meshes and method.

proposed approach provides the same accuracy as the DG method with GLegQ, with a lower cost induced by the matrix-vector products. In addition, the leading term of the cost for the DG methods with GLegQ and MixQ is identical but the proposed approach provides results twice as accurate. For non-constant velocity, the efficiency of GLobQ and MixQ seems to be equivalent, because the improved stability of the GLobQ method compensates for the improved precision of

p	h	ϵ_{GLegQ}	r_{GLegQ}	ϵ_{GLobQ}	r_{GLobQ}	ϵ_{MixQ}	r_{MixQ}	$\frac{\epsilon_{\text{GLobQ}}}{\epsilon_{\text{GLegQ}}}$	$\frac{\epsilon_{\text{MixQ}}}{\epsilon_{\text{GLegQ}}}$
1	0.5737	3.6×10^{-1}	–	3.6×10^{-1}	–	3.6×10^{-1}	–	1.0	1.0
1	0.2991	1.7×10^{-1}	1.2	1.7×10^{-1}	1.2	1.7×10^{-1}	1.2	1.0	1.0
1	0.1699	4.9×10^{-2}	2.2	4.9×10^{-2}	2.2	4.9×10^{-2}	2.2	1.0	1.0
1	0.0871	1.3×10^{-2}	2.0	1.3×10^{-2}	2.0	1.3×10^{-2}	2.0	1.0	1.0
1	0.0440	3.3×10^{-3}	2.0	3.5×10^{-3}	1.9	3.3×10^{-3}	2.0	1.1	1.0
1	0.0224	8.6×10^{-4}	2.0	1.0×10^{-3}	1.8	8.6×10^{-4}	2.0	1.2	1.0
1	0.0111	2.2×10^{-4}	2.0	3.2×10^{-4}	1.7	2.2×10^{-4}	1.9	1.5	1.0
2	0.5737	4.7×10^{-2}	–	4.7×10^{-2}	–	4.7×10^{-2}	–	1.0	1.0
2	0.2991	1.4×10^{-2}	1.9	1.4×10^{-2}	1.9	1.4×10^{-2}	1.9	1.0	1.0
2	0.1699	2.0×10^{-3}	3.4	2.1×10^{-3}	3.3	2.0×10^{-3}	3.4	1.0	1.0
2	0.0871	2.7×10^{-4}	3.0	2.9×10^{-4}	2.9	2.7×10^{-4}	3.0	1.1	1.0
2	0.0440	3.5×10^{-5}	3.0	6.0×10^{-5}	2.3	3.5×10^{-5}	3.0	1.7	1.0
2	0.0224	4.1×10^{-6}	3.2	8.2×10^{-6}	2.9	4.1×10^{-6}	3.2	2.0	1.0
2	0.0111	5.2×10^{-7}	2.9	1.1×10^{-6}	2.9	5.3×10^{-7}	2.9	2.0	1.0
3	0.5737	2.4×10^{-2}	–	2.4×10^{-2}	–	2.4×10^{-2}	–	1.0	1.0
3	0.2991	1.5×10^{-3}	4.3	1.5×10^{-3}	4.3	1.4×10^{-3}	4.3	1.0	1.0
3	0.1699	1.2×10^{-4}	4.5	1.2×10^{-4}	4.4	1.1×10^{-4}	4.5	1.1	1.0
3	0.0871	9.0×10^{-6}	3.8	1.2×10^{-5}	3.5	8.9×10^{-6}	3.8	1.3	1.0
3	0.0440	5.7×10^{-7}	4.0	1.0×10^{-6}	3.5	5.7×10^{-7}	4.0	1.8	1.0
3	0.0224	3.5×10^{-8}	4.1	7.5×10^{-8}	3.9	3.5×10^{-8}	4.1	2.1	1.0
3	0.0111	2.1×10^{-9}	4.0	4.4×10^{-9}	4.0	2.1×10^{-9}	4.0	2.1	1.0
4	0.5737	1.7×10^{-3}	–	1.5×10^{-3}	–	1.5×10^{-3}	–	0.9	0.9
4	0.2991	1.3×10^{-4}	4.0	1.2×10^{-4}	3.8	1.2×10^{-4}	3.9	1.0	0.9
4	0.1699	5.8×10^{-6}	5.4	6.5×10^{-6}	5.2	5.4×10^{-6}	5.4	1.1	0.9
4	0.0871	2.1×10^{-7}	5.0	2.9×10^{-7}	4.7	1.9×10^{-7}	5.0	1.4	0.9
4	0.0440	7.4×10^{-9}	4.9	1.4×10^{-8}	4.4	6.9×10^{-9}	4.8	1.9	0.9
4	0.0224	2.4×10^{-10}	5.1	4.5×10^{-10}	5.1	2.2×10^{-10}	5.1	1.9	0.9
4	0.0111	7.9×10^{-12}	4.9	1.5×10^{-11}	4.9	7.7×10^{-12}	4.8	1.9	1.0

Table 8: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ, GLobQ and MixQ for the 2D problem with non-constant velocity using non-Cartesian meshes.

the MixQ.

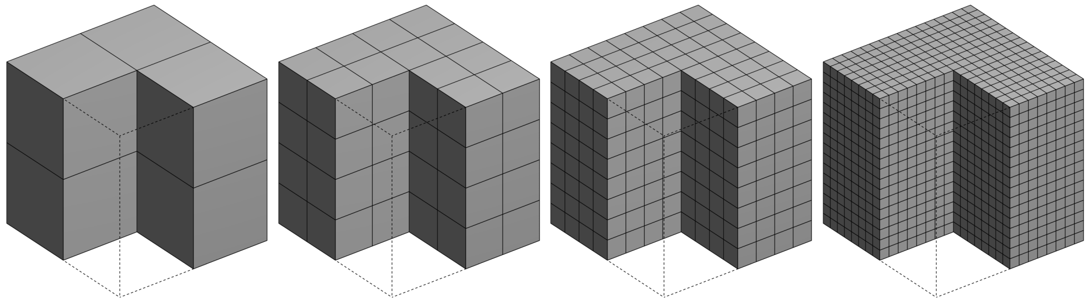
5.2 Three dimensional examples

The first three dimensional example considers the propagation of a sinusoidal wave at constant velocity in a Cartesian mesh. The domain is $\Omega = [0, 1]^3$ and the velocity field is given by $\boldsymbol{a} = [\cos(\pi/6)\sin(\pi/4), \sin(\pi/6)\sin(\pi/4), \cos(\pi/4)]$. As in the previous examples, the final time is $T = 1$ and the time step is selected small enough so that the error induced by the time marching scheme is negligible when compared to the error induced by the spatial discretisation.

A set of Cartesian meshes is utilised in three dimensions. The mesh corresponding to the i-th level of refinement is denoted by $\mathcal{M}^i$ and contains $2^i \times 2^i \times 2^i$ hexahedral elements with characteristic element size $h = 2^{-i}$, as shown in Figure 8.

Table 9 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm and denoted by ϵ, for the DG methods with GLegQ and GLobQ for a polynomial approximation of order ranging from $p = 1$ up to $p = 4$. The table also shows the rates of convergence, denoted by r and the ratio between the error of both DG approaches.

The optimal rate of convergence is shown for both DG approaches, that is a rate of approximately $p + 1$, in the $\mathcal{L}^2(\Omega)$ norm. As in the two dimensional example, the extra accuracy provided by the DG method with GLegQ is clearly observed. For the lowest order case, the use of GLegQ



(a) $\mathcal{M}^1$ (b) $\mathcal{M}^2$ (c) $\mathcal{M}^3$ (d) $\mathcal{M}^4$

Figure 8: Cartesian meshes of hexahedral elements.

p	h	$\epsilon_{\texttt{GLegQ}} = \epsilon_{\texttt{MixQ}}$	$r_{\texttt{GLegQ}} = r_{\texttt{MixQ}}$	ϵ_{GLobQ}	r_{GLobQ}	$\frac{\epsilon_{\text{GLobQ}}}{\epsilon_{\text{GLegQ}}}$
1	0.5000	3.1×10^{-1}	–	8.6×10^{-1}	–	2.8
1	0.2500	9.8×10^{-2}	1.6	4.7×10^{-1}	0.9	4.8
1	0.1250	2.5×10^{-2}	2.0	1.6×10^{-1}	1.6	6.4
1	0.0625	6.1×10^{-3}	2.0	4.3×10^{-2}	1.9	7.1
2	0.5000	5.3×10^{-2}	–	2.1×10^{-1}	–	3.9
2	0.2500	6.2×10^{-3}	3.1	3.0×10^{-2}	2.8	4.9
2	0.1250	7.6×10^{-4}	3.0	3.8×10^{-3}	3.0	5.0
2	0.0625	9.5×10^{-5}	3.0	4.8×10^{-4}	3.0	5.1
3	0.5000	6.3×10^{-3}	–	2.6×10^{-2}	–	4.1
3	0.2500	3.9×10^{-4}	4.0	1.7×10^{-3}	3.9	4.4
3	0.1250	2.5×10^{-5}	4.0	1.1×10^{-4}	4.0	4.5
3	0.0625	1.5×10^{-6}	4.0	6.9×10^{-6}	4.0	4.5
4	0.5000	6.7×10^{-4}	–	2.6×10^{-3}	–	3.9
4	0.2500	2.1×10^{-5}	5.0	8.6×10^{-5}	4.9	4.2
4	0.1250	6.5×10^{-7}	5.0	2.7×10^{-6}	5.0	4.2
4	0.0625	2.0×10^{-8}	5.0	8.5×10^{-8}	5.0	4.2

Table 9: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ and GLobQ for the 3D problem with constant velocity using Cartesian meshes.

provides an error almost nine times lower than that produced by the DG method with GLobQ. For higher orders,the use of GLegQ consistently provides an accuracy between four to five times higher than the DG method with GLobQ. It is worth noticing that for $p > 1$ the improvement due to the use of GLegQ is the same as for the two dimensional example with constant velocity and using Cartesian meshes.

In terms of the computational cost, the DG method with GLegQ requires $\mathcal{O}(6p^4)$ and $\mathcal{O}(3p^3)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. The DG method with GLobQ requires $\mathcal{O}(9p^4)$ and $\mathcal{O}(3p^2)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. Therefore, it is clear that the approach with GLegQ is beneficial as the dominant term, corresponding to the convection term, has a lower order of operations and provides more accurate results. As in two dimensions, note that stability tests (not shown here because the results are very similar in three dimensions) indicate that part of the efficiency of the DG method with GLegQ is lost because a time step twice smaller needs to be used.

As done in two dimensions, the next three dimensional example considers a problem with constant velocity by using non-Cartesian meshes. The meshes considered are shown in Figure 9 and are, again, generated by randomly perturbing the internal nodes of the Cartesian meshes employed in the previous three dimensional example.

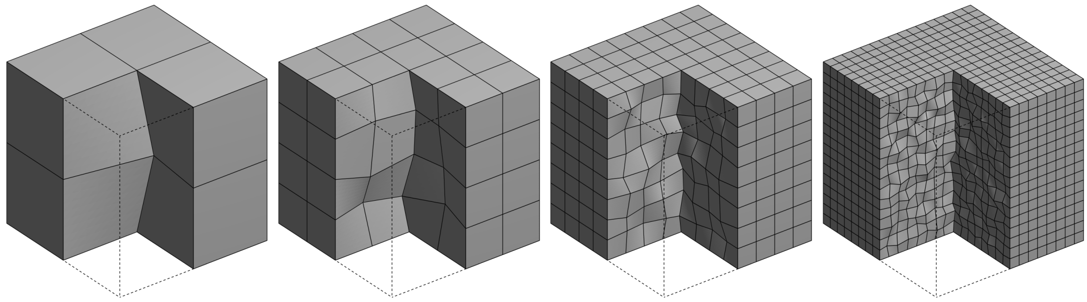


Figure 9: Non-Cartesian meshes of hexahedral elements.

Table 10 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm, for the DG methods with GLegQ, GLobQ and MixQ. The table also shows the rates of convergence and the ratio between the error of the DG methods with GLobQ and MixQ with respect to the GLegQ.

p	h	ϵ_{GLegQ}	r_{GLegQ}	ϵ_{GLobQ}	r_{GLobQ}	ϵ_{MixQ}	r_{MixQ}	$\frac{\epsilon_{\text{GLobQ}}}{\epsilon_{\text{GLegQ}}}$	$\frac{\epsilon_{\text{MixQ}}}{\epsilon_{\text{GLegQ}}}$
1	0.5750	3.1×10^{-1}	–	8.6×10^{-1}	–	3.1×10^{-1}	–	2.8	1.0
1	0.3482	1.0×10^{-1}	2.2	4.9×10^{-1}	1.1	1.1×10^{-1}	2.0	4.8	1.1
1	0.1758	2.7×10^{-2}	1.9	1.9×10^{-1}	1.4	3.6×10^{-2}	1.6	6.8	1.3
1	0.0897	6.8×10^{-3}	2.1	5.5×10^{-2}	1.8	1.3×10^{-2}	1.5	8.1	1.9
2	0.5750	6.5×10^{-2}	–	2.4×10^{-1}	–	6.8×10^{-2}	–	3.6	1.0
2	0.3482	7.4×10^{-3}	4.3	3.4×10^{-2}	3.9	7.8×10^{-3}	4.3	4.6	1.1
2	0.1758	9.9×10^{-4}	2.9	4.8×10^{-3}	2.9	1.1×10^{-3}	2.8	4.9	1.1
2	0.0897	1.2×10^{-4}	3.1	6.0×10^{-4}	3.1	1.5×10^{-4}	2.9	4.9	1.2
3	0.5750	6.5×10^{-3}	–	2.6×10^{-2}	–	6.5×10^{-3}	–	4.0	1.0
3	0.3482	4.6×10^{-4}	5.3	2.0×10^{-3}	5.1	4.8×10^{-4}	5.2	4.3	1.0
3	0.1758	3.3×10^{-5}	3.9	1.5×10^{-4}	3.8	3.6×10^{-5}	3.8	4.5	1.1
3	0.0897	2.2×10^{-6}	4.0	1.0×10^{-5}	4.0	2.6×10^{-6}	3.9	4.6	1.2
4	0.5750	7.9×10^{-4}	–	3.2×10^{-3}	–	8.0×10^{-4}	–	4.0	1.0
4	0.3482	2.8×10^{-5}	6.6	1.2×10^{-4}	6.6	2.9×10^{-5}	6.6	4.2	1.0
4	0.1758	9.9×10^{-7}	4.9	4.2×10^{-6}	4.9	1.0×10^{-6}	4.9	4.2	1.0
4	0.0897	3.4×10^{-8}	5.0	1.4×10^{-7}	5.0	3.6×10^{-8}	5.0	4.2	1.1

Table 10: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ, GLobQ and MixQ for the 3D problem with constant velocity using non-Cartesian meshes.

The results reveal that the proposed DG method with MixQ offers again almost identical results when compared to the DG method with GLegQ. This indicates that the numerical integration error introduced by the MixQ approach is below the discretisation error even when three dimensional elements with non-constant Jacobian are considered. In addition, the results in Table 10 indicate that the DG methods with GLegQ and MixQ offer between four and five times more accurate results when compared to the approach with GLobQ. This indicates a higher gain than what was observed in the two dimensional example.

In terms of the computational cost, the DG method with GLegQ requires $O(2p^6)$ and $O(2p^5)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. The DG method with GLobQ requires $O(9p^4)$ and $O(3p^2)$ operations for computing the matrix-vector products corresponding to the convection and face contributions to the residual, respectively. With the proposed DG method with MixQ $O(9p^4)$ and $O(2p^5)$ operations are required to compute the matrix-vector products of the convection and face contributions to the residual, respectively. It is worth noting that contrary to the observations made in two dimensions, for the DG method with MixQ the dominant term is now associated to the face term. However, when compared to the DG method with GLegQ, the proposed DG method still has a leading cost lower and provides the same accuracy. As in two dimensions, larger time steps need to be used so that the method is overall two to three times more efficient.

The last example considers a deformational flow in three dimensions. The non-constant velocity field is given by

$$\boldsymbol{a}(\boldsymbol{x},t) = \cos(\pi t/T) \begin{bmatrix} 0.1\sin^2(\pi x_1)\sin(2\pi x_2)\sin(\pi x_3)\cos(\pi x_3) \\ 0.2\sin(2\pi x_1)\sin^2(\pi x_2)\sin(\pi x_3)\cos(\pi x_3) \\ 0.05\pi\sin(2\pi x_1)\sin(2\pi x_2)\sin^2(\pi x_3) \end{bmatrix}^T \tag{62}$$

and the final time is $T = 1$.

The initial condition is given by $u_0(\boldsymbol{x}) = 0.5 + 0.5\sin(2\pi x_1)\sin(2\pi x_2)\sin(2\pi x_3)$ and periodic boundary conditions are considered on the boundary of the domain $\partial\Omega$. Analogously to the two dimensional deformation flow considered in Section 5.1, the cosine term in Eq. (62) ensures that the flow reverses at time $t = T/2$ and reaches the same state as the initial condition at time $t = T$.

Table 11 shows the relative error, measured in the $\mathcal{L}^2(\Omega)$ norm, for the three DG approaches considered using Cartesian meshes.

p	h	ϵ_{GLegQ}	r_{GLegQ}	ϵ_{GLobQ}	r_{GLobQ}	ϵ_{MixQ}	r_{MixQ}	$\frac{\epsilon_{\text{GLobQ}}}{\epsilon_{\text{GLegQ}}}$	$\frac{\epsilon_{\text{MixQ}}}{\epsilon_{\text{GLegQ}}}$
1	0.5000	2.3×10^{-1}	–	2.3×10^{-1}	–	2.3×10^{-1}	–	1.0	1.0
1	0.2500	1.5×10^{-1}	0.6	1.5×10^{-1}	0.6	1.5×10^{-1}	0.6	1.0	1.0
1	0.1250	4.8×10^{-2}	1.7	4.8×10^{-2}	1.7	4.8×10^{-2}	1.7	1.0	1.0
1	0.0625	1.3×10^{-2}	1.9	1.3×10^{-2}	1.9	1.3×10^{-2}	1.9	1.0	1.0
1	0.0313	3.3×10^{-3}	2.0	3.4×10^{-3}	1.9	3.3×10^{-3}	2.0	1.0	1.0
1	0.0156	8.4×10^{-4}	2.0	9.3×10^{-4}	1.9	8.4×10^{-4}	2.0	1.1	1.0
1	0.0078	2.1×10^{-4}	2.0	2.5×10^{-4}	1.9	2.1×10^{-4}	2.0	1.2	1.0
2	0.5000	3.8×10^{-2}	–	3.8×10^{-2}	–	3.8×10^{-2}	–	1.0	1.0
2	0.2500	1.2×10^{-2}	1.7	1.2×10^{-2}	1.7	1.2×10^{-2}	1.7	1.0	1.0
2	0.1250	1.5×10^{-3}	3.0	1.5×10^{-3}	3.0	1.5×10^{-3}	3.0	1.0	1.0
2	0.0625	1.9×10^{-4}	3.0	2.2×10^{-4}	2.8	1.8×10^{-4}	3.0	1.2	1.0
2	0.0313	2.2×10^{-5}	3.1	3.6×10^{-5}	2.6	2.2×10^{-5}	3.1	1.6	1.0
2	0.0156	2.6×10^{-6}	3.1	5.1×10^{-6}	2.8	2.6×10^{-6}	3.1	1.9	1.0
2	0.0078	3.2×10^{-7}	3.0	6.7×10^{-7}	2.9	3.1×10^{-7}	3.0	2.1	1.0
3	0.5000	2.2×10^{-2}	–	2.2×10^{-2}	–	2.2×10^{-2}	–	1.0	1.0
3	0.2500	9.6×10^{-4}	4.5	9.7×10^{-4}	4.5	9.6×10^{-4}	4.5	1.0	1.0
3	0.1250	7.6×10^{-5}	3.7	8.2×10^{-5}	3.6	7.3×10^{-5}	3.7	1.1	1.0
3	0.0625	5.4×10^{-6}	3.8	7.6×10^{-6}	3.4	5.4×10^{-6}	3.8	1.4	1.0
3	0.0313	3.6×10^{-7}	3.9	6.6×10^{-7}	3.5	3.6×10^{-7}	3.9	1.8	1.0
3	0.0156	2.2×10^{-8}	4.0	4.6×10^{-8}	3.8	2.2×10^{-8}	4.0	2.1	1.0
3	0.0078	1.2×10^{-9}	4.1	2.8×10^{-9}	4.1	1.2×10^{-9}	4.1	2.2	1.0
4	0.5000	8.2×10^{-4}	–	7.1×10^{-4}	–	7.2×10^{-4}	–	0.9	0.9
4	0.2500	8.9×10^{-5}	3.2	9.0×10^{-5}	3.0	8.3×10^{-5}	3.1	1.0	0.9
4	0.1250	3.3×10^{-6}	4.7	3.7×10^{-6}	4.6	2.9×10^{-6}	4.9	1.1	0.9
4	0.0625	1.2×10^{-7}	4.8	1.7×10^{-7}	4.4	1.0×10^{-7}	4.8	1.4	0.8
4	0.0313	3.8×10^{-9}	5.0	6.9×10^{-9}	4.7	3.2×10^{-9}	5.0	1.8	0.8
4	0.0156	1.2×10^{-10}	5.0	2.5×10^{-10}	4.8	1.0×10^{-10}	5.0	2.1	0.9

Table 11: Relative error in the $\mathcal{L}^2(\Omega)$ norm (ϵ), and rate of convergence (r) for the DG methods with GLegQ, GLobQ and MixQ for the 3D problem with non-constant velocity using Cartesian meshes.

As observed in previous examples, all methods show the optimal rate of convergence but the DG method with GLobQ requires very fine meshes to exhibit the asymptotic convergence. In this example the DG methods with GLegQ and MixQ offer an error two times lower than the

DG method with GLobQ. The results of the proposed approach, with MixQ, are in some cases marginally more accurate than the DG method with GLegQ, but this is attributed by the inexact integration of the complex velocity field considered.

It is worth noting that in three dimensions the dominant cost associated to the proposed approach is of lower order when compared to the DG method with GLegQ but the accuracy is almost identical. When comparing the proposed DG method with MixQ and GLobQ, the cost of the MixQ is only slightly higher but the extra accuracy is clearly observed. However, as in two dimensions, for non-constant velocity, it seems that the overall efficiency is the same for DG methods with MixQ and GLegQ because the weaker stability compensates for the lower cost for computing matrix-vector products.

6 Concluding remarks

A novel DG method for the solution of linear conservation laws has been presented. The approach uses a Gauss-Lobatto distribution of nodes to define the polynomial approximation and a mixture of Gauss-Lobatto and Gauss-Legendre quadratures for the numerical integration. The convection part of the residual of the system of ordinary differential equations is evaluated using Gauss-Lobatto quadratures, whereas the term corresponding to the face integrals is evaluated using Gauss-Legendre quadratures.

A formal proof is provided to show that in the case of constant Jacobian and constant coefficients (i.e., velocity field), the proposed method provides exactly the same results as a standard DG method with Gauss-Legendre quadratures, despite both the mass and convection matrices are underintegrated. For the general case, the numerical results show that the accuracy is also preserved, meaning that the numerical integration error is below the spatial discretisation error. This brings the advantages of both commonly used DG methods with Gauss-Lobatto and Gauss-Legendre quadratures. The proposed approach is as accurate as the standard DG method with Gauss-Legendre quadratures but at a lower cost, which is only slightly more expensive than the DG method with Gauss-Lobatto quadratures in three dimensions and as expensive in two dimensions. This conclusion remains generally correct when adding stability to the discussion, but, in some cases, the improved stability of the DG method with Gauss-Lobatto quadratures compensates for the improved precision of the method with mixed quadratures.

References

Ainsworth, M. (2004a). “Dispersive and dissipative behavior of high order discontinuous Galerkin finite element methods”. *Journal of Computational Physics* 198.1, pp. 106–130. DOI: 10.1016/j.jcp.2004.01.004

Ainsworth, M. (2004b). “Dispersive and dissipative behaviour of high order discontinuous Galerkin finite element methods”. *J. Comput. Phys.* 198.1, pp. 106–130

Ainsworth, M. and H. Abdul Wajid (2009). “Dispersive and dissipative behavior of the Spectral Element Method”. *SIAM Journal of Numerical Analysis* 47.5, pp. 3910–3937. DOI: 10.1137/080724976

Antonietti, P. F., I. Mazzieri, A. Quarteroni, and F. Rapetti (2012). “Non-conforming high order approximations of the elastodynamics equation”. *Computer Methods in Applied Mechanics and Engineering* 209-212, pp. 212–238. DOI: 10.1016/j.cma.2011.11.004

Arnold, D. N. (1982). “An interior penalty finite element method with discontinuous elements”. *SIAM J. Numer. Anal.* 19.4, pp. 742–760

Arnold, D. N., F. Brezzi, B. Cockburn, and L. D. Marini (2002). “Unified analysis of discontinuous Galerkin methods for elliptic problems”. *SIAM J. Numer. Anal.* 39.5, pp. 1749–1779

Atkins, H. L. and C. W. Shu (1998a). “Quadrature-free implementation of discontinuous Galerkin method for hyperbolic equations”. *AIAA Journal* 36.5, pp. 775–782

Atkins, H. L. and C. W. Shu (1998b). “Quadrature-free implementation of discontinuous Galerkin method for hyperbolic equations”. *AIAA Journal* 36.5, pp. 775–782

Balanis, C. A. (1989). *Advanced Engineering Electromagnetics.* New York: John Wiley and Sons

Beck, A. D., T. Bolemann, D. Flad, H. Frank, G. J. Gassner, F. Hindenlang, and C.-D. Munz (2014). "High-order Discontinous Galerkin Spectral Element Methods for transitional and turbulent flow simulations". *International Journal for Numerical Methods in Fluids* 76.8, pp. 522–548. DOI: 10.1002/fld.3943

Bey, K. S. and J. T. Oden (1996). "*hp*-version discontinuous Galerkin methods for hyperbolic conservation laws". *Computer Methods in Applied Mechanics and Engineering* 133.3-4, pp. 259–286. DOI: 10.1016/0045-7825(95)00944-2

Brezzi, F., G. Manzini, D. Marini, P. Pietra, and A. Russo (2000). "Discontinuous Galerkin approximations for elliptic problems". *Numer. Methods Partial Differential Equations* 16.4, pp. 365–378

Bui-Thanh, T. and O. Ghattas (2012). "Analysis of an hp-nonconforming Discontinuous Galerkin Spectral Element Method for wave propagation". *SIAM Journal of Numerical Analysis* 50.3, pp. 1801–1826. DOI: 10.1137/110828010

Canuto, C., M. Y. Hussaini, A. Quarteroni, and T. A. Zang (2007). *Spectral methods: evolution to complex geometries and applications to fluid dynamics*. Springer

Castel, N., G. Cohen, and M. Duruflé (2009). "Application of Discontinuous Galerkin spectral method on hexahedral elements for aeroacoustic". *Journal of Computational Acoustics* 17.2, pp. 175–196. DOI: 10.1142/S0218396X09003914

Castillo, P. (2002). "Performance of discontinuous Galerkin methods for elliptic PDEs". *SIAM J. Sci. Comp.* 24.2, pp. 524–547

Castillo, P. (2003). "A superconvergence result for Discontinuous Galerkin methods applied to elliptic problems". *Comput. Methods Appl. Mech. Engrg.* 192.41-42, pp. 4675–4685

Castillo, P. (2005). "An a posteriori error estimate for the local Discontinuous Galerkin method". *J. Sci. Comp.* 22.1, pp. 187–204

Castillo, P., B. Cockburn, I. Perugia, and D. Schötzau (2000). "An a priori error analysis of the local discontinuous Galerkin method for elliptic problems". *SIAM J. Numer. Anal.* 38.5, pp. 1676–1706

Castillo, P., B. Cockburn, D. Schotzau, and C. Schwab (2002). "Optimal a priori error estimates for the hp-version of the local Discontinuous Galerkin method for convection-diffusion problems". *Math. Comp.* 71.238, pp. 455–478

Chavent, G. and G. Salzano (1982). "A finite element method for the 1D water flooding problem with gravity". *J. Comput. Phys.* 45.3, pp. 307–344

Chavent, G. and B. Cockburn (1989). "The local projection P^0 P^1-Discontinuous Galerkin finite element method for scalar conservation laws". *M^2AN* 23.4, pp. 565–592

Chen, Z., B. Cockburn, J. W. Jerome, and C. W. Shu (1995). "Mixed-RKDG finite element methods for the 2-D hydrodynamics model for semiconductor device simulation". *VLSI Design* 3.2, pp. 145–158

Cockburn, B. (1999). "Discontinuous Galerkin methods for convection-dominated problems". *High-Order methods for computational physics*. Ed. by T. J. Barth and H. Deconinck. Vol. 9. Lecture Notes in Computational Science and Engineering. Springer, pp. 69–224. DOI: 10.1007/978-3-662-03882-6_2

Cockburn, B. (2003). "Discontinuous Galerkin methods. Plenary lecture presented at the 80th Annual GAMM Conference, Augsburg, 25-28 March 2002". *Z. Angew. Math. Mech.* 83.11, pp. 731–754

Cockburn, B. and C. W. Shu (1989a). "TVB Runge-Kutta Local Projection discontinuous Galerkin Finite Element Method For Conservation-Laws II. General Framework". *Math. Comp.* 52.186, pp. 411–435

Cockburn, B., S. Y. Lin, and C. W. Shu (1989b). "TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite-Element Method For Conservation-Laws III. One-Dimensional Systems". *J. Comput. Phys.* 84.1, pp. 90–113

Cockburn, B., S. C. Hou, and C. W. Shu (1990). "The Runge-Kutta Local Projection Discontinuous Galerkin Finite-Element Method For Conservation-Laws IV. The Multidimensional Case". *Math. Comp.* 54.190, pp. 545–581

Cockburn, B. and C. W. Shu (1991a). "The Runge-Kutta local projection P^1-Discontinuous Galerkin method for scalar conservation laws". *M^2AN* 25, pp. 337–361

Cockburn, B. and C. W. Shu (1991b). “The Runge-Kutta Local Projection Rho-1-Discontinuous Galerkin Finite-Element Method For Scalar Conservation-Laws”. *RAIRO-Mathematical Modelling and Numerical Analysi–Modélisation Mathématique et Analyse Numérique* 25.3, pp. 337–361

Cockburn, B. and C. W. Shu (1998a). “The local Discontinuous Galerkin method for time-dependent convection-diffusion systems”. *SIAM J. Numer. Anal.* 35.6, pp. 2440–2463

Cockburn, B. and C. W. Shu (1998b). “The Runge-Kutta Discontinuous Galerkin method for conservation laws V. Multidimensional systems”. *J. Comput. Phys.* 141.2, pp. 199–224

Cockburn, B., G. Kanschat, D. Schötzau, and C. Schwab (2002). “Local discontinuous Galerkin methods for the Stokes system”. *SIAM J. Numer. Anal.* 40, pp. 319–343

Cockburn, B., G. Karniadakis, and C. Shu (2005). *Discontinuous Galerkin methods. Theory, computation and applications.* Springer

Cockburn, B. (1998). “An introduction to the discontinuous Galerkin method for convection-dominated problems”. *Advanced numerical approximation of nonlinear hyperbolic equations*, pp. 151–268

Cockburn, B., G. E. Karniadakis, and C.-W. Shu (2000). “The development of discontinuous Galerkin methods”. *Discontinuous Galerkin Methods.* Springer, pp. 3–50

Delcourte, S. and N. Glinsky (2015). “Analysis of a high-order space and time Discontinuous Galerkin Method for elastodynamic equations. Applications to 3D wave propagation”. *ESAIM: Mathematical Modelling and Numerical Analysis* 49.4, pp. 1085–1126. DOI: 10.1051/m2an/2015001

Deville, M. O., P. F. Fischer, P. F. Fischer, E. Mund, et al. (2002). *High-order methods for incompressible fluid flow.* Vol. 9. Cambridge university press

Donea, J. and A. Huerta (2005). *Finite Element Methods for Flow Problems.* Wiley

Flad, D., A. D. Beck, and P. Guthke (2020). “A large eddy simulation method for DGSEM using non-linearly optimized relaxation filters”. *Journal of Computational Physics* 408.109303, pp. 1–12. DOI: 10.1016/j.jcp.2020.109303

Friedrich, L., A. R. Winters, D. C. Del Rey Fernández, G. J. Gassner, M. Parsani, and M. H. Carpenter (2018). “An entropy stable h/p non-conforming Discontinuous Galerkin Method with the summation-by-parts property”. *Journal of Scientific Computing* 77, pp. 689–725. DOI: 10.1007/s10915-018-0733-7

Gassner, G. J. and D. A. Kopriva (2011). “A comparison of the dispersion and dissipation errors of Gauss and Gauss-Lobatto Discontinuous Galerkin Spectral Element Methods”. *SIAM Journal of Scientific Computing* 33.5, pp. 2560–2579. DOI: 10.1137/100807211

Gassner, G. J. and A. D. Beck (2013). “On the accuracy of high-order discretizations for underresolved turbulence simulations”. *Theoretical and Computational Fluid Dynamics* 27, pp. 221–237. DOI: 10.1007/s00162-011-0253-7

Giraldo, F. X., J. S. Hesthaven, and T. Warburton (2002). “Nodal high-order Discontinuous Galerkin methods for the spherical shallow water equations”. *Journal of Computational Physics* 181.2, pp. 499–525. DOI: 10.1006/jcph.2002.7139

Giraldo, F. X. and M. Restelli (2008). “A study of spectral element and discontinuous Galerkin methods for the Navier-Stokes equations in nonhydrostatic mesoscale atmospheric modeling: equation sets and test cases”. *Journal of Computational Physics* 227.8, pp. 3849–3877. DOI: 10.1016/j.jcp.2007.12.009

Givoli, D. (1992). *Numerical Methods for Problems in Infinite Domains.* Amsterdam: Elsevier

Hairer, E., C. Lubich, and M. Roche (2006). *The numerical solution of differential-algebraic systems by Runge-Kutta methods.* Vol. 1409. Springer

Hesthaven, J. S. and T. Warburton (2002). “Nodal high-order methods on unstructured grids I. Time-domain solution of Maxwell’s equations”. *Journal of Computational Physics* 181.1, pp. 186–221

Hesthaven, J. S. and T. Warburton (2008a). “Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications”. Vol. 54. Texts in Applied Mathematics. New York: Springer

Hesthaven, J. S. and T. Warburton (2008b). *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications.* Vol. 54. Texts in Applied Mathematics. New York: Springer

Hesthaven, J. and T Warburton (2004). "Discontinuous Galerkin methods for the time-domain Maxwell's equations". *ACES Newsletter* 19, pp. 10–29

Houston, P., C. Schwab, and E. Süli (2002). "Discontinuous *hp*-Finite Element Methods for advection-diffusion-reaction problems". *SIAM Journal of Numerical Analysis* 39.6, pp. 2133–2163. DOI: 10.1137/S0036142900374111

Hu, F. Q., M. Y. Hussaini, and P. Rasetarinera (1999). "An analysis of the Discontinuous Galerkin Method for wave propagation problems". *Journal of Computational Physics* 151.2, pp. 921–946. DOI: 10.1006/jcph.1999.6227

Jameson, A., W. Schmidt, and E. Turkel (1981). "Numerical solution of the Euler equations using Runge-Kutta time-stepping schemes". *Proceedings of the 14th AIAA Fluid and Plasma Dynamic Conference*. AIAA. Palo Alto, California

Johnson, C. and J. Pitkaranta (1986). "An Analysis Of The discontinuous Galerkin Method For A Scalar Hyperbolic Equation". *Math. Comp.* 46.173, pp. 1–26

Kabakian, A. V., V. Shankar, and W. F. Hall (2004). "Unstructured grid-based discontinuous Galerkin method for broadband electromagnetic simulations". *Journal of Scientific Computing* 20.3, pp. 405–431

Kent, J. (2019). "Capturing the cross-terms in multidimensional advection schemes". *International Journal for Numerical Methods in Fluids* 91.2, pp. 49–62

Kopriva, D. A. and G. Gassner (2010). "On the quadrature and weak form choices in collocation type discontinuous Galerkin spectral element methods". *SIAM Journal of Scientific Computing* 44, pp. 136–155. DOI: 10.1007/s10915-010-9372-3

Laskowski, W., A. M. Rueda-Ramirez, G. Rubio, E. Valero, and E. Ferrer (2020). "Advantages of static condensation in implicit compressible Navier Stokes DGSEM solvers". *Computers & Fluids* 209.104646, pp. 1–17. DOI: 10.1016/j.compfluid.2020.104646

Lesaint, P. and P. A. Raviart (1974). "On a finite element method for solving the neutron transport equation". *Mathematical aspects of finite elements in PDE*. Ed. by C. de Boor. Academic Press, pp. 89–123

LeVeque, R. J. (1992). *Numerical methods for conservation laws*. Second. Lectures in Mathematics ETH Zürich. Basel: Birkhäuser Verlag, pp. x+214

Leveque, R. J. (1996). "High-resolution conservative algorithms for advection in incompressible flow". *SIAM Journal on Numerical Analysis* 33.2, pp. 627–665

LeVeque, R. J. (2002). *Finite volume methods for hyperbolic problems*. Cambridge university press

Li, B. Q. (2006). *Discontinuous Finite Elements in Fluid Dynamics and Heat Transfer (Computational Fluid and Solid Mechanics)*. Springer

Lilienthal, M. (2015). "Error controlled hp-adaptive Finite Element Methods for the time-dependent Maxwell equations". PhD thesis. Germany: Technische Universität Darmstadt

Malm, J., P. Schlatter, P. F. Fischer, and D. S. Henningson (2013). "Stabilization of the spectral element method in convection dominated flows by recovery of skew-symmetry". *Journal of Scientific Computing* 57, pp. 254–277

Noventa, G., F. Massa, F. Bassi, A. Colombo, N. Franchina, and A. Ghidoni (2016). "A high-order Discontinuous Galerkin solver for unsteady incompressible turbulent flows". *Computers & Fluids* 139, pp. 248–260

Osher, S. (1984). "Riemann Solvers, The Entropy Condition, And Difference Approximations". *SIAM J. Numer. Anal.* 21.2, pp. 217–235

Peraire, J. and P.-O. Persson (2008). "The Compact Discontinuous Galerkin (CDG) Method for Elliptic Problems". *SIAM J. Sci. Comput.* 30.4, pp. 1806–1824

Persson, P.-O. and J. Peraire (2006a). "An efficient low memory implicit DG algorithm for time dependent problems". *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA

Persson, P.-O. and J. Peraire (2006b). "Sub-cell shock capturing for discontinuous Galerkin Methods". *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA

Persson, P.-O. and J. Peraire (2009). "Curved Mesh Generation and Mesh Refinement using Lagrangian Solid Mechanics". *Proceedings of the 47th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA

Peterson, T. (1991). "A Note on the Convergence of the Discontinuous Galerkin Method for a Scalar Hyperbolic Equation". *SIAM J. Numer. Anal.* 28.1, pp. 133–140

Pomponet Oliveira, S. and S. Angelozi Leite (2018). "Error analysis of the spectral element method with Gauss-Lobatto-Legendre points for the acoustic wave equation in heterogeneous media". *Appl. Numer. Math.* 129, pp. 39–57. DOI: 10.1016/j.apnum.2018.02.007

Rasetarinera, P., M. Y. Hussaini, and F. Q. Hu (2000). "Some remarks on the accuracy of a Discontinuous Galerkin Method". *Discontinuous Galerkin methods. Theory, computation and applications.* Ed. by B. Cockburn, G. Karniadakis, and C. W. Shu. Springer-Verlag, pp. 407–412

Reed, W. H. and T. R. Hill (1973). *Triangular mesh methods for the neutron transport equation.* Tech. rep. LA-UR-73-479. Los Alamos: Los Alamos Scientific Laboratory

Renac, F. (2019). "Entropy stable DGSEM for nonlinear hyperbolic systems in nonconservative form with application to two-phase flows". *Journal of Computational Physics* 382, pp. 1–26. DOI: 10.1016/j.jcp.2018.12.035

Rider, W. J. and R. B. Lowrie (2002). "The use of classical Lax-Friedrichs Riemann solvers with discontinuous Galerkin methods". *Int. J. Numer. Meth. Engng. Fluids* 40.3-4, pp. 479–486

Sármány, D., M. A. Botchev, and J. J. W. van der Vegt (2007). "Dispersion and Dissipation Error in High-Order Runge-Kutta Discontinuous Galerkin Discretisations of the Maxwell Equations". *J. Sci. Comp.* 33.1, pp. 47–74

Sevilla, R., S. Fernández–Méndez, and A. Huerta (2011). "Comparison of high–order curved finite elements". *International Journal for Numerical Methods in Engineering* 87.8, pp. 719–734

Sevilla, R., O. Hassan, and K. Morgan (2014). "The Use of Hybrid Meshes to Improve the Efficiency of a Discontinuous Galerkin Method for the Solution of Maxwell's Equations". *Computers & Structures* 137, pp. 2–13

Sherwin, S. (2000). "Dispersion analysis of the continuous and Discontinuous Galerkin formulations". *Discontinuous Galerkin methods. Theory, computation and applications.* Ed. by B. Cockburn, G. Karniadakis, and C. W. Shu. Springer-Verlag, pp. 425–431

Sherwin, S. J., R. M. Kirby, J. Peiro, L. R. Taylor, and O. C. Zienkiewicz (2006). "On 2D elliptic discontinuous Galerkin methods". *Int. J. Numer. Meth. Engng.* 65.5, pp. 752–784

Shu, C. W. (2003). "High-order finite difference and finite volume WENO schemes and Discontinuous Galerkin methods for CFD". *Int. J. Comput. Fluid Dyn.* 17.2, pp. 107–118

Terrana, S., J.-P. Vilotte, and L. Guillot (2018). "A spectral hybridizable Discontinuous Galerkin Method for elastic-acoustic wave propagation". *Geophysical Journal International* 213.1, pp. 574–602. DOI: 10.1093/gji/ggx557

Ven, H. V. der and J. J. W. V. der Vegt (2002). "Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows II. Efficient flux quadrature". *Comput. Methods Appl. Mech. Engrg.* 191.41-42, pp. 4747–4780

Warburton, T. C., I. Lomtev, Y. Du, S. J. Sherwin, and G. E. Karniadakis (1999). "Galerkin and discontinuous Galerkin spectral/*hp* methods". *Computer Methods in Applied Mechanics and Engineering* 175.3-4, pp. 343–359. DOI: 10.1016/S0045-7825(98)00360-0

Wilcox, L. C., G. Stadler, C. Burstedde, and O. Ghattas (2010). "A high-order Discontinuous Galerkin Method for wave propagation through coupled elastic-acoustic media". *Journal of Computational Physics* 229.24, pp. 9373–9396. DOI: 10.1016/j.jcp.2010.09.008

Winters, A. R. and D. A. Kopriva (2014). "ALE-DGSEM approximation of wave reflection and transmission from a moving medium". *Journal of Computational Physics* 263, pp. 233–267. DOI: 10.1016/j.jcp.2014.01.022

Zienkiewicz, O. C. and R. L.Taylor (2000). *The Finite Element Method.* Fifth. Vol. 1. The basis. Butterwort-Heinemann

Zienkiewicz, O. C., R. L. Taylor, S. J. Sherwin, and J. Peiro (2003). "On discontinuous Galerkin methods". *Int. J. Numer. Meth. Engng.* 58.8, pp. 1119–1148

Zienkiewicz, O. C., R. L. Taylor, O. C. Zienkiewicz, and R. L. Taylor (1977). *The finite element method.* Vol. 3. McGraw-hill London