Discretely entropy stable weight-adjusted discontinuous Galerkin methods: curvilinear meshes and GPU acceleration

Jesse Chan, Lucas Wilcox

Abstract

Things to include: entropy conservation and stability on curvilinear meshes, choosing geometric terms to ensure constant state preservation, high order accuracy. Computational considerations for curvilinear meshes.

Contents

1 Introduction

Introducing WADG [1, 2].

Matrix WADG: GPU efficiency of WADG vs storing weighted inverses [3].

2 Systems of nonlinear conservation laws

This work addresses is high order schemes for the following system of n nonlinear conservation laws in d dimensions

$$\frac{\partial \boldsymbol{u}}{\partial t} + \sum_{j=1}^{d} \frac{\partial \boldsymbol{f}_j(\boldsymbol{u})}{\partial x_j} = 0, \qquad \boldsymbol{u} \in \mathbb{R}^n, \qquad \boldsymbol{f}_j : \mathbb{R}^n \to \mathbb{R}^n, \tag{1}$$

where $\boldsymbol{u}(\boldsymbol{x}, t)$ denote the *conservative variables* for this system. We will specify boundary conditions in a later section.

We are interested in nonlinear conservation laws for which an entropy function $U(\boldsymbol{u})$ exists, where $U(\boldsymbol{u})$ is convex with respect to the conservative variables $\boldsymbol{u}$. If this function exists, then it is possible to define *entropy variables* $\boldsymbol{v}(\boldsymbol{u}) = \frac{\partial U}{\partial \boldsymbol{u}}$. These functions symmetrize the system of nonlinear conservation laws (1) [4].

It can be shown (see, for example, [5]) that symmetrization is equivalent to the existence of an entropy flux function $F(\boldsymbol{u})$ and entropy potential ψ such that

$$\boldsymbol{v}^T \frac{\partial \boldsymbol{f}_j}{\partial \boldsymbol{u}} = \frac{\partial F_j(\boldsymbol{u})}{\partial \boldsymbol{u}}^T, \qquad \psi_j(\boldsymbol{v}) = \boldsymbol{v}^T \boldsymbol{f}_j(\boldsymbol{u}(\boldsymbol{v})) - F_j(\boldsymbol{u}(\boldsymbol{v})), \qquad \psi_j'(\boldsymbol{v}) = \boldsymbol{f}_j(\boldsymbol{u}(\boldsymbol{v})).$$

Smooth solutions of (1) can be shown to satisfy a conservation of entropy by multiplying (1) by $\boldsymbol{v}(\boldsymbol{u})$. Using the definition of the entropy variables, entropy flux, and the chain rule yields

$$\boldsymbol{v}^T \frac{\partial \boldsymbol{f}_j(\boldsymbol{u})}{\partial x_j} = \frac{\partial U(\boldsymbol{u})}{\partial \boldsymbol{u}}^T \frac{\partial \boldsymbol{f}_j(\boldsymbol{u})}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial x_j} = \frac{\partial F_j(\boldsymbol{u})}{\partial x_j}, \tag{2}$$

and that

$$\frac{\partial U(\boldsymbol{u})}{\partial t} + \sum_{j=1}^{d} \frac{\partial F_j(\boldsymbol{u})}{\partial x_j} = 0.$$

Let $\Omega\mathbb{R}^d$ now be a closed domain with boundary $\partial\Omega$. Integrating over Ω an using Gauss' theorem on the spatial derivative yields

$$\int_\Omega \frac{\partial U(\boldsymbol{u})}{\partial t} \,\mathrm{d}x + \int_{\partial\Omega} \sum_{j=1}^{d} \left(\boldsymbol{v}(\boldsymbol{u})^T \boldsymbol{f}_j(\boldsymbol{u}) - \psi_j\left(\boldsymbol{v}(\boldsymbol{u})\right) \right) n_j \,\mathrm{d}x = 0, \tag{3}$$

where $\boldsymbol{n} = (n_1, \ldots, n_d)^T$ denotes the unit outward normal vector on $\partial\Omega$.

General solutions (including non-smooth solutions such as shocks) satisfy an entropy *inequality*

$$\int_\Omega \frac{\partial U(\boldsymbol{u})}{\partial t} \,\mathrm{d}x + \int_{\partial\Omega} \sum_{j=1}^{d} \left(\boldsymbol{v}(\boldsymbol{u})^T \boldsymbol{f}_j(\boldsymbol{u}) - \psi_j\left(\boldsymbol{v}(\boldsymbol{u})\right) \right) n_j \,\mathrm{d}x \leq 0, \tag{4}$$

which results from considering solutions of an appropriate viscous form of the equations (1) and taking the limit as viscosity vanishes. In this work, schemes which satisfy a discrete form of (4) will be constructed by first enforcing a discrete version of entropy conservation (3), then adding an appropriate numerical dissipation which will enforce the entropy inequality (4).

2.1 Standard DG formulations for nonlinear conservation laws

We begin by reviewing the construction of standard high order DG formulations for (1).

2.1.1 Mathematical notation

Let the domain $\Omega \subset \mathbb{R}^d$ be decomposed into elements (subdomains) D^k, and let $\widehat{D}$ denote a d-dimensional reference element with boundary $\partial\widehat{D}$. Let $\widehat{\boldsymbol{x}}$ denote coordinates on $\widehat{D}$, and let $\widehat{n}_i$ denote and the ith component of the unit normal vector on $\partial\widehat{D}$. We assume that $\widehat{n}_i$ is constant; i.e., that the faces of the reference element are planar (this assumption holds for all commonly used reference elements [6]).

We will assume that each physical element D^k is the image of $\widehat{D}$ under some smoothly differentiable mapping $\boldsymbol{\Phi}_k(\widehat{\boldsymbol{x}})$ such that

$$\boldsymbol{x} = \boldsymbol{\Phi}_k(\widehat{\boldsymbol{x}}), \qquad \boldsymbol{x} \in D^k.$$

This also implies that integrals over physical elements can be mapped back to the reference element as follows

$$\int_{D^k} u \,\mathrm{d}\boldsymbol{x} = \int_{\widehat{D}} u J^k \,\mathrm{d}\widehat{\boldsymbol{x}},$$

where J^k denotes the determinant of the Jacobian of $\boldsymbol{\Phi}_k$. Integrals over physical faces of D^k can similarly be mapped back to reference faces.

We define an approximation space using degree N polynomials on the reference element. For example, on a d-dimensional reference simplex, the natural polynomial space are total degree N polynomials

$$P^N\left(\widehat{D}\right) = \left\{ \widehat{x}_1^{i_1} \dots \widehat{x}_d^{i_d}, \quad \widehat{\boldsymbol{x}} \in \widehat{D}, \quad 0 \leq \sum_{k=1}^{d} i_k \leq N \right\}.$$

Other element types possess different natural polynomial spaces [6], but typically contain the space of total degree N polynomials. This work is directly applicable to other elements and spaces as well. We denote the dimension of the approximation space P^N as $N_p = \dim\left(P^N\left(\widehat{D}\right)\right)$.

Finally, we define the L^2 norm and inner products over the reference element $\widehat{D}$ and the surface of the reference element $\partial\widehat{D}$

$$(\boldsymbol{u}, \boldsymbol{v})_{\widehat{D}} = \int_{\widehat{D}} \boldsymbol{u} \cdot \boldsymbol{v} \,\mathrm{d}\boldsymbol{x} = \int_{\widehat{D}} \boldsymbol{u} \cdot \boldsymbol{v} J^k \,\mathrm{d}\widehat{x}, \qquad \|\boldsymbol{u}\|_{\widehat{D}}^2 = (\boldsymbol{u}, \boldsymbol{u})_{\widehat{D}}, \qquad \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\partial\widehat{D}} = \int_{\partial\widehat{D}} \boldsymbol{u} \cdot \boldsymbol{v} \,\mathrm{d}\boldsymbol{x},$$

Finally, we introduce the continuous L^2 projection operator Π_N and lifting operator L. For $u \in L^2\left(\widehat{D}\right)$, the L^2 projection $\Pi_N u$ is defined through

$$\int_{\widehat{D}} \Pi_N u v \,\mathrm{d}\widehat{\boldsymbol{x}} = \int_{\widehat{D}} u v \,\mathrm{d}\widehat{\boldsymbol{x}}, \qquad \forall v \in P^N\left(\widehat{D}\right). \tag{5}$$

Likewise, for a boundary function $u \in L^2\left(\partial\widehat{D}\right)$, the lifting operator L [7, 8] is defined through

$$(Lu, v)_{\widehat{D}} = \langle u, v \rangle_{\partial\widehat{D}}, \qquad \forall v \in P^N\left(\widehat{D}\right). \tag{6}$$

2.1.2 Discontinuous Galerkin formulations and the L^2 projection

Discontinuous Galerkin methods have been very widely applied to systems of nonlinear conservation laws (1) [9, 10, 11, 12]. The development of new discontinuous Galerkin methods for nonlinear conservation laws have focused heavily on the choice of numerical flux [13] or the development of slope limiters [14, 15, 16] and artificial viscosity strategies [17, 18, 19]. However, the treatment of the underlying volume discretization remains relatively unchanged between each of these approaches.

Ignoring terms involving filters, limiters, or artificial viscosity, a semi-discrete "weak" DG formulation for (1) can be given locally over an element D^k: find $\boldsymbol{u} \in \left(P^N\left(D^k\right)\right)^n$ such that

$$\int_{D^k} \left(\frac{\partial \boldsymbol{u}}{\partial t} \cdot \boldsymbol{v} - \sum_{j=1}^{d} \boldsymbol{f}_j(\boldsymbol{u}) \cdot \frac{\partial \boldsymbol{v}}{\partial x_i} \right) \mathrm{d}\boldsymbol{x} + \sum_{j=1}^{d} \int_{\partial D^k} \left(\boldsymbol{f}_j^*\left(\boldsymbol{u}^+, \boldsymbol{u}\right) \right) \cdot \boldsymbol{v} n_j \,\mathrm{d}\boldsymbol{x} = 0, \qquad \forall \boldsymbol{v} \in \left(P^N\left(D^k\right)\right)^n, \tag{7}$$

where the numerical flux $\boldsymbol{f}^*$ is a function of the solution $\boldsymbol{u}$ on both D^k and neighboring elements.

Unfortunately, solutions to (7) do not (in general) obey a discrete version of the entropy inequality (4). Since (4) is a generalized statement of energy stability, the lack of a discrete entropy inequality implies that the discrete solution can blow up in finite time. The reason for this is due the fact that, in practice, the integrals in (7) are not computed exactly and are instead approximated using polynomially exact quadratures. This is compounded by the fact that the nonlinear flux function $\boldsymbol{f}_j(\boldsymbol{u})$ is often rational and impossible to integrate exactly using polynomial quadratures. To account for the inexactness of quadrature, we rewrite (7) in strong form using a discrete quadrature-based L^2 projection.

For polynomial approximation spaces, $\frac{\partial \boldsymbol{v}}{\partial x_i}$ is polynomial. Then, mapping (7) back to the reference element $\widehat{D}$ and using the L^2 projection and (5), we have that

$$\int_{D^k} \boldsymbol{f}_j(\boldsymbol{u}) \cdot \frac{\partial \boldsymbol{v}}{\partial x_i} \,\mathrm{d}\boldsymbol{x} = \int_{\widehat{D}} \Pi_N \boldsymbol{f}_j(\boldsymbol{u}) \cdot \frac{\partial \boldsymbol{v}}{\partial x_i} J^k \,\mathrm{d}\boldsymbol{x}.$$

Thus, integrating by parts (7) recovers a "strong" DG formulation involving the projection operator

$$\begin{aligned}
&\int_{D^k} \left(\frac{\partial \boldsymbol{u}}{\partial t} - \sum_{j=1}^{d} \frac{\partial \Pi_N \boldsymbol{f}_j(\boldsymbol{u})}{\partial x_j} \right) \cdot \boldsymbol{v} \,\mathrm{d}\boldsymbol{x} \\
&+ \sum_{j=1}^{d} \int_{\partial D^k} \left(\boldsymbol{f}_j^* \left(\boldsymbol{u}^+, \boldsymbol{u} \right) - \Pi_N \boldsymbol{f}_j(\boldsymbol{u}) \right) \cdot \boldsymbol{v} n_j \,\mathrm{d}\boldsymbol{x} = 0, \qquad \forall \boldsymbol{v} \in \left(P^N \left(D^k \right) \right)^n .
\end{aligned} \tag{8}$$

From this, we see that our discrete scheme does not differentiation the nonlinear flux function $\boldsymbol{f}_j(\boldsymbol{u})$ exactly, but instead differentiates the projection of $\Pi_N \boldsymbol{f}_j(\boldsymbol{u})$ onto polynomials of degree N. Because the L^2 projection operator is introduced, the chain rule no longer holds at the discrete level and step (2) of the proof of entropy conservation is no longer valid. Thus, ensuring discrete entropy stability will require a discrete formulation of the system of nonlinear conservation laws (1) from which we can prove a discrete entropy inequality without relying on the chain rule.

3 Discretely entropy stable DG methods on curved meshes

We will first show how to construct discretely entropy stable high order DG methods on curvilinear meshes, but will present this using a matrix formulation as opposed to a continuous formulation. This is to ensure that the effects of discretization, nonlinear, and quadrature are accounted for in the proof of semi-discrete entropy stability. We first introduce quadrature-based matrices, which we will then use to construct discretely entropy stable DG formulations.

3.1 Basis and quadrature rules

We now introduce quadrature-based matrices for the d-dimensional reference element $\widehat{D}$, which we will use to construct matrix-vector formulations of DG methods. Assuming $u(\boldsymbol{x}) \in P^N\left(\widehat{D}\right)$, it can be represented in terms of the vector of coefficients $\boldsymbol{u}$ using some polynomial basis ϕ_i of degree N and dimension N_p

$$u(\boldsymbol{x}) = \sum_{j=1}^{N_p} \boldsymbol{u}_j \phi_j(\widehat{\boldsymbol{x}}), \qquad P^N\left(\widehat{D}\right) = \operatorname{span}\left\{\phi_i(\widehat{x})\right\}_{i=1}^{N_p}.$$

We construct quadrature-based based on ϕ_i and appropriate volume and surface quadrature rules. The volume and surface quadrature rules are given by points and positive weights $\{(\boldsymbol{x}_i, w_i)\}_{i=1}^{N_q}$ and $\left\{(\boldsymbol{x}_i^f, w_i^f)\right\}_{i=1}^{N_q^f}$, respectively. We make the following assumptions on the strength of these quadratures:

Assumption 1 (Integration by parts under quadrature)**.** *The volume quadrature rule* $\{(\boldsymbol{x}_i, w_i)\}_{i=1}^{N_q}$ *is exact for polynomials of degree* $2N-1$*. Additionally, for any* $u, v \in P^N\left(\widehat{D}\right)$*, integration by parts*

$$\left(\frac{\partial u}{\partial x_i}, v\right)_{\widehat{D}} = \langle u, v n_i \rangle_{\partial \widehat{D}} - \left(u, \frac{\partial v}{\partial x_i}\right)_{\widehat{D}}$$

holds when volume and surface integrals are approximated using quadrature.

Assumption (1) holds, for example, for any surface quadrature rule which is exact for degree $2N$ polynomials on the boundary of the reference element $\partial\widehat{D}$.

3.2 Reference element matrices

Let $\boldsymbol{W}, \boldsymbol{W}_f$ denote diagonal matrices whose entries are volume and surface quadrature weights, respectively. The surface quadrature weights are given by quadrature weights on reference faces, which are mapped to faces of the reference element. We define the volume and surface quadrature interpolation matrices $\boldsymbol{V}_q$ and $\boldsymbol{V}_f$

$$\begin{aligned}
(\boldsymbol{V}_q)_{ij} &= \phi_j(\widehat{\boldsymbol{x}}_i), \qquad 1 \le j \le N_p, \qquad 1 \le i \le N_q, \\
(\boldsymbol{V}_f)_{ij} &= \phi_j(\widehat{\boldsymbol{x}}_i^f), \qquad 1 \le j \le N_p, \qquad 1 \le i \le N_q^f,
\end{aligned} \tag{9}$$

which map coefficients $\boldsymbol{u}$ to evaluations of u at volume and surface quadrature points.

Next, let $\boldsymbol{D}_i$ denote the differentiation matrix with respect to the ith coordinate, defined implicitly through the relation

$$u(\widehat{\boldsymbol{x}}) = \sum_{j=1}^{N_p} \boldsymbol{u}_j \phi_j(\widehat{\boldsymbol{x}}), \qquad \frac{\partial u}{\partial \widehat{\boldsymbol{x}}_i} = \sum_{j=1}^{N_p} \left(\boldsymbol{D}_i \boldsymbol{u}\right)_j \phi_j(\widehat{\boldsymbol{x}}).$$

The matrix $\boldsymbol{D}_i$ maps basis coefficients of some polynomial $u \in P^N$ to coefficients of its ith derivative with respect to the reference coordinate $\widehat{\boldsymbol{x}}$, and is sometimes referred to as a "modal" differentiation matrix (with respect to a general non-nodal "modal" basis [20]).

Using the volume quadrature interpolation matrix $\boldsymbol{V}_q$, we can compute a quadrature-based mass matrix $\boldsymbol{M}$ by evaluating L^2 inner products of different basis functions using quadrature

$$\boldsymbol{M} = \boldsymbol{V}_q^T \boldsymbol{W} \boldsymbol{V}_q, \qquad \boldsymbol{M}_{ij} = \sum_{k=1}^{N_q} w_k \phi_j(\boldsymbol{x}_k) \phi_i(\boldsymbol{x}_k) \approx \int_{\widehat{D}} \phi_j \phi_i \, \mathrm{d}\widehat{\boldsymbol{x}} = (\phi_j, \phi_i)_{\widehat{D}}.$$

The approximation in the formula for the mass matrix becomes an equality if the volume quadrature rule is exact for polynomials of degree $2N$. The mass matrix is symmetric and positive definite under Assumption 1; however, we do not make any distinctions between diagonal and dense (lumped) mass matrices in this work.

The mass matrix appears in the discretization of L^2 projection (5) and lift operators (6) using quadrature. The result are quadrature-based L^2 projection and lift operators $\boldsymbol{P}_q, \boldsymbol{L}_q$,

$$\boldsymbol{P}_q = \boldsymbol{M}^{-1} \boldsymbol{V}_q^T \boldsymbol{W}, \qquad \boldsymbol{L}_q = \boldsymbol{M}^{-1} \boldsymbol{V}_f^T \boldsymbol{W}_f, \tag{10}$$

which are discretizations of the continuous L^2 projection operator Π_N and continuous lift operator L. The matrix $\boldsymbol{P}_q$ maps a function (in terms of its evaluation at quadrature points) to coefficients of the L^2 projection in the basis $\phi_j(x)$, while the matrix $\boldsymbol{L}_q$ "lifts" a function (evaluated at surface quadrature points) from the boundary of an element to coefficients of a basis defined in the interior of the element.

Finally, we introduce quadrature-based operators $\boldsymbol{D}_N^i$ which will be used to construct discretizations of our nonlinear conservation laws. This operator was introduced in [21] as a "decoupled summation by parts" operator

$$\boldsymbol{D}_N^i = \begin{bmatrix} \boldsymbol{V}_q \boldsymbol{D}_i \boldsymbol{P}_q - \frac{1}{2} \boldsymbol{V}_q \boldsymbol{L}_q \mathrm{diag}\left(\widehat{\boldsymbol{n}}_i \widehat{\boldsymbol{J}}_f\right) \boldsymbol{V}_f \boldsymbol{P}_q & \frac{1}{2} \boldsymbol{V}_q \boldsymbol{L}_q \mathrm{diag}\left(\widehat{\boldsymbol{n}}_i \widehat{\boldsymbol{J}}_f\right) \\ -\frac{1}{2} \mathrm{diag}\left(\widehat{\boldsymbol{n}}_i \widehat{\boldsymbol{J}}_f\right) \boldsymbol{V}_f \boldsymbol{P}_q & \frac{1}{2} \mathrm{diag}\left(\widehat{\boldsymbol{n}}_i \widehat{\boldsymbol{J}}_f\right) \end{bmatrix} \tag{11}$$

where $\widehat{\boldsymbol{n}}_i$ is the vector containing values of the ith component of the unit normal on the surface of the reference element $\widehat{D}$, and $\widehat{\boldsymbol{J}}_f$ contains values of the face Jacobian factor $\widehat{J}_f$ which result from mapping a face of $\widehat{D}$ to a reference face. When combined with projection and lifting matrices, $\boldsymbol{D}_N^i$ produces a high order approximation of non-conservative products. Let $\boldsymbol{f}, \boldsymbol{g}$ denote vectors containing the evaluation of functions $f(\boldsymbol{x}), g(\boldsymbol{x})$ at both volume and surface quadrature points

$$\begin{bmatrix} \boldsymbol{P}_q & \boldsymbol{L}_q \end{bmatrix} \text{diag}\left(\boldsymbol{f}\right) \boldsymbol{D}_N^i \boldsymbol{g} \approx f \frac{\partial g}{\partial \widehat{x}_i}.$$

It was shown in [21] that the matrix $\boldsymbol{D}_N^i$ satisfies several key properties. First, it can be observed that $\boldsymbol{D}_N^i \mathbf{1} = 0$, where $\mathbf{1}$ is the vector of all ones. Second, $\boldsymbol{D}_N^i$ satisfies a summation-by-parts property. Let $\boldsymbol{Q}_N^i$ be the scaling of $\boldsymbol{D}_N^i$ by the diagonal matrix of volume and surface quadrature weights

$$\boldsymbol{Q}_N^i = \boldsymbol{W}_N \boldsymbol{D}_N^i, \qquad \boldsymbol{W}_N = \begin{pmatrix} \boldsymbol{W} & \\ & \boldsymbol{W}_f \end{pmatrix}.$$

Then, $\boldsymbol{Q}_N^i$ satisfies the following discrete analogue of integration by parts

$$\boldsymbol{Q}_N^i + \left(\boldsymbol{Q}_N^i\right)^T = \boldsymbol{B}_N^i, \qquad \boldsymbol{B}_N = \begin{pmatrix} \mathbf{0} & \\ & \boldsymbol{W}_f \text{diag}\left(\widehat{\boldsymbol{n}}_i \widehat{\boldsymbol{J}}_f\right) \end{pmatrix}. \tag{12}$$

3.3 Matrices on curved physical elements

The key difference between curvilinear and affine meshes is that geometric terms now vary spatially over each element. In practice, derivatives are computed over the reference element and mapped to the physical element D^k through a change of variables formula

$$\frac{\partial u}{\partial x_i} = \sum_{j=1}^{d} \boldsymbol{G}_{ij}^k \frac{\partial u}{\partial \widehat{x}_j}, \qquad \boldsymbol{G}_{ij}^k = \frac{\partial \widehat{x}_j}{\partial x_i}.$$

where we have defined the elements of the matrix $\boldsymbol{G}^k$ as the derivatives of the reference coordinates $\widehat{x}_j$ with respect to the physical coordinates x_i on D^k. We refer to the determinant of $\boldsymbol{G}^k$ as J^k, and denote evaluations of $J^k \boldsymbol{G}_{ij}^k$ at both volume and surface quadrature points as the vector $\boldsymbol{J}\boldsymbol{G}_{ij}^k$.

We assume in this work that the mesh is stationary. It can be shown at the continuous level that, for any differentiable and invertible mapping, the quantity $J^k \boldsymbol{G}^k$ satisfies a geometric conservation law (GCL) [22, 23]

$$\sum_{j=1}^{d} \frac{\partial}{\partial \widehat{x}_j} J^k \boldsymbol{G}_{ij}^k = 0, \tag{13}$$

or that $\widehat{\nabla} \cdot \left(\left(J^k \boldsymbol{G}^k\right)^T\right) = 0$. Using (13), the scaled physical derivative $J^k \frac{\partial u}{\partial x_i}$ can be computed via

$$J^k \frac{\partial u}{\partial x_i} = \frac{1}{2} \sum_{j=1}^{d} \left(J^k \boldsymbol{G}_{ij}^k \frac{\partial u}{\partial \widehat{x}_j} + \frac{\partial \left(J^k \boldsymbol{G}_{ij}^k u\right)}{\partial \widehat{x}_j} \right). \tag{14}$$

We will require two additional assumptions on the mesh, geometric terms, and normal vectors for this work:

Assumption 2 (Consistency of normals)**.** *We assume that normals are consistent across neighboring elements. Let $\boldsymbol{D}^k$ and $\boldsymbol{D}^{k,+}$ be adjacent elements sharing a face f. Then, the scaled outward normal vectors for each element are equal and opposite such that $\boldsymbol{n} J_f^k = -\boldsymbol{n}^+ J_f^{k,+}$.*

We also assume that the scaled matrix of geometric terms transforms scaled reference normal vectors to scaled physical normals, such that

$$\sum_{j=1}^{d} \boldsymbol{JG}_{ij} \widehat{n}_j \widehat{J}_f = n_i J_f^k, \qquad \left(\sum_{j=1}^{d} \left(\boldsymbol{JG}_{ij}^k \right) \circ \left(\widehat{\boldsymbol{n}}_j \widehat{\boldsymbol{J}}_f \right) \right) = \left(\boldsymbol{n}_i \boldsymbol{J}_f^k \right). \tag{15}$$

The property (15) holds at the continuous level [24], and thus at all points where the geometric terms are computed exactly. However, we will also consider cases where the geometric terms $J^k \boldsymbol{G}_{ij}^k$ are modified to enforce a discrete form of (13); in these situations, it will be important to ensure that (15) holds after modifications.

We define physical differentiation matrices based on the approximation of (14). Define $\boldsymbol{D}_k^i$ as

$$\boldsymbol{D}_k^i = \sum_{j=1}^{d} \text{diag}\left(\boldsymbol{JG}_{ij}^k \right) \boldsymbol{D}_N^j + \boldsymbol{D}_N^j \text{diag}\left(\boldsymbol{JG}_{ij}^k \right).$$

Using properties of the Hadamard product [25], we can rewrite $\boldsymbol{D}_k^i$ as

$$\boldsymbol{D}_k^i = \sum_{j=1}^{d} \left(\boldsymbol{D}_N^j \circ \left\{ \left\{ \boldsymbol{JG}_{ij}^k \right\} \right\} \right), \qquad \left\{ \left\{ \boldsymbol{JG}_{ij}^k \right\} \right\}_{mn} = \frac{1}{2} \left(\left(\boldsymbol{JG}_{ij}^k \right)_m + \left(\boldsymbol{JG}_{ij}^k \right)_n \right), \tag{16}$$

where $\left\{ \left\{ \boldsymbol{JG}_{ij}^k \right\} \right\}$ denotes the matrix of averages between each of the entries of $\boldsymbol{JG}_{ij}^k$. From (16), it is straightforward to show that (because $\left\{ \left\{ \boldsymbol{JG}_{ij}^k \right\} \right\}$ is symmetric) $\boldsymbol{Q}_k^i = \boldsymbol{W}_N \boldsymbol{D}_k^i$ also satisfies a summation by parts property

$$\boldsymbol{Q}_k^i + \left(\boldsymbol{Q}_k^i \right)^T = \boldsymbol{B}_k^i, \qquad \boldsymbol{B}_k^i = \begin{pmatrix} \boldsymbol{0} & \\ & \boldsymbol{W}_f \text{diag}\left(\boldsymbol{n}_i \boldsymbol{J}_f^k \right) \end{pmatrix}, \tag{17}$$

where $\boldsymbol{n}_i$ and $\boldsymbol{J}_f^k$ are vectors containing evaluations of the physical unit normals and face Jacobian factors for D^k at surface quadrature points, respectively.

Curvilinear mappings also imply that integrals over each physical element D^k are no longer simple scalings of integrals over $\widehat{D}$. The L^2 projection of $u \in L^2\left(D^k\right)$ over a curvilinear element D^k is defined through

$$\left(\Pi_N^k u, v \right)_{D^k} = \left(u, v \right)_{D^k}. \tag{18}$$

Mapping integrals to the reference element $\widehat{D}$ yields

$$\left(\Pi_N^k u, vJ \right)_{\widehat{D}} = \left(u, vJ \right)_{\widehat{D}}. \tag{19}$$

For affine elements, J is constant and can be cancelled. Thus, the L^2 projection over affine elements is equivalent to simply taking the L^2 projection of a function over the reference element. However, for curved elements, J acts as a spatially varying weight within the L^2 inner product.

Discretizing (19) requires a weighted mass matrix. We define a curved mass matrix over an element D^k by weighting the discrete L^2 norm with values of J at quadrature points

$$\boldsymbol{M}^k = \boldsymbol{V}_q^T \boldsymbol{W}_{J^k} \boldsymbol{V}_q, \qquad \left(\boldsymbol{W}_J \right)_{ij} = \delta_{ij} w_i J^k(\boldsymbol{x}_i), \qquad i = 1, \ldots, N_q. \tag{20}$$

Then, a curvilinear L^2 projection and lift matrices can be defined in a manner analogous to (10)

$$\boldsymbol{P}_q^k = \left(\boldsymbol{M}^k \right)^{-1} \boldsymbol{V}_q^T \boldsymbol{W} \text{diag}\left(\boldsymbol{J}^k \right), \qquad \boldsymbol{L}_q^k = \left(\boldsymbol{M}^k \right)^{-1} \boldsymbol{V}_f^T \boldsymbol{W}_f \text{diag}\left(\boldsymbol{J}_f^k \right). \tag{21}$$

These matrices are distinct from element to element, reflecting the fact that problem (19) is distinct from element to element.

3.4 A discretely entropy stable DG formulation on curved meshes

Given the matrices in Section 3.3, we can now define a local entropy stable DG formulation on an element D^k. Here, we seek an approximation solution $\boldsymbol{u}_N(\boldsymbol{x},t)$ to (1), which is represented using vector-valued coefficients $\boldsymbol{u}_h(t)$ such that

$$\boldsymbol{u}_N(\boldsymbol{x},t) = \sum_{j=1}^{N_p} \left(\boldsymbol{u}_h(t)\right)_j \phi_j(\boldsymbol{x}), \qquad \left(\boldsymbol{u}_h(t)\right)_j \in \mathbb{R}^n.$$

Since the coefficients are vector valued, we assume that all matrices act component-wise on $\boldsymbol{u}_h$ in the Kronecker product sense.

We first define the numerical fluxes $\boldsymbol{f}_{i,S}\left(\boldsymbol{u}_L,\boldsymbol{u}_R\right)$ as the bivariate function of "left" and "right" conservative variable states $\boldsymbol{u}_L,\boldsymbol{u}_R$. Such a numerical flux is referred to as entropy conservative (or entropy stable) if it satisfies the following conditions:

Definition 1. The numerical flux $\boldsymbol{f}_{i,S}\left(\boldsymbol{u}_L,\boldsymbol{u}_R\right)$ is entropy conservative (or entropy stable) if it satisfies the following conditions

1. $\boldsymbol{f}_{i,S}\left(\boldsymbol{u}_L,\boldsymbol{u}_R\right) = \boldsymbol{f}_{i,S}\left(\boldsymbol{u}_R,\boldsymbol{u}_L\right)$ (symmetry),
2. $\boldsymbol{f}_{i,S}\left(\boldsymbol{u},\boldsymbol{u}\right) = \boldsymbol{f}_i\left(\boldsymbol{u}\right)$ (consistency),
3. $\boldsymbol{f}_{i,S}$ is referred to as entropy conservative if it satisfies conditions 1, 2, and
$$\left(\boldsymbol{v}_L - \boldsymbol{v}_R\right)^T \boldsymbol{f}_{i,S}\left(\boldsymbol{u}_L,\boldsymbol{u}_R\right) = \psi_i\left(\boldsymbol{u}_L\right) - \psi_i\left(\boldsymbol{u}_R\right).$$
4. $\boldsymbol{f}_{i,S}$ is referred to as entropy stable if it satisfies conditions 1, 2, and
$$\left(\boldsymbol{v}_L - \boldsymbol{v}_R\right)^T \boldsymbol{f}_{i,S}\left(\boldsymbol{u}_L,\boldsymbol{u}_R\right) \leq \psi_i\left(\boldsymbol{u}_L\right) - \psi_i\left(\boldsymbol{u}_R\right).$$

We now introduce the L^2 projection of the entropy variables $\boldsymbol{v}_h$ and the entropy-projected conservative variables $\widetilde{\boldsymbol{u}}$

$$\boldsymbol{u}_q = \boldsymbol{V}_q\boldsymbol{u}_h, \qquad \boldsymbol{v}_h = \boldsymbol{P}_q\boldsymbol{v}\left(\boldsymbol{u}_q\right), \qquad \widetilde{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{V}_q \\ \boldsymbol{V}_f \end{bmatrix} \boldsymbol{v}_h, \qquad \widetilde{\boldsymbol{u}} = \begin{bmatrix} \widetilde{\boldsymbol{u}}_q \\ \widetilde{\boldsymbol{u}}_f \end{bmatrix} = \boldsymbol{u}\left(\widetilde{\boldsymbol{v}}\right). \tag{22}$$

In (22), the entropy-projected conservative variables $\widetilde{\boldsymbol{u}}$ denote the evaluation of the conservative variables in terms of the projected entropy variables at volume and face quadrature points. We note that, under an appropriate choice of quadrature on quadrilaterals and hexahedra, this approach is equivalent to the approach taken in [26], where the entropy variables are evaluated at Gauss nodes, then interpolated to a different set of nodes and used to compute the nonlinear fluxes.

We now introduce a semi-discrete DG formulation for $\boldsymbol{u}_h$

$$\begin{aligned}
&\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} + \begin{bmatrix} \boldsymbol{P}_q^k & \boldsymbol{L}_q^k \end{bmatrix} \sum_{j=1}^{d} \left(\boldsymbol{D}_k^j \circ \boldsymbol{F}_S\right) \boldsymbol{1} + \sum_{j=1}^{d} \boldsymbol{L}_q^k \mathrm{diag}\left(\boldsymbol{n}_i\right)\left(\boldsymbol{f}^* - \boldsymbol{f}(\widetilde{\boldsymbol{u}}_f)\right) = 0, \\
&\left(\boldsymbol{F}_S\right)_{ij} = \boldsymbol{f}_S\left(\left(\widetilde{\boldsymbol{u}}\right)_i, \left(\widetilde{\boldsymbol{u}}\right)_j\right), \qquad 1 \leq i,j \leq N_q + N_q^f, \\
&\boldsymbol{f}^* = \boldsymbol{f}_S(\widetilde{\boldsymbol{u}}_f^+, \widetilde{\boldsymbol{u}}_f) \text{ on interior faces},
\end{aligned} \tag{23}$$

where $\widetilde{\boldsymbol{u}}^+$ denotes the values of the entropy-projected conservative variables on the neighboring element across each face of D^k, and $\boldsymbol{f}^*$ on the boundary denotes some numerical flux through which boundary conditions are imposed.

Define the boundary quadrature matrix $\boldsymbol{W}_{\partial\Omega}$ such that

$$(\boldsymbol{W}_{\partial\Omega})_{ii} = \begin{cases} \boldsymbol{W}_f \text{diag}\left(\boldsymbol{J}_f^k\right), & \text{if } \boldsymbol{x}_i^f \text{ is on the } \partial\Omega \\ 0, & \text{otherwise.} \end{cases}$$

We have the following semi-discrete statement of entropy conservation:

Theorem 1. *Assume that $\boldsymbol{Q}_k^j \mathbf{1} = 0$ for $j = 1, \dots, d$ over each element D^k. Then, (23) is entropy conservative in the sense that*

$$\sum_k \mathbf{1}^T \boldsymbol{J}^k \boldsymbol{W} \frac{\mathrm{d}U(\boldsymbol{u})}{\mathrm{dt}} = \sum_k \sum_{j=1}^d \mathbf{1}^T \boldsymbol{W}_{\partial\Omega} \left(\psi_j\left(\widetilde{\boldsymbol{u}}_f\right) - \widetilde{\boldsymbol{v}}_f^T \boldsymbol{f}_j^*\right).$$

Proof. Using that $\boldsymbol{Q}_k^j \mathbf{1} = 0$ for $i = 1, \dots, d$ over each element and (17), the proof of entropy conservation is identical to that of [21]. Expand out! Do spatial term only, punt for time term. □

An entropy stable scheme can be constructed by adding an entropy-dissipating penalty term, such as a Lax-Friedrichs penalization or the matrix dissipation terms introduced in [27, 28]. maybe add more here?

3.5 Enforcing the discrete geometric conservation law

An important aspect of Theorem 1 is the assumption that $\boldsymbol{Q}_k^j \mathbf{1} = 0$ for $j = 1, \dots, d$. However, this is not always guaranteed to hold for $\boldsymbol{Q}_k^j$ as defined through (16). In this section, we discuss methods of constructing the geometric terms $J\boldsymbol{G}_{ij}$ for curvilinear meshes in a way that ensures $\boldsymbol{Q}_k^j \mathbf{1} = 0$.

From (16), the condition $\boldsymbol{Q}_k^j \mathbf{1} = 0$ is equivalent to

$$\begin{aligned} \boldsymbol{Q}_k^j \mathbf{1} = \boldsymbol{W}_N \sum_{j=1}^d \boldsymbol{D}_N^j \circ \left\{\{J\boldsymbol{G}_{ij}^k\}\right\} \mathbf{1} &= \frac{1}{2} \boldsymbol{W}_N \sum_{j=1}^d \left(\text{diag}\left(J\boldsymbol{G}_{ij}^k\right) \boldsymbol{D}_N^j \mathbf{1} + \boldsymbol{D}_N^j \text{diag}\left(J\boldsymbol{G}_{ij}^k\right) \mathbf{1}\right) \\ &= \frac{1}{2} \boldsymbol{W}_N \sum_{j=1}^d \boldsymbol{D}_N^j \left(J\boldsymbol{G}_{ij}^k\right) = 0, \end{aligned}$$

where we have used that $\boldsymbol{D}_N^j \mathbf{1} = 0$ to eliminate the first term. Since $\boldsymbol{W}_N$ is a diagonal matrix with positive entries, $\boldsymbol{Q}_k^j \mathbf{1} = 0$ is equivalent to ensuring that a discrete version of the GCL (13) holds

$$\sum_{j=1}^d \boldsymbol{D}_N^j \left(J\boldsymbol{G}_{ij}^k\right) = 0. \tag{24}$$

This condition is required to ensure that free-stream preservation holds at the discrete level. In other words, we wish to ensure that the semi-discrete scheme preserves (for $\boldsymbol{u}$ constant)

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}) = \frac{\partial \boldsymbol{u}}{\partial t} = 0.$$

For isoparametric geometric mappings (where the degree of the mapping matches the degree of the polynomial approximation) in two dimensions, the GCL is naturally enforced by noting that the scaled metric terms $J\boldsymbol{G}_{ij}$ are exactly polynomials of degree N. As a result, computing the metric terms exactly automatically enforces that both the continuous GCL (13) and the discrete GCL (24) are satisfied. However, the discrete GCL is not always maintained at the discrete level in 3D.

In three dimensions, geometric terms are typically computed in "cross-product" form

$$\begin{bmatrix} J^k \boldsymbol{G}^k_{i1} \\ J^k \boldsymbol{G}^k_{i2} \\ J^k \boldsymbol{G}^k_{i3} \end{bmatrix} = \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_j} \times \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_k}, \qquad i,j,k = 1,2,3, \qquad (i,j,k) \text{ cyclic.} \tag{25}$$

This formula can be used to compute the geometric terms exactly at volume and surface quadrature points. However, because $\frac{\partial \boldsymbol{x}}{\partial \widehat{x}_j}, \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_k} \in P^{N-1}$, the curl of this term is a polynomial of degree P^{2N-2}. The discrete GCL condition holds only if $J^k \boldsymbol{G}^k_{ij} \in P^{2N-2}$ are differentiated exactly; however, because applying $\boldsymbol{D}^j_N$ involves the L^2 projection, and because $J^k \boldsymbol{G}^k_{ij}$ and its L^2 projection onto degree N polynomials can differ, the discrete GCL (24) does not hold in general [23].

This can be remedied by using an alternative form of the geometric terms, which ensures that (24) is satisfied a-priori [29, 23]. The geometric terms $J\boldsymbol{G}_{ij}$ can also be computed in a "conservative curl" form

$$\begin{bmatrix} J^k \boldsymbol{G}^k_{i1} \\ J^k \boldsymbol{G}^k_{i2} \\ J^k \boldsymbol{G}^k_{i3} \end{bmatrix} = \widehat{\nabla} \times \left(\frac{\partial \boldsymbol{x}}{\partial \widehat{x}_j} \times \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_k} \right), \qquad i,j,k \text{ cyclic}, \tag{26}$$

where $\widehat{\nabla}\times$ denotes the curl computed with respect to the reference element coordinates. A similar "invariant curl form" is also described in [23]. From (26), it can be observed that, because the divergence of a curl vanishes, the continuous GCL condition (13) holds

$$\widehat{\nabla} \cdot \begin{bmatrix} J^k \boldsymbol{G}^k_{i1} \\ J^k \boldsymbol{G}^k_{i2} \\ J^k \boldsymbol{G}^k_{i3} \end{bmatrix} = \sum_{j=1}^{d} \frac{\partial}{\partial \widehat{x}_j} J^k \boldsymbol{G}^k_{ij} = 0. \tag{27}$$

The idea of Kopriva is to approximate the geometric terms through

$$\begin{bmatrix} J^k \boldsymbol{G}^k_{i1} \\ J^k \boldsymbol{G}^k_{i2} \\ J^k \boldsymbol{G}^k_{i3} \end{bmatrix} \approx \widehat{\nabla} \times I_N \left(\frac{\partial \boldsymbol{x}}{\partial \widehat{x}_j} \times \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_k} \right), \qquad i,j,k \text{ cyclic}, \tag{28}$$

where I_N denotes the degree N polynomial interpolation operator. However, because the geometric terms are computed by applying the curl, the geometric terms are approximated as degree $(N-1)$ polynomials rather than degree N polynomials, which can reduce accuracy.

We adopt a slight modification of (28) in this work, approximating geometric terms by using the interpolation operator I_{N+1} onto degree $(N+1)$ polynomials, then interpolating back to degree N polynomials in the end

$$\begin{bmatrix} J^k \boldsymbol{G}^k_{i1} \\ J^k \boldsymbol{G}^k_{i2} \\ J^k \boldsymbol{G}^k_{i3} \end{bmatrix} \approx \widehat{\nabla} \times I_{N+1} \left(\frac{\partial \boldsymbol{x}}{\partial \widehat{x}_j} \times \frac{\partial \boldsymbol{x}}{\partial \widehat{x}_k} \right), \qquad i,j,k \text{ cyclic}. \tag{29}$$

We note that, since the curl of a degree $(N+1)$ polynomial is degree N, the approximation generated by (29) is a degree N polynomial.

Theorem 2. *Let the geometric terms $\widetilde{J^k \boldsymbol{G}^k_{ij}}$ denote the approximation of $J^k \boldsymbol{G}^k_{ij}$ using (29), and let the scaled normals $\widetilde{n_i J^k_f}$ be computed using $\widetilde{J^k \boldsymbol{G}^k_{ij}}$ and (15). Then, $\widetilde{J^k \boldsymbol{G}^k_{ij}}$ is an $O(h^{N+1})$ accurate approximation which satisfies both the discrete GCL condition (24) and Assumption 2.*

Proof. The satisfaction of (24) relies on the fact that $J^k \boldsymbol{G}^k_{ij} \in P^N$, and that $J^k G^k_{ij}$ is equal to its own L^2 projection. Then,

$$\begin{bmatrix} \boldsymbol{V}_q \\ \boldsymbol{V}_f \end{bmatrix} \boldsymbol{P}_q J\boldsymbol{G}^k_{ij} = J\boldsymbol{G}^k_{ij}$$

by the fact that the discrete L^2 projection operator applied to $J\boldsymbol{G}^k_{ij}$ returns its exact polynomial coefficients. Finish □

Remark. *We note that the accuracy of (29) depends on the choice of interpolation points. It is well known that interpolating at equispaced points can result in inaccurate polynomial approximations. One can determine good interpolation point sets by optimizing over some measure of interpolation quality (such as the Lebesgue constant), and in practice, sets of interpolation points are pre-computed for some polynomial degrees $N = 1, \dots, N_{\max}$ on the reference element and stored [30, 31]. Interpolation points can also be explicitly computed as the image of equispaced points under an appropriately defined mapping [32, 33, 34].*

We briefly outline how to compute $\widetilde{J^k \boldsymbol{G}^k_{ij}}$ in three dimensions. Let $\{\widehat{\boldsymbol{x}}_i^N\}_{i=1}^{N_p}$ denote the set of degree N interpolation points, and let $\ell_i^N(\widehat{\boldsymbol{x}})$ denote the ith degree N Lagrange basis function on the reference element. We define interpolation matrices $\boldsymbol{V}_N^{N+1}$ and $\boldsymbol{V}_{N+1}^N$ between degree N and $N+1$ polynomials such that

$$\begin{aligned} \left(\boldsymbol{V}_N^{N+1}\right)_{ij} &= \ell_j^N(\widehat{\boldsymbol{x}}_i^{N+1}), \qquad 1 \le i \le N_p, \qquad 1 \le i \le (N+1)_p \\ \left(\boldsymbol{V}_{N+1}^{N}\right)_{ij} &= \ell_j^{N+1}(\widehat{\boldsymbol{x}}_i^{N}), \qquad 1 \le i \le (N+1)_p, \qquad 1 \le i \le N_p, \end{aligned} \tag{30}$$

where $N_p, (N+1)_p$ denotes the number of interpolation points for degree N and $N+1$ polynomials, respectively. Next, let $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ denote vectors containing x, y, z coordinates of degree N interpolation points on a curved physical element D^k, and let $\widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2, \widetilde{\boldsymbol{x}}_3$ denote their degree $N+1$ interpolation such that

$$\widetilde{\boldsymbol{x}}_1 = \boldsymbol{V}_N^{N+1}\boldsymbol{x}_1, \qquad \widetilde{\boldsymbol{x}}_2 = \boldsymbol{V}_N^{N+1}\boldsymbol{x}_2, \qquad \widetilde{\boldsymbol{x}}_3 = \boldsymbol{V}_N^{N+1}\boldsymbol{x}_3. \tag{31}$$

Let $\widetilde{\boldsymbol{D}}_i^{N+1}$ denote the nodal differentiation matrix of degree $N+1$ with respect to the ith coordinate direction. The geometric factors are computed as follows:

$$\begin{aligned}
\boldsymbol{JG}^{\boldsymbol{k}}_{11} &= \boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{12} &= \boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{13} &= \boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{21} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{22} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{23} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right) - \widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_1\right)\circ\widetilde{\boldsymbol{x}}_3\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{31} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right) - \widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{32} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_3^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right) - \widetilde{\boldsymbol{D}}_3^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right)\right) \\
\boldsymbol{JG}^{\boldsymbol{k}}_{33} &= -\boldsymbol{V}_{N+1}^N \left(\widetilde{\boldsymbol{D}}_2^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_1^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right) - \widetilde{\boldsymbol{D}}_1^{N+1}\left(\left(\widetilde{\boldsymbol{D}}_2^{N+1}\widetilde{\boldsymbol{x}}_2\right)\circ\widetilde{\boldsymbol{x}}_1\right)\right).
\end{aligned} \tag{32}$$

Remark. *We note that the discrete GCL (24) can be enforced directly through a local constrained minimization problem [35, 36], which yields a solution in terms of a pseudo-inverse. However, we have not found a straightforward way to simultaneously enforce both the discrete GCL condition (24) and Assumption 2 using this approach.*

4 Discretely stable and low storage DG methods on curved meshes

A disadvantage of the formulation (23) is high storage costs, especially at high orders of approximation. While the matrices $\boldsymbol{Q}_k^i$ can be applied to a vector without needing to explicitly store the matrix, the projection and lifting matrices (21) differ from element to element, necessitating either explicit pre-computation and storage

or the assembly and inversion of a weighted mass matrix for each right hand side evaluation. The latter option is computationally expensive, while the former option increases storage costs. This increase in storage can result in suboptimal performance on modern computational architectures [3], due to the increasing cost of memory operations and data movement compared to arithmetic operations.

In this section, we present a discretely entropy stable scheme which avoids this high storage cost through the use of a low-storage weight-adjusted approximation to the inverse of a weighted mass matrix. To ensure a discrete entropy conservation or a discrete entropy inequality, we also modify the formulation (23) to take into account the use of a weight-adjusted mass matrix.

4.1 A weight-adjusted approximation to the curvilinear mass matrix

The presence of the weighted L^2 inner product $\left(J^k \Pi_N^k u, v\right)_{\widehat{D}}$ in (19) results in the presence of a weighted mass matrix. Because the weight J^k varies spatially over each element, the inverse of a weighted mass matrix is no longer a scaling of the inverse reference mass matrix. The motivation for the weight-adjusted mass matrix is to replace the inversion of weighted mass matrices over each element with the application of inverse reference mass matrices and quadrature-based operations involving the spatially varying weights J^k [1, 2].

To define a weight-adjusted approximation to the curvilinear L^2 inner product, we first define the operator $T_w^{-1} : L^2 \to P^N$ as follows

$$\left(w T_w^{-1} u, v\right)_{\widehat{D}} = (u, v)_{\widehat{D}}, \qquad \forall v \in P^N. \tag{33}$$

Roughly speaking, $T_w u$ approximates u/w. Thus, taking $w = 1/J^k$ provides an approximation of the curvilinear L^2 inner product

$$\left(J^k u, v\right)_{\widehat{D}} \approx \left(T_{1/J^k}^{-1} u, v\right)_{\widehat{D}}.$$

Computing $T_{1/J^k}^{-1} u$ requires solving (33). Let $u \in P^N$, and let $\boldsymbol{u}_J$ denote coefficients for the polynomial $T_{1/J^k}^{-1} u$. This results in the following matrix system

$$\boldsymbol{M}_{1/J^k} \boldsymbol{u}_J = \boldsymbol{M} \boldsymbol{u}, \qquad \boldsymbol{M}_{1/J^k} = \boldsymbol{V}_q^T \boldsymbol{W} \text{diag}\left(1/\boldsymbol{J}^k\right) \boldsymbol{V}_q,$$

which implies that, when restricted to polynomials, the matrix form of T_{1/J^k}^{-1} is $\boldsymbol{M}_{1/J^k}^{-1} \boldsymbol{M}$. Then, the weight-adjusted mass matrix is the Gram matrix with respect to the weight-adjusted inner product $\left(T_{1/J^k}^{-1} u, v\right)_{\widehat{D}}$, such that

$$\boldsymbol{M}^k \approx \boldsymbol{M} \boldsymbol{M}_{1/J^k}^{-1} \boldsymbol{M}, \qquad \left(\boldsymbol{M}^k\right)^{-1} \approx \boldsymbol{M}^{-1} \boldsymbol{M}_{1/J^k} \boldsymbol{M}^{-1}.$$

The inverse of the weight-adjusted mass matrix can be applied in a matrix-free fashion by using quadrature to form $\boldsymbol{M}_{1/J^k}$. This requires storage of the inverse reference mass matrix and the values of $\boldsymbol{J}^k$ at quadrature points. Assuming that the number of quadrature points scales as $O(N^d)$ in d dimensions, this yields a storage cost of $O(N^d)$ per-element compared to an $O(N^{2d})$ per element storage cost required for the storage of inverse weighted mass matrices $\left(\boldsymbol{M}^k\right)^{-1}$. This application of the weight-adjusted mass matrix is typically applied using the L^2 projection matrix $\boldsymbol{P}_q$ as follows

$$\boldsymbol{M}^{-1} \boldsymbol{M}_{1/J^k} \boldsymbol{M}^{-1} = \boldsymbol{P}_q \text{diag}\left(1/\boldsymbol{J}^k\right) \boldsymbol{V}_q \boldsymbol{M}^{-1}.$$

When evaluating the right hand side of a semi-discrete formulation such as (23), the inverse mass matrix is typically merged into operations on the right hand side, such that the main work in applying the weight-adjusted mass matrix consists of applying the interpolation matrix $\boldsymbol{V}_q$, scaling by pointwise values of $1/\boldsymbol{J}^k$ at quadrature points, and multiplying by the L^2 projection matrix $\boldsymbol{P}_q$.

4.2 A discretely entropy stable low storage DG formulation on curved meshes

Given the weight-adjusted inverse mass matrix, we can also define a weight-adjusted version of the L^2 projection over a curved element D^k. We refer to this operator as $P_N^k : L^2 \to P^N$, which satisfies

$$\left(T_{1/J^k}^{-1} P_N^k u, v\right)_{\widehat{D}} = \left(uJ^k, v\right)_{\widehat{D}}, \qquad \forall v \in P^N\left(\widehat{D}\right).$$

It was shown in [2] that P_N^k is given explicitly by

$$P_N^k u = \Pi_N\left(\frac{1}{J^k}\Pi_N\left(uJ^k\right)\right), \tag{34}$$

where Π_N is the L^2 projection operator on the reference element $\widehat{D}$. We can discretize P_N^k using quadrature to yield a weight-adjusted projection matrix $\widetilde{\boldsymbol{P}}_q$

$$\begin{aligned}\widetilde{\boldsymbol{P}}_q^k &= \boldsymbol{M}^{-1}\boldsymbol{M}_{1/J^k}\boldsymbol{M}^{-1}\boldsymbol{V}_q^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right) = \boldsymbol{M}^{-1}\boldsymbol{V}_q^T\boldsymbol{W}\mathrm{diag}\left(1/\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{P}_q\mathrm{diag}\left(\boldsymbol{J}^k\right)\boldsymbol{V}_q \\ &= \boldsymbol{P}_q\mathrm{diag}\left(1/\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{P}_q\mathrm{diag}\left(\boldsymbol{J}^k\right).\end{aligned} \tag{35}$$

We can similarly define a weight-adjusted lifting matrix $\widetilde{\boldsymbol{L}}_q$ by replacing the weighted mass matrix in (21) with the weight-adjusted mass matrix

$$\widetilde{\boldsymbol{L}}_q^k = \boldsymbol{P}_q\mathrm{diag}\left(1/\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{L}_q\mathrm{diag}\left(\boldsymbol{J}_f^k\right). \tag{36}$$

We can now introduce the weight-adjusted projection of the entropy variables $\boldsymbol{v}_h$ and the corresponding entropy-projected conservative variables $\widetilde{\boldsymbol{u}}$

$$\boldsymbol{u}_q = \boldsymbol{V}_q\boldsymbol{u}_h, \qquad \boldsymbol{v}_h = \widetilde{\boldsymbol{P}}_q\boldsymbol{v}\left(\boldsymbol{u}_q\right), \qquad \widetilde{\boldsymbol{v}} = \begin{bmatrix}\boldsymbol{V}_q \\ \boldsymbol{V}_f\end{bmatrix}\boldsymbol{v}_h, \qquad \widetilde{\boldsymbol{u}} = \begin{bmatrix}\widetilde{\boldsymbol{u}}_q \\ \widetilde{\boldsymbol{u}}_f\end{bmatrix} = \boldsymbol{u}\left(\widetilde{\boldsymbol{v}}\right). \tag{37}$$

A semi-discrete DG formulation for $\boldsymbol{u}_h$ can be constructed using the variables defined in (37)

$$\begin{aligned}&\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} + \begin{bmatrix}\widetilde{\boldsymbol{P}}_q^k & \widetilde{\boldsymbol{L}}_q^k\end{bmatrix}\sum_{j=1}^d\left(\boldsymbol{D}_k^j \circ \boldsymbol{F}_S\right)\boldsymbol{1} + \sum_{j=1}^d \widetilde{\boldsymbol{L}}_q^k\mathrm{diag}\left(\boldsymbol{n}_i\right)\left(\boldsymbol{f}^* - \boldsymbol{f}(\widetilde{\boldsymbol{u}}_f)\right) = 0, \\ &\left(\boldsymbol{F}_S\right)_{ij} = \boldsymbol{f}_S\left(\left(\widetilde{\boldsymbol{u}}\right)_i, \left(\widetilde{\boldsymbol{u}}\right)_j\right), \qquad 1 \le i,j \le N_q + N_q^f, \\ &\boldsymbol{f}^* = \boldsymbol{f}_S(\widetilde{\boldsymbol{u}}_f^+, \widetilde{\boldsymbol{u}}_f) \text{ on interior faces.}\end{aligned} \tag{38}$$

Since the weight-adjusted mass matrix inverse is low-storage, and since the matrices $\boldsymbol{D}_k^j$ in (16) can be assembled from reference matrices $\boldsymbol{D}_N^i$ and the values of geometric terms at quadrature points, the overall scheme requires only $O(N^d)$ storage per element. We can additionally show that formulation (38) is entropy conservative in the same sense as Theorem 3:

Theorem 3. *Assume that $\boldsymbol{Q}_k^j\boldsymbol{1} = 0$ for $j = 1, \ldots, d$ over each element D^k. Then, (38) is entropy conservative in the sense that*

$$\sum_k \boldsymbol{1}^T\boldsymbol{J}^k\boldsymbol{W}\frac{\mathrm{d}U(\boldsymbol{u})}{\mathrm{dt}} = \sum_k\sum_{j=1}^d \boldsymbol{1}^T\boldsymbol{W}_{\partial\Omega}\left(\psi_j\left(\widetilde{\boldsymbol{u}}_f\right) - \widetilde{\boldsymbol{v}}_f^T\boldsymbol{f}^*\right).$$

Proof. The proof is similar to that of Theorem 1. First, we note that

$$\begin{aligned}\widetilde{\boldsymbol{P}}_q^k &= \boldsymbol{P}_q\mathrm{diag}\left(1/\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{P}_q\mathrm{diag}\left(\boldsymbol{J}^k\right) = \boldsymbol{M}^{-1}\boldsymbol{V}_q^T\boldsymbol{W}\mathrm{diag}\left(1/\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{M}^{-1}\boldsymbol{V}_q^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right) \\ &= \boldsymbol{M}^{-1}\boldsymbol{M}_{1/J^k}\boldsymbol{M}^{-1}\boldsymbol{V}_q^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right)\end{aligned} \tag{39}$$

$$\widetilde{\boldsymbol{L}}_q^k = \boldsymbol{M}^{-1}\boldsymbol{M}_{1/J^k}\boldsymbol{M}^{-1}\boldsymbol{V}_f^T\boldsymbol{W}_f\mathrm{diag}\left(\boldsymbol{J}_f^k\right). \tag{40}$$

Then, multiplying by the weight-adjusted mass matrix $\boldsymbol{M}\boldsymbol{M}_{1/J^k}^{-1}\boldsymbol{M}$ on both sides and using (39), (40) yields

$$\boldsymbol{M}\boldsymbol{M}_{1/J^k}^{-1}\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} + \begin{bmatrix} \boldsymbol{V}_q^T & \boldsymbol{V}_f^T \end{bmatrix} \sum_{j=1}^{d}\left(\boldsymbol{Q}_k^j \circ \boldsymbol{F}_S\right)\boldsymbol{1} + \sum_{j=1}^{d}\boldsymbol{V}_f^T\boldsymbol{W}_f \mathrm{diag}\left(\boldsymbol{n}_i\right)\left(\boldsymbol{f}^* - \boldsymbol{f}(\widetilde{\boldsymbol{u}}_f)\right) = 0.$$

Testing with the weight-adjusted projection of the entropy variables $\boldsymbol{v}_h = \widetilde{\boldsymbol{P}}_q\boldsymbol{v}\left(\boldsymbol{u}_q\right)$ and using (39) then yields for the time term

$$\begin{aligned}\left(\widetilde{\boldsymbol{P}}_q\boldsymbol{v}\left(\boldsymbol{u}_q\right)\right)^T\boldsymbol{M}\boldsymbol{M}_{1/J^k}^{-1}\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} &= \boldsymbol{v}\left(\boldsymbol{u}_q\right)^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right)\boldsymbol{V}_q\boldsymbol{M}^{-1}\boldsymbol{M}_{1/J^k}\boldsymbol{M}^{-1}\boldsymbol{M}\boldsymbol{M}_{1/J^k}^{-1}\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} \\ &= \boldsymbol{v}\left(\boldsymbol{u}_q\right)^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right)\frac{\mathrm{d}\boldsymbol{V}_q\boldsymbol{u}_h}{\mathrm{dt}} = \boldsymbol{1}^T\boldsymbol{W}\mathrm{diag}\left(\boldsymbol{J}^k\right)\boldsymbol{v}\left(\boldsymbol{u}_q\right)^T\frac{\mathrm{d}\boldsymbol{u}_q}{\mathrm{dt}} \\ &= \boldsymbol{1}^T\boldsymbol{J}^k\boldsymbol{W}\frac{\mathrm{d}U(\boldsymbol{u})}{\mathrm{dt}}.\end{aligned}$$

The remainder of the proof is identical to that of Theorem 1 and [21]. □

4.3 Analysis of weight-adjusted projection

The construction of the discretely entropy stable weight-adjusted DG formulation (38) replaces the L^2 projection operator Π_N^k with the weight-adjusted projection operator P_N^k. While this preserves entropy stability, it is unclear whether P_N^k is high order accurate. In this section, we prove that the weight-adjusted projection is high order accurate in the sense that, for a fixed geometric mapping, $\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(\Omega)} = O(h^{N+2})$. Because the difference between the L^2 and weight-adjusted projection is super-convergent, the results are typically indistinguishable for a fixed geometric mapping [37].

We first note that $P_N^k = \Pi_N\left(\frac{1}{J^k}\Pi_N\left(uJ^k\right)\right)$ is self-adjoint with respect to the J-weighted L^2 inner product

$$\left(J^k P_N^k u, v\right) = \left(\Pi_N\left(\frac{1}{J^k}\Pi_N\left(uJ^k\right)\right), vJ^k\right) = \left(uJ^k, \Pi_N\left(\frac{1}{J^k}\Pi_N\left(vJ^k\right)\right)\right) = \left(uJ^k, P_N^k v\right). \tag{41}$$

Furthermore, using that the operator T_{1/J^k}^{-1} is self-adjoint for $v \in P^N$ with respect to the L^2 inner product [1], we find that a projection-like property holds for the weight-adjusted L^2 inner product

$$\left(T_{1/J^k}^{-1}P_N^k u, v\right) = \left(\frac{1}{J^k}\Pi_N(uJ^k), T_{1/J^k}^{-1}v\right) = \left(\Pi_N(uJ^k), v\right) = \left(uJ^k, v\right), \qquad \forall v \in P^N. \tag{42}$$

To prove $\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(\Omega)} = O(h^{N+2})$, we use results from [1, 2]. The first theorem we need shows that $T_{1/J}^{-1}$ can be used to approximate weighted curvilinear L^2 inner products with order $(2N+2)$ accuracy.

Theorem 4. *Let $u \in W^{N+1,2}\left(D^k\right)$ and $v \in P^N\left(D^k\right)$. Then,*

$$\begin{aligned}&\left|(u, vJ)_{\widehat{D}} - \left(T_{1/J}^{-1}u, v\right)_{\widehat{D}}\right| \leq \\ &Ch^{2N+2}\left\|J\right\|_{L^\infty(D^k)}\left\|\frac{1}{J}\right\|_{L^\infty(D^k)}^2\left\|J\right\|_{W^{N+1,\infty}(D^k)}^2\left\|u\right\|_{W^{N+1,2}(D^k)}\left\|v\right\|_{W^{N+1,2}(D^k)}.\end{aligned}$$

Proof. The proof involves straightforward adaptations of Theorem 4, Theorem 5, and Theorem 6 in [1] to the reference element $\widehat{D}$. □

The next result we need is a generalized inverse inequality.

Lemma 1. *Let $v \in P^N\left(D^k\right)$, and let $h = \operatorname{diam}\left(D^k\right)$. Then,*

$$\|v\|_{W^{N+1,2}(D^k)} \leq C_N h^{-N} \|v\|_{L^2(D^k)}.$$

where C_N is independent of h.

Proof. Formalize this...

The result is the consequence of a scaling argument $\frac{\partial^N u}{\partial x_i^N} = O(h^{-N})\frac{\partial^N u}{\partial \hat{x}_i^N}$ and a Rayleigh quotient bound involving the largest eigenvalue of the generalized eigenvalue problem

$$\boldsymbol{K}_N \boldsymbol{u} = \lambda \boldsymbol{M} \boldsymbol{u},$$

where $\boldsymbol{M}$ is the L^2 mass matrix over D^k and $\boldsymbol{K}_N$ is the Gram matrix corresponding to the Sobolev inner product on the reference element $W^{N+1,2}\left(\widehat{D}\right)$. We emphasize, however, that the constant C_N depends on the largest eigenvalue, which in turn depends on the order N and dimension d. □

We can now prove that $P_N^k u$ is super-convergent to the curvilinear L^2 projection $\Pi_N^k u$ for sufficiently smooth u:

Theorem 5. *Let $u \in W^{N+1,2}\left(D^k\right)$. The difference between the L^2 projection $\Pi_N^k u$ and the weight-adjusted projection $P_N^k u$ is*

$$\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(D^k)} \leq C_N \|J\|_{L^\infty(D^k)} \left\|\frac{1}{J}\right\|_{L^\infty(D^k)}^2 \|J\|_{W^{N+1,\infty}(D^k)}^2 h^{N+2} \|u\|_{W^{N+1,2}(D^k)}.$$

where C_N is a mesh-independent constantwhich depends on N, d.

Proof. Let $\Pi_N^k u$ be the L^2 projection over the curved element D^k, such that

$$\left(J\Pi_N^k u, v\right) = \left(uJ, v\right), \qquad \forall v \in P^N.$$

Using the fact that $P_N^k u$ satisfies an analogous property (42), we can rewrite the norm of the difference between the weight-adjusted and L^2 projections

$$\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(D^k)}^2 = \left(\Pi_N^k u - P_N^k u, vJ\right)_{\widehat{D}}, \qquad v = \Pi_N^k u - P_N^k u.$$

Because $v \in P^N\left(D^k\right)$, we can also evaluate the squared error as

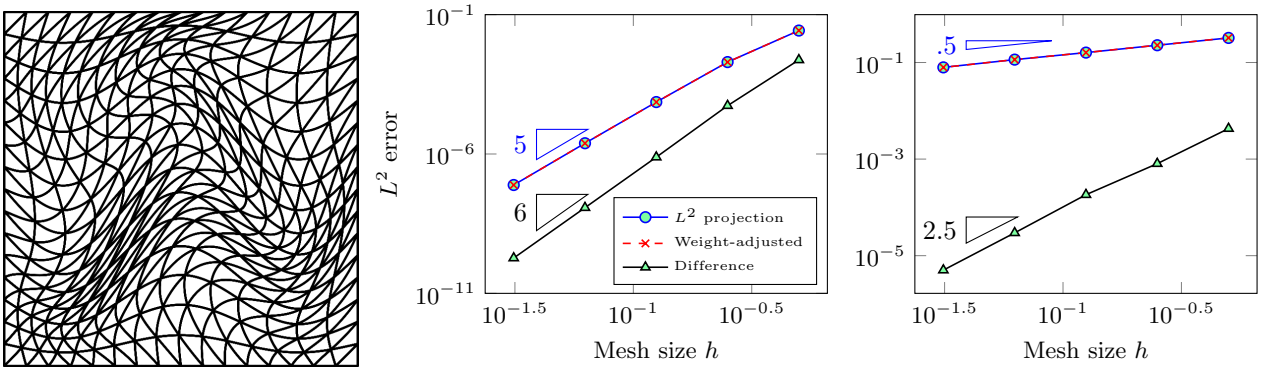
$$\begin{aligned}\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(D^k)}^2 &= \left|\left(\Pi_N^k, vJ\right)_{\widehat{D}} - \left(P_N^k u, vJ\right)_{\widehat{D}}\right| \\ &= \left|(u, vJ)_{\widehat{D}} - \left(P_N^k u, vJ\right)_{\widehat{D}}\right| = \left|\left(T_{1/J}^{-1} P_N^k u, v\right)_{\widehat{D}} - \left(JP_N^k u, v\right)_{\widehat{D}}\right|.\end{aligned}$$

Applying Theorem 4 and then Lemma 1 yields

$$\begin{aligned}\left\|\Pi_N^k u - P_N^k u\right\|_{L^2(D^k)}^2 &\leq Ch^{2N+2} \|J\|_{L^\infty(D^k)} \left\|\frac{1}{J}\right\|_{L^\infty(D^k)}^2 \|J\|_{W^{N+1,\infty}(D^k)}^2 \|u\|_{W^{N+1,2}(D^k)} \|v\|_{W^{N+1,2}\left(\widehat{D}\right)} \\ &\leq Ch^{N+2} \|J\|_{L^\infty(D^k)} \left\|\frac{1}{J}\right\|_{L^\infty(D^k)}^2 \|J\|_{W^{N+1,\infty}(D^k)}^2 \|u\|_{W^{N+1,2}(D^k)} \|v\|_{L^2(D^k)}.\end{aligned}$$

Dividing through by $\|v\|_{L^2(D^k)} = \left\|\Pi_N^k u - P_N^k u\right\|_{L^2(D^k)}$ gives the desired result. □

We note that, while Theorem 5 requires sufficient smoothness of u, numerical experiments in Section 5 demonstrate that even for discontinuous functions, the L^2 and weight-adjusted projections are indistinguishable from each other on a curved element.



(a) Warped curvilinear mesh

(b) $f(\boldsymbol{x}) = e^{x+y}\sin(\pi x)\sin(\pi y)$

(c) $f(\boldsymbol{x}) = H(x + y - \sin(\pi x))$

Figure 1: L^2 errors in approximating both smooth and discontinuous functions $f(\boldsymbol{x})$ using L^2 and weight-adjusted projections on a curved mesh. The approximation order is $N = 4$, and a degree $2N$ quadrature rule is used to compute integrals over the reference triangle.

4.3.1 A locally conservative high order accurate correction

It should be noted that, while the weight-adjusted approximation of the mass matrix is high order accurate and efficient, it does not preserve the average over a physical element, which is equivalent to the J-weighted average over the reference element. This is due to the fact that, in general,

$$\int_{\widehat{D}} uJ \, \mathrm{d}\widehat{\boldsymbol{x}} - \int_{\widehat{D}} T_{1/J}^{-1} u \, \mathrm{d}\widehat{\boldsymbol{x}} = \mathbf{1}^T \boldsymbol{M}_J \boldsymbol{u} - \mathbf{1}^T \boldsymbol{M} \boldsymbol{M}_{1/J}^{-1} \boldsymbol{M} \boldsymbol{u} \neq 0. \tag{43}$$

Taking $v = 1$ in Theorem 4 shows that the differnece between the true mean and weight-adjusted mean in (43) converges extremely fast at a rate of $O(h^{2N+2})$. However, for systems of conservation laws, it is often desired that the element average is preserved exactly up to machine precision. In [2], it was shown that a rank one correction of the weight-adjusted mass matrix inverse preserves the mean exactly while avoiding any loss of precision in numerical experiments. This requires the use of the Sherman-Morrison formula for the inverse of a rank one update.

We present an explicit formula for a conservative correction here which is simpler than this algebraic correction. Consider a vector $\boldsymbol{u}$ which is the

$$\widetilde{P}_N^k u = P_N^k u + \frac{\int_{\widehat{D}} f - J P_N^k u}{\int_{\widehat{D}} J}.$$

This approach is applicable to an arbitrary weight, and can be generalized to matrix-valued weights as well [3].

5 Numerical experiments

5.1 Super-convergence of the weight-adjusted projection

We begin by verifying Theorem 5. Figure 1

H is the Heaviside function.

5.2 Compressible Euler equations

5.2.1 Two dimensional experiments

Shock propagation on a curved mesh: testing entropy conservation
Isentropic vortex on curved and non-curved meshes

5.2.2 Three dimensional experiments

Isentropic vortex on curved and non-curved meshes

6 Computational results

The semi-discrete evolution equation is as follows

$$\frac{\partial \boldsymbol{u}}{\partial t} = \begin{bmatrix} \boldsymbol{P}_q & \boldsymbol{L}_q \end{bmatrix} \left(\begin{bmatrix} 2\boldsymbol{D}_q & \boldsymbol{V}_q\boldsymbol{L}_q \\ \boldsymbol{V}_f\boldsymbol{P}_q & \boldsymbol{I} \end{bmatrix} \circ \begin{bmatrix} \boldsymbol{f}_S(\boldsymbol{u},\boldsymbol{u}_f) & \boldsymbol{f}_S(\boldsymbol{u},\boldsymbol{u}_f) \\ \boldsymbol{f}_S(\boldsymbol{u},\boldsymbol{u}_f) & \boldsymbol{f}_S(\boldsymbol{u}_f^+,\boldsymbol{u}_f) \end{bmatrix} \boldsymbol{1} \right).$$

where $\boldsymbol{u}, \boldsymbol{u}_f$ are evaluations at volume and surface quadrature points.

The method is outlined as follows: we store the conservation variables $\boldsymbol{u}$ at quadrature points, and compute projected entropy variables $\boldsymbol{v}_u = \Pi_N(\boldsymbol{v}(\boldsymbol{u}))$

1. Apply volume flux differencing ($\boldsymbol{D}_q$ and $\boldsymbol{V}_q\boldsymbol{L}_q$) and projection ($\boldsymbol{V}_q\boldsymbol{P}_q$) (volume kernel)
2. Apply surface flux differencing ($\boldsymbol{V}_f\boldsymbol{P}_q$) and lifting ($\boldsymbol{V}_q\boldsymbol{L}_q$) (volume kernel)
3. Apply WADG (scale by $1/J$ and apply $\boldsymbol{V}_q\boldsymbol{P}_q$ and update $\boldsymbol{u}$ at volume quad points. Compute curvilinear projected entropy variables $\boldsymbol{v}_u = P_N^k\boldsymbol{v}(\boldsymbol{u}) = \Pi_N\left(\frac{1}{J}\Pi_N(\boldsymbol{v}(\boldsymbol{u})J)\right)$, interpolate to volume quad points, and compute conservative variable volume values (update)
4. Interpolate entropy variables at surface quadrature points and write out conservative variable surface values (face).

7 Conclusions

8 Acknowledgements

The authors thank Mark H. Carpenter and David C. Del Rey Fernandez for informative discussions.

References

[1] Jesse Chan, Russell J Hewett, and T Warburton. Weight-adjusted discontinuous Galerkin methods: wave propagation in heterogeneous media. *SIAM Journal on Scientific Computing*, 39(6):A2935–A2961, 2017.

[2] Jesse Chan, Russell J Hewett, and T Warburton. Weight-adjusted discontinuous Galerkin methods: curvilinear meshes. *SIAM Journal on Scientific Computing*, 39(6):A2395–A2421, 2017.

[3] Jesse Chan. Weight-adjusted discontinuous Galerkin methods: Matrix-valued weights and elastic wave propagation in heterogeneous media. *International Journal for Numerical Methods in Engineering*, pages n/a–n/a. nme.5720.

[4] Thomas JR Hughes, LP Franca, and M Mallet. A new finite element formulation for computational fluid dynamics: I. Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics. *Computer Methods in Applied Mechanics and Engineering*, 54(2):223–234, 1986.

[5] Michael S Mock. Systems of conservation laws of mixed type. *Journal of Differential equations*, 37(1):70–88, 1980.

[6] Jesse Chan, Zheng Wang, Axel Modave, Jean-Francois Remacle, and T Warburton. GPU-accelerated discontinuous Galerkin methods on hybrid meshes. *Journal of Computational Physics*, 318:142–168, 2016.

[7] Jan S Hesthaven and Tim Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*, volume 54. Springer, 2007.

[8] Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69. Springer Science & Business Media, 2011.

[9] Bernardo Cockburn and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework. *Mathematics of computation*, 52(186):411–435, 1989.

[10] Bernardo Cockburn and Chi-Wang Shu. The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems. *Journal of Computational Physics*, 141(2):199–224, 1998.

[11] Bernardo Cockburn. Devising discontinuous Galerkin methods for non-linear hyperbolic conservation laws. *Journal of Computational and Applied Mathematics*, 128(1-2):187–204, 2001.

[12] George Karniadakis and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

[13] Jianxian Qiu, Boo Cheong Khoo, and Chi-Wang Shu. A numerical study for the performance of the Runge–Kutta discontinuous Galerkin method based on different numerical fluxes. *Journal of Computational Physics*, 212(2):540–565, 2006.

[14] Lilia Krivodonova. Limiters for high-order discontinuous Galerkin methods. *Journal of Computational Physics*, 226(1):879–896, 2007.

[15] Xiangxiong Zhang, Yinhua Xia, and Chi-Wang Shu. Maximum-principle-satisfying and positivity-preserving high order discontinuous Galerkin schemes for conservation laws on triangular meshes. *Journal of Scientific Computing*, 50(1):29–62, 2012.

[16] Michael Dumbser, Olindo Zanotti, Raphaël Loubère, and Steven Diot. A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws. *Journal of Computational Physics*, 278:47–75, 2014.

[17] Per-Olof Persson and Jaime Peraire. Sub-cell shock capturing for discontinuous Galerkin methods. *AIAA*, 112, 2006.

[18] Garrett E Barter and David L Darmofal. Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation. *Journal of Computational Physics*, 229(5):1810–1827, 2010.

[19] Andreas Klöckner, Tim Warburton, and Jan S Hesthaven. Viscous shock capturing in a time-explicit discontinuous Galerkin method. *Mathematical Modelling of Natural Phenomena*, 6(3):57–83, 2011.

[20] Jason E Hicken, David C Del Rey Fernndez, and David W Zingg. Multidimensional summation-by-parts operators: General theory and application to simplex elements. *SIAM Journal on Scientific Computing*, 38(4):A1935–A1958, 2016.

[21] Jesse Chan. On discretely entropy conservative and entropy stable discontinuous Galerkin methods. *arXiv preprint arXiv:1708.01243*, 2017. Accepted to JCP.

[22] PD Thomas and CK Lombard. Geometric conservation law and its application to flow computations on moving grids. *AIAA journal*, 17(10):1030–1037, 1979.

[23] David A Kopriva. Metric identities and the discontinuous spectral element method on curvilinear meshes. *Journal of Scientific Computing*, 26(3):301–327, 2006.

[24] Philippe G Ciarlet. *The finite element method for elliptic problems*. Elsevier, 1978.

[25] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[26] Matteo Parsani, Mark H Carpenter, Travis C Fisher, and Eric J Nielsen. Entropy Stable Staggered Grid Discontinuous Spectral Collocation Methods of any Order for the Compressible Navier–Stokes Equations. *SIAM Journal on Scientific Computing*, 38(5):A3129–A3162, 2016.

[27] Praveen Chandrashekar. Kinetic energy preserving and entropy stable finite volume schemes for compressible Euler and Navier-Stokes equations. *Communications in Computational Physics*, 14(5):1252–1286, 2013.

[28] Andrew R Winters, Dominik Derigs, Gregor J Gassner, and Stefanie Walch. A uniquely defined entropy stable matrix dissipation operator for high Mach number ideal MHD and compressible Euler simulations. *Journal of Computational Physics*, 332:274–289, 2017.

[29] Miguel R Visbal and Datta V Gaitonde. On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1):155–185, 2002.

[30] Qi Chen and Ivo Babuška. The optimal symmetrical points for polynomial interpolation of real functions in the tetrahedron. *Computer methods in applied mechanics and engineering*, 137(1):89–94, 1996.

[31] Jan S Hesthaven. From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex. *SIAM Journal on Numerical Analysis*, 35(2):655–676, 1998.

[32] MG Blyth and C Pozrikidis. A Lobatto interpolation grid over the triangle. *IMA journal of applied mathematics*, 71(1):153–169, 2006.

[33] T Warburton. An explicit construction of interpolation nodes on the simplex. *Journal of engineering mathematics*, 56(3):247–262, 2006.

[34] Jesse Chan and T Warburton. A comparison of high order interpolation nodes for the pyramid. *SIAM Journal on Scientific Computing*, 37(5):A2151–A2170, 2015.

[35] David C Fernández, Jason E Hicken, and David W Zingg. Simultaneous approximation terms for multi-dimensional summation-by-parts operators. *arXiv preprint arXiv:1605.03214*, 2016.

[36] Jared Crean, Jason E Hicken, David C Del Rey Fernández, David W Zingg, and Mark H Carpenter. Entropy-stable summation-by-parts discretization of the Euler equations on general curved elements. *Journal of Computational Physics*, 356:410–438, 2018.

[37] Jesse Chan and John A Evans. Multi-patch discontinuous Galerkin isogeometric analysis for wave propagation: Explicit time-stepping and efficient mass matrix inversion. *Computer Methods in Applied Mechanics and Engineering*, 333:22–54, 2018.