Mortar-based entropy stable discontinuous Galerkin methods on non-conforming quadrilateral and hexahedral meshes

Jesse Chan, Mario Bencomo, David C. Del Rey Fernandez

Abstract High order entropy stable discontinuous Galerkin (DG) methods for nonlinear conservation laws reproduce a discrete entropy inequality by combining entropy conservative finite volume fluxes with summation-by-parts (SBP) discretization matrices. On tensor product (quadrilateral and hexahedral) elements, SBP matrices are typically constructed by collocating at Lobatto quadrature points. Recent work has extended the construction of entropy stable DG schemes to collocation at more accurate Gauss quadrature points [1].

In this work, we extend entropy stable Gauss collocation schemes to non-conforming meshes. Entropy stable DG schemes require computing entropy conservative numerical fluxes between volume and surface quadrature nodes. On conforming tensor product meshes where volume and surface nodes are aligned, flux evaluations are required only between "lines" of nodes. However, on non-conforming meshes, volume and surface nodes are no longer aligned, resulting in a larger number of flux evaluations. We reduce this expense by introducing an entropy stable mortar-based treatment of non-conforming interfaces via a face-local correction term, and provide necessary conditions for high order accuracy. Numerical experiments confirm the stability and accuracy of this approach.

1 Introduction

Discretely entropy stability has emerged as a methodology for designing high order schemes for nonlinear conservation laws. Entropy stable discretizations ensure the satisfaction of a semi-discrete entropy inequality by combining specific finite volume numerical fluxes with summation-by-parts (SBP) discretization matrices. Compared to traditional high order methods, the resulting schemes demonstrate significantly improved robustness in the presence of under-resolved solution features such as shocks or turbulence while retaining high order accuracy.

Entropy stable discontinuous Galerkin (DG) methods were originally constructed for quadrilateral and hexahedral meshes based on nodal collocation at Lobatto quadrature points [3–5]. Entropy stable schemes were later extended to simplicial and more general elements using tailored volume and surface quadrature

———————————
Address(es) of author(s) should be given

rules [6, 7]. More general quadrature rules were addressed in [1, 2, 8, 9], including entropy stable collocation schemes on quadrilateral and hexahedral elements based on more accurate Gauss quadrature rules and generalized SBP operators [1].

The work presented here focuses on geometrically non-conforming meshes. Such meshes may arise when applying domain decomposition techniques to a complex geometry (e.g. meshing sub-domains independently) [10] or performing local mesh refinement. Entropy stable Lobatto collocation schemes have been constructed on non-conforming meshes in [11] using SBP projection operators. In this work, we extend entropy stable Gauss collocation to non-conforming quadrilateral and hexahedral meshes. While it is straightforward to construct Gauss collocation schemes on non-conforming meshes, the treatment of non-conforming interfaces results in signficantly increased computational costs. We reduce such costs by adopting a mortar-based treatment of non-conforming interfaces. Moreover, while the cost of the proposed scheme is similar to that of [11] in 2D, the mortar-based approach is more computationally efficient for both Lobatto and Gauss collocation schemes on 3D non-conforming hexahedral meshes.

The paper is organized as follows: Section 2 briefly reviews the derivation of an entropy inequality for a system of nonlinear conservation laws. Section 3 introduces "hybridized" SBP operators as a unified way to treat both Lobatto and Gauss collocation schemes on conforming meshes of tensor product elements. Section 4 describes a naive extension to non-conforming meshes and illustrates why this formulation results in an increase in computational costs. Section 5 introduces a mortar-based formulation on curved meshes which addresses such costs, and characterizes the accuracy of the resulting formulations. We conclude with numerical validation of theoretical results in Section 7.

2 Entropy stability for systems of nonlinear conservation laws

We are interested in the numerical approximation of solutions to systems of nonlinear conservation laws

$$\frac{\partial \boldsymbol{u}}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{f}_i(\boldsymbol{u})}{\partial x_i} = 0. \tag{1}$$

Here, $\boldsymbol{u}$ denotes the conservative variables and $\boldsymbol{f}_i(\boldsymbol{u})$ are nonlinear fluxes. We briefly review entropy stability for systems of conservation laws in d dimensions. We assume there exists a convex scalar entropy $S(\boldsymbol{u})$ associated with (1). We then define the entropy variables $\boldsymbol{v}(\boldsymbol{u})$ as the gradient of the entropy $S(\boldsymbol{u})$ with respect to the conservative variables

$$\boldsymbol{v} = \frac{\partial S(\boldsymbol{u})}{\partial \boldsymbol{u}}.$$

For $S(\boldsymbol{u})$ convex, $\boldsymbol{v}(\boldsymbol{u})$ defines an invertible mapping between the conservative and entropy variables, whose inverse (from entropy to conservative variables) we denote by $\boldsymbol{u}(\boldsymbol{v})$. Viscosity solutions to (1) satisfy an integrated form of the entropy inequality [12]

$$\int_{\Omega} \frac{\partial S(\boldsymbol{u})}{\partial t} + \int_{\partial \Omega} \sum_{i=1}^{d} \boldsymbol{n}_i \left(\boldsymbol{v}^T \boldsymbol{f}_i(\boldsymbol{u}) - \psi_i(\boldsymbol{u}) \right) \le 0, \tag{2}$$

where F_i denotes the ith scalar entropy flux function, $\psi_i(\boldsymbol{u}) = \boldsymbol{v}^T \boldsymbol{f}_i(\boldsymbol{u}) - F_i(\boldsymbol{u})$ denotes the ith entropy potential, $\partial\Omega$ denotes the boundary of Ω and $\boldsymbol{n}_i$ denotes the ith component of the outward normal on $\partial\Omega$. This can be interpreted as implying that the time rate of change of entropy is bounded by the entropy flux through the boundary.

3 Entropy stable collocation DG methods and hybridized SBP operators

In this section, we summarize the work of [1] on the construction of entropy stable collocation DG methods based on generalized summation by parts (GSBP) operators. These constructions are applicable to collocation schemes based on either Lobatto and Gauss nodes.

3.1 Hybridized SBP operators in 1D

We begin by introducing collocation discretization matrices on the reference interval $[-1, 1]$. We assume the solution is collocated at $(N + 1)$ quadrature points x_i with associated quadrature weights w_i, and consider primarily Lobatto or Gauss quadrature points. The collocation assumption is equivalent to approximating the solution using a degree N Lagrange basis $\ell_j(x)$ at the $(N + 1)$ quadrature points.

We define mass and integrated differentiation matrices $\boldsymbol{M}, \boldsymbol{Q}$

$$\boldsymbol{M}_{ij} = \int_{-1}^{1} \ell_i(x)\ell_j(x), \qquad \boldsymbol{Q}_{ij} = \int_{-1}^{1} \frac{\partial \ell_j}{\partial x} \ell_i.$$

We assume that all integrals are computed using the collocated quadrature rule, such that

$$\boldsymbol{M}_{ij} = \int_{-1}^{1} \ell_i(x)\ell_j(x) \approx \sum_{k=1}^{N+1} \ell_i(x_k)\ell_j(x_k)w_k = \delta_{ij} w_i$$

$$\boldsymbol{Q}_{ij} = \int_{-1}^{1} \frac{\partial \ell_j}{\partial x} \ell_i \approx \sum_{k=1}^{N+1} \ell_i(x_k) \left.\frac{\partial \ell_j}{\partial x}\right|_{x_k} w_k.$$

In other words, the mass matrix is diagonal with entries equal to the quadrature weights. Since the integrands of $\boldsymbol{M}$ are degree $2N$ polynomials, the collocation approximation of $\boldsymbol{M}$ is exact for Gauss quadrature, but not for Lobatto quadrature. For both Gauss and Lobatto quadrature, the matrix $\boldsymbol{Q}$ is exact under collocation quadrature.

We introduce the $2 \times (N+1)$ matrix $\boldsymbol{E}$ which interpolates values at collocation nodes to values at the endpoints $x = -1$ and $x = 1$. These two matrices are defined entrywise as

$$(\boldsymbol{E})_{1i} = \ell_i(-1), \qquad (\boldsymbol{E})_{2i} = \ell_i(1).$$

The mass and differentiation matrices satisfy a generalized summation by parts property [13]

$$\boldsymbol{Q} = \boldsymbol{E}^T \boldsymbol{B} \boldsymbol{E} - \boldsymbol{Q}^T, \qquad \boldsymbol{B} = \begin{bmatrix} -1 & \\ & 1 \end{bmatrix}. \tag{3}$$

The GSBP property holds for both Lobatto and Gauss nodes, and switching between these two nodal sets simply requires redefining the matrices $\boldsymbol{D}, \boldsymbol{E}$. For Gauss nodes, $\boldsymbol{E}$ is dense. For Lobatto nodes, since the collocation nodes include boundary points, the interpolation matrix $\boldsymbol{E}$ reduces to the matrix which extracts nodal values associated with the left and right endpoints

$$\boldsymbol{E} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 \end{bmatrix}.$$

It is possible to construct energy stable high order discretizations of linear hyperbolic systems using GSBP operators based on Gauss nodes [13]. However, for nonlinear conservation laws, the presence of the dense $\boldsymbol{E}$ matrix in the GSBP property (3) complicates the imposition of boundary conditions and computation of inter-element numerical fluxes [1, 2, 7]. This can be avoided by using "hybridized" (also referred to as decoupled) SBP operators [2, 9, 14], which are defined as the block matrix

$$\boldsymbol{Q}_h = \frac{1}{2} \begin{bmatrix} \boldsymbol{Q} - \boldsymbol{Q}^T & \boldsymbol{E}^T \boldsymbol{B} \\ -\boldsymbol{B}\boldsymbol{E} & \boldsymbol{B} \end{bmatrix}.$$

The hybridized SBP operator satisfies a block form of the SBP property

$$\boldsymbol{Q}_h + \boldsymbol{Q}_h^T = \boldsymbol{B}_h, \qquad \boldsymbol{B}_h = \begin{bmatrix} \boldsymbol{0} & \\ & \boldsymbol{B} \end{bmatrix}. \tag{4}$$

where $\boldsymbol{E}$ does not appear in the block boundary matrix on the right hand side. Here, we have used $\boldsymbol{0}, \boldsymbol{1}$ to denote a matrix or vector of zeros or ones, where the size is inferred from context. Note that the matrix $\boldsymbol{Q}_h$ also satisfies

$$\boldsymbol{Q}_h \boldsymbol{1} = \frac{1}{2} \begin{bmatrix} \boldsymbol{Q}\boldsymbol{1} - \boldsymbol{Q}^T\boldsymbol{1} + \boldsymbol{E}^T \boldsymbol{B}\boldsymbol{1} \\ -\boldsymbol{B}\boldsymbol{E}\boldsymbol{1} - \boldsymbol{B}\boldsymbol{1} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \left(-\boldsymbol{Q}^T + \boldsymbol{E}^T \boldsymbol{B} \right) \boldsymbol{1} \\ \boldsymbol{0} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \boldsymbol{Q}\boldsymbol{1} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{0} \tag{5}$$

where we have used that $\boldsymbol{E}\boldsymbol{1} = \boldsymbol{1}$ (since $\boldsymbol{E}$ is a high order accurate boundary interpolation matrix), $\boldsymbol{Q}\boldsymbol{1} = \boldsymbol{0}$ (since $\boldsymbol{Q}$ is a differentiation matrix), and the GSBP property $-\boldsymbol{Q}^T + \boldsymbol{E}^T \boldsymbol{B}\boldsymbol{E} = \boldsymbol{Q}$

3.2 Hybridized SBP operators in higher dimensions

The formulation (2) can be extended to higher dimensions through a tensor product construction. For simplicity, we illustrate this for 2D quadrilateral elements (the extension to 3D hexahedral elements is straightforward). Let $\boldsymbol{M}_{1\mathrm{D}}, \boldsymbol{Q}_{1\mathrm{D}}$ denote one-dimensional generalized SBP norm (mass) and differentiation matrices, and let $\boldsymbol{E}_{1\mathrm{D}}$ denote the 1D face interpolation matrix. We define multi-dimensional mass and differentiation matrices in terms of Kronecker products

$$\widehat{\boldsymbol{Q}}_1 = \boldsymbol{Q}_{1\mathrm{D}} \otimes \boldsymbol{M}_{1\mathrm{D}}, \qquad \widehat{\boldsymbol{Q}}_2 = \boldsymbol{M}_{1\mathrm{D}} \otimes \boldsymbol{Q}_{1\mathrm{D}}, \qquad \widehat{\boldsymbol{M}} = \boldsymbol{M}_{1\mathrm{D}} \otimes \boldsymbol{M}_{1\mathrm{D}}. \tag{6}$$

We also construct 2D face interpolation matrices from Kronecker products. Let $\boldsymbol{E}_{1\mathrm{D}}$ denote the one-dimensional face interpolation matrix, and let $\boldsymbol{B}_{1\mathrm{D}}$ denote

the 1D boundary matrix in (3). For a specific ordering of the face points, the two-dimensional face interpolation matrix $\boldsymbol{E}$ and reference boundary matrices $\widehat{\boldsymbol{B}}_1, \widehat{\boldsymbol{B}}_2$ are given by

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{E}_{1\mathrm{D}} \otimes \boldsymbol{I}_2 \\ \boldsymbol{I}_2 \otimes \boldsymbol{E}_{1\mathrm{D}} \end{bmatrix}, \qquad \widehat{\boldsymbol{B}}_1 = \begin{bmatrix} \boldsymbol{B}_{1\mathrm{D}} \otimes \boldsymbol{M}_{1\mathrm{D}} \\ & \boldsymbol{0} \end{bmatrix}, \qquad \widehat{\boldsymbol{B}}_2 = \begin{bmatrix} \boldsymbol{0} \\ & \boldsymbol{M}_{1\mathrm{D}} \otimes \boldsymbol{B}_{1\mathrm{D}} \end{bmatrix}, \quad (7)$$

where $\boldsymbol{I}_2$ is the 2×2 identity matrix. Recall that $\boldsymbol{M}_{1\mathrm{D}}$ is the diagonal matrix of quadrature weights, so $\widehat{\boldsymbol{B}}_i$ are diagonal.

The construction of operators in (6) and (7) correspond to the use of tensor product points quadrature points and *aligned* surface points. For example, if Gauss points are used for volume quadrature, we assume that Gauss points are also used for surface quadrature. This ensures that the operators $\widehat{\boldsymbol{Q}}_i, \widehat{\boldsymbol{B}}_i, \boldsymbol{E}$ satisfy the analogous higher dimensional generalized SBP property $\widehat{\boldsymbol{Q}}_i = \boldsymbol{E}^T \widehat{\boldsymbol{B}}_i \boldsymbol{E} - \widehat{\boldsymbol{Q}}_i^T$.

The 2D differentiation and interpolation matrices $\widehat{\boldsymbol{Q}}_i, \boldsymbol{E}$ can now be used to construct 2D hybridized SBP operators. Let $\widehat{\boldsymbol{Q}}_{i,h}$ denote the hybridized SBP operator for the ith coordinate on the reference element, where $\widehat{\boldsymbol{Q}}_{i,h}$ is defined as

$$\widehat{\boldsymbol{Q}}_{i,h} = \frac{1}{2} \begin{bmatrix} \widehat{\boldsymbol{Q}}_i - \widehat{\boldsymbol{Q}}_i^T & \boldsymbol{E}^T \widehat{\boldsymbol{B}}_i \\ -\widehat{\boldsymbol{B}}_i \boldsymbol{E} & \widehat{\boldsymbol{B}}_i \end{bmatrix}. \tag{8}$$

These matrices satisfy the following multi-dimensional properties:

Lemma 1 *Let $\widehat{\boldsymbol{Q}}_i$ be defined as in (6) and let $\widehat{\boldsymbol{Q}}_{i,h}$ be defined as in (8). Then,*

$$\widehat{\boldsymbol{Q}}_{i,h} + \widehat{\boldsymbol{Q}}_{i,h}^T = \begin{bmatrix} \boldsymbol{0} \\ & \widehat{\boldsymbol{B}}_i \end{bmatrix}, \qquad \widehat{\boldsymbol{Q}}_{i,h} \boldsymbol{1} = \boldsymbol{0}. \tag{9}$$

The proof is found in [1]. The SBP property follows from the construction of $\widehat{\boldsymbol{Q}}_{i,h}$, and $\widehat{\boldsymbol{Q}}_{i,h} \boldsymbol{1} = \boldsymbol{0}$ is a consequence of properties of the Kronecker product and the same arguments used to derive (5). The construction of 3D differentiation and interpolation matrices proceeds similarly, and the matrices satisfy the same properties as described in Lemma 1.

We now construct an entropy conservative formulation on an unstructured curved mesh. Suppose the domain Ω is decomposed into non-overlapping elements D^k which are images of the reference mapping, such that D^k is a differentiable mapping of the reference element $\widehat{D} = [-1, 1]^d$. Let $\boldsymbol{x}_i$ denote the ith physical coordinate on D^k and let $\widehat{\boldsymbol{x}}_j$ denote the jth reference coordinate. The mapping between reference and physical element induces scaled geometric terms

$$G_{ij} = J \frac{\partial \widehat{\boldsymbol{x}}_j}{\partial \boldsymbol{x}_i},$$

where J denotes the determinant of the Jacobian of the physical-to-reference mapping. We refer to J as the Jacobian from here onwards. These geometric terms also relate the normal vectors on reference and physical elements. Let $\widehat{\boldsymbol{n}} \widehat{J}_f$ denote the outward normal vector on $\widehat{D}$, scaled by the reference face Jacobians. Then, the (scaled) physical normal vectors on D^k are related to $\widehat{\boldsymbol{n}} \widehat{J}_f$ through

$$n_i J_f = \sum_{j=1}^d G_{ij} \widehat{n}_j \widehat{J}_f. \tag{10}$$

We assume that the mesh is watertight, such that the scaled normal vectors are equal and opposite across shared faces between two elements. We also assume that the geometric terms G_{ij} are constructed such that they are polynomials of degree less than or equal to N and satisfy a discrete geometric conservation law (GCL)

$$\sum_{j=1}^{d} \frac{\partial}{\partial \widehat{x}_j} G_{ij} = 0. \tag{11}$$

The discrete GCL is satisfied automatically for isoparametric mappings in 2D, and there exist several techniques to enforce the satisfaction of a discrete GCL on various three-dimensional domains [7, 8, 15–18]. We will discuss specific approaches for non-conforming meshes in Section 3.3 and Appendix A.

We can now assemble physical mass and differentiation matrices on D^k from reference mass and differentiation matrices using the chain rule. Let $\widehat{Q}_{j,h}$ denote the jth reference differentiation matrix on $\widehat{D}$. We can define the physical differentiation matrices $Q_{i,h}$ on D^k via the skew-symmetric splitting

$$Q_{i,h} = \frac{1}{2} \sum_{j=1}^{d} \mathrm{diag}\,(G_{ij})\, \widehat{Q}_{j,h} + \widehat{Q}_{j,h} \mathrm{diag}\,(G_{ij})\,, \tag{12}$$

where G_{ij} denotes the vector of values of $G_{ij} = J \frac{\partial \widehat{x}_j}{\partial x_i}$ at both volume and surface points. Since G_{ij} is assumed to be a polynomial of degree less than or equal to N, G_{ij} is constructed using polynomial interpolation.

Because the reference matrices $\widehat{Q}_{j,h}$ satisfy the reference SBP properties (4), one can show [8] that the physical matrices $Q_{i,h}$ satisfy analogous SBP properties.

Lemma 2 *Suppose the geometric terms G_{ij} satisfy the discrete GCL (11), the normals are constructed via (10), and that $Q_{i,h}$ is constructed using (12). Then,*

$$Q_{i,h} + Q_{i,h}^T = \begin{bmatrix} \mathbf{0} & \\ & B_i \end{bmatrix}, \qquad Q_{i,h}\mathbf{1} = \mathbf{0},$$

where n_i is a vector whose entries are values of the ith component of the scaled normals $n_i J_f$ at face points, w_f is the vector of face quadrature weights w_i, and $B_i = \mathrm{diag}\,(n_i \circ w_f)$.

The proof is given in [8].

3.3 Entropy conservative formulations on conforming meshes of mapped elements

The aforementioned matrices can now be used to construct high order accurate entropy stable and entropy conservative schemes. We first introduce an entropy conservative numerical flux [19]. Let u_L, u_R denote left and right states in the conservative variables. Then, an entropy conservative numerical flux is a vector-valued function $f_{i,S}$ for $i = 1, \ldots, d$ which satisfies the following three properties

$$f_{i,S}\,(u, u) = f_i(u), \qquad \text{(consistency)}$$

$$f_{i,S}\,(u_L, u_R) = f_{i,S}(u_R, u_L), \qquad \text{(symmetry)}$$

$$(v_L - v_R)^T f_{i,S}\,(u_L, u_R) = \psi_i(u_L) - \psi_i(u_R), \qquad \text{(conservation)}.$$

Here, $\boldsymbol{v}_L, \boldsymbol{v}_R$ are the entropy variables evaluated at $\boldsymbol{u}_L, \boldsymbol{u}_R$, and ψ_i is the ith entropy potential which appears in (2). Then, an entropy conservative scheme on a mapped element is given by

$$\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} + \begin{bmatrix}\boldsymbol{I}\\\boldsymbol{E}\end{bmatrix}^T \sum_{i=1}^{d}\left(2\boldsymbol{Q}_{i,h}\circ\boldsymbol{F}_i\right)\boldsymbol{1} + \boldsymbol{E}^T\boldsymbol{B}_i\left(\boldsymbol{f}_i^* - \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_f)\right) = 0 \qquad (13)$$

$$\left(\boldsymbol{F}_i\right)_{jk} = \boldsymbol{f}_S\left(\widetilde{\boldsymbol{u}}_j, \widetilde{\boldsymbol{u}}_k\right), \qquad \boldsymbol{f}_i^* = \boldsymbol{f}_{i,S}\left(\widetilde{\boldsymbol{u}}_f^+, \widetilde{\boldsymbol{u}}_f\right),$$

where $\boldsymbol{M} = \widehat{\boldsymbol{M}} \circ \mathrm{diag}\left(\boldsymbol{J}\right)$ is the diagonal mass matrix scaled by the determinant of the geometric Jacobian J evaluated at nodal points. Here, the exterior state $\widetilde{\boldsymbol{u}}_f^+$ used to evaluate the numerical flux $\boldsymbol{f}_i^*$ corresponds either to interface values on a neighboring element or an exterior state used to enforce boundary conditions, and $\circ$ denotes the matrix Hadamard product. Assuming continuity in time, the formulation (13) is entropy conservative in the sense that

$$\boldsymbol{1}^T\boldsymbol{M}\frac{\mathrm{d}S(\boldsymbol{u}_h)}{\mathrm{dt}} + \sum_{i=1}^{d}\boldsymbol{1}^T\boldsymbol{B}_i\left(\boldsymbol{v}_f^T\boldsymbol{f}_i^* - \psi_i(\widetilde{\boldsymbol{u}}_f)\right) = 0, \qquad \boldsymbol{v}_f = \boldsymbol{E}\boldsymbol{v}(\boldsymbol{u}_h).$$

The proof uses similar techniques as proofs in other papers [2, 6, 7, 9] The proof of entropy conservation relies mainly on Lemma 2, which states that $\boldsymbol{Q}_{i,h}$ is conservative (e.g., exact for constants) and satisfies the SBP property. Extensions to the non-conforming setting will utilize the same properties.

All entropy conservative schemes described here can be made entropy stable by introducing entropy dissipation through mechanisms such as physical or artificial viscosity [20, 21]. We introduce dissipation by incorporating a penalization term into the interface flux [22]. For example, Lax-Friedrichs dissipation can be added by modifying the interface flux term $\boldsymbol{B}_i\boldsymbol{f}_i^*$

$$\boldsymbol{B}_i\boldsymbol{f}_i^* \Longrightarrow \boldsymbol{B}_i\boldsymbol{f}_i^* - \frac{\lambda}{2}[\![\widetilde{\boldsymbol{u}}_f]\!], \qquad [\![\widetilde{\boldsymbol{u}}_f]\!] = \widetilde{\boldsymbol{u}}_f^+ - \widetilde{\boldsymbol{u}}_f,$$

where λ is an estimate of the maximum wave speed [2, 6].

For the remainder of this work, we will construct formulations and prove they are entropy conservative, with the understanding that they can be made entropy stable by incorporating an entropy dissipative penalty term.

4 Non-conforming meshes

Section 3 describes the construction of entropy stable schemes on geometrically conforming meshes, where each element shares at most one neighbor across a face. We extend the construction of stable schemes to meshes containing geometric non-conformity, where an element can share a face with two or more neighboring elements. To ensure stability, the coupling conditions imposed at this non-conforming face must be handled appropriately. For DG discretizations, this is most naturally achieved by combining composite quadrature rules on non-conforming faces with appropriate evaluations of volume terms [17]. However, for entropy stable schemes, the naive use of composite quadrature at non-conforming interfaces can significantly increase the computational cost.

(a) Flux evaluations required for aligned surface nodes

(b) Flux evaluations required for non-aligned surface nodes
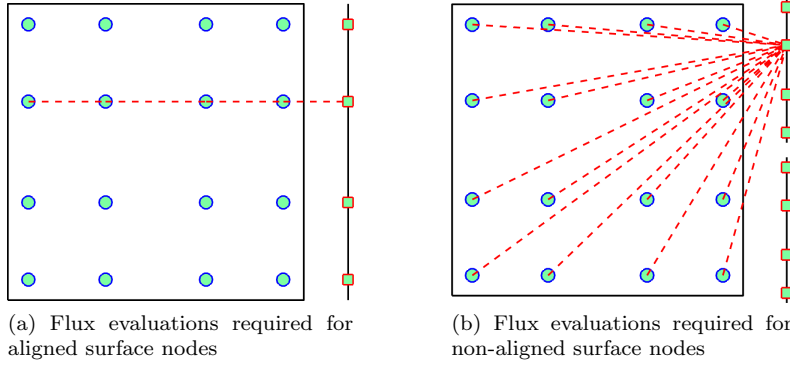
Fig. 1: Nodes between which flux evaluations are required for Gauss nodes. Aligned surface nodes (conforming interfaces) require evaluations between each surface node and a line of volume nodes, while non-aligned surface nodes (non-conforming interfaces) require flux evaluations between a surface node and *all* volume nodes.

For quad and hex elements under tensor product volume quadrature, the differentiation matrices are Kronecker products of 1D differentiation matrices and diagonal mass matrices, which result in sparse operators. As a result, flux evaluations are only required between "lines" of volume nodes [1, 4]. Flux evaluations also follow the sparsity pattern of the matrix E, which maps from volume nodes to surface nodes. For Lobatto or Gauss collocation methods on conforming quadrilateral and hexahedral meshes, E is sparse if the surface quadrature nodes are aligned with volume quadrature nodes [1]. In such cases, flux evaluations are required only between lines of volume nodes and adjacent surface nodes, as shown in Figure 1a.

Composite quadrature rules on non-conforming meshes, however, are not aligned with volume nodes. Suppose that D^k is a quadrilateral element with two neighbors across each face, such that every face is non-conforming and utilizes a composite quadrature rule. The entropy conservative formulation (13) can be extended to the non-conforming case by redefining the interpolation matrix E as the matrix which interpolates from volume nodes to composite surface nodes. However, this version of E is fully dense, and evaluating the formulation (13) requires flux evaluations between each surface node and *all* volume nodes, as illustrated in Figure 1b. This greatly increases computational costs, especially at high orders of approximation.

5 Entropy stable mortar formulations

The goal of this work is to reduce computational costs for Gauss collocation schemes in the presence of non-conforming interfaces. This can be done by treating composite quadrature nodes as a layer of "mortar" nodes which are coupled directly to surface nodes, but not directly to the volume nodes, as illustrated in Figure 2. This results in modifications of the matrices involved in the entropy stable formulation (13). These modifications preserve both high order accuracy and entropy stablility, and yield an implementation which is identical to that of (13) except for a face-local correction to the numerical flux.
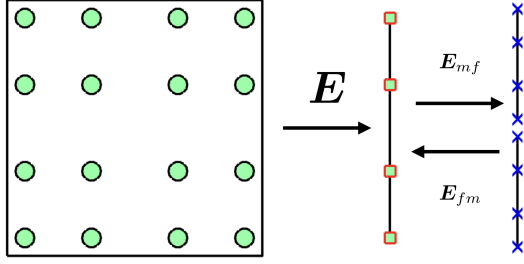
Fig. 2: Illustration of mortar operators for a Gauss collocation scheme. The matrix $\boldsymbol{E}$ maps from volume quadrature points to surface quadrature points, $\boldsymbol{E}_m$ maps from surface to mortar surface points, and $\widetilde{\boldsymbol{E}}_m$ maps from mortar surface points to surface points. Fix notation after Mario standardizes it.

We introduce new matrices on the reference element $\widehat{D}$. Let $\widehat{\boldsymbol{x}}$ denote volume collocation points, and let $\widehat{\boldsymbol{x}}_f$ denote face (surface) points on the boundary $\partial\widehat{D}$. To simplify notation, we assume from this point onwards that the nodes are ordered face-by-face. In 2D, this implies that $\widehat{\boldsymbol{x}}_f$ is

$$\widehat{\boldsymbol{x}}_f = \begin{bmatrix} \widehat{\boldsymbol{x}}_{f,1} \ \widehat{\boldsymbol{x}}_{f,2} \ \widehat{\boldsymbol{x}}_{f,3} \ \widehat{\boldsymbol{x}}_{f,4} \end{bmatrix}^T, \tag{14}$$

where $\widehat{\boldsymbol{x}}_{f,j}$ denotes the vector of 1D nodal positions on the jth face of the reference quadrilateral $[-1,1]^2$. Recall that the matrix $\boldsymbol{E}$ as defined in (7) interpolates from volume collocation points $\widehat{\boldsymbol{x}}$ to a specific ordering of the surface points $\widehat{\boldsymbol{x}}_f$. The ordering (14) simply corresponds to a permutation of the interpolation matrix $\boldsymbol{E}$.

We now introduce a second set of mortar points $\widehat{\boldsymbol{x}}_m$ on a face of the reference element. We assume these points also correspond to quadrature nodes with corresponding weights $\boldsymbol{w}_m$. For example, mortar nodes can be composite Gauss or Lobatto quadrature nodes (for h non-conforming interfaces), higher degree Gauss or Lobatto nodes (for p non-conforming interfaces), or identical to the surface nodes $\widehat{\boldsymbol{x}}_f$ (non-mortar interface). For simplicity of notation, we assume that each face has the same set of mortar nodes; however, it is straightforward to extend this to the case when the mortar nodes vary face-by-face.

We define the interpolation matrix $\boldsymbol{E}_{mf}$ as the operator which maps values at surface nodes to values at mortar nodes. In 2D, since the face of a quadrilateral is a 1D line, we can define $\boldsymbol{E}_{mf}$ as the block matrix acting on the surface nodes on the four faces

$$\boldsymbol{E}_{mf} = \boldsymbol{I}_{4\times 4} \otimes \boldsymbol{E}_m,$$

where $\boldsymbol{I}_{4\times 4}$ is the 4-by-4 identity matrix and $\boldsymbol{E}_m$ is the mortar interpolation matrix over the reference face (interval) $[-1,1]$. The matrix $\boldsymbol{E}_m$ is defined in terms of $\widehat{x}_{m,i}$, the mortar nodes mapped to the reference interval $[-1,1]$,

$$\boldsymbol{E}_{mf} = \ell_j(\widehat{x}_{m,i}), \qquad 1 \le i \le \text{num. mortar points}, \quad 1 \le j \le (N+1).$$

We can define an analogous matrix $\boldsymbol{E}_{fm}$ which maps from the mortar nodes back to the surface nodes. We first define the face mass matrix as the block diagonal

matrix whose blocks are 1D diagonal mass matrices

$$\boldsymbol{M}_f = \boldsymbol{I}_{4\times4} \otimes \boldsymbol{M}_{1\mathrm{D}}.$$

These matrices are defined in 2D for simplicity, but are straightforward to extend to 3D. The matrix $\boldsymbol{E}_{fm}$ can now be defined through a quadrature-based L^2 projection

$$\boldsymbol{E}_{fm} = \boldsymbol{M}_f^{-1} \boldsymbol{E}_{mf}^T \boldsymbol{M}_m, \qquad \boldsymbol{M}_m = \boldsymbol{I}_{4\times4} \otimes \mathrm{diag}\left(\boldsymbol{w}_m\right). \tag{15}$$

Finally, we introduce diagonal boundary matrices on surface and mortar nodes

$$\widehat{\boldsymbol{B}}_{i,f} = \mathrm{diag}\left(\widehat{\boldsymbol{n}}_{i,f}\right)\boldsymbol{M}_f \qquad \widehat{\boldsymbol{B}}_{i,m} = \mathrm{diag}\left(\widehat{\boldsymbol{n}}_{i,m}\right)\boldsymbol{M}_m, \tag{16}$$

where $\widehat{\boldsymbol{n}}_{i,f}, \widehat{\boldsymbol{n}}_{i,m}$ are vectors containing components of the scaled reference normals at face and mortar points, respectively.

Note that $\boldsymbol{E}_{fm}$ exactly recovers polynomials of a certain degree on the reference face, where the degree of the polynomial is related to the accuracy of the surface and mortar quadratures.

Lemma 3 *Suppose the face (surface) quadrature is exact for polynomials of degree $N + N_f$ and mortar quadratures are exact for degree $N + N_m$ polynomials. Then, $\boldsymbol{E}_{fm}$ exactly recovers polynomials of degree $\min(N, N_f, N_m)$.*

Proof Let $u(\boldsymbol{x})$ be a polynomial of degree $\min(N, N_f, N_m)$ or less, and let $\boldsymbol{u}_f$ be its values on the face (surface) nodes. Then, $\boldsymbol{u}_m = \boldsymbol{E}_{mf}\boldsymbol{u}_f$ are the interpolated values of the polynomial on the mortar nodes. Applying $\boldsymbol{E}_{fm}$ yields

$$\boldsymbol{E}_{fm}\boldsymbol{u}_m = \boldsymbol{M}_f^{-1}\boldsymbol{E}_{mf}^T\boldsymbol{M}_m\boldsymbol{E}_{mf}\boldsymbol{u}_f.$$

The entries of $\boldsymbol{E}_{mf}^T\boldsymbol{M}_m\boldsymbol{E}_{mf}$ are integrals of products of Lagrange basis functions with $u(\boldsymbol{x})$, where all integrals are approximated using mortar quadrature.

Since u is degree $\min(N, N_f, N_m)$ and each Lagrange basis function is degree N, the integrand is computed exactly under the mortar quadrature. Moreover, since u is degree $\min(N, N_f, N_m)$, it is also computed exactly using the face (surface) quadrature. Thus, $\boldsymbol{E}_{mf}^T\boldsymbol{M}_m\boldsymbol{E}_{mf}\boldsymbol{u}_f = \boldsymbol{M}_f\boldsymbol{u}_f$ by exactness of the face and mortar quadratures, and $\boldsymbol{E}_{fm}\boldsymbol{u}_m = \boldsymbol{u}_f$. $\square$

5.1 Mortar-based hybridized SBP operators

The matrices $\boldsymbol{E}_{mf}, \boldsymbol{E}_{fm}, \widehat{\boldsymbol{B}}_{i,f}, \widehat{\boldsymbol{B}}_{i,m}$ can now be used to construct SBP operators which involve mortar nodes. We define the mortar-based hybridized SBP operator on the reference element $\widehat{\boldsymbol{Q}}_{i,m}$ to be

$$\widehat{\boldsymbol{Q}}_{i,m} = \frac{1}{2}\begin{bmatrix} \widehat{\boldsymbol{Q}}_i - \widehat{\boldsymbol{Q}}_i^T & \boldsymbol{E}^T\widehat{\boldsymbol{B}}_{i,f} & \\ -\widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E} & & \widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E}_{fm} \\ & -\widehat{\boldsymbol{B}}_{i,m}\boldsymbol{E}_{mf} & \widehat{\boldsymbol{B}}_{i,m} \end{bmatrix}. \tag{17}$$

Since it is not clear at first glance how the hybridized operator $\widehat{\boldsymbol{Q}}_{i,h}$ or its mortar-based variant $\widehat{\boldsymbol{Q}}_{i,m}$ can be used to perform differentiation, we review intuitive explanations of each operator.

Let $\boldsymbol{u}$ denote basis coefficients for some function $u(\boldsymbol{x})$, and let f, g denote two functions on the reference element. Suppose $\boldsymbol{u}$ satisfies the following matrix system involving the standard hybridized operator $\widehat{\boldsymbol{Q}}_{i,h}$

$$\widehat{\boldsymbol{M}}\boldsymbol{u} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \end{bmatrix}^T \operatorname{diag}\left(\boldsymbol{f}\right) \widehat{\boldsymbol{Q}}_{i,h} \operatorname{diag}\left(\boldsymbol{g}\right), \qquad \boldsymbol{f} = \begin{bmatrix} f(\widehat{\boldsymbol{x}}) \\ f(\widehat{\boldsymbol{x}}_f) \end{bmatrix}, \qquad \boldsymbol{g} = \begin{bmatrix} g(\widehat{\boldsymbol{x}}) \\ g(\widehat{\boldsymbol{x}}_f) \end{bmatrix}. \quad (18)$$

It was shown in [2, 9] that this corresponds to a high order accurate approximation of $f\frac{\partial g}{\partial x_i}$. Let $\boldsymbol{v}$ denote basis coefficients for some arbitrary degree N polynomial test function. Expanding out the different blocks of (18) and using generalized summation by parts for $\widehat{\boldsymbol{Q}}_i$ yields

$$\widehat{\boldsymbol{M}}\boldsymbol{u} = \operatorname{diag}\left(f(\widehat{\boldsymbol{x}})\right) \widehat{\boldsymbol{Q}}_i g(\widehat{\boldsymbol{x}}) + \frac{1}{2} \left(\operatorname{diag}\left(f(\widehat{\boldsymbol{x}})\right) \boldsymbol{E}^T + \boldsymbol{E}^T f(\widehat{\boldsymbol{x}}_f) \right) \widehat{\boldsymbol{B}}_i \left(g(\widehat{\boldsymbol{x}}_f) - \boldsymbol{E} g(\widehat{\boldsymbol{x}}) \right).$$

The latter expression can be interpreted as a boundary correction term. Since $\boldsymbol{E}$ is a high order accurate boundary interpolation operator, this correction term vanishes if g is a degree N polynomial.

The mortar-based hybridized SBP operator can be similarly interpreted as a high order differentiation operator. Recall that $\boldsymbol{E}_{mf}$ interpolates from face nodes to mortar nodes, and that $\boldsymbol{E}$ interpolates from volume nodes to face nodes. Thus, the matrix $\boldsymbol{E}_{mf}\boldsymbol{E}$ interpolates from volume nodes to mortar nodes. We can use this matrix to replicate (18) for the mortar-based operator $\widehat{\boldsymbol{Q}}_{i,m}$. Let $\boldsymbol{u}$ solve the following system

$$\widehat{\boldsymbol{M}}\boldsymbol{u} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{mf}\boldsymbol{E} \end{bmatrix}^T \operatorname{diag}\left(\boldsymbol{f}\right) \widehat{\boldsymbol{Q}}_{i,m} \operatorname{diag}\left(\boldsymbol{g}\right), \qquad \boldsymbol{f} = \begin{bmatrix} f(\widehat{\boldsymbol{x}}) \\ f(\widehat{\boldsymbol{x}}_f) \\ f(\widehat{\boldsymbol{x}}_m) \end{bmatrix}, \quad \boldsymbol{g} = \begin{bmatrix} g(\widehat{\boldsymbol{x}}) \\ g(\widehat{\boldsymbol{x}}_f) \\ g(\widehat{\boldsymbol{x}}_m) \end{bmatrix}. \quad (19)$$

Expanding out terms yields a similar expression involving multiple correction terms

$$
\begin{aligned}
\widehat{\boldsymbol{M}}\boldsymbol{u} = {} & \operatorname{diag}\left(f(\widehat{\boldsymbol{x}})\right) \widehat{\boldsymbol{Q}}_i g(\widehat{\boldsymbol{x}}) \qquad\qquad\qquad\qquad\qquad\qquad (20) \\
& + \frac{1}{2} \operatorname{diag}\left(f(\widehat{\boldsymbol{x}})\right) \boldsymbol{E}^T \widehat{\boldsymbol{B}}_{i,f} \left(g(\widehat{\boldsymbol{x}}_f) - \boldsymbol{E} g(\widehat{\boldsymbol{x}}) \right) \\
& + \frac{1}{2} \boldsymbol{E}^T \operatorname{diag}\left(f(\widehat{\boldsymbol{x}}_f)\right) \widehat{\boldsymbol{B}}_{i,f} \left(\boldsymbol{E}_{fm} g(\widehat{\boldsymbol{x}}_m) - \boldsymbol{E} g(\widehat{\boldsymbol{x}}) \right) \\
& + \frac{1}{2} \boldsymbol{E}^T \boldsymbol{E}_{mf}^T \operatorname{diag}\left(f(\widehat{\boldsymbol{x}}_m)\right) \widehat{\boldsymbol{B}}_{i,m} \left(g(\widehat{\boldsymbol{x}}_m) - \boldsymbol{E}_{mf} g(\widehat{\boldsymbol{x}}_f) \right)
\end{aligned}
$$

We can show that these correction terms vanish for polynomial functions g.

Lemma 4 *Let $f(\boldsymbol{x}) = 1$ and suppose that the face (surface) quadrature is exact for polynomials of degree $N + N_f$ and mortar quadratures are exact for degree $N + N_m$ polynomials. Then, derivative approximations given by the equations (19) and (20) are exact if $g(\boldsymbol{x})$ is a degree $\min(N, N_f, N_m)$ polynomial.*

Proof Following [2], if $f = 1$ and $g(\boldsymbol{x})$ is a degree N polynomial, then (20) is exact if the correction terms vanish. The first and third correction terms are zero by the fact that $\boldsymbol{E}, \boldsymbol{E}_{mf}$ are degree N interpolation operators, and the second correction term vanishes since Lemma 3 implies that $\boldsymbol{E}_{fm}$ exactly recovers polynomials of degree $\min(N, N_f, N_m)$. $\qquad\square$

Finally, we note that the mortar-based hybridized SBP operators satisfy a summation by parts property.

Lemma 5 *Let $\widehat{\boldsymbol{Q}}_{i,m}$ be defined as in (17). Then,*

$$\widehat{\boldsymbol{Q}}_{i,m} + \widehat{\boldsymbol{Q}}_{i,m}^T = \begin{bmatrix} \boldsymbol{0} & & \\ & \boldsymbol{0} & \\ & & \widehat{\boldsymbol{B}}_{i,m} \end{bmatrix}, \qquad \widehat{\boldsymbol{Q}}_{i,m}\boldsymbol{1} = \boldsymbol{0}.$$

Proof The SBP property holds if

$$\widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E}_{fm} = (\widehat{\boldsymbol{B}}_{i,m}\boldsymbol{E}_{mf})^T = \boldsymbol{E}_{mf}^T\widehat{\boldsymbol{B}}_{i,m}, \tag{21}$$

where we have used that $\widehat{\boldsymbol{B}}_{i,m}$ is diagonal. Recall that $\boldsymbol{E}_{fm} = \boldsymbol{M}_f^{-1}\boldsymbol{E}_{mf}^T\boldsymbol{M}_m$. Then, by the definition of $\widehat{\boldsymbol{B}}_{i,m}, \widehat{\boldsymbol{B}}_{i,h}$ in (16), we have that

$$\widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E}_{fm} = \operatorname{diag}\left(\widehat{\boldsymbol{n}}_{i,f}\right)\boldsymbol{M}_f\boldsymbol{M}_f^{-1}\boldsymbol{E}_{mf}^T\boldsymbol{M}_m = \operatorname{diag}\left(\widehat{\boldsymbol{n}}_{i,f}\right)\boldsymbol{E}_{mf}^T\boldsymbol{M}_m.$$

Then, since the scaled outward normals $\widehat{n}_i\widehat{J}_f$ are constant over each face of the reference element and $\boldsymbol{E}_{mf}, \boldsymbol{M}_m$ are block diagonal matrices (with each block corresponding to a face), $\operatorname{diag}\left(\widehat{\boldsymbol{n}}_{i,f}\right)$ commutes and

$$\operatorname{diag}\left(\widehat{\boldsymbol{n}}_{i,f}\right)\boldsymbol{E}_{mf}^T\boldsymbol{M}_m = \boldsymbol{E}_{mf}^T\boldsymbol{M}_m\operatorname{diag}\left(\widehat{\boldsymbol{n}}_{i,m}\right) = \boldsymbol{E}_{mf}^T\widehat{\boldsymbol{B}}_{i,m}.$$

$\square$

5.2 Mortar-based SBP operators and entropy conservative formulations on mapped elements

We can extend mortar-based SBP operators to mapped curvilinear elements by constructing physical operators in a manner akin to (12). Abusing notation, we use $\boldsymbol{G}_{ij}$ to now denote the vector containing values of $G_{ij} = J\frac{\partial\widehat{\boldsymbol{x}}_j}{\partial\boldsymbol{x}_i}$ at volume, surface, and mortar points. Then, we define physical operators $\boldsymbol{Q}_{i,m}$ via

$$\boldsymbol{Q}_{i,m} = \frac{1}{2}\sum_{j=1}^d \operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\widehat{\boldsymbol{Q}}_{j,m} + \widehat{\boldsymbol{Q}}_{j,m}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right). \tag{22}$$

Proofs of entropy conservation (e.g., for the formulation (13)) require that the hybridized SBP operators are conservative (e.g., exact for constants) and satisfy the SBP property. Entropy conservation for mortar-based formulations on mapped elements will require similar properties. However, in contrast to hybridized SBP operators, conservation for mapped mortar-based operators requires additional constraints on the accuracy of surface and mortar quadratures relative to the polynomial degree of the geometric mapping.

We assume now that each geometric mapping from reference element $\widehat{D}$ to physical element D^k is a polynomial of degree $N_{\text{geo}} \leq N$. We also introduce the space of tensor product polynomials in d dimensions $Q^{N_1,\ldots,N_d}$ as

$$Q^{N_1,\ldots,N_d} = \left\{\widehat{x}_1^{i_1}\widehat{x}_2^{i_2}\ldots\widehat{x}_d^{i_d}, \qquad 0 \leq i_k \leq N_k, \quad k = 1,\ldots,d.\right\}$$

We denote the isotropic tensor product space $Q^N = Q^{N,\dots,N}$ for conciseness.

In [9], it was shown that the geometric terms G_{ij} are tensor product polynomials of specific degrees. For mapped quadrilateral elements, G_{ij} satisfy

$$G_{i1} \in Q^{N_{\text{geo}}, N_{\text{geo}}-1}$$
$$G_{i2} \in Q^{N_{\text{geo}}-1, N_{\text{geo}}}.$$

for $i = 1, 2$, and the geometric terms naturally satisfy the GCL condition (11).

For hexahedral elements, it is more challenging to construct geometric terms G_{ij} which satisfy the GCL while retaining high order accuracy. This is further complicated by the fact that we must take into account the polynomial degrees of G_{ij} when proving conservation for mortar-based SBP operators. We consider two approaches for computing hexahedral geometric terms in this work:

1. Approach 1: the construction of G_{ij} from [16], which yields

$$G_{ij} \in Q^{N_{\text{geo}}}, \qquad i, j = 1, 2, 3.$$

2. Approach 2: the construction of G_{ij} from [17] (Appendix C.3, see also Footnote 3 in [9]), which yields

$$G_{i1} \in Q^{N_{\text{geo}}, N_{\text{geo}}-1, N_{\text{geo}}-1}$$
$$G_{i2} \in Q^{N_{\text{geo}}-1, N_{\text{geo}}, N_{\text{geo}}-1}$$
$$G_{i3} \in Q^{N_{\text{geo}}-1, N_{\text{geo}}-1, N_{\text{geo}}}, \qquad i = 1, 2, 3.$$

These approaches are described in more detail in Appendix A.

Lemma 6 *Suppose D^k is a tensor product element element with tensor product surface quadratures and that $\boldsymbol{Q}_{i,m}$ is constructed using (22). Suppose also that the surface quadrature is exact for Q^{N+N_f}, the mortar quadrature is exact for Q^{N+N_m}. If*

1. *D^k is a quadrilateral element and $N_{\text{geo}} \le \min(N, N_f + 1, N_m + 1)$, or*
2. *D^k is a hexahedral element, G_{ij} is constructed via Approach 1 [16], and $N_{\text{geo}} \le \min(N, N_f, N_m)$, or*
3. *D^k is hexahedral element, G_{ij} is constructed via Approach 2 [17], and $N_{\text{geo}} \le \min(N, N_f + 1, N_m + 1)$,*

then the following properties hold:

$$\boldsymbol{Q}_{i,m} + \boldsymbol{Q}_{i,m}^T = \boldsymbol{B}_{i,m}, \quad \text{(SBP property)}$$
$$\boldsymbol{Q}_{i,m}\mathbf{1} = \mathbf{0}, \quad \text{(conservation)},$$

where the boundary matrix

$$\boldsymbol{B}_{i,m} = \text{diag}\left(\boldsymbol{n}_{i,m} \circ \boldsymbol{w}_m\right) \tag{23}$$

is the diagonal matrix whose entries consist of the scaled physical normals.

Proof Using the SBP property of $\widehat{\boldsymbol{Q}}_{i,m}$, proving the SBP property follows the same steps as the proof of Lemma 2. Expanding out $\boldsymbol{Q}_{i,m} + \boldsymbol{Q}_{i,m}^T$ yields

$$\frac{1}{2}\sum_{j=1}^{d}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\widehat{\boldsymbol{Q}}_{j,m} + \widehat{\boldsymbol{Q}}_{j,m}^T\operatorname{diag}\left(\boldsymbol{G}_{ij}\right) + \widehat{\boldsymbol{Q}}_{j,m}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right) + \operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\widehat{\boldsymbol{Q}}_{j,m}^T.$$

For each term in the sum, we use the SBP property of $\widehat{\boldsymbol{Q}}_{j,m}$. The volume terms cancel, leaving only surface terms

$$\frac{1}{2}\sum_{j=1}^{d}\left(\widehat{\boldsymbol{B}}_{j,m}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right) + \operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\widehat{\boldsymbol{B}}_{j,m}\right) = \begin{bmatrix}\boldsymbol{0} & & \\ & \boldsymbol{0} & \\ & & \boldsymbol{B}_{i,m}\end{bmatrix}.$$

where we have used (10) and that $\widehat{\boldsymbol{B}}_{j,m}$ and $\operatorname{diag}\left(\boldsymbol{G}_{ij}\right)$ are diagonal matrices.

To show conservation, expanding out $\boldsymbol{Q}_{i,m}\boldsymbol{1}$ yields

$$\boldsymbol{Q}_{i,m}\boldsymbol{1} = \frac{1}{2}\sum_{j=1}^{d}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\widehat{\boldsymbol{Q}}_{j,m}\boldsymbol{1} + \widehat{\boldsymbol{Q}}_{j,m}\operatorname{diag}\left(\boldsymbol{G}_{ij}\right)\boldsymbol{1} = \frac{1}{2}\sum_{j=1}^{d}\widehat{\boldsymbol{Q}}_{j,m}\boldsymbol{G}_{ij}$$

since $\widehat{\boldsymbol{Q}}_{j,m}\boldsymbol{1} = \boldsymbol{0}$ by Lemma 5. Expanding out remaining terms using (20) yields

$$\sum_{j=1}^{d}\widehat{\boldsymbol{Q}}_{j}G_{ij}(\widehat{\boldsymbol{x}}) + \frac{1}{2}\boldsymbol{E}^T\widehat{\boldsymbol{B}}_{j,f}\left(G_{ij}(\widehat{\boldsymbol{x}}_f) - \boldsymbol{E}G_{ij}(\widehat{\boldsymbol{x}})\right)$$

$$+ \frac{1}{2}\boldsymbol{E}^T\widehat{\boldsymbol{B}}_{j,f}\left(\boldsymbol{E}_{fm}G_{ij}(\widehat{\boldsymbol{x}}_m) - \boldsymbol{E}G_{ij}(\widehat{\boldsymbol{x}})\right)$$

$$+ \frac{1}{2}\boldsymbol{E}^T\boldsymbol{E}_{mf}^T\widehat{\boldsymbol{B}}_{j,m}\left(G_{ij}(\widehat{\boldsymbol{x}}_m) - \boldsymbol{E}_{mf}G_{ij}(\widehat{\boldsymbol{x}}_f)\right).$$

where $G_{ij}(\widehat{\boldsymbol{x}}), G_{ij}(\widehat{\boldsymbol{x}}_f), G_{ij}(\widehat{\boldsymbol{x}}_m)$ denote the values of G_{ij} at volume, surface, and mortar points. By the fact that $G_{ij} \in Q^N$ and $\boldsymbol{E}, \boldsymbol{E}_{mf}$ are degree N interpolation operators, the first and third correction terms vanish. The remaining boundary correction terms are

$$\boldsymbol{E}^T\widehat{\boldsymbol{B}}_{j,f}\left(\boldsymbol{E}_{fm}G_{ij}(\widehat{\boldsymbol{x}}_m) - \boldsymbol{E}G_{ij}(\widehat{\boldsymbol{x}})\right). \tag{24}$$

Recall that $\widehat{\boldsymbol{B}}_{i,f}$ is a diagonal matrix whose entries are $\widehat{\boldsymbol{n}}_{i,m}$ scaled by the mortar quadrature weights. On tensor product elements, $\widehat{\boldsymbol{n}}_{i,m} = \pm 1$ on faces where $\widehat{x}_i = \pm 1$ and zero otherwise. Thus, restricting $\widehat{x}_1 = \pm 1$ yields the boundary values of G_{11} on faces where the correction term (24) is non-zero. We can now show that (24) vanishes for $i = j = 1$ (the cases of $i = 2, \ldots, d$ are similar).

1. If D^k is a quadrilateral element, $G_{11} \in Q^{N_{\mathrm{geo}}, N_{\mathrm{geo}}-1}$, and the surface traces of G_{11} are in the 1D trace space $P^{N_{\mathrm{geo}}-1}$.
2. If D^k is a hexahedral element and G_{ij} is constructed using Approach 1 [17], $G_{11} \in Q^{N_{\mathrm{geo}}, N_{\mathrm{geo}}-1, N_{\mathrm{geo}}-1}$, and the surface traces of G_{11} are in the quadrilateral trace space $Q^{N_{\mathrm{geo}}-1, N_{\mathrm{geo}}-1}$.

Since $N_{\mathrm{geo}} \leq N_m + 1$, by Lemma 3 and the assumption that $N_{\mathrm{geo}} \leq \min(N, N_f + 1, N_m + 1)$, $\boldsymbol{E}_{fm}G_{ij}(\widehat{\boldsymbol{x}}_m) = \boldsymbol{E}G_{ij}(\widehat{\boldsymbol{x}})$ and (24) vanishes.

If Approach 1 [16] is used, then surface traces of G_{11} are contained in $Q^{N_{\mathrm{geo}}, N_{\mathrm{geo}}}$. Then, (24) vanishes since $N_{\mathrm{geo}} \leq \min(N, N_f, N_m)$. In both cases, the remaining terms vanish assuming that G_{ij} satisfies the discrete GCL (11). $\qquad\square$

An entropy conservative mortar formulation on a mapped element is then

$$\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}} + \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{mf}\boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^{d} (2\boldsymbol{Q}_{i,m} \circ \boldsymbol{F}_i)\,\mathbf{1} + \boldsymbol{E}^T\boldsymbol{E}_{mf}^T\boldsymbol{B}_{i,m}\left(\boldsymbol{f}_i^* - \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_m)\right) = 0 \quad (25)$$

$$\widetilde{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u}_h \\ \widetilde{\boldsymbol{u}}_f \\ \widetilde{\boldsymbol{u}}_m \end{bmatrix}, \qquad \widetilde{\boldsymbol{u}}_m = \boldsymbol{u}\left(\boldsymbol{E}_{mf}\boldsymbol{E}\boldsymbol{v}(\boldsymbol{u}_h)\right), \qquad \boldsymbol{f}_i^* = \boldsymbol{f}_{i,S}\left(\widetilde{\boldsymbol{u}}_m^+, \widetilde{\boldsymbol{u}}_m\right),$$

Again, we assume geometric terms on curved elements are approximated using polynomials, and that the normals are constructed via (10) and polynomial interpolation.

Theorem 1 *Assuming continuity in time, the local formulation (25) satisfies*

$$\mathbf{1}^T\boldsymbol{M}\frac{\mathrm{d}S(\boldsymbol{u}_h)}{\mathrm{dt}} + \sum_{i=1}^{d}\mathbf{1}^T\boldsymbol{B}_{i,m}\left(\boldsymbol{v}_m^T\boldsymbol{f}_i^* - \psi_i(\widetilde{\boldsymbol{u}}_m)\right) = 0, \qquad \boldsymbol{v}_m = \boldsymbol{E}_{mf}\boldsymbol{E}\boldsymbol{v}(\boldsymbol{u}_h). \quad (26)$$

Proof The main steps are algebraically very similar to proofs in [1, 8, 9], with Lemma 6 used in place of Lemma 2. We begin by testing (13) with the vector of entropy variables evaluated at nodal points $\boldsymbol{v}(\boldsymbol{u}_h)$. We apply the chain rule in time and use that $\boldsymbol{M}$ is diagonal to show that

$$\boldsymbol{v}(\boldsymbol{u}_h)^T\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{dt}} = \mathbf{1}^T\boldsymbol{M}\left(\boldsymbol{v}(\boldsymbol{u}_h) \circ \frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{dt}}\right) = \mathbf{1}^T\boldsymbol{M}\frac{\mathrm{d}S(\boldsymbol{u}_h)}{\mathrm{dt}}.$$

For the volume term, we use the SBP property (Lemma 6) to arrive at

$$\boldsymbol{v}(\boldsymbol{u}_h)^T \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{mf}\boldsymbol{E} \end{bmatrix}^T (2\boldsymbol{Q}_{i,m} \circ \boldsymbol{F}_i)\,\mathbf{1} =$$

$$\widetilde{\boldsymbol{v}}^T\left(\left(\boldsymbol{Q}_{i,m} - \boldsymbol{Q}_{i,m}^T\right) \circ \boldsymbol{F}_i\right)\mathbf{1} + (\boldsymbol{E}\boldsymbol{v}(\boldsymbol{u}_h))^T\left(\begin{bmatrix} \mathbf{0} \\ & \mathbf{0} \\ & & \boldsymbol{B}_{i,m} \end{bmatrix} \circ \boldsymbol{F}_i\right)\mathbf{1},$$

where we have introduced

$$\widetilde{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{mf}\boldsymbol{E} \end{bmatrix}\boldsymbol{v}(\boldsymbol{u}_h).$$

Note that, by the consistency of the entropy conservative flux $\boldsymbol{f}_{i,S}$, the diagonal of $\boldsymbol{F}_i$ is the flux evaluated at volume, surface, and mortar points. Thus, the latter term reduces to

$$(\boldsymbol{E}\boldsymbol{v}(\boldsymbol{u}_h))^T\left(\begin{bmatrix} \mathbf{0} \\ & \mathbf{0} \\ & & \boldsymbol{B}_{i,m} \end{bmatrix} \circ \boldsymbol{F}_i\right)\mathbf{1} = \boldsymbol{v}_m^T\boldsymbol{B}_{i,m}\boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_m) = \mathbf{1}^T\boldsymbol{B}_{i,m}\left(\boldsymbol{v}_m^T\boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_m)\right).$$

where we have used that $\boldsymbol{B}_{i,m}$ is diagonal. The final part of the proof uses that

$$\widetilde{\boldsymbol{v}}^T \left(\left(\boldsymbol{Q}_{i,m} - \boldsymbol{Q}_{i,m}^T \right) \circ \boldsymbol{F}_i \right) \mathbf{1} = \sum_j \widetilde{\boldsymbol{v}}_j \left(\boldsymbol{Q}_{i,m} - \boldsymbol{Q}_{i,m}^T \right)_{jk} (\boldsymbol{F}_i)_{jk}$$

$$= \sum_j (\boldsymbol{Q}_{i,m})_{jk} \left(\widetilde{\boldsymbol{v}}_j - \widetilde{\boldsymbol{v}}_k \right)^T \boldsymbol{f}_{i,S} \left(\widetilde{\boldsymbol{u}}_j, \widetilde{\boldsymbol{u}}_k \right)$$

$$= \sum_j (\boldsymbol{Q}_{i,m})_{jk} \left(\psi(\widetilde{\boldsymbol{u}}_j) - \psi(\widetilde{\boldsymbol{u}}_k) \right) = \widetilde{\boldsymbol{u}}^T \boldsymbol{Q}_{i,m} \mathbf{1} - \mathbf{1}^T \boldsymbol{Q}_{i,m} \widetilde{\boldsymbol{u}}$$

By Lemma 6, $\boldsymbol{Q}_{i,m}\mathbf{1} = \mathbf{0}$. The proof is completed by applying the SBP property (4) to the remaining term, which reduces to $-\mathbf{1}^T \boldsymbol{Q}_{i,m} \widetilde{\boldsymbol{u}} = -\mathbf{1}^T \boldsymbol{B}_{i,m} \widetilde{\boldsymbol{u}}_m$. □

Global entropy conservation can be derived by summing up (26) over each element

$$\sum_k \mathbf{1}^T \boldsymbol{M}^k \frac{\mathrm{d}S(\boldsymbol{u}_h^k)}{\mathrm{dt}} + \sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\left(\boldsymbol{v}_m^k \right)^T \boldsymbol{f}_i^{k,*} - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right) = 0,$$

where have introduced the superscript k to denote operators and local solutions on the kth element D^k. The first term corresponds to the average entropy over the entire domain. Assume without loss of generality a periodic domain, such that all interfaces are interior interfaces. Sharing interface contributions involving the numerical flux between neighboring elements yields

$$\sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\left(\boldsymbol{v}_m^k \right)^T \boldsymbol{f}_i^{k,*} - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right) = \quad (27)$$

$$\sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\frac{1}{2} \left(\boldsymbol{v}_m^k - \boldsymbol{v}_m^{k,+} \right)^T \boldsymbol{f}_i^{k,*} - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right)$$

where $\boldsymbol{v}_m^{k,+}$ denotes the exterior value of the entropy variables on the neighboring element. If the numerical flux is computed using the entropy conservative flux $\boldsymbol{f}_i^{k,*} = \boldsymbol{f}_{i,S} \left(\widetilde{\boldsymbol{u}}_m^k, \widetilde{\boldsymbol{u}}_m^{k,+} \right)$, then

$$\left(\boldsymbol{v}_m^k - \boldsymbol{v}_m^{k,+} \right)^T \boldsymbol{f}_{i,S} \left(\widetilde{\boldsymbol{u}}_m^k, \widetilde{\boldsymbol{u}}_m^{k,+} \right) = \psi_i(\widetilde{\boldsymbol{u}}_m^k) - \psi_i(\widetilde{\boldsymbol{u}}_m^{k,+}).$$

Returning the contributions $\psi_i(\widetilde{\boldsymbol{u}}_m^{k,+})$ to the neighboring element simplifies (27) as follows

$$\sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\frac{1}{2} \left(\boldsymbol{v}_m^k - \boldsymbol{v}_m^{k,+} \right)^T \boldsymbol{f}_i^{k,*} - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right)$$

$$= \sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\frac{1}{2} \left(\psi_i(\widetilde{\boldsymbol{u}}_m^k) - \psi_i(\widetilde{\boldsymbol{u}}_m^{k,+}) - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right) \right)$$

$$= \sum_k \sum_{i=1}^d \mathbf{1}^T \boldsymbol{B}_{i,m}^k \left(\psi_i(\widetilde{\boldsymbol{u}}_m^k) - \psi_i(\widetilde{\boldsymbol{u}}_m^k) \right) = 0.$$

When the domain is non-periodic, the interface contributions do not vanish at the boundary. These boundary terms can be made non-negative (e.g., entropy conservative or entropy stable) if boundary conditions are imposed in an appropriate manner [6, 23].

5.3 Comparison of Lobatto and Gauss nodes: requirements for entropy conservation (stability) and accuracy

This section summarizes theoretical differences between entropy conservative (stable) Lobatto and Gauss collocation schemes on non-conforming meshes based on Lemma 6 and Lemma 4. The main differences are that the use of Lobatto nodes induces additional requirements on the construction of geometric terms for curved meshes and lowers the expected order of accuracy by one degree.

High order entropy stable collocation schemes on tensor product elements typically utilize tensor product quadrature rules based on $(N+1)$-point Lobatto nodes [4, 5] or $(N+1)$-point Gauss nodes [1]. On non-conforming meshes, the most natural choice of mortar nodes is a composite Lobatto or Gauss rule. Recall that an $(N+1)$-point Lobatto rule is exact for polynomials of degree $(2N-1)$, while an $(N+1)$-point Gauss rule is exact for polynomials of degree $(2N+1)$.

First, we discuss the expected accuracy of each scheme. We have not conducted a rigorous error analysis; however, Lemma 4 implies that a mortar-based SBP operator under Lobatto quadrature is exact for degree $(N-1)$ polynomials, while a mortar-based SBP operator under Gauss quadrature is exact for degree N polynomials. In other words, the order of accuracy of a Gauss collocation scheme can be up to one degree higher than for a Lobatto collocation scheme.

We note that order reduction for Lobatto nodes is not consistently evident in numerical experiments, both for conforming [1, 24] and non-conforming meshes [25]. In [11], order reduction with respect to the L^2 error was observed using Lobatto nodes on 2D non-conforming meshes. However, in [25], L^2 order reduction is not observed using Lobatto nodes on 3D non-conforming hexahedral meshes, though order reduction is observed for the L^∞ error. Similar inconsistent behavior is observed for the 2D and 3D experiments in this paper when using Lobatto nodes. We do note observe order reduction for Gauss nodes in any of our experiments.

Next, we discuss requirements on the construction of geometric terms. On non-conforming quadrilateral meshes, Theorem 6 implies that both Lobatto and Gauss collocation schemes are stable for $N_{\mathrm{geo}} \leq N$, i.e., for isoparametric curved mappings. On non-conforming hexahedral meshes, Theorem 6 implies that Gauss nodes are also stable for $N_{\mathrm{geo}} \leq N$, but that Lobatto nodes are stable only for $N_{\mathrm{geo}} \leq (N-1)$ if Approach 1 [16] is used to compute geometric terms.

The restriction on $N_{\mathrm{geo}} \leq (N-1)$ for Lobatto nodes is not necessary for entropy conservation (stability) on conforming hexahedral meshes. However, when using Approach 1 to compute geometric terms, the introduction of non-aligned mortar nodes at non-conforming curved interfaces induces this additional restriction. Moreover, this condition is consistent with Theorem 3 in [25]. One can remove this constraint on the degree of the geometric mapping by using Approach 2 [17] to compute geometric terms. If Approach 2 is used, then Lobatto collocation schemes remain entropy conservative (stable) on curved hexahedral meshes for all $N_{\mathrm{geo}} \leq N$. Appendix A describes Approach 1 [16] and Approach 2 [17] in more

detail, and demonstrates numerically that both approaches achieve similar high order accuracy in approximating geometric terms.

6 A mortar-based implementation

While the formulation (25) is convenient for analysis, it is less convenient to implement. However, using properties of discretization matrices, we can show that the mortar-based formulation (25) is equivalent to a conforming formulation (13) with a modified numerical flux involving face-local correction terms.

First, we reformulate volume terms by incorporating geometric terms into the definition of the flux matrices. Note that $\boldsymbol{Q}_{i,m}$ can alternatively be defined as [5]

$$\boldsymbol{Q}_{i,m} = \sum_{j=1}^{d} \widehat{\boldsymbol{Q}}_{j,m} \circ \left\{\!\left\{\boldsymbol{G}_{ij}\right\}\!\right\}, \qquad \left\{\!\left\{\boldsymbol{G}_{ij}\right\}\!\right\}_{kl} = \frac{1}{2}\left((\boldsymbol{G}_{ij})_k + (\boldsymbol{G}_{ij})_l\right),$$

where k, l are indices over the total number of volume, surface, and mortar points, and $\boldsymbol{G}_{ij}$ is the vector containing values of G_{ij} at volume, surface, and mortar points. The volume term of (25) can then be rewritten as

$$\begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_m \boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^{d} \left(2\boldsymbol{Q}_{i,m} \circ \boldsymbol{F}_i\right)\mathbf{1} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_m \boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^{d} \left(2\widehat{\boldsymbol{Q}}_{i,m} \circ \widehat{\boldsymbol{F}}_i\right)\mathbf{1} \qquad (28)$$

$$\widehat{\boldsymbol{F}}_i = \sum_{j=1}^{d} \left\{\!\left\{\boldsymbol{G}_{ij}\right\}\!\right\} \circ \boldsymbol{F}_i.$$

We next decompose $\widehat{\boldsymbol{F}}_i$ into interactions between volume nodes, surface nodes, and mortar nodes

$$\widehat{\boldsymbol{F}}_i = \begin{bmatrix} \widehat{\boldsymbol{F}}_i^{vv} & \widehat{\boldsymbol{F}}_i^{vf} & \widehat{\boldsymbol{F}}_i^{vm} \\ \widehat{\boldsymbol{F}}_i^{fv} & \widehat{\boldsymbol{F}}_i^{ff} & \widehat{\boldsymbol{F}}_i^{fm} \\ \widehat{\boldsymbol{F}}_i^{mv} & \widehat{\boldsymbol{F}}_i^{mf} & \widehat{\boldsymbol{F}}_i^{mm} \end{bmatrix}$$

By expanding out the definition of the mortar-based SBP operator $\widehat{\boldsymbol{Q}}_{i,m}$ and using properties of the matrices $\widehat{\boldsymbol{B}}_{i,m}, \widehat{\boldsymbol{F}}_i$, (28) can be rewritten as

$$\begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{mf}\boldsymbol{E} \end{bmatrix}^T \left(\begin{bmatrix} \widehat{\boldsymbol{Q}}_i - \widehat{\boldsymbol{Q}}_i^T & \boldsymbol{E}^T\widehat{\boldsymbol{B}}_i & \\ -\widehat{\boldsymbol{B}}_i\boldsymbol{E} & & \widehat{\boldsymbol{B}}_i\boldsymbol{E}_{fm} \\ & -\widehat{\boldsymbol{B}}_{i,m}\boldsymbol{E}_{mf} & \end{bmatrix} \circ \widehat{\boldsymbol{F}}_i\right)\mathbf{1} + \boldsymbol{E}^T\boldsymbol{E}_{mf}^T\left(\widehat{\boldsymbol{B}}_{i,m}\circ\widehat{\boldsymbol{F}}_i^{mm}\right)\mathbf{1}$$

$$= \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \end{bmatrix}^T \left(\begin{bmatrix} \widehat{\boldsymbol{Q}}_i - \widehat{\boldsymbol{Q}}_i^T & \boldsymbol{E}^T\widehat{\boldsymbol{B}}_i \\ -\widehat{\boldsymbol{B}}_i\boldsymbol{E} & \widehat{\boldsymbol{B}}_i \end{bmatrix} \circ \widehat{\boldsymbol{F}}_i\right)\mathbf{1}$$

$$+ \boldsymbol{E}^T\left(\boldsymbol{E}_{mf}^T\left(\widehat{\boldsymbol{B}}_{i,m}\circ\widehat{\boldsymbol{F}}_i^{mm}\right)\mathbf{1} - \left(\boldsymbol{B}_i\circ\widehat{\boldsymbol{F}}_i^{ff}\right)\mathbf{1}\right)$$

$$+ \boldsymbol{E}^T\left(\left(\widehat{\boldsymbol{B}}_i\boldsymbol{E}_{fm}\right)\circ\widehat{\boldsymbol{F}}_i^{fm}\right)\mathbf{1} - \boldsymbol{E}^T\boldsymbol{E}_{mf}^T\left(\left(\widehat{\boldsymbol{B}}_i\boldsymbol{E}_{mf}\right)\circ\widehat{\boldsymbol{F}}_i^{mf}\right)\mathbf{1}.$$

The above expressions can be simplified. First, evaluating $\widehat{\boldsymbol{F}}_i^{mm}, \widehat{\boldsymbol{F}}_i^{ff}$ yields

$$\widehat{\boldsymbol{F}}_i^{mm} = \mathrm{diag}\left(\sum_{j=1}^{d} G_{ij}(\widehat{\boldsymbol{x}}_m)\circ\boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_m)\right), \quad \widehat{\boldsymbol{F}}_i^{ff} = \mathrm{diag}\left(\sum_{j=1}^{d} G_{ij}(\widehat{\boldsymbol{x}}_f)\circ\boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_f)\right),$$
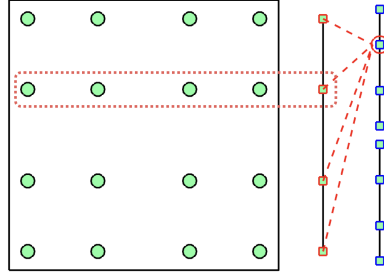
Fig. 3: Coupling between volume, surface, and mortar nodes for formulation (29).

Moreover, recall that the physical normals are defined as products of G_{ij} with reference normals (10). Since $\widehat{\boldsymbol{B}}_{i,m}, \widehat{\boldsymbol{B}}_i$ contain scaled normals on the reference element on the diagonal, we can rewrite the correction terms using the physical normal boundary matrices $\boldsymbol{B}_{i,m}, \boldsymbol{B}_i$

$$\boldsymbol{E}^T \left(\boldsymbol{E}_{mf}^T \left(\widehat{\boldsymbol{B}}_{i,m} \circ \widehat{\boldsymbol{F}}_i^{mm} \right) \boldsymbol{1} - \left(\boldsymbol{B}_i \circ \widehat{\boldsymbol{F}}_i^{ff} \right) \boldsymbol{1} \right)$$
$$= \boldsymbol{E}^T \left(\boldsymbol{E}_{mf}^T \boldsymbol{B}_{i,m} \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_m) - \boldsymbol{B}_i \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_f) \right).$$

The remaining correction terms can also be simplified by noting that $\boldsymbol{E}_{fm}$ are block diagonal matrices, with each block corresponding to a face, and that the scaled reference normals (i.e., the diagonal entries of $\widehat{\boldsymbol{B}}_{i,m}, \widehat{\boldsymbol{B}}_i$) are constant over each face. Then, using (10), we can rewrite the final set of correction terms as

$$\boldsymbol{E}^T \left(\left(\widehat{\boldsymbol{B}}_i \boldsymbol{E}_{fm} \right) \circ \widehat{\boldsymbol{F}}_i^{fm} \right) \boldsymbol{1} - \boldsymbol{E}^T \boldsymbol{E}_{mf}^T \left(\left(\widehat{\boldsymbol{B}}_{i,m} \boldsymbol{E}_{mf} \right) \circ \widehat{\boldsymbol{F}}_i^{mf} \right) \boldsymbol{1}$$
$$= \boldsymbol{E}^T \left(\boldsymbol{E}_{fm} \circ \boldsymbol{F}_i^{fm} \circ \{\!\{\boldsymbol{n}_i\}\!\}_{fm} \right) \boldsymbol{1} - \boldsymbol{E}^T \boldsymbol{E}_{mf}^T \left(\boldsymbol{E}_{mf} \circ \boldsymbol{F}_i^{mf} \circ \{\!\{\boldsymbol{n}_i\}\!\}_{mf} \right) \boldsymbol{1}$$

where $\boldsymbol{F}_i^{fm} = \left(\boldsymbol{F}_i^{mf} \right)^T$ are the flux matrices whose entries are evaluations of $\boldsymbol{f}_{i,S}$ between solution values at mortar and surface nodes, and $\{\!\{\boldsymbol{n}_i\}\!\}_{fm} = \{\!\{\boldsymbol{n}_i\}\!\}_{mf}^T$ are matrices whose entries are arithmetic averages of $n_i J_f$ (the ith component of the scaled normal vector) between each mortar and surface point.

Adding these correction terms back into the formulation (25) yields

$$\boldsymbol{M} \frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{d}t} + \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^d \left(2\boldsymbol{Q}_{i,h} \circ \boldsymbol{F}_i \right) \boldsymbol{1} + \boldsymbol{E}^T \left(\boldsymbol{f}_i^* - \boldsymbol{B}_i \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_f) \right) = 0 \qquad (29)$$
$$\boldsymbol{f}_i^* = \boldsymbol{E}_{mf}^T \boldsymbol{B}_{i,m} \boldsymbol{f}_{i,m}^* + \boldsymbol{\delta f},$$
$$\boldsymbol{\delta f} = \left(\boldsymbol{E}_{fm} \circ \boldsymbol{F}_i^{fm} \circ \{\!\{\boldsymbol{n}_i\}\!\}_{fm} \right) \boldsymbol{1} - \boldsymbol{E}_{mf}^T \left(\boldsymbol{E}_{mf} \circ \boldsymbol{F}_i^{mf} \circ \{\!\{\boldsymbol{n}_i\}\!\}_{mf} \right) \boldsymbol{1}$$

where $\boldsymbol{f}_{i,m} = \boldsymbol{f}_{i,S}(\widetilde{\boldsymbol{u}}_m^+, \widetilde{\boldsymbol{u}}_m)$ is the numerical flux computed at mortar points.

The formulation (29) is nearly equivalent to the hybridized mortar formulation, with the main difference being the presence of skew-symmetric correction terms in the definition of the numerical flux. Note that in this formulation, the only interactions between surface and mortar nodes occur within the computation of $\boldsymbol{f}_i^*$, and involve only computations which are local to mortar interfaces.

Remark 1 On conforming meshes, the surface and mortar nodes are the same. Then,

$$\boldsymbol{E}_{mf} = \boldsymbol{E}_{fm} = \boldsymbol{I}, \qquad \{\{\boldsymbol{n}_i\}\}_{fm} = \{\{\boldsymbol{n}_i\}\}_{mf}$$

and the correction terms in the definition of $\boldsymbol{f}_i^*$ cancel each other out. The formulation (29) then reverts to a Gauss collocation scheme [2] on conforming meshes.

6.1 Improving efficiency on non-conforming hexahedral meshes

In 3D, the mortar-based approach enables a more efficient treatment of non-conforming hexahedral interfaces compared to existing approaches [11]. The cost of entropy stable schemes based on flux differencing usually scales with the number of flux evaluations, which in turn scales by the number of non-zero entries present in differentiation and interpolation matrices. For a hexahedral element, both Gauss and Lobatto node differentiation operators contain $O(N^4)$ non-zeros due to their tensor product (Kronecker) structure.
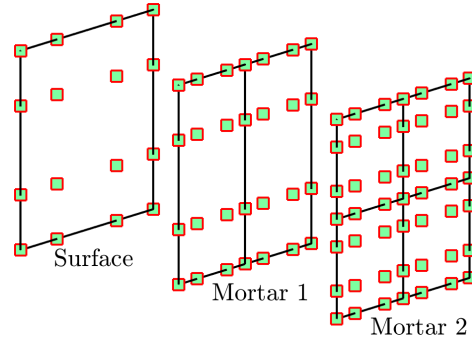
Fig. 4: Two-layer mortars for reduced $O(N^3)$ computational cost on non-conforming hexahedral interfaces.

The interpolation matrices are generally dense with $O(N^4)$ non-zero entries, as they interpolate from $(N+1)^2$ surface points to $4(N+1)^2$ non-aligned mortar points (for an isotropically refined face). However, we can expose sparsity by using a two-step interpolation from surface to mortar points. Figure 4 illustrates this process. First, the $(N+1)^2$ surface points are interpolated to an anisotropically refined interface. Because the points are aligned along one coordinate direction, this requires interpolation only along lines of nodes and results in an interpolation matrix with $O(N^3)$ non-zero entries. The anisotropically refined nodes are then interpolated to the isotropically refined nodes. Again, because nodes are aligned, the resulting interpolation matrix has $O(N^3)$ non-zero entries.

This two-step procedure can be interpreted as a sparse factorization of the interpolation matrix from $(N+1)^2$ surface points to $4(N+1)^2$ points. Let $\boldsymbol{x}_f$ denote the surface points, and let $\boldsymbol{x}_{m_1}, \boldsymbol{x}_{m_2}$ denote the anisotropically and isotropically refined mortar nodes (e.g., the first and second mortar layers), respectively. We first introduce $\boldsymbol{E}_{1D}$ be the 1D polynomial interpolation matrix which maps from
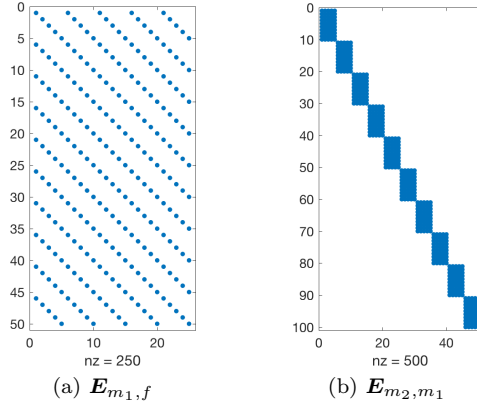
(a) $\boldsymbol{E}_{m_1,f}$

(b) $\boldsymbol{E}_{m_2,m_1}$

Fig. 5: Sparsity patterns of the interpolation operators $\boldsymbol{E}_{m_1,f}, \boldsymbol{E}_{m_2,f}$.

Gauss or Lobatto points to split (composite) Gauss or Lobatto points. Let $\boldsymbol{E}_{m_1,f}$ denote the interpolation operator from surface to nodes in the first mortar layer, and let $\boldsymbol{E}_{m_2,m_1}$ denote the interpolation operator from nodes in the first mortar layer to the second mortar layer. These operators are defined

$$\boldsymbol{E}_{m_1,f} = \boldsymbol{E}_{1\mathrm{D}} \otimes \boldsymbol{I}_{N+1}, \qquad \boldsymbol{E}_{m_2,m_1} = \boldsymbol{I}_{2(N+1)} \otimes \boldsymbol{E}_{1\mathrm{D}}.$$

where $\boldsymbol{I}_{N+1}, \boldsymbol{I}_{2(N+1)}$ denote the $(N+1) \times (N+1)$ and $2(N+1) \times 2(N+1)$ identity matrices, respectively. Then, $\boldsymbol{E}_{m_2,f} = \boldsymbol{E}_{m_2,m_1} \boldsymbol{E}_{m_1,f}$ interpolates from surface nodes to the second layer of mortar nodes. However, the operators $\boldsymbol{E}_{m_2,m_1}, \boldsymbol{E}_{m_1,f}$ are sparse (as shown in Figure 5), while $\boldsymbol{E}_{m_2,f}$ is completely dense.

We can also define projection operators which transfer from the second to the first mortar layer and from the first mortar layer to surface nodes. Let $\boldsymbol{M}_{m_i}$ denote the diagonal matrix containing quadrature weights for the ith layer of mortar nodes, respectively. Then, we can define $\boldsymbol{E}_{f,m_1}, \boldsymbol{E}_{m_1,m_2}$ via

$$\boldsymbol{E}_{f,m_1} = \boldsymbol{M}_f^{-1} \boldsymbol{E}_{m_1,f}^T \boldsymbol{M}_{m_1}$$
$$\boldsymbol{E}_{m_1,m_2} = \boldsymbol{M}_{m_1}^{-1} \boldsymbol{E}_{m_2,m_1}^T \boldsymbol{M}_{m_2}.$$

The definitions of $\boldsymbol{E}_{f,m_1}, \boldsymbol{E}_{m_1,m_2}$ are analogous to the definition of $\boldsymbol{E}_{fm}$ in (15). The operator $\boldsymbol{E}_{f,m_1}$ can be interpreted as projecting onto degree N polynomials on a quadrilateral face, while $\boldsymbol{E}_{m_1,m_2}$ can be interpreted as projecting onto the space of piecewise degree N polynomials on the two quadrilateral faces of the first mortar layer in Figure 4.

Putting these pieces together allows us to define two-layer mortar-based SBP operators. We introduce boundary matrices

$$\widehat{\boldsymbol{B}}_{j,m_i} = \mathrm{diag}\left(\widehat{\boldsymbol{n}}_{i,m_i}\right) \boldsymbol{M}_{m_i}, \qquad i = 1, 2, \qquad j = 1, \ldots, d.$$

where $\widehat{\boldsymbol{n}}_{i,m_i}$ denote scaled outward normals evaluated at the ith layer of mortar points. Then, the two-layer mortar-based SBP operator is

$$\widehat{\boldsymbol{Q}}_{i,m} = \frac{1}{2}\begin{bmatrix} \widehat{\boldsymbol{Q}}_i - \widehat{\boldsymbol{Q}}_i^T & \boldsymbol{E}^T\widehat{\boldsymbol{B}}_{i,f} & & \\ -\widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E} & & \widehat{\boldsymbol{B}}_{i,f}\boldsymbol{E}_{f,m_1} & \\ & -\widehat{\boldsymbol{B}}_{i,m_1}\boldsymbol{E}_{m_1,f} & & \widehat{\boldsymbol{B}}_{i,m_1}\boldsymbol{E}_{m_1,m_2} \\ & & -\widehat{\boldsymbol{B}}_{i,m_2}\boldsymbol{E}_{m_2,m_1} & \widehat{\boldsymbol{B}}_{i,m_2} \end{bmatrix}. \tag{30}$$

It is straightforward to show that, for both Gauss and Lobatto nodes, $\widehat{\boldsymbol{Q}}_{i,m}$ satisfies SBP and conservation properties

$$\widehat{\boldsymbol{Q}}_{i,m} + \widehat{\boldsymbol{Q}}_{i,m}^T = \begin{bmatrix} \boldsymbol{0} & & & \\ & \boldsymbol{0} & & \\ & & \boldsymbol{0} & \\ & & & \boldsymbol{B}_{i,m_2} \end{bmatrix}, \qquad \widehat{\boldsymbol{Q}}_{i,m}\mathbf{1} = \boldsymbol{0}.$$

The accuracy conditions for two-layer mortar operators are also identical to those of Lemma 4. We can now define physical SBP operators via the formula (22). These also satisfy SBP and conservation properties under the conditions of Lemma 6.

Using the operators introduced so far, an entropy conservative formulation using two-layer mortar operators is given by

$$\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{d}t} + \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \\ \boldsymbol{E}_{m_1,f}\boldsymbol{E} \\ \boldsymbol{E}_{m_2,m_1}\boldsymbol{E}_{m_1,f}\boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^d \left(2\boldsymbol{Q}_{i,m} \circ \boldsymbol{F}_i\right)\mathbf{1} \tag{31}$$

$$+ \boldsymbol{E}^T\boldsymbol{E}_{m_1,f}^T\boldsymbol{E}_{m_2,m_1}^T\boldsymbol{B}_{i,m_2}\left(\boldsymbol{f}_i^* - \boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_{m_2})\right) = 0. \tag{32}$$

These formulations look very similar to (25). The two-layer mortar formulation can also be reformulated in a way which involves only face-local correction terms to the numerical flux:

$$\boldsymbol{M}\frac{\mathrm{d}\boldsymbol{u}_h}{\mathrm{d}t} + \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E} \end{bmatrix}^T \sum_{i=1}^d \left(2\boldsymbol{Q}_{i,h} \circ \boldsymbol{F}_i\right)\mathbf{1} + \boldsymbol{E}^T\left(\boldsymbol{f}_i^* - \boldsymbol{B}_i\boldsymbol{f}_i(\widetilde{\boldsymbol{u}}_f)\right) = 0 \tag{33}$$

$$\boldsymbol{f}_i^* = \boldsymbol{E}_{m_1,f}^T\boldsymbol{E}_{m_2,m_1}^T\boldsymbol{B}_{i,m_2}\boldsymbol{f}_{i,m_2} + \boldsymbol{\delta f},$$

$$\boldsymbol{\delta f} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{E}_{m_1,f} \\ \boldsymbol{E}_{m_2,m_1}\boldsymbol{E}_{m_1,f} \end{bmatrix}^T \left(\begin{bmatrix} & \boldsymbol{E}_{f,m_1} & \\ -\boldsymbol{E}_{m_1,f} & & \boldsymbol{E}_{m_1,m_2} \\ & -\boldsymbol{E}_{m_2,m_1} & \end{bmatrix} \circ \boldsymbol{F}_i^{f,m_1,m_2} \circ \{\!\{\boldsymbol{n}_i\}\!\}\right)\mathbf{1}$$

where $\boldsymbol{\delta f}$ is a face-local correction term for the numerical flux, and the matrix $\boldsymbol{F}_i^{f,m_1,m_2}$ is the matrix of entropy conservative flux evaluations between solution values at surface nodes and nodes in the two mortar layers. The term $\{\!\{\boldsymbol{n}_i\}\!\}$ denote the averaged scaled normals between each surface point and each point in the two mortar layers. The derivation of this face-local formulation is similar to the derivation in Section 6, and we omit it for brevity.

7 Numerical experiments

This section presents numerical examples for the compressible Euler equations, which are given in d dimensions as follows:

$$\frac{\partial \rho}{\partial t} + \sum_{j=1}^{d} \frac{\partial (\rho u_j)}{\partial x_j} = 0,$$

$$\frac{\partial \rho u_i}{\partial t} + \sum_{j=1}^{d} \frac{\partial (\rho u_i u_j + p \delta_{ij})}{\partial x_j} = 0, \qquad i = 1, \dots, d$$

$$\frac{\partial E}{\partial t} + \sum_{j=1}^{d} \frac{\partial (u_j (E + p))}{\partial x_j} = 0.$$

Here, ρ is density, u_i denotes the ith component of the velocity, and E is the total energy. The pressure p and specific internal energy ρe are given by

$$p = (\gamma - 1) \left(E - \frac{1}{2} \rho \|\boldsymbol{u}\|^2 \right), \qquad \rho e = E - \frac{1}{2} \rho \|\boldsymbol{u}\|^2, \qquad \|\boldsymbol{u}\|^2 = \sum_{j=1}^{d} u_j^2.$$

We enforce the entropy conservation (stability) with respect to the unique entropy $S(\boldsymbol{u})$ for the viscous compressible Navier-Stokes equations [26] given by

$$S(\boldsymbol{u}) = -\frac{\rho s}{\gamma - 1},$$

where $s = \log \left(\frac{p}{\rho^\gamma} \right)$ is the specific entropy, and $d = 2, 3$ is the dimension. The entropy variables with respect to $S(\boldsymbol{u})$ are also derived in [26]

$$v_1 = \frac{\rho e (\gamma + 1 - s) - E}{\rho e}, \qquad v_{1+i} = \frac{\rho u_i}{\rho e}, \qquad v_{d+2} = -\frac{\rho}{\rho e}$$

for $i = 1, \dots, d$. The inverse mapping is given by

$$\rho = -(\rho e) v_{d+2}, \qquad \rho u_i = (\rho e) v_{1+i}, \qquad E = (\rho e) \left(1 - \frac{\sum_{j=1}^{d} v_{1+j}^2}{2 v_{d+2}} \right),$$

where $i = 1, \dots, d$, and ρe and s in terms of the entropy variables are

$$\rho e = \left(\frac{(\gamma - 1)}{(-v_{d+2})^\gamma} \right)^{1/(\gamma - 1)} e^{\frac{-s}{\gamma - 1}}, \qquad s = \gamma - v_1 + \frac{\sum_{j=1}^{d} v_{1+j}^2}{2 v_{d+2}}.$$

Entropy conservative fluxes involve both standard and logarithmic averages. Let f denote some piecewise continuous function, and let f^+ denote the exterior value of f across an element face. The average and logarithmic averages are

$$\{\{f\}\} = \frac{f^+ + f}{2}, \qquad \{\{f\}\}^{\log} = \frac{f^+ - f}{\log (f^+) - \log (f)}.$$

The average and logarithmic average are assumed to act component-wise on vector-valued functions. We evaluate the logarithmic average using the numerically stable

algorithm of [27]. Explicit expressions for entropy conservative numerical fluxes in two dimensions are given by Chandrashekar [28]

$$
\begin{aligned}
&f_{1,S}^1(\boldsymbol{u}_L, \boldsymbol{u}_R) = \{\{\rho\}\}^{\log}\{\{u_1\}\}, && f_{2,S}^1(\boldsymbol{u}_L, \boldsymbol{u}_R) = \{\{\rho\}\}^{\log}\{\{u_2\}\}, \\
&f_{1,S}^2(\boldsymbol{u}_L, \boldsymbol{u}_R) = f_{1,S}^1\{\{u_1\}\} + p_{\text{avg}}, && f_{2,S}^2(\boldsymbol{u}_L, \boldsymbol{u}_R) = f_{2,S}^1\{\{u_1\}\}, \\
&f_{1,S}^3(\boldsymbol{u}_L, \boldsymbol{u}_R) = f_{2,S}^2, && f_{2,S}^3(\boldsymbol{u}_L, \boldsymbol{u}_R) = f_{2,S}^1\{\{u_2\}\} + p_{\text{avg}}, \\
&f_{1,S}^4(\boldsymbol{u}_L, \boldsymbol{u}_R) = (E_{\text{avg}} + p_{\text{avg}})\{\{u_1\}\}, && f_{2,S}^4(\boldsymbol{u}_L, \boldsymbol{u}_R) = (E_{\text{avg}} + p_{\text{avg}})\{\{u_2\}\},
\end{aligned}
$$

where we have introduced the auxiliary quantities $\beta = \frac{\rho}{2p}$ and

$$
p_{\text{avg}} = \frac{\{\{\rho\}\}}{2\{\{\beta\}\}}, \qquad E_{\text{avg}} = \frac{\{\{\rho\}\}^{\log}}{2\{\{\beta\}\}^{\log}(\gamma - 1)} + \frac{u_{\text{avg}}^2}{2}, \qquad u_{\text{avg}}^2 = \sum_{j=1}^d u_j u_j^+. \quad (34)
$$

where u_j, u_j^+ denote the trace and exterior values of u_j across a face. Expressions for entropy conservative numerical fluxes for the three-dimensional compressible Euler equations can also be explicitly written as

$$
\boldsymbol{f}_{1,S} = \begin{pmatrix} \{\{\rho\}\}^{\log}\{\{u_1\}\} \\ \{\{\rho\}\}^{\log}\{\{u_1\}\}^2 + p_{\text{avg}} \\ \{\{\rho\}\}^{\log}\{\{u_1\}\}\{\{u_2\}\} \\ \{\{\rho\}\}^{\log}\{\{u_1\}\}\{\{u_3\}\} \\ (E_{\text{avg}} + p_{\text{avg}})\{\{u_1\}\} \end{pmatrix}, \qquad
\boldsymbol{f}_{2,S} = \begin{pmatrix} \{\{\rho\}\}^{\log}\{\{u_2\}\} \\ \{\{\rho\}\}^{\log}\{\{u_1\}\}\{\{u_2\}\} \\ \{\{\rho\}\}^{\log}\{\{u_2\}\}^2 + p_{\text{avg}} \\ \{\{\rho\}\}^{\log}\{\{u_2\}\}\{\{u_3\}\} \\ (E_{\text{avg}} + p_{\text{avg}})\{\{u_2\}\} \end{pmatrix},
$$

$$
\boldsymbol{f}_{3,S} = \begin{pmatrix} \{\{\rho\}\}^{\log}\{\{u_3\}\} \\ \{\{\rho\}\}^{\log}\{\{u_1\}\}\{\{u_3\}\} \\ \{\{\rho\}\}^{\log}\{\{u_2\}\}\{\{u_3\}\} \\ \{\{\rho\}\}^{\log}\{\{u_3\}\}^2 + p_{\text{avg}} \\ (E_{\text{avg}} + p_{\text{avg}})\{\{u_3\}\} \end{pmatrix}.
$$

For all numerical experiments, we use an explicit low storage RK-45 time-stepper [29] and estimate the timestep dt using J, J_f^k, and degree-dependent L^2 trace constants C_N

$$
dt = C_{\text{CFL}}\frac{h}{aC_N}, \qquad h = \frac{1}{\|J^{-1}\|_{L^\infty}\left\|J_f^k\right\|_{L^\infty}},
$$

where a is an estimate of the maximum wave speed, h estimates the mesh size, and C_{CFL} is some user-defined CFL constant. For isotropic elements, the ratio of J^k to J_f^k scales as the mesh size h, while C_N captures the dependence of the maximum stable timestep on the polynomial degree N. For hexahedral elements, C_N varies depending on the choice of quadrature. It was shown in [30] that

$$
C_N = \begin{cases} d\frac{N(N+1)}{2} & \text{for GLL nodes} \\ d\frac{(N+1)(N+2)}{2} & \text{for Gauss nodes} \end{cases}.
$$

Thus, based on this conservative estimate of the maximum stable timestep, GLL collocation schemes should be able to take a timestep which is roughly $(1 + 2/N)$ times larger than the maximum stable timestep for Gauss collocation schemes. We do not account for this discrepancy in this work, and instead set the timestep for

both GLL and Gauss collocation schemes based on the more conservative Gauss collocation estimate of dt. All results are computed using $C_{\mathrm{CFL}} = 1/2$ and Lax-Friedrichs interface dissipation.

Suppose $\boldsymbol{u}_\Omega$ denotes the global solution vector; then, an entropy conservative DG scheme can be rewritten as

$$\boldsymbol{M}_\Omega \frac{\mathrm{d}\boldsymbol{u}_\Omega}{\mathrm{dt}} = \boldsymbol{F}(\boldsymbol{u}_\Omega).$$

where $\boldsymbol{M}_\Omega$ is the global mass matrix, and $\boldsymbol{F}(\boldsymbol{u}_\Omega)$ denotes the global right hand side. Let $\boldsymbol{v}_\Omega$ denote the global vector of entropy variables. The proof of entropy conservation implies that the "spatial entropy" $\boldsymbol{v}_\Omega^T \boldsymbol{F}(\boldsymbol{u}_\Omega) = 0$ [7]. We verify that both 2D and 3D implementations are entropy conservative (and thus entropy stable if interface dissipation is added) by computing the spatial entropy for a discontinuous initial condition. In the absence of interface dissipation, all schemes yield a spatial entropy of magnitude $O\left(10^{-14}\right)$, which suggests that the implementations are semi-discretely entropy conservative up to machine precision.

7.1 Two-dimensional results

We begin by examining high order convergence of the proposed methods in two dimensions using the isentropic vortex problem [31, 32]. The analytical solution is

$$\rho(\boldsymbol{x}, t) = \left(1 - \frac{\frac{1}{2}(\gamma-1)(\beta e^{1-r(\boldsymbol{x},t)^2})^2}{8\gamma\pi^2}\right)^{\frac{1}{\gamma-1}}, \qquad p = \rho^\gamma, \qquad (35)$$

$$u_1(\boldsymbol{x}, t) = 1 - \frac{\beta}{2\pi}e^{1-r(\boldsymbol{x},t)^2}(y - y_0), \qquad u_2(\boldsymbol{x}, t) = \frac{\beta}{2\pi}e^{1-r(\boldsymbol{x},t)^2}(x - x_0 - t),$$

where u_1, u_2 are the x and y velocity, $r(\boldsymbol{x}, t) = \sqrt{(x - x_0 - t)^2 + (y - y_0)^2} x_0 = 5, y_0 = 0$, and $\beta = 5$. We solve on a periodic rectangular domain $[0, 15] \times [-5, 5]$ until final time $T = 5$ and compute total L^2 errors over all solution fields.

For a degree N scheme, we approximate the L^2 error using an $(N + 2)$ point Gauss quadrature rule. We also examine the influence of element curvature for both GLL and Gauss collocation schemes by examining L^2 errors on affine and curvilinear meshes (see Figure 6). These warpings are constructed by modifying nodal positions according to the following mapping

$$\widetilde{x} = x + L_x \alpha \cos\left(\frac{\pi}{L_x}\left(x - \frac{L_x}{2}\right)\right)\cos\left(\frac{3\pi}{L_y}y\right),$$

$$\widetilde{y} = y + L_y \alpha \sin\left(\frac{4\pi}{L_x}\left(\widetilde{x} - \frac{L_x}{2}\right)\right)\cos\left(\frac{\pi}{L_y}y\right),$$

where $\alpha = 1/16$, and $L_x = 15, L_y = 10$ denote the lengths of the domain in the x and y directions, respectively.

Figure 7 shows results for the 2D compressible Euler equations by the PI on non-conforming quadrilateral meshes constructed by refining in a checkerboard pattern (see Figure 6), with volume, surface, and mortar quadratures constructed from one-dimensional Lobatto or Gauss quadrature rules. As noted in Section 5.3 and Lemma 4, Gauss quadrature produces a degree N differentiation operator,
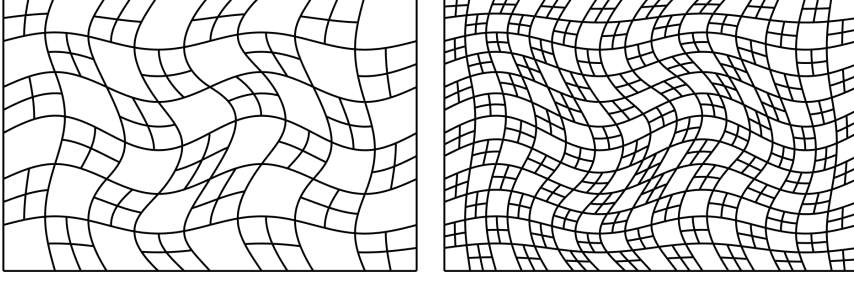
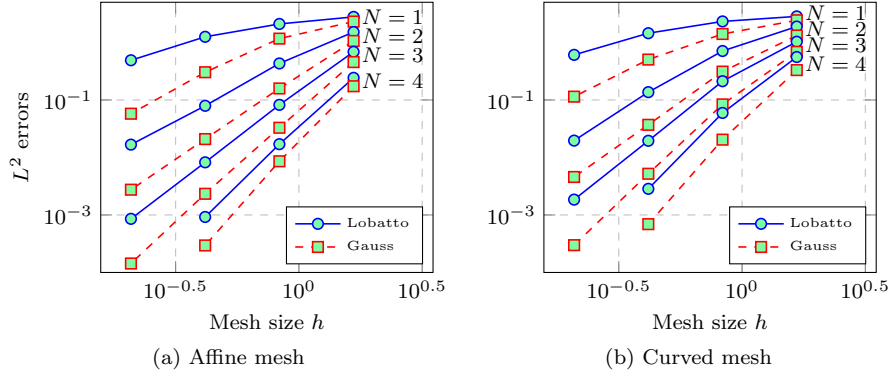Fig. 6: Two curved $N = 3$ meshes used in the grid refinement studies.

(a) Affine mesh

(b) Curved mesh

Fig. 7: L^2 errors for Lobatto and Gauss collocation schemes on affine and curved meshes under h-refinement.

while Lobatto quadrature produces a degree $(N-1)$ differentiation operator. We observe a similar discrepancy in L^2 errors for Lobatto and Gauss quadratures on non-conforming meshes in Figure 7.

7.2 Three-dimensional results

We test the accuracy of the mortar-based scheme in 3D using an extruded 2D vortex propagating in the y direction, with analytic expressions given in [33]

$$\rho(\boldsymbol{x}, t) = \left(1 - \frac{(\gamma - 1)}{2}\Pi^2\right)^{\frac{1}{\gamma - 1}}$$

$$u_i(\boldsymbol{x}, t) = \Pi r_i + \delta_{i2},$$

$$E(\boldsymbol{x}, t) = \frac{p_0}{\gamma - 1}\left(1 - \frac{\gamma - 1}{2}\Pi^2\right)^{\frac{\gamma}{\gamma - 1}} + \frac{\rho}{2}\sum_{i=1}^{d}u_i^2.$$
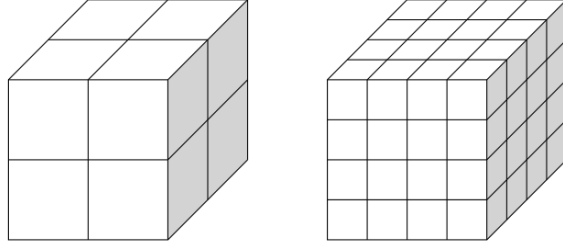
Fig. 8: Construction of a non-conforming hexahedral mesh by combining two conforming Cartesian meshes.

where $(u_1, u_2, u_3)^T$ is the three-dimensional velocity vector and

$$\Pi = \Pi_{\max} e^{\frac{1 - \sum_{i=1}^{d} r_i^2}{2}}, \qquad \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} -(x_2 - c_2 - t) \\ x_1 - c_1 \\ 0 \end{pmatrix}.$$

We take $c_1 = c_2 = 7.5$, $p_0 = 1/\gamma$, and $\Pi_{\max} = 0.4$, and solve on the domain $[0, 15] \times [0, 20] \times [0, 1]$.

We construct a non-conforming mesh by gluing together two uniform hexahedral meshes of different resolutions (see Figure 8). The non-conforming interface runs parallel to the y-axis, such that the vortex propagates along the direction of the non-conforming interface. The solution is evolved until final time $T = 1$. Figure 9 shows computed estimates of L^2 and L^∞ errors, where we estimate both by evaluating the solution using a tensor product Gauss quadrature rule with $N + 3$ points in each coordinate direction. For the L^2 error, we observe that Gauss quadrature is more accurate than Lobatto quadrature. However, Gauss quadrature delivers significantly more accurate L^∞ errors than Lobatto quadrature, which is consistent with numerical experiments in [25].

Finally, we discuss computed L^2 convergence rates. For $N = 1$, we observe convergence rates of 1.365 and 1.939 for Lobatto and Gauss, respectively. This order reduction is less prominent for $N = 3$, with Lobatto and Gauss quadratures achieving rates of 3.468 and 3.977, respectively. Order reduction is observed for both Gauss and Lobatto nodes for $N = 2$, with both choices achieving convergence rates of around 2.3. This may indicate that the meshes have not yet been sufficiently refined to reach the asymptotic regime, or may be a consequence of the use of Lax-Friedrichs penalization [24].

TODO: add curved results and results using two-layer mortars.

8 Conclusions and extensions of the mortar-based approach

We have introduced an entropy stable mortar-based interface treatment which reduces the computational cost of high order entropy stable Lobatto and Gauss collocation DG schemes on non-conforming meshes. Mortars are incorporated into entropy stable DG schemes using mortar-based extensions of hybridized SBP operators introduced in [1, 8, 9], and admit efficient face-local implementations.
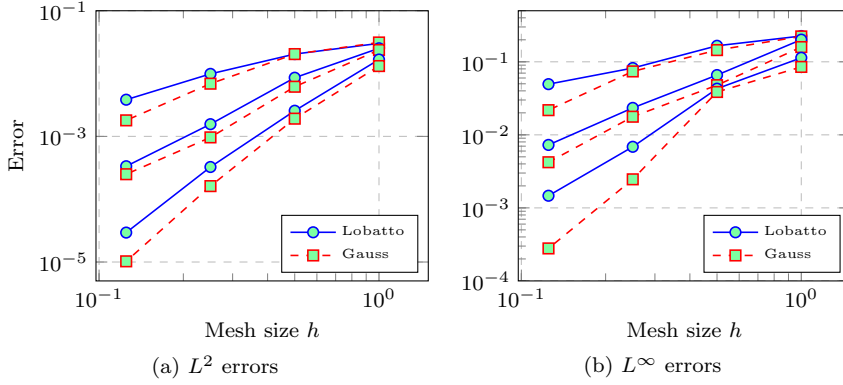
(a) L^2 errors   (b) L^∞ errors

Fig. 9: L^2 and L^∞ errors for entropy stable Lobatto and Gauss collocation schemes on non-conforming Cartesian (affine) hexahedral meshes under h-refinement for $N = 1, 2, 3$. Note the different axis limits.

We conclude by discussing some extensions and generalizations of this mortar-based approach. First, the extension to p and hp non-conforming meshes is straightforward. As with h non-conforming meshes, the mortar ndoes are taken to be the nodes on the refined side of a hp non-conforming interfaces. The two-layer mortar approach of Section 6.1 can also be extended to hp-nonconforming interfaces in a straightforward fashion.

This approach can also be generalized to non-hexahedral elements. However, the mortar-based SBP operators in this paper require node alignment to expose more sparsity. Unlike tensor product elements, simplicial elements, neither mortar nor surface nodes are "aligned" with volume nodes. Thus, the mortar-based approach in this work is not expected to be more efficient than the direct use of hybridized SBP operators on non-tensor product elements [34].

9 Acknowledgments

A Geometric terms for curved non-conforming meshes

Entropy conservation (stability) requires that the geometric terms be constructed appropriately. In addition to requiring satisfaction of the discrete geometric conservation law (GCL) [7, 8, 16], the schemes derived in this work require that the polynomial degree of the geometric terms be related to the accuracy of the surface and mortar quadrature rules.

As discussed in Section 5.3, the standard construction of geometric terms (using a cross product formula [16, 35]) on 2D isoparametric curved elements are sufficient to ensure the GCL and thus entropy conservation (stability) for Lobatto and Gauss collocation schemes. For 3D isoparametric curved hexahedra, the standard construction of geometric terms does not ensure satisfaction of the GCL. We consider two alternative methods to construct geometric terms: Approach 1 from [16], and Approach 2 from [17].

We note that entropy conservation (or stability) on non-conforming meshes also requires that the scaled outward normals are equal and opposite across each mortar face. This holds if the mesh is *watertight* or *well-constructed*, such that there are no gaps between neighboring elements. Several conditions which guarantee that a curved non-conforming mesh is well-constructed are described in [18]. For the two-dimensional numerical experiments in this work, we construct watertight curved non-conforming meshes by first curving a conforming mesh. We locally refine to construct a non-conforming mesh, with the refined child elements inheriting the geometric mapping of the parent elements. For the three-dimensional numerical experiments in this work, we adopt the following approach.

A.1 Construction of geometric terms for curved hexahedra

Let $\boldsymbol{D}_{1\mathrm{D}}$ denote the $(N_{\mathrm{geo}} + 1) \times (N_{\mathrm{geo}} + 1)$ one-dimensional nodal differentiation matrix at degree N_{geo} Lobatto nodes. Multi-dimensional differentiation matrices on the reference hexahedron can then be constructed via

$$\boldsymbol{D}_1 = \boldsymbol{D}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}}$$
$$\boldsymbol{D}_2 = \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{D}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}}$$
$$\boldsymbol{D}_3 = \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{D}_{1\mathrm{D}},$$

where $\boldsymbol{I}_{1\mathrm{D}}$ is the $(N_{\mathrm{geo}} + 1) \times (N_{\mathrm{geo}} + 1)$ identity matrix.

Let $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ denote the vectors of physical nodal coordinates of a curved degree N_{geo} element. Approach 1 in [16] constructs scaled geometric terms $G_{ij} = J\frac{\partial \widehat{x}_j}{\partial x_i}$ by computing the curl of geometric "potentials" $\boldsymbol{f}_{ij}$

$$\boldsymbol{G}_{i1} = \alpha_i \left(\boldsymbol{D}_3 \boldsymbol{f}_{i2} - \boldsymbol{D}_2 \boldsymbol{f}_{i3} \right),$$
$$\boldsymbol{G}_{i2} = \alpha_i \left(\boldsymbol{D}_1 \boldsymbol{f}_{i3} - \boldsymbol{D}_3 \boldsymbol{f}_{i1} \right),$$
$$\boldsymbol{G}_{i3} = \alpha_i \left(\boldsymbol{D}_2 \boldsymbol{f}_{i1} - \boldsymbol{D}_1 \boldsymbol{f}_{i2} \right)$$

where $\alpha_1 = 1$ and $\alpha_2 = \alpha_3 = -1$, and the potentials $\boldsymbol{f}_{ij}$ are defined as

$$\boldsymbol{f}_{1j} = (\boldsymbol{D}_j \boldsymbol{y}) \circ \boldsymbol{z}, \qquad \boldsymbol{f}_{2j} = (\boldsymbol{D}_j \boldsymbol{x}) \circ \boldsymbol{z}, \qquad \boldsymbol{f}_{3j} = (\boldsymbol{D}_j \boldsymbol{y}) \circ \boldsymbol{x}. \qquad (36)$$

The resulting geometric terms are tensor product polynomials of degree $G_{ij} \in Q^{N_{\mathrm{geo}}}$. We note that any geometric terms constructed by applying the discrete curl operator automatically satisfy the geometric conservation law.

On non-conforming meshes, the resulting construction does not guarantee that the scaled normal vectors (10) are equal and opposite across faces [18]. This is remedied by enforcing continuity of the geometric potentials $\boldsymbol{f}_{ij}$ across all non-conforming interfaces (for example, by interpolating values of $\boldsymbol{f}_{ij}$ from the coarse side to the refined side).

Approach 2 in [17] (see also [9], Footnote 3) further modifies this approach by slightly by reducing the degree of each geometric potential along the corresponding coordinate direction. Let $\boldsymbol{I}_{N_{\mathrm{geo}}}^{N_{\mathrm{geo}}-1}$ denote the interpolation operator from degree N_{geo} to degree $N_{\mathrm{geo}} - 1$ Lobatto nodes, and let $\boldsymbol{I}_{N_{\mathrm{geo}}-1}^{N_{\mathrm{geo}}}$ denote the interpolation operator from degree $N_{\mathrm{geo}} - 1$ to degree N_{geo} Lobatto nodes. Define the degree reduction operator $\boldsymbol{F}_{1\mathrm{D}}$ as the product of these interpolation operators

$$\boldsymbol{F}_{1\mathrm{D}} = \boldsymbol{I}_{N_{\mathrm{geo}}-1}^{N_{\mathrm{geo}}} \boldsymbol{I}_{N_{\mathrm{geo}}}^{N_{\mathrm{geo}}-1}.$$

Multiplication by $\boldsymbol{F}_{1\mathrm{D}}$ lowers the degree of a polynomial from N_{geo} to $N_{\mathrm{geo}} - 1$ while leaving the boundary values unchanged. We can define multi-dimensional degree reduction operators

$$\boldsymbol{F}_1 = \boldsymbol{F}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}}$$
$$\boldsymbol{F}_2 = \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{F}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}}$$
$$\boldsymbol{F}_3 = \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{I}_{1\mathrm{D}} \otimes \boldsymbol{F}_{1\mathrm{D}}.$$

N	1	2	3	4
Approach 1	2.8472	4.0772	4.9896	6.0879
Approach 2	2.8494	4.0238	4.9608	6.0769

Table 1: Computed L^2 error convergence rates for Approach 1 [16] and Approach 2 [17]. Both approaches achieve $O(h^{N+2})$ rates, which were proven in [7, 8] for simplicial elements.

Suppose $u \in Q^{N_{\text{geo}}}$ is represented using nodal values $\boldsymbol{u}$; then, $\boldsymbol{F}_1 \boldsymbol{u}$ corresponds to a polynomial in $Q^{N_{\text{geo}}-1,N_{\text{geo}},N_{\text{geo}}}$. In other words, $\boldsymbol{F}_i$ lowers the polynomial degree in the ith coordinate by 1. Approach 2 computes the geometric terms $\boldsymbol{G}_{ij}$ via

$$\boldsymbol{G}_{i1} = \alpha_i \left(\boldsymbol{D}_3 \widetilde{\boldsymbol{f}}_{i2} - \boldsymbol{D}_2 \widetilde{\boldsymbol{f}}_{i3} \right),$$

$$\boldsymbol{G}_{i2} = \alpha_i \left(\boldsymbol{D}_1 \widetilde{\boldsymbol{f}}_{i3} - \boldsymbol{D}_3 \widetilde{\boldsymbol{f}}_{i1} \right),$$

$$\boldsymbol{G}_{i3} = \alpha_i \left(\boldsymbol{D}_2 \widetilde{\boldsymbol{f}}_{i1} - \boldsymbol{D}_1 \widetilde{\boldsymbol{f}}_{i2} \right)$$

where $\widetilde{\boldsymbol{f}}_{ij} = \boldsymbol{F}_j \boldsymbol{f}_{ij}$ are degree-reduced geometric potentials, and the original geometric potentials $\boldsymbol{f}_{ij}$ are computed via (36). Since multiplication by $\boldsymbol{D}_i$ reduces the degree by 1 in the ith coordinate by 1, the resulting geometric terms G_{ij} satisfy

$$G_{i1} \in Q^{N_{\text{geo}},N_{\text{geo}}-1,N_{\text{geo}}-1}$$
$$G_{i2} \in Q^{N_{\text{geo}}-1,N_{\text{geo}},N_{\text{geo}}-1}$$
$$G_{i3} \in Q^{N_{\text{geo}}-1,N_{\text{geo}}-1,N_{\text{geo}}}, \qquad i = 1,2,3.$$

The difference between Approach 1 and Approach 2 appears in Lemma 6. For Lobatto nodes, the mortar-based operators $\boldsymbol{Q}_{i,m}$ satisfy an SBP property (and can be used to construct entropy stable formulations) for $N_{\text{geo}} < N$ under Approach 1, while Approach 2 relaxes this condition to $N_{\text{geo}} \leq N$.

While Approach 1 is marginally more accurate than Approach 2 for $N > 1$, both approaches achieve nearly identical accuracy and convergence rates. We analyze L^2 errors using Approach 1 and Approach 2 by computing exact values of G_{ij} using the cross product formula [16, 35]. We construct a curved mesh by transforming a Cartesian hexahedral grid on $[-1,1]^3$ by interpolating the global curved mapping at Lobatto nodes

$$\widetilde{x} = x + \frac{1}{4} \cos(x)\sin(y)\sin(z)$$

$$\widetilde{y} = y + \frac{1}{4} \sin(x)\cos(y)\sin(z)$$

$$\widetilde{z} = z + \frac{1}{4} \sin(x)\sin(y)\cos(z),$$

where x, y, z denote coordinates on the Cartesian grid and $\widetilde{x}, \widetilde{y}, \widetilde{z}$ denote coordinates on the mapped curved domain. Figure 10 shows the convergence of the L^2 error over all geometric terms under mesh refinement. Both Approach 1 and Approach 2 achieve very similar errors for $N = 1, \ldots, 4$, with Approach 1 achieving slightly lower errors for all $N > 1$. Table 1 shows computed convergence rates, which match the $O(h^{N+2})$ rates proven in [7, 8] for simplicial elements.

References

1. Jesse Chan, David C Del Rey Fernández, and Mark H Carpenter. Efficient entropy stable Gauss collocation methods. *SIAM Journal on Scientific Computing*, 41(5):A2938–A2966, 2019.
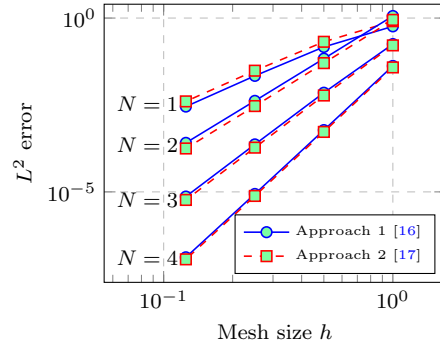
Fig. 10: L^2 errors under mesh refinement for Approach 1 [16] and Approach 2 [17].

2. Jesse Chan. On discretely entropy conservative and entropy stable discontinuous Galerkin methods. *Journal of Computational Physics*, 362:346 – 374, 2018.

3. Travis C Fisher and Mark H Carpenter. High-order entropy stable finite difference schemes for nonlinear conservation laws: Finite domains. *Journal of Computational Physics*, 252:518–557, 2013.

4. Mark H Carpenter, Travis C Fisher, Eric J Nielsen, and Steven H Frankel. Entropy Stable Spectral Collocation Schemes for the Navier–Stokes Equations: Discontinuous Interfaces. *SIAM Journal on Scientific Computing*, 36(5):B835–B867, 2014.

5. Gregor J Gassner, Andrew R Winters, and David A Kopriva. Split form nodal discontinuous Galerkin schemes with summation-by-parts property for the compressible Euler equations. *Journal of Computational Physics*, 327:39–66, 2016.

6. Tianheng Chen and Chi-Wang Shu. Entropy stable high order discontinuous Galerkin methods with suitable quadrature rules for hyperbolic conservation laws. *Journal of Computational Physics*, 345:427–461, 2017.

7. 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.

8. Jesse Chan and Lucas C Wilcox. Discretely entropy stable weight-adjusted discontinuous Galerkin methods on curvilinear meshes. *Journal of Computational Physics*, 378:366 – 393, 2019.

9. Jesse Chan. Skew-Symmetric Entropy Stable Modal Discontinuous Galerkin Formulations. *Journal of Scientific Computing*, 81(1):459–485, Oct 2019.

10. Christine Bernardi, Yvon Maday, and Anthony T Patera. Domain decomposition by the mortar element method. In *Asymptotic and numerical methods for partial differential equations with critical parameters*, pages 269–286. Springer, 1993.

11. Lucas Friedrich, Andrew R Winters, David C Del Rey Fernández, Gregor J Gassner, Matteo Parsani, and Mark H Carpenter. An entropy stable h/p non-conforming discontinuous Galerkin method with the summation-by-parts property. *Journal of Scientific Computing*, pages 1–37, 2017.

12. Constantine M Dafermos. *Hyperbolic conservation laws in continuum physics*. Springer, 2005.

13. David C Del Rey Fernández, Pieter D Boom, and David W Zingg. A generalized framework for nodal first derivative summation-by-parts operators. *Journal of Computational Physics*, 266:214–239, 2014.

14. Tianheng Chen and Chi-Wang Shu. Review of entropy stable discontinuous Galerkin methods for systems of conservation laws on unstructured simplex meshes, 2019. Accessed July 25, 2019.

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

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

17. Jeremy E Kozdon and Lucas C Wilcox. An energy stable approach for discretizing hyperbolic equations with nonconforming discontinuous Galerkin methods. *Journal of Scientific Computing*, 76(3):1742–1784, 2018.

18. David A Kopriva, Florian J Hindenlang, Thomas Bolemann, and Gregor J Gassner. Free-Stream Preservation for Curved Geometrically Non-conforming Discontinuous Galerkin Spectral Elements. *Journal of Scientific Computing*, 79(3):1389–1408, 2019.

19. Eitan Tadmor. The numerical viscosity of entropy stable schemes for systems of conservation laws. I. *Mathematics of Computation*, 49(179):91–103, 1987.

20. Eitan Tadmor and Weigang Zhong. Entropy stable approximations of Navier–Stokes equations with no artificial numerical viscosity. *Journal of Hyperbolic Differential Equations*, 3(03):529–559, 2006.

21. Johnathon Upperman and Nail K Yamaleev. Entropy stable artificial dissipation based on Brenner regularization of the Navier-Stokes equations. *Journal of Computational Physics*, 393:74–91, 2019.

22. 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.

23. Magnus Svärd and Hatice Özcan. Entropy-stable schemes for the Euler equations with far-field and wall boundary conditions. *Journal of Scientific Computing*, 58(1):61–89, 2014.

24. Florian J Hindenlang and Gregor J Gassner. On the order reduction of entropy stable DGSEM for the compressible Euler equations. *arXiv preprint arXiv:1901.05812*, 2019.

25. David C Fernandez, Mark H Carpenter, Lisandro Dalcin, Stefano Zampini, and Matteo Parsani. Entropy Stable h/p-Nonconforming Discretization with the Summation-by-Parts Property for the Compressible Euler and Navier-Stokes Equations. *arXiv preprint arXiv:1910.02110*, 2019.

26. 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.

27. Andrew R Winters, Christof Czernik, Moritz B Schily, and Gregor J Gassner. Entropy stable numerical approximations for the isothermal and polytropic Euler equations. *BIT Numerical Mathematics*, pages 1–34, 2019.

28. 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.

29. Mark H Carpenter and Christopher A Kennedy. Fourth-order 2n-storage Runge-Kutta schemes. Technical Report NASA-TM-109112, NAS 1.15:109112, NASA Langley Research Center, 1994.

30. 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.

31. Chi-Wang Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In *Advanced numerical approximation of nonlinear hyperbolic equations*, pages 325–432. Springer, 1998.

32. Jared Crean, Jason E Hicken, David C Del Rey Fernández, David W Zingg, and Mark H Carpenter. High-Order, Entropy-Stable Discretizations of the Euler Equations for Complex Geometries. In *23rd AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, 2017.

33. David M Williams and Antony Jameson. Nodal points and the nonlinear stability of high-order methods for unsteady flow problems on tetrahedral meshes. In *21st AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, June 2013.

34. Siavosh Shadpey and David W Zingg. Energy-and Entropy-Stable Multidimensional Summation-by-Parts Discretizations on Non-Conforming Grids. In *AIAA Aviation 2019 Forum*, page 3204, 2019.

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