Computational Astrophysics and Cosmology
a SpringerOpen Journal

RESEARCH Open Access

Technologies for supporting high-order geodesic mesh frameworks for computational astrophysics and space sciences

Vladimir Florinski[1*], Dinshaw S. Balsara[2], Sudip Garain[2,3] and Katharine F. Gurski[4]

Abstract

Many important problems in astrophysics, space physics, and geophysics involve flows of (possibly ionized) gases in the vicinity of a spherical object, such as a star or planet. The geometry of such a system naturally favors numerical schemes based on a spherical mesh. Despite its orthogonality property, the polar (latitude-longitude) mesh is ill suited for computation because of the singularity on the polar axis, leading to a highly non-uniform distribution of zone sizes. The consequences are (a) loss of accuracy due to large variations in zone aspect ratios, and (b) poor computational efficiency from a severe limitations on the time stepping. Geodesic meshes, based on a central projection using a Platonic solid as a template, solve the anisotropy problem, but increase the complexity of the resulting computer code. We describe a new finite volume implementation of Euler and MHD systems of equations on a triangular geodesic mesh (TGM) that is accurate up to fourth order in space and time and conserves the divergence of magnetic field to machine precision. The paper discusses in detail the generation of a TGM, the domain decomposition techniques, three-dimensional conservative reconstruction, and time stepping.

Keywords: Geodesic mesh; Finite volume scheme; Divergence free MHD

1 Introduction

Objects in the universe tend to assume a spherical shape owing to the central nature of the gravitational force. Common examples include globular star clusters, stars and stellar-like objects, planets, and the larger planetary satellites. Modeling such objects' interior, surface, or atmospheric processes is most conveniently done in a spherical coordinate system because it is perfectly adapted to the shape of the object. A three-dimensional spherical coordinate system has radial distance from the center of the sphere as one of its coordinates. In a spherical *polar* coordinate system the two remaining coordinates are the polar angle, or co-latitude, and the azimuthal angle. Implementing a computational mesh based on the polar spherical system incurs only a modest increase in algorithmic complexity compared with Cartesian meshes because both meshes are logically orthogonal. Unfortunately, this simplicity comes at a price: spherical polar meshes have a singularity on the polar axis where the planes of constant azimuth converge to a single line. As a result the sizes of the computational zones become progressively smaller toward the poles. A polar mesh therefore provides a very non-uniform coverage of the surface of the sphere, which is a highly undesirable property. Because the time step used in a simulation is proportional to the smallest dimension of the zone, a simulation based on a polar mesh is quite inefficient.

*Correspondence: vaf0001@uah.edu
[1]Space Science Department, University of Alabama in Huntsville, Huntsville, USA
Full list of author information is available at the end of the article

Polar singularities can be avoided by using a composite mesh, consisting of multiple partially overlapping patches of structured mesh, where each patch is singularity free (Phillips 1959; Browning et al. 1989; Kageyama and Sato 2004; Feng et al. 2010; Usmanov et al. 2012). In this approach the different meshes must be synchronized in their regions of overlap, which involves interpolation and could result in a loss of accuracy or conservation. Another approach, first introduced in the work of Sadourny et al. (1968), uses a mesh that covers the surface of the sphere without gaps or overlaps, known as a tesselation. Each "tile" in the tesselatation is a spherical polygon such as a triangle, a quadrilateral, a pentagon, or a hexagon. The lines connecting adjacent vertices on the sphere are usually (but not always) great circle arcs, which are geodesic lines on the sphere (hence the name, "geodesic mesh"). A well chosen tesselation method can provide a nearly uniform coverage of the surface of the sphere which greatly improves computational efficiency.

A geodesic mesh is constructed from a regular polyhedron (Platonic solid) inscribed inside a sphere used as a template. The most common method of generating such a mesh is to project the edges of the polyhedron to the sphere and recursively subdivide each spherical polygon into smaller polygonal faces until the desired level of discretization is achieved. A cube can be used to generate a cube-sphere mesh whose faces are quadrilaterals (Ronchi et al. 1996; Koldoba et al. 2002; Choblet et al. 2007; Putman and Lin 2007; Ivan et al. 2015; Ullrich and Taylor 2015). Such a mesh is topologically Cartesian within each of the six faces of the cube, requiring special treatment only in the vicinity of the eight corners. It is also possible to construct a mesh out of triangles using an octahedron (Feng et al. 2007), dodecahedron (Nakamizo et al. 2009), or an icosahedron (Giraldo 1997; Pudykiewicz 2006; Bernard et al. 2009) as the base solid. A variation of this approach uses a hexagon based dual tesselation, obtained by replacing the vertices of the triangular mesh with face circumcenters and vice versa (Heikes and Randall 1995a; Du et al. 2003; Feng et al. 2007; Miura 2007; Florinski et al. 2013).

Non geodesic tesselations also exist; one prominent example being the HEALPix mesh used for numerical analysis of astrophysical data on the sphere (Gorski et al. 2005). For three-dimensional problems the tesselation is extruded radially, producing a three-dimensional spherical geodesic mesh. A 3D mesh based on a geodesic tesselation has a very useful property that some of its faces (the so-called r-faces, see below) are flat, which greatly simplifies the numerical scheme. By contrast, all faces of non-geodesic meshes are curved, making such meshes less convenient for use with 3D problems.

In this paper we describe a powerful new framework for finite volume simulations on a triangular geodesic mesh (TGM) with second, third, and fourth orders of accuracy. At this time the software is developed to solve MHD problems with up to fourth order of accuracy in space and time, while conserving the divergence of the magnetic field down to machine precision. Several of the underlying numerical algorithms have been previously published and we refer the interested reader to these papers. However, implementation of these algorithms on a geodesic mesh requires a novel perspective. This is because a geodesic mesh possesses properties of both structured and unstructured meshes. A number of innovative techniques need to be brought together in order to efficiently carry out CFD type simulations on TGMs. The goal of this paper is to describe in detail the techniques that enable efficient implementation of MHD algorithms on spherical geodesic meshes.

2 Mesh construction

The choice of spherical polygons used to tile the sphere consists of triangles, quadrilaterals, and hexagons (with a small mix of pentagons), but not all combinations result in a high quality mesh. It is desirable to have a mesh that is both highly uniform (or isotropic) and nestable. The first property demands that the faces should be approximately of the same shape and size, while the second ensures strict parent-child relationship between the recursive subdivisions, which is a critical property for domain decomposition (and hence efficient parallelization) as well as adaptive refinement. A regular polyhedron is perfectly uniform: the edges are all of equal length, the faces have the same area, and the vertex angles are the same (see the upper left panel in Fig. 1). However, the very first subdivision breaks this perfect symmetry because the four daughter faces are of a slightly different shape and size. For example, in a triangular mesh shown in the lower left panel of Fig. 1 the daughter face in the middle of the parent face is slightly different in size from the three daughter faces at the corners. Consequently, higher division meshes are somewhat less uniform that those at lower division. This departure from uniformity is greatest near the vertices of the base polyhedron. In addition, the uniform connectivity of the mesh is violated near these singular points. As an example, consider a mesh constructed from a base hexahedron with quadrilateral faces (i.e., the cube sphere). While commonly each vertex is shared by four faces, only three meet at the eight singular points. As a result the quadrilaterals adjacent to these vertices are diamond shaped, rather than square.

The mesh described in this paper is constructed from an icosahedron and has triangular shaped faces. The upper right panel in Fig. 1 shows that there are twelve singular points in this mesh, where five triangles meet instead of the usual six, but the anisotropy so introduced is not as prominent because the defects are distributed over a larger number of sites. This is the reason that an icosahedron produces a superior mesh compared to a tetrahedron or an octahedron. A dodecahedron can in principle be used,

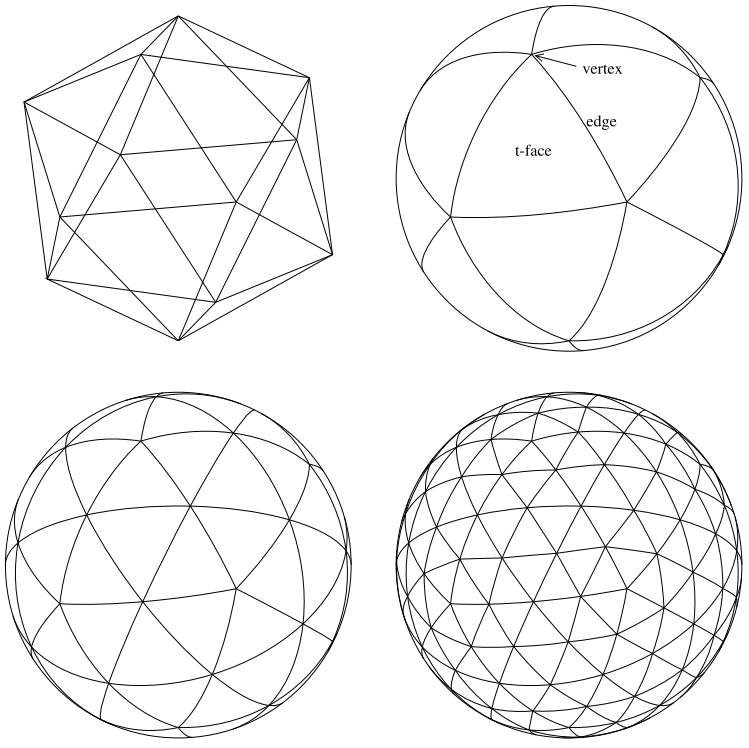

Figure 1 Recursive icosahedral mesh generation. Shown are the inscribed icosahedron (top left) and the division 0 (top right), 1(bottom left), and 2(bottom right) triangular tesselations. The edges of the tesselation are repeatedly bisected until a desired level of refinement is reached

but it lacks a division 0 triangular tesselation, consisting instead of pentagons, and is less convenient for practical use. A hexagonal mesh like that used by Florinski et al. (2013) has good uniformity, but is not nestable.

Construction of a TGM begins with inscribing an icosahedron inside a sphere (in the rest of this paper we will always assume that the sphere has a unit radius, unless stated otherwise) and centrally projecting its edges to the surface of the sphere, see the top row of Fig. 1. This projection generates a *division* 0 tesselation that includes 12 vertices, 20 triangular faces, called *t-faces* and 30 edges, called *t-edges* (these names are chosen to distinguish them from the faces and edges oriented in the radial direction produced by the radial extrusion of the mesh that bear the prefix "r"). For the sake of efficiency, all calculations on the sphere are performed in Cartesian coordinates using vector operations on the vertices. The input to the mesh generator consists of the coordinates of the icosahedron's vertices, vertex-vertex (VV) neighbor information, and face-vertex (FV) connectivity information.

At each division, the complete mesh connectivity information is computed and stored. For vertices, this includes the list (VV) of six neighbor vertices (five at division 0), six(five) t-edges meeting at the vertex (VE) and six(five) t-faces sharing the vertex (VF). For edges, connectivity information includes the two vertices at the ends (EV) and the two t-faces sharing this t-edge (EF). Finally, for faces we compute the list of three vertices at the corners (FV), the list of three edges (FE) and the list of three face neighbors (FF), for the total of eight connectivity tables. Table 1 shows the order of connectivity table generation and the methods used for construction. Note that at division zero the VV and FV information is already available and steps 1 and 3 are therefore omitted. To facilitate search operations FV, FE, FF, and VF lists are ordered in the counterclockwise direction, while the remaining tables are not ordered. None of the steps of the mesh generation process require a full search, and the algorithm is linear in the number of elements.

Table 1 Connectivity table construction methods

Step	Table	Prerequisite	Method of construction
1	VV	parent division	Based on numbering scheme
2	EV	VV	Insert edge per VV entry with no duplicates
3	FV	parent division	Based on numbering scheme
4	VE	EV	Inverse of EV
5	VF	FV	Inverse of FV
6	EF	EV, VF	Match two faces sharing this edge's vertices
7	FE	EF	Inverse of EF
8	FF	EF, FE	Find the other face sharing each edge

To produce a division 1 tesselation shown in Fig. 1 (bottom-left) new vertices are inserted at the midpoints of division 0 edges. These vertices are then connected with new edges (great circle arcs) that divide each spherical triangle into four smaller triangles. The process is repeated until the desired level of refinement is achieved. It can be easily verified that the number of vertices, edges, and faces in the tesselation at division d are

$$N_v(d) = 2 + 10 \times 2^{2d}, \qquad N_e(d) = 30 \times 2^{2d}, \qquad N_f(d) = 20 \times 2^{2d}. \tag{1}$$

It should be pointed out that the mesh construction algorithm described above is not restricted to icosahedral meshes, but can in principle start with any one of the five Platonic solids. Only steps 1 and 3 in Table 1 need to be adjusted. This property permits writing highly modular geodesic mesh generation algorithms for the sphere.

The nonuniformity of the mesh can be assessed by computing the ratios between the largest and the smallest measurement of edge lengths, vertex angles, and face areas. A high quality mesh would have these ratios as close to unity as possible. Table 2 documents the properties of triangular icosahedral tesselations at divisions zero through eight. Note that the ratios quickly converge to their asymptotic values. The largest face is only 30% larger than the smallest face, so the disparity in zone sizes will not noticeably affect the time step. Figure 2 compares the geometric properties of the icosahedral TGM and the hexahedral quadrilateral geodesic mesh (QGM), also known as the gnomonic cube sphere. Shown are the edge, angle, and area largest-to-smallest ratios that should be a close to unity as possible. One can see that the icosahedral mesh has superior uniformity of every property compared with the QGM.

The simple mesh does have a few deficiencies, mainly related to the fact that the centroids of the faces are distinct from the circumcenters, as pointed out by Heikes and Randall (1995b). Several numerical optimization algorithms have been proposed to improve the mesh, including the spring dynamics model (Tomita et al. 2001) and the centroidal generation algorithm (Du et al. 1999). Numerical optimization methods usually improve a certain mesh property at the expense of another. For example, an algorithm could trade face area uniformity for vertex angle disparity. Another problem with numerically modified meshes is that the optimization process is specific to each division and the resulting meshes lose their nestable property, i.e., become unsuitable for mesh refinement (Putman and Lin 2007). Because we anticipate such development in the future, and because we have not observed any adverse effects from using the simple recursive mesh, it is our preferred method of construction.

The triangular tesselation is extruded radially over a number of concentric spherical layers called shells, to produce the three-dimensional TGM. The software stores the reciprocal connectivity tables for every element on the sphere (vertex, edge, or face) at all divisions, up to the maximum allowed. In addition, there are tree structures describing the parent-child relationships between the faces. For the purpose of domain decomposition, a face subdivided into higher division faces is called a *sector* and a layer

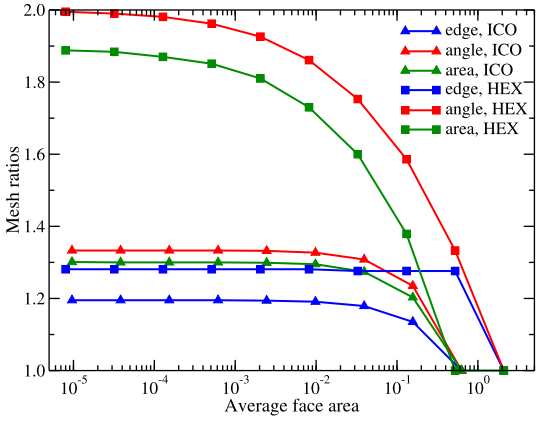

Figure 2 Uniformity measures of the icosahedral mesh (triangles) and the hexahedral mesh (squares) for division 0–8 (former) and 0–9 (latter). Blue lines show edge ratios, red angle ratios, and green area ratios. The icosahedral TGM is measurably more uniform than the gnomonic cube sphere

Table 2 Triangular geodesic mesh properties at divisions 0–8

Div	Vertices	Edges	Faces	Avg. edge	Avg. angle	Average area	Edge ratio	Angle ratio	Area ratio
0	12	30	20	63.4°	72.0°	6.28×10^{-1}	1.00	1.00	1.00
1	42	120	80	33.9°	63.0°	1.57×10^{-1}	1.14	1.24	1.20
2	162	480	320	17.2°	60.8°	3.93×10^{-2}	1.18	1.31	1.28
3	642	1920	1280	8.64°	60.2°	9.82×10^{-3}	1.19	1.33	1.29
4	2562	7680	5120	4.33°	60.0°	2.45×10^{-3}	1.19	1.33	1.30
5	10,242	30,720	20,480	2.16°	60.0°	6.14×10^{-4}	1.19	1.33	1.30
6	40,962	122,880	81,920	1.08°	60.0°	1.53×10^{-4}	1.19	1.33	1.30
7	163,842	491,520	327,680	0.54°	60.0°	3.84×10^{-5}	1.19	1.33	1.30
8	655,362	1,966,080	1,310,720	0.27°	60.0°	1.16×10^{-5}	1.19	1.33	1.30

of consecutive shells is called a *slab*. An intersection between a sector and a slab is called a *block*, which is the computational unit on this mesh. Each computational zone has the shape of a truncated triangular pyramid also known as a *frustum*.

Locating an arbitrary vector (i.e., finding the zone containing the vector) on the TGM follows a simple procedure valid for any nested polyhedral tesselation. Once the shell number has been determined (via a mapping function or bisection search), the vector is normalized to unity. The nearest division 0 vertex is found by computing the largest scalar product with all 12 vertices at that division. Next, the algorithm tests which of the five surrounding t-faces the vector belongs to, and then recursively tests the four daughter faces at each division. A test for the t-face interior consists of computing the triple products of the vector with two consecutive vertices (1–2, 2–3, and 3–1). If all three triple products are positive, the point belongs to the interior of the t-face with counter-clockwise vertex ordering.

Partitioning the mesh into sectors and slabs enables efficient domain decomposition and offers many opportunities for parallelization. The software framework uses MPI and MPI-derived libraries and achieves essentially linear weak scaling (Balsara et al. 2019). We will next concentrate on a single triangular block and describe its partitioning into computational zones, generating stencils, and performing reconstruction of zone based mesh variables with a desired order of accuracy.

3 Grid blocks

The tree numbering system for the faces, edges, and vertices is too slow to be used for zone access within a sector, for which we introduce a flat, two-dimensional "triangular addressing scheme", or TAS. The face numbering pattern is illustrated in Fig. 3 which shows one block of a mesh whose sector division d_s is three less than its face division ($\Delta d = d - d_s = 3$). In this example the sector has two layers of ghost zones around its interior. The numbering starts from the base vertex identified by the tesselation; the sector is always drawn in an orientation where the principal vertex is in the SW corner. The first coordinate index runs from W to E and the second index runs from SE to NW. The alternating color shading in Fig. 3 is used to distinguish faces with opposite orientations; many of the vector operations are performed with the opposite signs for the shaded (yellow) and unshaded (white) faces.

The number of vertices, t-edges, and t-faces in a sector with N_g layers of ghost zones are

$$N_v = \frac{(L+1)(L+2)}{2}, \qquad N_e = \frac{3L(L+1)}{2}, \qquad N_f = L^2, \tag{2}$$

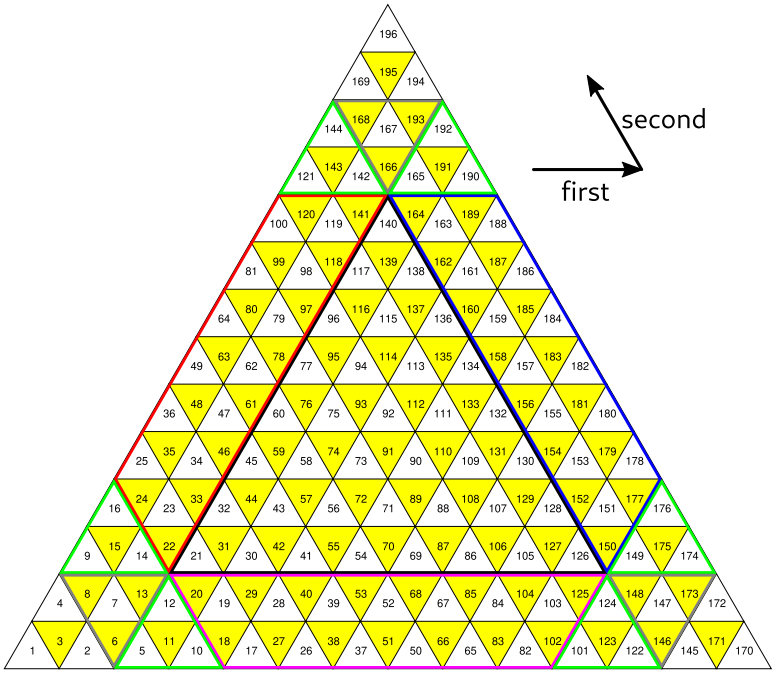

Figure 3 A single sector of the mesh. In this example the face division is equal to the sector division plus three. The black arrows show the directions of the first and second TAS coordinates. Two layers of ghost faces are visible. The three trapezoidal and nine small triangular pieces marked with different border colors are the areas subject to boundary exchange with neighboring blocks

where

$$L = 2^{\Delta d} + 3N_g \tag{3}$$

is the length of the side of the sector. Note that the number of t-edges is three times the number of unshaded faces; it is often convenient to access the edges using a loop on unshaded faces only. The numbering scheme used for the t-edges and vertices is similar to that used for the faces. The edges are numbered in a specific order: first all NE edges, then all NW edges, and finally all S edges (relative to the respective unshaded t-face).

Figure 3 draws with different colors the boundaries of the blocks of ghost zones used to exchange information with the neighboring sectors. The boundary exchange process is discussed in some detail in Sect. 8. Here we only mention that the grey bordered triangular regions may be absent if the block contains one or more *penta-corners*, which are the vertices of the original icosahedron. These vertices have only five neighbor elements rather than six, and care must be taken to adjust stencil generation procedure and boundary exchanges between blocks near these special points. For example, if the principal vertex of the block shown in Fig. 3 is a penta-corner, t-faces 6, 7, 8, and 13 are absent, and the mesh must be closed along the *cut line* that appears in place of the missing faces.

Grid blocks also maintain a set of local connectivity tables similar to those listed in Table 2. These tables have a very regular pattern and are much simpler to construct than the tesselation tables; all neighbors are ordered in counter-clockwise direction. The t-edge orientation is defined with respect to its unshaded neighbor face, which fixes the directions of the normal and tangent vectors on the mesh.

Each grid block needs to know the coordinates of every vertex in the local grid. Because the tesselation numbers its t-faces and vertices differently from the grid blocks, a routine is provided to assemble a list of vertices that lie in a requested sector with ghost cells in the TAS format. The convention is that the base vertex is the first vertex in the FV set of the sector. The mapping routine walks the sector, including the ghost t-faces, from W to E and from SE to NW, storing the coordinates of the vertices encountered along the path. Three step operators are defined, all relative to the base vertex of the t-face, shown in Fig. 4. A type 1 step moves from the initial t-face (t_i) to the final face (t_f) in the S direction and the new base vertex (v_f) is to the E of the old base vertex (v_i) on the common edge. A type 2 step moves diagonally to the NE, and the new base vertex is opposite to the initial base vertex. Finally, a type 3 step moves to the NW, but the new base vertex belongs to the common edge. In Fig. 4, the vertex moves are shown with orange arrows and the face moves with red arrows. These

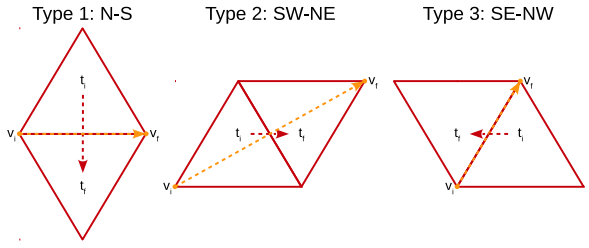

Figure 4 Step operators on the mesh. The three unshaded to shaded step operators shown are used to walk the sector with its ghost t-faces. The step is from the vertex-face pair v_i, t_i to v_f, t_f. The shaded to unshaded steps are obtained by switching the origin and destination vertex-face pairs

three operations apply to unshaded to shaded t-face movement. The shaded to unshaded step operators are algorithmically identical to those, and correspond to switching the initial and the final t-faces and vertices, and reversing the arrow directions.

The sector walk routine works as follows. From the base vertex of the sector, the code first walks to the NW until it encounters the left side of the block (t-face 25 in Fig. 3). Then the code walks to the SW until it reaches the corner of the grid block (face number 1 in the grid block's numbering scheme). From there, the code makes a step to the right followed by i steps diagonally (SE-NW), where i is the index of the horizontal step. That way every cell in the block is visited once. Note that the alternating pattern of shaded and unshaded t-faces is broken across the cut line, and special versions of the step operators are needed to move between the faces of the same shading.

4 Representing spherical geometry

In principle, it is possible to perform all calculations on a TGM by directly using spherical geometry. We found, however, that using isoparametric mapping from a reference zone, which in this case is a right triangular (equilateral) prism, offers significant advantages. In particular, integration on spherical triangles is difficult, requiring a large number of quadrature points at higher orders (Beckmann et al. 2012). Integration on the reference element is straightforward by comparison.

The physical zone and its reference image are shown in Fig. 5. The left panel shows the physical zone that has the shape of a truncated triangular pyramid, also called a frustum. The spherical top and bottom caps are the t-faces, and the annular sides are the r-faces. The frustum therefore has three r-faces and two t-faces. The edges of the t-faces are called t-edges, and the edges connecting the bottom and top t-faces are called r-edges. There are six t-edges, three r-edges, and six vertices per zone. The vertices belonging to a t-face are numbered counter-clockwise in its connectivity tables, 1 through 3, and the t-edges of each t-face are

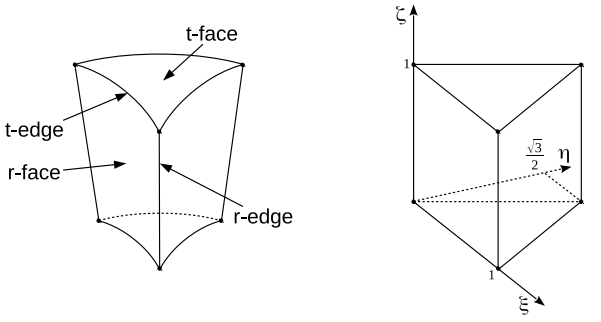

Figure 5 A computational zone (left) and the reference element (right). The reference shape is a right equilateral triangular prism which is mapped into the frustum in physical space

also numbered counter-clockwise, 1 through 3. By convention, a vertex has the same index as the opposite t-edge.

A point in reference space is addressed with a coordinate triplet (ξ, η, ζ). The bottom and the top faces of the prism lie in the planes $\zeta = 0$ and $\zeta = 1$, respectively. The area of the t-face in reference coordinates is $\sqrt{3}/4$, the area of the r-face is one, and the volume of the prism is $\sqrt{3}/4$. It is convenient to work with barycentric coordinates in the $\xi\eta$ plane $(\Lambda_1, \Lambda_2, \Lambda_3)$, defined in Chap. 8 of Zienkiewicz et al. (2013) as

$$\begin{aligned} \xi &= \Lambda_1\xi_1 + \Lambda_2\xi_2 + \Lambda_3\xi_3, \\ \eta &= \Lambda_1\eta_1 + \Lambda_2\eta_2 + \Lambda_3\eta_3, \\ 1 &= \Lambda_1 + \Lambda_2 + \Lambda_3, \end{aligned} \tag{4}$$

where ξ_i and η_i, $i = 1, 2, 3$, are the ξ and η components of the vertices of the triangle in the reference space. The barycentric coordinates are equal to the partial areas of the sub-triangles formed by the point (ξ, η) and the three vertices of the reference triangle (please note that this is not true for the areas of the respective curved triangles). For the equilateral reference triangle the inverse of (4) is

$$\begin{aligned} \Lambda_1 &= 1 - \xi - \frac{\eta}{\sqrt{3}}, \\ \Lambda_2 &= \xi - \frac{\eta}{\sqrt{3}}, \\ \Lambda_3 &= \frac{2\eta}{\sqrt{3}}. \end{aligned} \tag{5}$$

We next introduce a set of two-dimensional linearly independent Lagrange basis functions associated with the nodal points on the curved triangular faces that fix the mapping from reference space to the physical space. It is convenient to compute the nodal point coordinates on the unit sphere; the physical coordinates are obtained simply by rescaling to the desired radial distance. We denote vectors that lie on the unit sphere with the superscript "*u*". All coordinates are factored as

$$\mathbf{x}(\xi, \eta, \zeta) = r(\zeta)\mathbf{x}^u(\xi, \eta), \tag{6}$$

where

$$r(\zeta) = r_b + \zeta(r_t - r_b) = r_b\big[1 + \zeta(\rho - 1)\big], \tag{7}$$

where r_b and r_t are the radial distances of the nodal points on the bottom and the top t-face, respectively, and ρ is their ratio. As discussed below, this factoring enables a more efficient implementation of the reconstruction algorithm on the TGM compared with fully unstructured tetrahedral meshes. To perform integration we also require a set of curvilinear unnormalized basis vectors

$$\begin{aligned} \mathbf{h}_\xi &= \frac{\partial \mathbf{x}}{\partial \xi} = r\frac{\partial \mathbf{x}^u}{\partial \xi}, \qquad \mathbf{h}_\eta = \frac{\partial \mathbf{x}}{\partial \eta} = r\frac{\partial \mathbf{x}^u}{\partial \eta}, \\ \mathbf{h}_\zeta &= \frac{\partial \mathbf{x}}{\partial \zeta} = (r_t - r_b)\mathbf{x}^u. \end{aligned} \tag{8}$$

Given N nodal points on a triangle, there are N Lagrange basis functions $\psi_i(\xi, \eta)$ that satisfy

$$\psi_i(\xi_j, \eta_j) = \delta_{ij}, \tag{9}$$

where (ξ_j, η_j) are the coordinates of nodal point j on the unit sphere. A position vector $\mathbf{x}^u$ can be represented as an expansion over the basis functions

$$\mathbf{x}^u(\xi, \eta) = \sum_{i=1}^{N} \mathbf{v}_i^u \psi_i(\xi, \eta). \tag{10}$$

The coefficients in this expansion are the physical coordinates of the nodal points $\mathbf{v}_i^u$. Figure 6 shows the locations of the nodes on the reference triangle. These elements use $N = 3$, 6, and 10 for linear, quadratic, and cubic basis functions, respectively. The explicit formulas for the basis functions on the equilateral triangle are given in Appendix A. Note that the maps from two adjacent t-faces are continuous at the shared t-edge by virtue of the use of barycentric coordinates for their construction.

An expansion similar to (10) is used for the t-edges. Points from the surface element lying on that edge are used (see Fig. 6) and the corresponding basis functions are simply restriction of the facial bases functions for one of the barycentric coordinate equal to zero. It is convenient to introduce an auxilliary variable δ that measures distance along the edge in the counter-clocksise direction. Its relation to barycentric coordinates is shown in Table 3. The basis functions $\phi_i(\delta)$ for the edges can be also found in Appendix A.

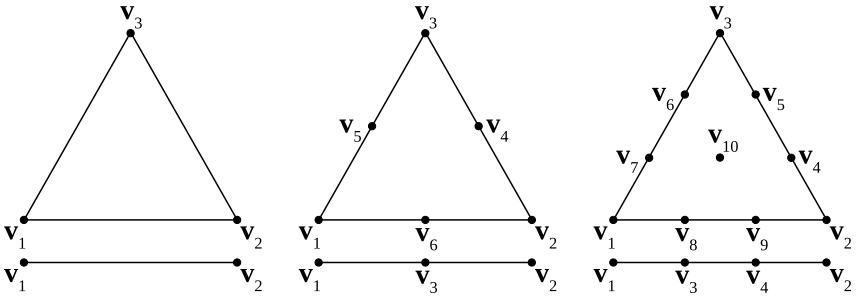

Figure 6 Node locations on t-faces and t-edges for linear (left), quadratic (middle), and cubic (right) mapping. The first order surface element contains three nodes coincident with the vertices. The second order surface element includes all nodes from the first order elements plus the edge midpoints for the total of 6 nodes. The third order surface element includes all nodes from the first order elements plus the points at the thirds of each edge and the centroid, 10 nodes in total. The nodes of line elements representing t-edges are shown below each surface element

Table 3 The choice of the auxilliary variable δ

t-edge/r-face	Λ_1	Λ_2	Λ_3
1	0	$1-\delta$	δ
2	δ	0	$1-\delta$
3	$1-\delta$	δ	0

It is instructive to evaluate the disparity between the mapped surface given by Eq. (10) and the ideal surface, i.e., the unit sphere. Below we compute the error in the radial coordinate, $1-r^u$ for a mapped equilateral spherical triangle with a circumcircle radius of 5°. Figure 7 shows the error distribution for element orders one, two, and three. Obviously, the first order element with its planar faces is unable to reproduce the spherical shape resulting in a large error near the center. Switching to the second order element improves the accuracy by three orders of magnitude, while going to third order yields another factor of ~ 20. It is evident that both second or third order elements reproduce spherical geometry with remarkable accuracy.

It is worth mentioning that Ivan et al. (2013) have previously developed an isoparametric cube sphere model based on a cubic reference element. However, their trilinear mapping anchored at the four corners of the quadrilateral t-face is not capable of truly reproducing a spherical surface because it has only one extra degree of freedom compared with the linear map. For example, when all four vertices lie in the same plane, the trilinear map yields a surface that is flat instead of curved.

5 Evaluation of integrals on a geodesic mesh

A finite volume scheme requires evaluating multi-dimensional integrals in the initial setup phase and during time updates of the conserved variables. This requires, at a minimum, volume and surface integrals. The use of constrained transport scheme to advance the magnetic field requires, in addition, evaluation of integrals along the edges (see Sect. 7 below). We will therefore define the following integral operations: volume integration over a zone, surface integration on t-faces and r-faces, and line integration on t-edges and r-edges. For a three-dimensional vector variable $\mathbf{V}$ these are defined as

$$\iiint_{\text{zone}} \mathbf{V}(\mathbf{x})\, dV = \iiint \mathbf{V}(\xi,\eta,\zeta)(\mathbf{h}_\xi \times \mathbf{h}_\eta)\cdot \mathbf{h}_\zeta \, d\xi\, d\eta\, d\zeta, \tag{11}$$

$$\iint_{\text{t-face}} \mathbf{V}(\mathbf{x})\cdot \mathbf{dS} = \iint_{\Delta} \mathbf{V}(\xi,\eta)\cdot(\mathbf{h}_\xi \times \mathbf{h}_\eta)\, d\xi\, d\eta, \tag{12}$$

$$\iint_{\text{r-face}} \mathbf{V}(\mathbf{x})\cdot \mathbf{dS} = \int_0^1\int_0^1 \mathbf{V}(\delta,\zeta)\cdot(\mathbf{h}_\delta \times \mathbf{h}_\zeta)\, d\delta\, d\zeta, \tag{13}$$

$$\int_{\text{t-edge}} \mathbf{V}(\mathbf{x})\cdot \mathbf{dl} = \int_0^1 \mathbf{V}(\delta)\cdot \mathbf{h}_\delta \, d\delta, \tag{14}$$

$$\int_{\text{r-edge}} \mathbf{V}(\mathbf{x})\cdot \mathbf{dl} = \int_0^1 \mathbf{V}(\zeta)\cdot \mathbf{h}_\zeta \, d\zeta, \tag{15}$$

where the symbol 'Δ' designates integration over a triangle. We will now describe our strategy for evaluating the integrals using quadrature rules. Consider a single zone in the mesh addressed with a t-face index f whose top and bottom vertices lie at r and ρr, respectively. Further, suppose e is the index of one of the t-edges of the zone, and v is one of the vertices.

5.1 Integration on r-edges

R-edges are addressed by the vertex index with specified r and ρ. Because r-faces are always straight, the integrals can be evaluated directly using Gauss–Legendre quadrature points. Define such set of points on the reference in-

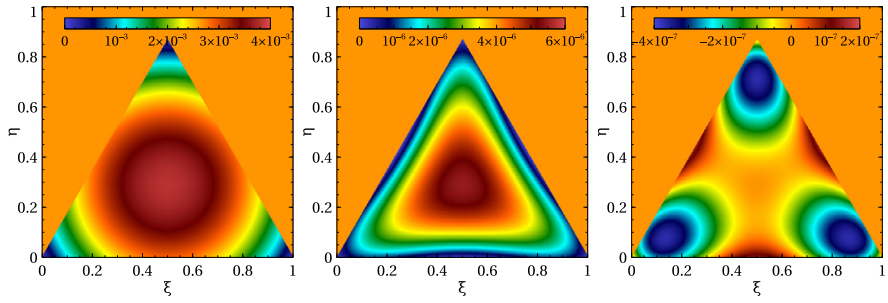

Figure 7 Linear (left), quadratic (center), and cubic (right) mapping accuracy. Shown is the distance between the mapped surface and the perfect sphere computed for an equilateral triangle with a circumcircle radius of 5°

terval $\zeta = [0,1]$ as Q_{1r}. Each quadrature point q has position ζ_q and weight w_q. Then it is evident that

$$\int_{\text{r-edge}} \mathbf{V}(\mathbf{x}) \cdot \mathbf{dl} \approx r(\rho - 1) \sum_{q \in Q_{1r}} w_q \mathbf{V}\left(r\zeta'_q \mathbf{x}^u_q\right) \cdot \mathbf{x}^u_v, \tag{16}$$

where $\mathbf{x}^u_v$ is the position of the vertex v on the unit sphere and

$$\zeta'_q = 1 + (\rho - 1)\zeta_q \tag{17}$$

is the elevated radial position. The code uses 1 quadrature point for integrating polynomials of degrees zero and one, 2 points for degrees two and three, 3 points for fourth and fifth degree polynomials, etc.

5.2 Integration on t-edges

T-edges are addressed by the edge index with fixed r. These edges are curved (except when using linear basis functions) and the quadrature weights are therefore multiplied by the Jacobian equal to the length of the tangent vector $\mathbf{h}_\delta$. Again, designate the set of Gauss–Legendre points on the reference interval $\delta = [0,1]$ as Q_{1t} (which may or may not be the same as Q_{1r}). Using the definition (8) we can write

$$\int_{\text{t-edge}} \mathbf{V}(\mathbf{x}) \cdot \mathbf{dl} \approx r \sum_{q \in Q_{1r}} w_q \mathbf{V}\left(r\mathbf{x}^u_q\right) \cdot \frac{\partial \mathbf{x}^u_e(\delta_q)}{\partial \delta}, \tag{18}$$

where δ_q are the locations of the quadrature points on the reference interval and

$$\mathbf{x}^u_q = \mathbf{x}^u_e(\delta_q). \tag{19}$$

Here the subscript 'e' refers to the fact that the map specific for edge e is used to evaluate the coordinate and its derivative. We use the same number of points for t-edge integration as for r-edge integration. In practice, the values of the point coordinates and tangent vectors on the unit sphere are precomputed for each t-edge at the start of a simulation for fast retrieval.

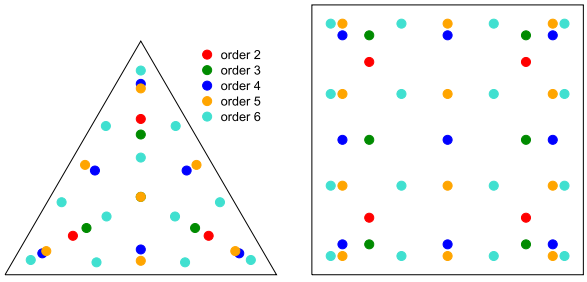

Figure 8 Quadrature points on t-faces (left) and r-faces (right) of the prism. Three and four points are used to integrate a quadratic function exactly on t-faces and r-faces, respectively, four and six for cubic, six and nine for quartic, seven and twelve for quintic, and twelve and sixteen for sextic functions. The four point rule is not used on t-faces; instead the six point rule is used for third degree polynomials

5.3 Integration on r-faces

R-faces are addressed by the edge index with specified r and ρ and approximate annular regions (trapezoids for elements of order 1). The position is specified via the (δ, ζ) pair of coordinates. We now introduce quadrature points on the reference square $(\delta, \zeta) = [0,1] \times [0,1]$ as Q_{2r}. These points are conveniently computed as tensor products of the Gauss–Legendre quadrature points. The quadrature rule for r-faces can be written as

$$\int_{\text{r-face}} \mathbf{V}(\mathbf{x}) \cdot \mathbf{dS} \approx r^2(\rho - 1) \sum_{q \in Q_{2r}} w_q \zeta'_q \mathbf{V}\left(r\zeta'_q \mathbf{x}^u_q\right) \cdot \left(\frac{\partial \mathbf{x}^u_e(\delta_q)}{\partial \delta} \times \mathbf{x}^u_q\right) \tag{20}$$

with $\mathbf{x}^u_q$ given by Eq. (19).

The right panel of Fig. 8 shows the locations of the points on the reference square. On a rectangle, three points are sufficient for exactly integrating a quadratic polynomial, four for cubic, and six for quartic. However, it is our intention to maintain exact polynomial integration rules for first

order elements, where the r-face is a trapezoid. Its Jacobian is linear in the ζ coordinate, and the order of accuracy is reduced by one. For this reason we use four, six, and nine point rules to integrate polynomials of degrees two, three, and four, respectively.

5.4 Integration on t-faces

T-faces are addressed by the face index with fixed r. They approximate spherical triangles (flat triangles for linear coordinate transformation). The position is specified via the (ξ, η) pair of coordinates, and the set of quadrature points defined on a unit equilateral triangle is designated as Q_{2t}. Here we use the symmetric quadrature rules given in Dunavant (1985) with quadrature point locations shown in the left panel of Fig. 8. The integration algorithm for t-faces is

$$\int_{\text{t-face}} \mathbf{V}(\mathbf{x}) \cdot \mathbf{dS} \approx r^2 \sum_{q\in Q_{2t}} w_q \mathbf{V}\left(r\mathbf{x}_q^u\right) \cdot \left(\frac{\partial \mathbf{x}_f^u(\xi_q,\eta_q)}{\partial \xi} \times \frac{\partial \mathbf{x}_f^u(\xi_q,\eta_q)}{\partial \eta}\right), \tag{21}$$

where

$$\mathbf{x}_q^u = \mathbf{x}_f^u(\xi_q, \eta_q). \tag{22}$$

Three, four, and six points are sufficient to integrate a quadratic, cubic, and quartic polynomial exactly on a flat triangle. The four-point rule should be avoided because it has a negative weight, and we use the six point rule at third order. These points and the normal vectors are also precomputed for each t-face.

5.5 Integration on frustums

A frustum can be addressed by the face index with specified r and ρ. Defining a position requires all three reference coordinates (ξ, η, ζ). We arrange the quadrature points in p "planes", where each plane corresponds to a triangular quadrature rule with a set of points Q_{2t} described in the previous subsection. The planes themselves are located at ζ_p corresponding to the Gauss–Legendre points on $[0, 1]$ that we designate as P_1 with the plane weights given by w_p. Then a volume integral can be evaluated as

$$\int_{\text{zone}} \mathbf{V}(\mathbf{x})\, dV \approx r^3(\rho-1) \sum_{p\in P_1} w_p \zeta_p'^2 \sum_{q\in Q_{2t}} w_q \mathbf{V}\left(r\zeta_p' \mathbf{x}_q^u\right) \left(\frac{\partial \mathbf{x}_f^u(\xi_q,\eta_q)}{\partial \xi} \times \frac{\partial \mathbf{x}_f^u(\xi_q,\eta_q)}{\partial \eta}\right) \cdot \mathbf{x}_q^u, \tag{23}$$

where $\mathbf{x}_q^u$ is given by Eq. (22). We use two quadrature planes for polynomials of degrees 0 and 1, three for polynomials of degrees 2 and 3 and four for degrees 4 and 5.

In curved spaces the total degree of the reconstruction polynomial increases significantly upon transformation to the reference coordinates. For example, a third degree polynomial in $\mathbf{x}$ on a quadratic surface element gives an integrand or degree $3^2 + 2 = 8$ in α and β, where the Jacobian adds two extra powers. The same polynomial on a cubic element gives an integrand of degree $3^3 + 4 = 13$. However, it is quite unnecessary to match the order of the quadrature algorithm to the resulting total degree of the polynomial in the reference space because the truncation error decreases at the rate imposed by the quadrature scheme alone. The magnitude of error depends on the details of the coordinate mapping, but the order of convergence does not.

6 Conservative reconstruction on a geodesic mesh

The TGM framework presented here is intended to be used primarily with finite volume schemes for systems of PDEs. These methods usually operate on conserved (extrinsic) physical variables associated with each zone in the mesh. Conserved variables are advanced in time using the fluxes evaluated at the zone boundaries. The fluxes may be generated by means of a Riemann solver that computes, often approximately, the self-similar wave pattern developed from an interaction of two or more constant states. The Riemann solver may be invoked for a set of points in each face, and the total flux is evaluated as the average over these points. The invocation of multiple Riemann solvers at suitably placed quadrature points within each face of the mesh contributes to the high order accuracy of the scheme.

The constant states fed to the Riemann solver are obtained via high-order spatial reconstruction of the conserved variables, which amounts to finding a functional form of the variable within each zone consistent with a given piecewise distribution at the beginning of the time step. Reconstruction is performed on a set of stencils associated with each zone (the principal zone of that stencil) that include zones in a certain proximity to the principal. We use conservative polynomial reconstruction (known in one-dimensional or directionally split applications as reconstruction via primitive functions) from multiple stencils for each computational zone.

6.1 Stencil construction

We now discuss the reconstruction strategy focusing on the TGM specific issues. At the start of a simulation a set of stencils is built for each computational zone. The number of zones in a stencil cannot be smaller than the number of degrees of freedom in the polynomial reconstruction, given by

$$D(M) = \frac{(M+1)(M+2)(M+3)}{6}, \tag{24}$$

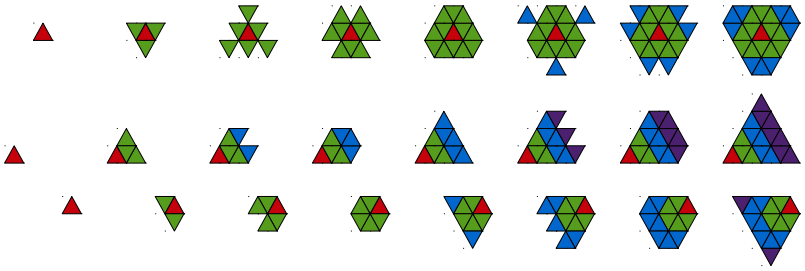

Figure 9 Stencil shapes in the plane. The top, middle, and bottom rows show the central, forward-biased, and backward-biased stencils, respectively. The principal zone is shown in red. Green, blue, and purple colors represent first, second, and third von Neumann neighbors of the principal zone, respectively

where M is the degree of the reconstruction polynomial. It has been argued that using the number of zones in the stencil equal to $D(M)$ does not always produce satisfactory results (Ollivier-Gooch and Van Altena 2002). For this reason we use over-determined stencils that are larger than the minimal size. With such stencils the conservative property of the reconstruction is enforced in the least squares sense.

The zones in the TGM are arranged in a regular pattern (see Fig. 3) allowing us to design universal stencils valid on any mesh. The zones in a stencil are arranged in "planes" that correspond to different radial shells. Each plane consists of a two-dimensional t-face stencil. The principal plane contains the largest 2D stencil and the other planes contain progressively smaller stencils. Figure 9 shows the choice of 2D stencils available in the code. In addition to the symmetric central stencil that is used in regions where the solution is smooth (top row of Fig. 9), twelve directional stencils are defined to be used in situations where the central stencil produces a large variation due to a sharp gradient or discontinuity in the solution (Käser and Iske 2005). Directional stencils can point in the in- or outward radial direction and along six directions in a plane (three forward-biased and three backward-biased, see Käser and Iske (2005) for the explanation of these terms). We use three, five, and seven planes per central stencil and two, three, and four planes per directional stencil for $M = 1$, 2, and 3, respectively. The code can use all thirteen stencils, but can also be run without backward stencils which nearly halves the execution time of the reconstruction step.

Contrary to the fully unstructured meshes, stencils on the TGM can be generated using pre-defined patterns and in principle need not rely on mesh connectivity information. The exception to this rule are the penta-corners, where some of the neighbors may be missing. Figure 10 shows some examples of stencils in the principal plane that could be used for third order polynomial reconstruction. The central stencil, shown in the top panel, clearly contains a penta-corner. The middle row shows the forward and the bottom row the backward stencils. Notice how the first of the backward stencils has a different shape than the other two. If a stencil is found to be defective (i.e., contains fewer zones than required), the software will repeatedly upgrade to the next largest stencil until the order condition is fulfilled.

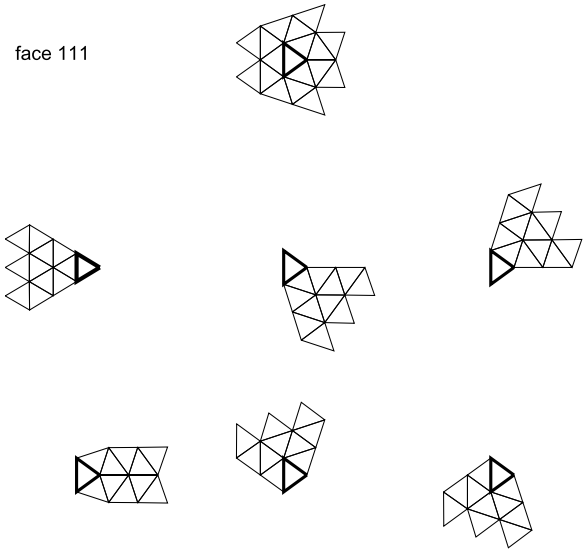

Figure 10 Stencils of one selected face at division four near a penta-corner. Shown are the central stencil (top), the forward stencils (middle row) and the backward stencils (bottom row). The principal face is drawn with thick lines. Because of the penta-corner to the right of the principal face, some of the stencils have a different shape

Consider a conservative mesh variable $\mathbf{U}$ defined via its averages over each zone i, $\bar{\mathbf{U}}_i$. A reconstruction of this variable in zone i using Mth degree polynomials can be written as

$$\mathbf{U}_i(\mathbf{x}) = \sum_{|\alpha|=0}^{D(M)-1} \mathbf{U}_i^{\alpha} \left(x^{\alpha} - \left\langle x^{\alpha} \right\rangle_i \right), \tag{25}$$

where multi-index notation is used with $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, $|\alpha| = \alpha_1 + \alpha_2 + \alpha_3$, and $x^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} x_3^{\alpha_3}$. The term $\langle x^\alpha \rangle_i$ denotes the moment of zone i, divided by the volume of the zone, and $\mathbf{U}_i^\alpha$ is the coefficient (or mode) in the reconstruction. To enforce the conservation property $\langle \mathbf{U}(\mathbf{x}) \rangle_i = \mathbf{U}_i^{(0,0,0)} = \bar{\mathbf{U}}_i$ one must formally set $\langle x^{(0,0,0)} \rangle_i = 0$ in (25). The remaining moments are computed using high order quadratures given by Eq. (23). The moments are computed in Cartesian coordinates. These moments are transformed into the center of mass frame of the zone using the parallel axis theorem (for details, see Balsara et al. 2019) and scaled by the characteristic length determined by the dimensions of the zone.

An optimal choice of stencils should achieve a balance between accuracy and performance. To find this balance we have performed a statistical study of error in the reconstruction with cubic polynomials (i.e., at fourth order of accuracy) using only the central stencil, as a function of the number of zones in the stencil. The results, presented in Appendix B, demonstrated that where as the L_∞ error can become unacceptably large for the minimal stencil, the deficiency is cured by increasing the stencil size by as little as 15%. Past this point, both the L_1 and the maximum errors have a weak increasing trend previously noted in Ivan et al. (2015). Based on these results, we introduced an adjustable parameter in the code to set the minimum number of zones in the stencil to be slightly larger than $D(M)$.

6.2 Utilizing radial similarity

A spherical mesh commonly has shell thickness varying with radial distance to satisfy the needs of the particular computational problem. Let us introduce a dimensionless variable $\chi \in [0,1]$ and a mapping $r(\chi)$ that satisfies $r(0) = r_{\min}$, $r(1) = r_{\max}$, where $r_{\min}$ and $r_{\max}$ are the inner and the outer boundaries of the entire simulation domain, not including the ghost shells. One example of such a mapping is a power law

$$r(\chi) = r_{\min}\left\{1 + \left[\left(\frac{r_{\max}}{r_{\min}}\right)^{1/b} - 1\right]\chi\right\}^b, \tag{26}$$

where b is some positive real number. The interior of the simulation domain is partitioned into L shells of equal width $\Delta\chi = L^{-1}$ that map physical shells of variable widths $\Delta r(r)$. Suppose the zone i is indexed by shell s and face f. In physical coordinates the zones corresponding to the same f but different s have different aspect ratios. For example, for the mapping (26) the zones closer to the origin will be more radially elongated than those at larger distances (for $b > 1$).

One particular function of χ preserves the zone aspect ratio, such that $\Delta r/r = \text{const}$. This is the exponential mapping,

$$r'(\chi) = r_{\min}\left(\frac{r_{\max}}{r_{\min}}\right)^\chi, \tag{27}$$

(e.g., Koldoba et al. 2002), that also satisfies $r'(0) = r_{\min}$, $r'(1) = r_{\max}$. One can then introduce exponential coordinates given by

$$x_1' = r' x_1^u, \qquad x_2' = r' x_2^u, \qquad x_3' = r' x_3^u, \tag{28}$$

where, as before, the coordinates with the superscript 'u' are measured on the unit sphere.

A conserved mesh variable $\mathbf{U}(\mathbf{x})$ is defined via

$$\int_{(i)} \mathbf{U}(\mathbf{x}) r^2\, dr\, d\Omega = \bar{\mathbf{U}}_i V_i. \tag{29}$$

Integration over the solid angle Ω corresponds to integration on the unit sphere. Equation (29) can be rewritten in exponential coordinates as

$$\int_{(i)} \mathbf{U}(\mathbf{x}) \frac{r^2}{r'^3}\frac{dr}{d\chi} r'^2\, dr'\, d\Omega = \frac{\Omega_f (r_{s+1}^3 - r_s^3)}{3} \ln\left(\frac{r_{\max}}{r_{\min}}\right)\bar{\mathbf{U}}_i, \tag{30}$$

where Ω_f is the area of face f on the unit sphere. We next introduce a three-dimensional polynomial reconstruction of the quantity $\mathbf{W} = \mathbf{U} r^2/r'^3 (dr/d\chi)$ in the zone i

$$\mathbf{W}_i(\tilde{\mathbf{x}}_s) = \sum_{|\alpha|=0}^{D(M)-1} \mathbf{W}_i^\alpha \left(\tilde{x}_s^\alpha - \langle \tilde{x}^\alpha \rangle_f\right), \tag{31}$$

where $\tilde{\mathbf{x}}_s = \mathbf{x}'/r_s'$ is the position vector expressed in the exponential normalized coordinates (ENC). Here they are normalized to the exponential distance to the bottom of the zone. The moments of any zone with a face index f are

$$\langle \cdots \rangle_f = \frac{3}{\Omega_f[(1+\Delta\tilde{r})^3 - 1)]} \times \int_1^{1+\Delta\tilde{r}} \tilde{r}^2\, d\tilde{r} \int_{(f)} (\cdots)\, d\Omega, \tag{32}$$

where

$$\Delta\tilde{r} = \frac{r_{s+1}' - r_s'}{r_s'}, \tag{33}$$

which is the same for all shells s. In the ENC the moments are independent of the shell, so the index s is dropped for them. It is evident that

$$\mathbf{W}_i^{(0,0,0)} = \frac{r_{s+1}^3 - r_s^3}{r_s'^3[(1+\Delta\tilde{r})^3 - 1)]} \ln\left(\frac{r_{\max}}{r_{\min}}\right)\bar{\mathbf{U}}_i. \tag{34}$$

To obtain the remaining modes a geometry matrix is computed for each three-dimensional stencil. Suppose S_i denotes the set of zones comprising the stencil, and that the

zone j that belongs to this stencil, $j \in S_i$, $j \neq i$ is indexed by shell σ and face ϕ. Using the fact that

$$\tilde{x}_s^\alpha = \left(\frac{r'_\sigma}{r'_s}\right)^{|\alpha|} \tilde{x}_\sigma^\alpha, \tag{35}$$

averaging (31) over a given zone in the stencil, which uses the ENC specific for its own shell σ, rather than the principal shell s, yields

$$\sum_{|\alpha|=0}^{D(M)-1} \mathbf{W}_i^\alpha \left[(1+\Delta\tilde{r})^{|\alpha|(\sigma-s)} \langle \tilde{x}^\alpha \rangle_\phi - \langle \tilde{x}^\alpha \rangle_f\right] = \mathbf{W}_j^{(0,0,0)} - \mathbf{W}_i^{(0,0,0)}. \tag{36}$$

This is a linear system for $\mathbf{W}_i^\alpha$. The geometry matrix on the LHS has the number of rows equal to the number of zones in the stencil, without counting the principal zone, and its column count is $D(M)-1$. The geometry matrix's coefficients only depend on the relative shell displacement in the stencil, $\sigma - s$, and are identical for any zone with the same face index because the corresponding stencils all have the same structure.

The advantage of the described scheme is that the amount of storage is significantly reduced (by the factor equal to the number of shells in the block) compared with the method that treats each zone as unique. Only a single copy of each moment and the geometry matrix are needed per t-face. This also permits us to precompute the LU decomposition or inverse of each geometry matrix and store it to perform reconstruction with a different RHS in (36) at each time step. The physical variable is then recovered via

$$\mathbf{U}_i(\mathbf{x}) = \mathbf{W}_i(\tilde{\mathbf{x}}_s) \frac{r'^3}{r^2} \left(\frac{dr}{d\chi}\right)^{-1}. \tag{37}$$

6.3 Limiting the reconstruction

The code performs reconstruction on all thirteen (or seven) stencils and stores the resulting modes for each stencil. The solutions from multiple stencils are combined in a nonlinear fashion into a single reconstruction polynomial using the weighted essentially non-oscillatory (WENO) method (Harten and Osher 1987; Shu and Osher 1988; Liu et al. 1994; Jiang and Shu 1995; Friedrich 1998; Balsara and Shu 2000; Dumbser and Käser 2007). The nonlinear hybridization helps to stabilize the WENO scheme when local discontinuities develop in the flow.

Suppose there are S stencils associated with face i, with the central stencil bearing the index 1, and the directional stencils numbered 2 through $S = 7, 13$. The central stencil is the most accurate and therefore carries the largest linear weight, $\gamma_1 = \in [0.85, 0.95]$, where as the remaining stencils have $\gamma_s = (1-\gamma_1)/(S-1)$. Suppose we need to perform a reconstruction of a scalar variable $U(\mathbf{x})$. The WENO procedure computes a weighted average of the reconstruction polynomials derived on each of the stencils with preference given to stencils achieving a smoother reconstruction (roughly speaking, having smaller absolute values of the modes U_{is}^α where $|\alpha| > 0$ and $s = 1, \ldots, S$). The scheme is biased by the smoothness indicators that can be estimated simply as

$$\beta_{is} = \sum_\alpha \left(U_{is}^\alpha\right)^2. \tag{38}$$

We have implemented plain second and third order WENO schemes and an adaptive order WENO-AO(4,3) scheme within the TGM framework. The plain WENO procedure computes the nonlinear weights as

$$w_{is} = \frac{\gamma_s}{(\beta_{is}+\epsilon)^2}, \tag{39}$$

where $\epsilon \sim 10^{-12}$ is used to avoid possible division by zero. The weights are then normalized so that they add up to unity. The normalized weights are obtained as

$$\bar{w}_{is} = \frac{w_{is}}{\sum_{s=1}^S w_{is}}. \tag{40}$$

The coefficients of the hybrid reconstruction polynomial are computed as

$$U_{i,\mathrm{WENO}}^\alpha = \sum_{s=1}^S \bar{w}_{is} U_{is}^\alpha. \tag{41}$$

At fourth order of accuracy we have used an adaptive order method to avoid the excessive computational cost of performing high order reconstruction on all thirteen stencils. The WENO-AO method has been described in great detail in Balsara et al. (2016), while its implementation on unstructured meshes was presented in Balsara et al. (2019, 2020). Here we only discuss some specifics of its implementation on the geodesic mesh. The WENO-AO(4,3) method uses, in addition to the set of stencils used to perform third-order reconstruction, a large central stencil that we assign the index of 0 to avoid relabeling of the third-order stencils. This large stencil is used to perform reconstruction of polynomial degree 3 and carries the linear weight $\gamma_0 = \in [0.85, 0.95]$. For example, the third order central stencil may be the stencil shown in the fourth or fifth column of Fig. 9, while the fourth-order stencil will be from column seven or eight of that figure. The linear weights γ'_s of the adaptive order scheme are given by

$$\gamma'_0 = \gamma_0, \qquad \gamma'_1 = (1-\gamma_0)\gamma_1, \qquad \gamma'_s = \frac{(1-\gamma_0)(1-\gamma_1)}{S-1}, \quad s = 2, \ldots, S \tag{42}$$

(note that the number of stencils used in this case is $S+1$). The smoothness indicators and nonlinear weights are obtained according to (38) and (39), respectively using γ_s' in place of γ_s, where $s = 0, \ldots, S$. The normalized nonlinear weights are given by

$$\bar{w}_{is} = \frac{w_{is}}{\sum_{s=0}^{S} w_{is}}. \tag{43}$$

The coefficients of the hybrid reconstruction polynomial in the adaptive case are computed as

$$U^{\alpha}_{i,\text{WENO-AO}} = \frac{\bar{w}_{i0}}{\gamma_0'}\left(U^{\alpha}_{i0} - \sum_{s=1}^{S}\gamma_s' U^{\alpha}_{is}\right) + \sum_{s=1}^{S}\bar{w}_{is}U^{\alpha}_{is}. \tag{44}$$

Expression (44) reduces to U^{α}_{i0} in the limit that the solution is smooth on all stencils and therefore $\bar{w}_{is} \to \gamma_s'$. This choice yields the most accurate reconstruction because it is based entirely on the large central stencil.

The reconstruction procedure is carried out in each zone lying in the interior of the block and in two more layers of ghost zones. The latter is needed by the slope flattening procedure that scales down the reconstruction coefficients within the zones lying near strong density enhancements. The stencils shown in Fig. 9 extend a distance equal to the degree of the reconstruction polynomial beyond the principal zone. As a result we use three layers of ghost zones at second order of accuracy, four at third order and five at fourth order.

7 Constrained reconstruction of the magnetic field

For MHD problems, it is essential to keep the magnetic field divergence free. The most successful technique to maintain $\nabla \cdot \mathbf{B} = 0$ is the constrained transport method (Evans and Hawley 1988; DeVore 1991; Ryu et al. 1998; Balsara and Spicer 1999) that is based on the Yee type staggered mesh. In this approach the magnetic field is a face based variable, unlike the zone averaged mass, momentum, and total energy conserved variables. More specifically, the variable is a normally projected, face averaged value of the magnetic field that will be called $\bar{B}$, possibly with a subscript of the face where it is defined. This magnetic field is initialized using the vector potential

$$\mathbf{B} = \nabla \times \mathbf{A}, \tag{45}$$

and is updated in time via Faraday's law,

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}, \tag{46}$$

where $\mathbf{E}$ is the electric field and SI units are used. Integrating equations (45) and (46) requires edge based vector potential and electric field, respectively, in applying the Stokes theorem.

Let us focus on a single zone with index i in the mesh. Denote by F_i the set of faces that belong to this zone. The set can be further partitioned into three r-faces (set R_i) and two t-faces (set T_i). By convention, the normals $\hat{\mathbf{n}}_j$ for $j \in R_i$ are directed outward as viewed from a zone corresponding to an unshaded t-face (and hence inward as viewed from a shaded face, see Fig. 3), where as the normals for $j \in T_i$ always point in the outward direction (direction of increasing r). Further, suppose E_j is the set of edges that comprise the boundary of face j. For $j \in R_i$, the boundary consists of two t-edges and two r-edges; while faces $j \in T_i$ have three t-edges. The tangent vectors to the t-edges are assumed to be directed counter-clockwise relative to the unshaded face while the r-edge tangents point outward.

Using the above conventions, the face-based magnetic field initialization procedure is written as

$$\bar{B}_j S_j = \iint_{\text{face } j} \mathbf{B}\cdot\mathbf{dS} = \sum_{k\in E_j}\int_{\text{edge } k}\mathbf{A}\cdot\mathbf{dl} = \sum_{k\in E_j}\bar{A}_k l_k, \tag{47}$$

where S_j is the area of face j, l_k is the length of the edge k, and $\bar{A}_k$ is the average over the edge k of the vector potential dotted with the tangent vector to that edge. The line integral in (47) is evaluated using formulae (16) and (18). In addition, the integral divergence free condition for $\mathcal{D} = \nabla \cdot \mathbf{B}$ may be written as

$$\bar{\mathcal{D}}_i V_i = \sum_{j\in F_i}\iint_{\text{face } j}\mathbf{B}\cdot\mathbf{dS} = \sum_{j\in F_i}\bar{B}_j S_j = 0, \tag{48}$$

where V_i is the volume of zone i. In practice, the numerical code defines variables of zone, face, and edge types and the curl and divergence integral operations to perform "conversions" between the types.

Following Balsara and Dumbser (2015a) the model presented here uses a supplementary zone based vector variable $\mathbf{B}'$. At the start of the simulation, this variable must be initialized in each zone i in some way consistent with the primary field $\bar{B}$ defined on F_i. One possibility is to use the least squares fit

$$\iint_{\text{face } j}\mathbf{B}_i'\cdot\mathbf{dS} = \bar{B}_j S_j, \quad j \in F_i. \tag{49}$$

The integral in the above equation is evaluated using (20) and (21), giving five equations (one per face) for the three unknown field components. The alternative is to initialize $\mathbf{B}'$ directly as a zone variable using the expression for the field rather than the potential. The resulting $\mathbf{B}'$ is subsequently treated like any other zone variable. In particular, it is subjected to the same volume reconstruction procedure described in the previous section. This reconstruction is not functionally divergence free, and an additional procedure, described below, is applied to obtain a constrained

reconstruction. This approach represents a low computational cost alternative to a face based reconstruction.

Suppose the preliminary, non-divergence-free reconstruction, computed as discussed in the previous section, is given by

$$\mathbf{B}'_i(\mathbf{x}) = \sum_{|\alpha|=0}^{D(M)-1} \mathbf{B}'^{\alpha}_i \left(x^\alpha - \langle x^\alpha \rangle_i\right), \tag{50}$$

where $\mathbf{B}'^{\alpha}_i$ are the modes. We seek a constrained polynomial reconstruction for the magnetic field $\tilde{\mathbf{B}}(\mathbf{r})$ as

$$\begin{aligned}
\tilde{B}_{1i}(\mathbf{x}) &= \sum_{\substack{|\alpha|=0 \\ \alpha_2,\alpha_3\le M}}^{D(M+1)-1} \tilde{B}^{\alpha}_{1i}\left(x^\alpha - \langle x^\alpha \rangle_i\right), \\
\tilde{B}_{2i}(\mathbf{x}) &= \sum_{\substack{|\alpha|=0 \\ \alpha_1,\alpha_3\le M}}^{D(M+1)-1} \tilde{B}^{\alpha}_{2i}\left(x^\alpha - \langle x^\alpha \rangle_i\right), \\
\tilde{B}_{3i}(\mathbf{x}) &= \sum_{\substack{|\alpha|=0 \\ \alpha_1,\alpha_2\le M}}^{D(M+1)-1} \tilde{B}^{\alpha}_{3i}\left(x^\alpha - \langle x^\alpha \rangle_i\right).
\end{aligned} \tag{51}$$

These reconstructions have $\tilde{D}(M) = 2D(M) - D(M-1)$ degree of freedoms, which is larger than $D(M)$. While the degree of the reconstruction polynomials (51) is one higher than of (50), not every additional high order mode is present. The need for the extra modes will be demonstrated shortly. We now describe the five separate constraints imposed on the magnetic field modes that ensure that the magnetic field remains divergence-free not only in the integral sense (zero total flux through all faces of a zone), but also functionally at any location within the zone.

7.1 Constraint 1

This step ensures that the polynomial reconstruction of the magnetic field has zero divergence everywhere in the zone. Taking the divergence of Eq. (51) and making the resulting polynomial expression equal to zero yields $D(M)$ equations of the form

$$\alpha_1 \tilde{B}^{\alpha_1}_{1i} + \alpha_2 \tilde{B}^{\alpha_2}_{2i} + \alpha_3 \tilde{B}^{\alpha_3}_{3i} = 0. \tag{52}$$

Clearly, $\tilde{B}_1$, $\tilde{B}_2$, and $\tilde{B}_3$ modes with $\alpha_1 = 0$, $\alpha_2 = 0$, and $\alpha_3 = 0$, respectively, do not contribute to (52). Only the extra modes that contain powers of x_1 for $\tilde{B}_1$, x_2 for $\tilde{B}_2$, and x_3 for $\tilde{B}_3$ are included. For instance, at third order of accuracy ($M = 2$) the extra modes present in the first equation of (51) are those containing x_1^3, $x_1^2x_2$, $x_1^2x_3$, $x_1x_2^2$, $x_1x_2x_3$, and $x_1x_3^2$, where as the second equation includes $x_1^2x_2$, $x_1x_2^2$, $x_1x_2x_3$, x_2^3, $x_2^2x_3$, and $x_2x_3^2$ terms. The remaining high order modes do not contribute to the local divergence-free conditions.

7.2 Constraint 2

The second constraint imposed on the reconstruction (51) is the requirement that its normal component, evaluated from any two adjacent zones sharing the face j and averaged over that face must be equal to $\bar{B}_j$, namely

$$\iint_{\text{face } j} (\tilde{B}_{1i}\, dS_1 + \tilde{B}_{2i}\, dS_2 + \tilde{B}_{3i}\, dS_3) = \bar{B}_j, \tag{53}$$

where the integral is evaluated according to the rules (20)–(21). This is the requirement of zero divergence in the integral sense. The order of the quadrature rule need not be very high, but only sufficient to match the order of the overall scheme. For example, for a third order scheme that uses polynomials of up to third degree, we use six point quadratures on all faces.

It should be pointed out that because of (53) one constraint in (52) is redundant. This is readily demonstrated by computing the divergence of $\tilde{\mathbf{B}}$ (Eq. (51)) analytically, integrating over the volume of the zone, and setting the integral to zero. For the sake of symmetry, we chose to discard the first equation in (52), so that system's equation count is reduced to $D(M) - 1$.

7.3 Constraint 3

Balsara and Dumbser (2015a) proposed a method seeking to match, at each face, complete polynomial reconstructions of the normal component of the magnetic field. Here we use a weaker requirement that the reconstructions of the normal component should approximately match at the facial quadrature points used to perform integration on that face. This procedure nonetheless ensures a very close matching of the modes of the magnetic field within each face.

The matching procedure starts by evaluating $\mathbf{B}'_i(\mathbf{x})$ from (51) at each quadrature point of face $j \in F_i$ and projecting it onto the unit normal to the face at that point. Initially this normal component is not continuous at the zone boundaries, so there are two values of the normal component, B_{iq} and B_{kq}, at each facial quadrature point q contributed by two adjacent zones i and k, where $j \in F_i, F_k$. The common normal component at each quadrature point q, B_q, is evaluated in two steps as

$$B^*_q = \bar{B}_q + \text{minmod}(B_{iq} - \bar{B}_j, B_{kq} - \bar{B}_j), \tag{54}$$

$$B_q = B^*_q - \langle B^* \rangle_j + \bar{B}_j, \tag{55}$$

where B^* is the intermediate value of the common normal component of the field at the interface and the angular brackets denote its average over the face j. The normal component of the magnetic field given by (55) is continuous and its average over the face matches the respective

value of the primary variable $\bar{B}_j$. Therefore, the third set of constraints can be written as

$$\tilde{B}_{1i}(\mathbf{x}_q)\hat{n}_1 + \tilde{B}_{2i}(\mathbf{x}_q)\hat{n}_2 + \tilde{B}_{3i}(\mathbf{x}_q)\hat{n}_3 \approx B_q, \tag{56}$$

for each quadrature point q, with $j \in F_i$. The number of conditions in (56) is equal to the total number of quadrature points on all five faces of the frustum.

7.4 Constraint 4

Next, we demand that the divergence-free reconstruction (51) should be as close to the volume reconstruction of $\mathbf{B}'$ as possible, i.e.,

$$\tilde{B}^{\alpha}_{1i} \approx B'^{\alpha}_{1i}, \quad \tilde{B}^{\alpha}_{2i} \approx B'^{\alpha}_{2i}, \quad \tilde{B}^{\alpha}_{3i} \approx B'^{\alpha}_{3i}, \quad |\alpha| < D(M). \tag{57}$$

Equation (52) is based on the observation that the initial (unconstrained) volume reconstruction is the best possible starting point for determining the constrained modes. With this condition the convergence order of the constrained reconstruction stays close to the order of convergence of the unconstrained volume reconstruction.

7.5 Constraint 5

In the same spirit it is desirable that the "extra" high order modes should be small, i.e.,

$$\tilde{B}^{\alpha}_{1i} \approx 0, \qquad \tilde{B}^{\alpha}_{2i} \approx 0, \qquad \tilde{B}^{\alpha}_{3i} \approx 0, \quad |\alpha| \geq D(M). \tag{58}$$

Table 4 provides the counts of the degrees of freedom and the number of equations contributed by formulae (52), (53), (56), (57), and (58) for schemes of second, third, and fourth orders of spatial convergence. Since there are more equations than unknowns, only the local and global divergence-free conditions (52) and (53) are strictly enforced; the remaining conditions can only be satisfied approximately, in the least squares sense (in principle, at third and fourth order of accuracy the constraints (56) can also be strictly imposed). This constitutes a constrained linear least square (CLSQ) problem. Figure 11 illustrates the structure of the LLS and constraints matrices at fourth order. From Table 3, the rank of the Karush–Kuhn–Tucker (KKT) matrix of the CLSQ problem is 29, 62, and 114 at second, third, and fourth order of accuracy. Note that despite the sparsity of the LLS and the constraint matrices, the KKT matrix is largely dense.

Table 4 The number of unknowns (the "Unknowns" column) vs. the number of conditions of each type (C1 through C5) in magnetic field reconstruction

Order	$D(M)$	$\bar{D}(M)$	Unknowns	C1	C2	C3	C4	C5
2	4	7	21	3	5	15	12	9
3	10	16	48	9	5	24	30	18
4	20	30	90	19	5	30	60	30

Based on the results of the previous section, it may be expected that only a single KKT matrix needs to be constructed and inverted per t-face. Unfortunately, the difficulty here is with the global divergence-free condition (Constraint 1), which is, in general, incompatible with the reconstruction (31). Coordinate factorization is still possible, but only if the mesh is directly exponentially rationed, i.e., $r = r'$ and $\mathbf{W} = \ln(r_{\max}/r_{\min})\mathbf{U}$. This is the mesh that was used for all MHD applications discussed below.

At the end of the magnetic field reconstruction step, the previously obtained unconstrained modes are discarded and replaced with the constrained version. This ensures synchronization between the primary and supplementary magnetic field variables used by the code.

8 Time advance and boundary exchange

The complete finite volume method is implemented as follows. First, the zone-based variables (including $\mathbf{B}'$) are reconstructed to the quadrature points on the faces as described in the previous two sections. Pairs of states from each side of the interface are fed into a Riemann solver. We employ the HLL family of nonlinear solvers (Harten et al. 1983; Einfeldt 1988) that are very robust and usually positivity preserving as long as the speeds of the extremal waves are properly estimated. The popular HLLC solver (Batten et al. 1997; Gurski 2004; Li 2005) consists of four states separated by two fast shocks and a tangential discontinuity. The HLLD solver (Miyoshi and Kusano 2005) adds a pair of rotational discontinuities, and is therefore less dissipative than HLLC, but is somewhat less robust and can fail for certain combinations of input states. Our approach is to start with the least diffusive solver, downgrading to the more dissipative solver when the former fails to deliver a positive resolved state. The fluxes are evaluated at each quadrature point and combined together to obtain the total flux through a face. These fluxes update the conserved variables in the zone using a TVD Runge–Kutta scheme (Shu and Osher 1988). A version of the code is also available that uses the so-called arbitrary derivative (ADER) update technique (Dumbser et al. 2008; Balsara et al. 2009). The ADER implementation on the geodesic mesh has been reported elsewhere (Balsara et al. 2019).

Unlike the zone variables, the magnetic fields are reconstructed to the quadrature points lying on the *edges*. A single point is sufficient at second order and two points at third and fourth orders. The constrained magnetic field is used here in place of the non-constrained reconstruction. Each t-edge receives four states and each r-face five or six states depending on whether it is a penta-corner or not. These states are fed into a multi-state two-dimensional Riemann solver (Balsara 2010, 2012, 2014; Balsara and Dumbser 2015b) generating the electric field at the edges (the remaining flux components are discarded). The 2D Riemann solver used here is of the HLLI type (Dumbser and Balsara 2016) that can include every MHD wave,

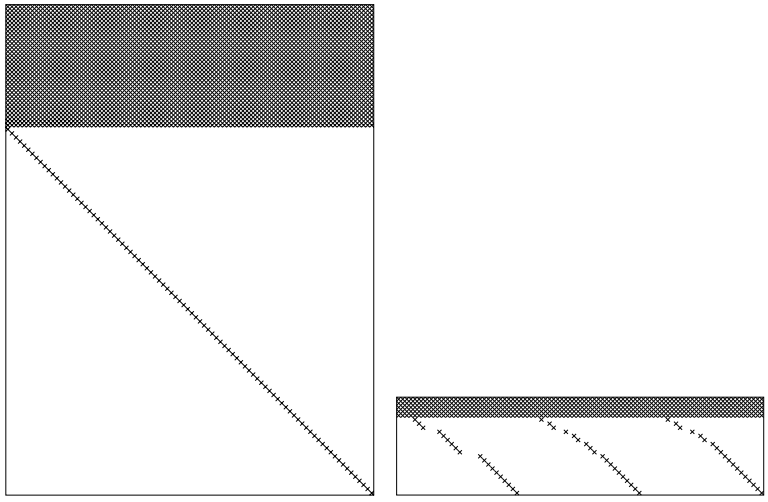

Figure 11 Matrix structure for divergence-free reconstruction. The left panel shows the nonzero elements (marked with x's) of the least squares matrix and the right panel shows the same for the constraint matrix for fourth order reconstruction. The corresponding KKT matrix is 114 × 114

including the Alfvén and slow magnetosonic waves. The face-based magnetic field is updated via the same Runge–Kutta procedure. This operation conserves the divergence of **B** to the machine precision.

A correct implementation of the above scheme must ensure that all variables are properly synchronized at the block boundaries. Each block can have up to 38 neighbors which at some point in the calculation must send some of their zone, face, or edge based boundary data and received equivalent data in return to fill in the ghost mesh element or synchronize the common boundaries. The implementation described here does not use neighbor lists, instead delegating all bookkeeping tasks to the message passing library.

Figure 12 demonstrates the typical mesh topology. Ten out of 20 blocks are shown in this cutout view, shaded using different colors. This corresponds to the smallest decomposition of the computational domain, confined between two concentric spheres with $r_{\max}/r_{\min} = 2$, and using a single slab.

To formalize our communication strategy we define the concept of "exchange site" that could be a face, edge, or corner (vertex) of the block. Each exchange site maintains a list of blocks that share the site. The list consists of two elements for any face site, four for a t-edge, six or five for an r-edge, and twelve or ten for a vertex site. Each site further defines a number of exchanges that occur at the site as lists of participant blocks. A block maintains a list of exchanges it needs to perform during a time step and its own order in that exchange. All exchanges of the same kind are started at once on every participating block in the non-blocking regime; we therefore rely on the message passing library's own scheduling facilities to achieve optimal utilization of the interconnection network.

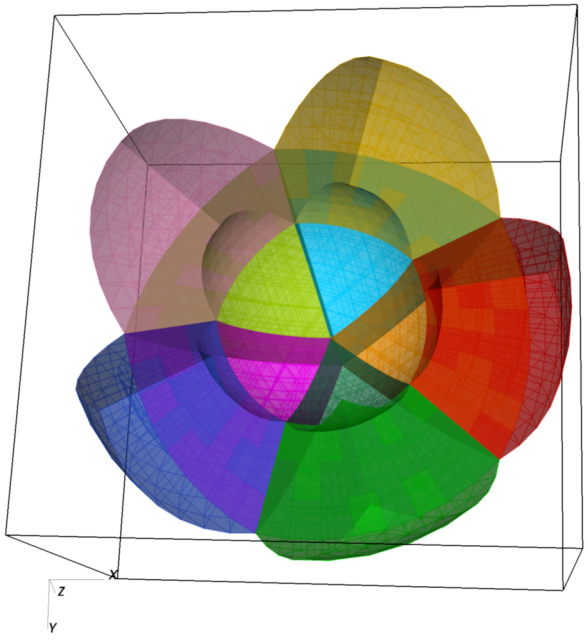

Figure 12 Cutout view of the mesh showing ten blocks out of 20 (the smallest number of blocks on this mesh, consisting of a single slab and 20 division 0 sectors). Each block is shaded using a unique color

A block maintains a set of buffers and corresponding rules to pack a part of the block destined for exchange into contiguous memory of the buffer as well as the inverse (unpack) operation. Because neighboring blocks can have any of the three possible orientations, packing and unpacking

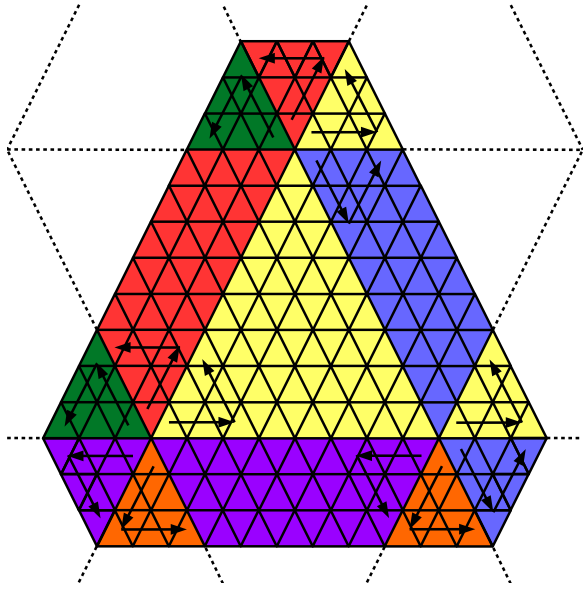

Figure 13 Neighbor arrangement around a block. The pairs of arrows indicate the directions of the first and the second TAS coordinates, respectively. The different rotations of the TAS are shown with different color shading

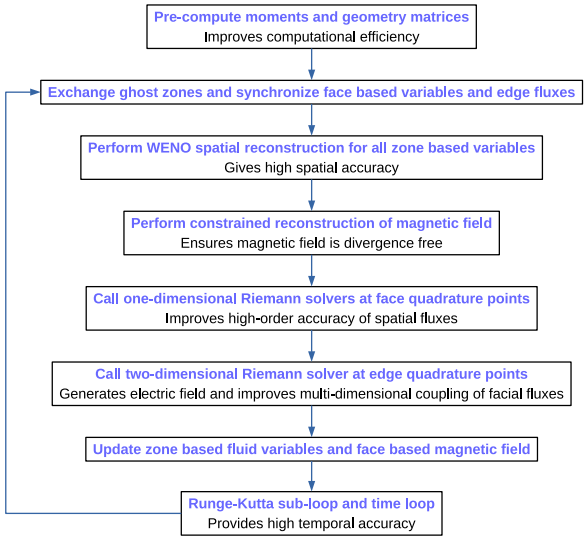

Figure 14 Sequence of steps in one iteration of the time loop

must be done in a way that is independent of the choice of the base vertex. Care must also be taken to synchronize the variables at locations that are shared among blocks, such as face-based magnetic field values and fluxes, and edge-based electric fields. This synchronization is needed to eliminate possible divergence between neighboring blocks owing to roundoff errors.

Figure 13 shows the block surrounded by twelve neighbors that belong to the same slab. The large yellow triangle is the interior area of the block, while the surrounding smaller trapezoidal or triangular areas represent the *receive buffers* that correspond to the ghost zones of the block. To extract the data received from each neighbor from the corresponding buffer requires rotated TAS coordinates that are represented in Fig. 13 by pairs of black arrows showing the directions of the first and the second TAS coordinates, respectively. The convention for packing and unpacking a trapezoid is that the principal vertex is in the lower left corner with the trapezoid resting on its wider base. For small triangles, the principal vertex is the vertex shared with the block's interior. This convention automatically ensures that unpacking of a buffer is done in the same order as it was packed by the neighbor block.

Figure 14 shows the structure of the complete simulation loop based on the Runge–Kutta time advance. The initial setup involving pre-computing the geometry matrices is time consuming, but the subsequent computation is sped up dramatically as a result.

The production version of the code was written in Fortran, and a development version writen in C++ is also available. The input and output is handled by the open source SILO library (https://wci.llnl.gov/simulation/computer-codes/silo). The library features a simple parallel I/O implementation that groups the blocks (which could number in the hundreds of thousands) into a smaller number of SILO files. An assembly file is then generated describing the relationship between the blocks for the visualizer. We use the VisIt visualizer (https://wci.llnl.gov/simulation/computer-codes/visit) for 3D rendering of the model output; several of the figures in this paper were produced with VisIt.

9 Numerical tests

Here we present two simple tests validating the accuracy of the model. The first test in the "manufactured" solution of Ivan et al. (2013) that describes an interaction between a point source and a uniform flow; which is the most basic model of an astrosphere (an interface produced by a stellar wind expanding into a moving interstellar medium). This steady state, current-free field-aligned flow is given by

$$\rho = \rho_0 \left(\frac{r_0}{r}\right)^{5/2}, \tag{59}$$

$$\mathbf{u} = \frac{u_0 \mathbf{x}}{(r_0 r)^{1/2}} + u_1 \left(\frac{r}{r_0}\right)^{5/2} \hat{\mathbf{e}}_3, \tag{60}$$

$$p = p_0 \left(\frac{r_0}{r}\right)^{5/2}, \tag{61}$$

$$\mathbf{B} = \frac{B_0 r_0^2 \mathbf{x}}{r^3} + \frac{B_0 u_1}{u_0} \hat{\mathbf{e}}_3. \tag{62}$$

The source terms corresponding to Eqs. (59)–(62) that appear in the conservative MHD equations are

$$Q_\rho = 0, \tag{63}$$

$$\mathbf{Q}_u = \left[\rho_0 u_0\left(\frac{u_0}{r} - \frac{u_1 z}{r^2}\right) - \frac{5p_0 r_0}{r^2}\right]\frac{r_0^{3/2}\mathbf{x}}{2r^{5/2}} + \left(7u_0 + \frac{5u_1 zr}{r_0^2}\right)\frac{\rho_0 u_1 \hat{\mathbf{e}}_3}{2(r_0 r)^{1/2}}, \tag{64}$$

$$Q_e = \frac{\rho_0 u_0^2}{2r}\left(\frac{u_0 r_0}{r} + \frac{7u_1 z}{r_0}\right) + \frac{\rho_0 u_0 u_1^2(7r^2 + 4z^2)}{r_0^3} + \frac{5\rho_0 u_1^3 z r^3}{2r_0^5}, \tag{65}$$

$$\mathbf{Q}_B = 0. \tag{66}$$

The analytic solution is independent of the adiabatic index γ. Following Ivan et al. (2013), the zero-subscripted constants are all set to unity, $u_1/u_0 = 0.017$, $\gamma = 1.4$, and the simulation is performed in the region between $r_{\min} = 2$ and $r_{\max} = 3.5$. The radial shell width has an exponential dependence on r. Figure 15 shows the rate of convergence for the L_1 and L_∞ norms of the error in the density, total energy and one component of the magnetic field. Simulations were performed on division 5 through division 8 meshes with 32 through 256 radial shells and the same number of zones per sector side.

As is demonstrated in Fig. 15, the L_1 error decreases at the nominal convergence rate of the scheme in each case. The L_∞ norm displays the nominal convergence rate at second order, but decreases slower than the nominal rate and third and fourth order. This is further quantified in Tables 5 through 7 that show the numerical values of the order of convergence for the manufactured solution problem. The rates for density and energy (both zone based variables) are very similar, while magnetic field shows a different behavior. The imposition of constraints described in Sect. 7 affects the accuracy of reconstruction. It seems to

Table 5 Actual order of convergence for the manufactured solution problem using the nominally second order scheme. Density (ρ), total energy (e), and one component of magnetic field (B_x) are shown

	ρ			e			B_x		
Division	5	6	7	5	6	7	5	6	7
L_1	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
L_∞	2.0	2.0	1.8	2.0	2.0	2.0	1.8	1.8	1.8

Table 6 Actual order of convergence for the manufactured solution problem using the nominally third order scheme. Density (ρ), total energy (e), and one component of magnetic field (B_x) are shown

	ρ			e			B_x		
Division	5	6	7	5	6	7	5	6	7
L_1	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
L_∞	3.0	2.7	2.6	3.0	2.7	2.5	2.3	2.3	2.3

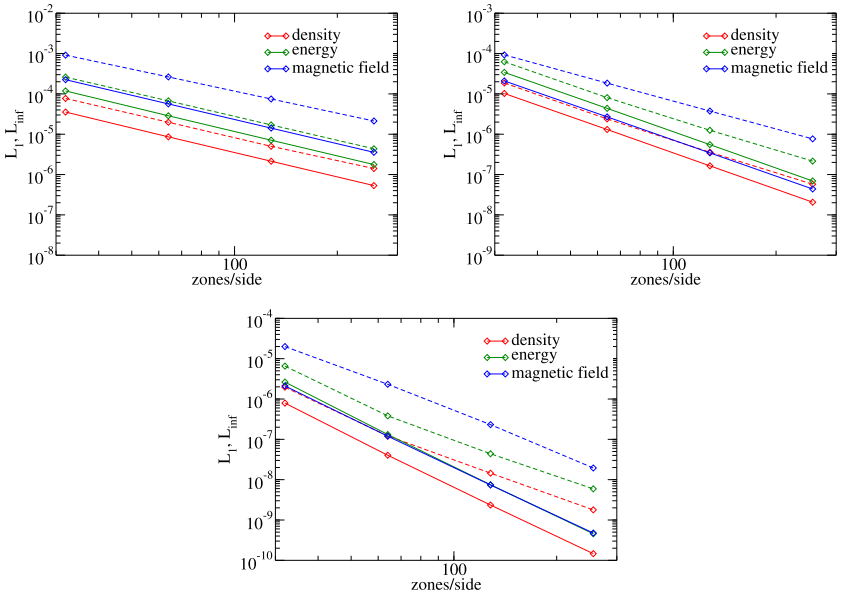

Figure 15 L_1 (solid lines) and L_∞ (dashed lines) norms of the error of density, total energy, and the x component of the magnetic field for the manufactured solution on division 4–7 TGM. The three panels correspond to second through fourth order schemes

be detrimental at lower order of accuracy but is surprisingly beneficial at fourth order.

Figure 16 shows the velocity magnitude (left panel) and the distribution of the error on the sphere at $r = 2.75$ for a simulation on a division 6 mesh using the fourth order scheme. The flow geometry resembles that of a potential flow of gas around a point source, although the velocity field is not irrotational here. The right panel shows the error distribution in one spherical layer. In common with other geodesic meshes, the error distribution shows a distinctive imprint of the mesh. The errors are the largest near the penta-corners and at the boundaries of division 0 and division 1 sectors. Similar phenomena have been reported by Tomita et al. (2001), Weller et al. (2012) and Peixoto and Barros (2013) in the context of the shallow water equations.

It is expected that the error becomes more concentrated near singular points with increasing refinement. For any division mesh, only 60 zones have a singular point as a vertex. Because the ratio of the number of large error zones to the total number of zones decreases with increased resolution, the L_1 norm is not affected even if the convergence order in high error zones is one lower than elsewhere in the mesh; this is supported by the numbers from Table 7.

Table 7 Actual order of convergence for the manufactured solution problem using the nominally fourth order scheme. Density (ρ), total energy (e), and one component of magnetic field (B_x) are shown

	ρ			e			B_x		
Division	5	6	7	5	6	7	5	6	7
L_1	4.3	4.1	4.0	4.3	4.1	4.0	4.1	4.0	4.0
L_∞	4.0	3.1	3.0	4.1	3.1	2.9	3.1	3.3	3.6

The second test is a time-dependent blast problem from (Florinski et al. 2013). The initial conditions are piecewise constant within each of the two concentric shells, $r_{\min} \le r \le r_1$ and $r_1 \le r \le r_{\max}$. Both states have $\rho_0 = 1$ and $\mathbf{u}_0 = 0$, while the pressure is set to $p_0 = 10$ ($r < r_1$) and $p_0 = 0.1$ ($r > r_1$). The initial magnetic field is a superposition of a dipole and a uniform fields,

$$\mathbf{B} = \mathbf{B}_0\left(1 + \frac{r_0^3}{2r^3}\right) - \frac{3r_0^3(\mathbf{B}_0 \cdot \mathbf{x})\mathbf{x}}{2r^5}. \tag{67}$$

We set $\mathbf{B}_0 = 10/\sqrt{3}(\hat{\mathbf{e}}_1 + \hat{\mathbf{e}}_2 + \hat{\mathbf{e}}_3)$, $\gamma = 1.4$, and $r_1 = 0.1$. The simulation was performed in the region between $r_{\min} = r_0 = 0.01$ and $r_{\max} = 0.5$ until the time $t = 0.07$ with third order reconstruction on a division 6 mesh with 256 exponentially spaced radial shells. A reflective condition was used at the internal boundary and the fixed initial state maintained at the external boundary.

The magnetic field obtained for this problem is shown in Fig. 17. The flow consists of a fast shock wave and two dense shells of material elongated along the magnetic field. The result is in excellent agreement with that reported in Florinski et al. (2013).

10 Summary

This paper documents many of the original techniques and innovations that were incorporated into our newly developed computational model for MHD equations based on an icosahedral triangular geodesic mesh. The new geodesic framework features numerous improvements compared with our earlier icosahedral hexagonal model reported in Florinski et al. (2013) that was used successfully to simulate the interaction between the solar wind and the surrounding interstellar medium (Guo and Florinski 2016; Guo et al. 2018). These improvements are summarized below.

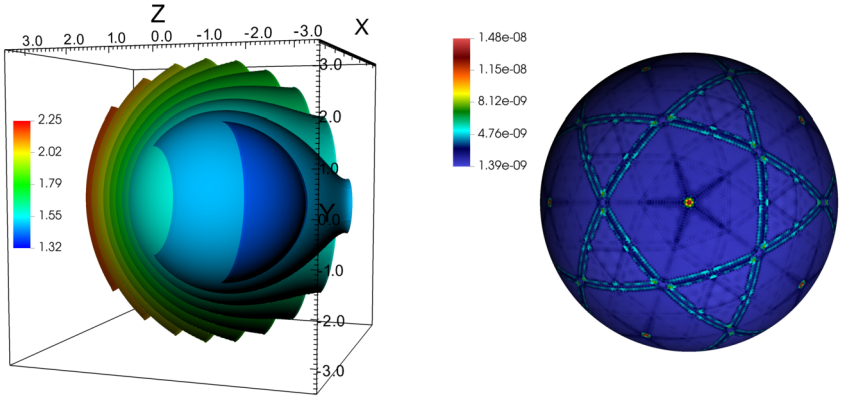

Figure 16 Left: velocity magnitude isosurfaces from the manufactured solution problem. The solution has a resemblance to an interaction between a stellar wind an a uniform flow. A fourth order scheme was used on division 6 mesh with 64 radial shells. Right: density error distribution in a spherical layer at $r = 2.75$

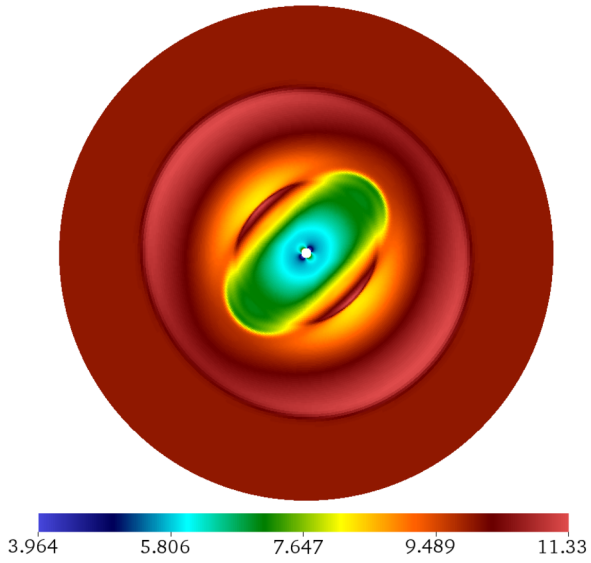

Figure 17 Magnitude of the magnetic field from the solution to the blast wave problem at $t = 0.07$. A linear color scale is used

Triangle based mesh. Using triangles instead of hexagons paved the way to efficient decomposition of the computational domain into sectors in addition to radial shells (shell decomposition was the only one available in Florinski et al. (2013)). As a result, the new code scales up to tens of thousands of CPU cores with almost linear weak scaling (Balsara et al. 2019). The second advantage of the TGM is that it is amenable to adaptive mesh refinement, which is not possible with a hexagonal mesh.

Increased order of accuracy. The new framework provides second, third, and fourth orders of accuracy for the MHD equations. High accuracy was achieved by using larger stencils and multiple families of stencils including symmetric central and asymmetric directional stencils. This is a major improvement over our earlier geodesic model that was only second order capable. The only other fourth order geodesic mesh MHD model we are aware of was reported in Ivan et al. (2015); however it was based on a cube-sphere rather than a TGM.

Accurate representation of spherical geometry. The new framework uses linear, quadratic, and cubic Lagrangian basis function to perform coordinate transformations from a reference element (a right prism) to the physical computational zone. These maps are based on the serendipity family of triangular finite elements with three, six, and ten nodes, respectively. This approach allows a very accurate representation of the spherical surface without the drawback of dealing directly with spherical coordinates. While the accuracy of the geometry representation does not improve the convergence order of the scheme, it could be potentially very important for the models of thin shells, such as planetary atmospheres.

Divergence free MHD. The new model features the first implementation of the constraint transport method on a geodesic mesh. In this approach the magnetic field is maintained divergence free because of exact cancellation of all contributions to divergence in a zone during the time update. In addition, the model features pointwise and functional divergence-free reconstruction of the magnetic field.

Implementation flexibility. All geodesic meshes are based on the same set of underlying principles used in mesh generation and spatial reconstruction. We have found that the present framework can be adapted, with a very limited number of changes, to build a model around any of the five regular polyhedra. We have developed an initial implementation of the geometry framework component for the hexahedral QGM. This will eventually permit a direct comparison between geodesic meshes of different types.

Appendix A: Basis functions for triangular faces

Linear elements: nodal points at vertices,

$$\begin{aligned}
\psi_1 &= 1 - \xi - \frac{1}{\sqrt{3}}\eta, \\
\psi_2 &= \xi - \frac{1}{\sqrt{3}}\eta, \\
\psi_3 &= \frac{2}{\sqrt{3}}\eta, \\
\phi_1 &= 1 - \delta, \\
\phi_2 &= \delta.
\end{aligned} \tag{68}$$

Quadratic elements: nodal points at vertices and edge midpoints,

$$\begin{aligned}
\psi_1 &= 1 - 3\xi - \sqrt{3}\eta + 2\xi^2 + \frac{4}{\sqrt{3}}\xi\eta + \frac{2}{3}\eta^2, \\
\psi_2 &= -\xi + \frac{1}{\sqrt{3}}\eta + 2\xi^2 - \frac{4}{\sqrt{3}}\xi\eta + \frac{2}{3}\eta^2, \\
\psi_3 &= -\frac{2}{\sqrt{3}}\eta + \frac{8}{3}\eta^2, \\
\psi_4 &= \frac{8}{\sqrt{3}}\xi\eta - \frac{8}{3}\eta^2, \\
\psi_5 &= \frac{8}{\sqrt{3}}\eta - \frac{8}{\sqrt{3}}\xi\eta - \frac{8}{3}\eta^2, \\
\psi_6 &= 4\xi - \frac{4}{\sqrt{3}}\eta - 4\xi^2 + \frac{4}{3}\eta^2, \\
\phi_1 &= 1 - 3\delta + 2\delta^2, \\
\phi_2 &= -\delta + 2\delta^2, \\
\phi_3 &= 4\delta - 4\delta^2.
\end{aligned} \tag{69}$$

Cubic elements: nodal points at vertices, edge thirds, and geometric center,

$$\psi_1 = 1 - \frac{11}{2}\xi - \frac{11}{2\sqrt{3}}\eta + 9\xi^2 + 6\sqrt{3}\xi\eta + 3\eta^2 - \frac{9}{2}\xi^3 - \frac{9\sqrt{3}}{2}\xi^2\eta - \frac{9}{2}\xi\eta^2 - \frac{\sqrt{3}}{2}\eta^3,$$

$$\psi_2 = \xi - \frac{1}{\sqrt{3}}\eta - \frac{9}{2}\xi^2 + 3\sqrt{3}\xi\eta - \frac{3}{2}\eta^2 + \frac{9}{2}\xi^3 - \frac{9\sqrt{3}}{2}\xi^2\eta + \frac{9}{2}\xi\eta^2 - \frac{\sqrt{3}}{2}\eta^3,$$

$$\psi_3 = \frac{2}{\sqrt{3}}\eta - 6\eta^2 + 4\sqrt{3}\eta^3,$$

$$\psi_4 = -3\sqrt{3}\xi\eta + 3\eta^2 + 9\sqrt{3}\xi^2\eta - 18\xi\eta^2 + 3\sqrt{3}\eta^3,$$

$$\psi_5 = -3\sqrt{3}\xi\eta + 3\eta^2 + 18\xi\eta^2 - 6\sqrt{3}\eta^3,$$

$$\psi_6 = -3\sqrt{3}\eta + 3\sqrt{3}\xi\eta + 21\eta^2 - 18\xi\eta^2 - 6\sqrt{3}\eta^3,$$

$$\psi_7 = 6\sqrt{3}\eta - 15\sqrt{3}\xi\eta - 15\eta^2 + 9\sqrt{3}\xi^2\eta + 18\xi\eta^2 + 3\sqrt{3}\eta^3, \tag{70}$$

$$\psi_8 = 9\xi - 3\sqrt{3}\eta - \frac{45}{2}\xi^2 + \frac{15}{2}\eta^2 + \frac{27}{2}\xi^3 + \frac{9\sqrt{3}}{2}\xi^2\eta - \frac{9}{2}\xi\eta^2 - \frac{3\sqrt{3}}{2}\eta^3,$$

$$\psi_9 = -\frac{9}{2}\xi + 3\frac{3\sqrt{3}}{2}\eta + 18\xi^2 - 9\sqrt{3}\xi\eta + 3\eta^2 - \frac{27}{2}\xi^3 + \frac{9\sqrt{3}}{2}\xi^2\eta + \frac{9}{2}\xi\eta^2 - \frac{3\sqrt{3}}{2}\eta^3,$$

$$\psi_{10} = 18\sqrt{3}\xi\eta - 18\eta^2 - 18\sqrt{3}\xi^2\eta - 6\sqrt{3}\eta^3,$$

$$\phi_1 = 1 - \frac{11}{2}\delta + 9\delta^2 - \frac{9}{2}\delta^3,$$

$$\phi_2 = \delta - \frac{9}{2}\delta^2 + \frac{9}{2}\delta^3,$$

$$\phi_3 = 9\delta - \frac{45}{2}\delta^2 + \frac{27}{2}\delta^3,$$

$$\phi_4 = -\frac{9}{2}\delta + 18\delta^2 - \frac{27}{2}\delta^3.$$

Appendix B: Reconstruction accuracy vs. stencil size

To determine whether the stencil configuration affects the accuracy of the reconstructed solution, we conducted a statistical test by initializing a single grid block with an ensemble of $N_w = 10$ waves with isotropically distributed wavevectors $\mathbf{k}_j$ and random phases φ_j,

$$U(\mathbf{x}) = \frac{1}{N_w}\sum_{j=1}^{N_w}\cos(\mathbf{k}_j\cdot\mathbf{x} + \varphi_j), \tag{71}$$

where U is the scalar variable to be reconstructed. The wavelengths $\lambda_j = 2\pi/k_j$ were logarithmically distributed between $\lambda_{\min} = 1$ and $\lambda_{\max} = 10$. For this test we used fourth order of accuracy because it offers the largest choice of stencils. A single division 1 block with 16 shells, $r_{\min} = 1.71$, $r_{\max} = 2.92$, and division 5 t-faces was used. Shell spacing was exponential as given by Eq. (27).

Figure 18 shows the average L_1 error, its standard deviation, and the largest error over all trials, as a function of the number of zones in the stencil that varied between $D(3) = 20$ and $3D(3) = 60$. Interestingly, both the average and the maximum errors are slowly increasing with the stencil size, although a zero trend would also be consistent with the error bars. This trend was previously observed by Ivan et al. (2015), who suggested that smaller stencils provide higher accuracy because the reconstruction data is more local to the zone. The only exception is the first point corresponding to the stencil of the smallest possible size. It would seem reasonable, then, to use smaller stencils as long as they have a few extra zones to benefit from the least square procedure. One has to remember, however, that this result may not hold for every problems or mesh configuration.

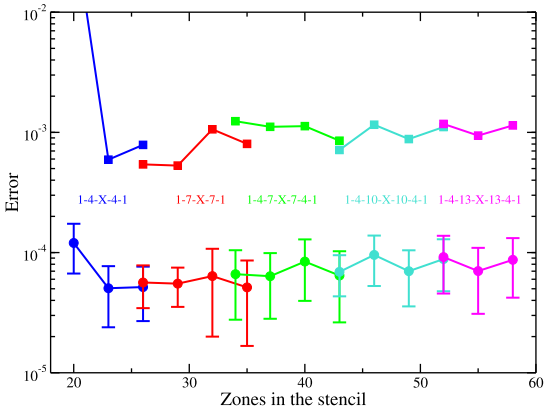

Figure 18 Fourth order reconstruction error for different size stencils. Average of L_1 reconstruction error (circles) with standard deviation, shown as error bars, and the largest error, shown with square symbols, on a division 1 block with division 5 faces and 16 radial shells. Averaging was done over 1000 trials, each initialized with a random ensemble of ten waves with isotropically distributed wavevectors. Stencil configuration is color-coded; the value of X can be deduced by subtracting the sum of zones in all planes, save the principal plane, from the total number of zones on the horizontal axis

Acknowledgements
Not applicable.

Funding
This work was supported, in part, by NSF grant DMS-1361219 and by NASA grant NNX17AB85G. DBS acknowledges support via NSF grants ACI-1533850, DMS-1622457, ACI-1713765 and DMS-1821242. Support from a grant by Notre Dame International is also acknowledged.

Abbreviations
2D, Two dimensional; 3D, Three dimensional; ADER, Arbitrary high order scheme using Riemann problem to advect high-order Derivatives; CFD, Computational Fluid Dynamics; CLSQ, Constrained Least Square (system); CPU, Central Processing Unit; E, East; EF, Edge-face (connectivity); ENC, Exponential Normalized Coordinates; EV, Edge-vertex (connectivity); FE, Face-edge (connectivity); FF, Face-face (connectivity); FV, Face-vertex (connectivity); HLL, Harten–Lax–van Leer (Riemann solver); HLLC, Harten–Lax–van Leer with Contact discontinuity (Riemann solver); HLLD, Harten–Lax–van Leer with multiple Discontinuities (Riemann solver); HLLI, Harten–Lax–van Leer with Intermediate characteristic fields (Riemann solver); KKT, Karush–Kuhn–Tucker (matrix); LHS, Left Hand Side; LLS, Linear Least Square (system); LU, Lower–Upper; MHD, Magneto-hydrodynamic; MPI, Message Passing Interface; N, North; NE, North-East; NW, North-West; PDE, Partial Differential Equation; QGM, Quadrilateral Geodesic Mesh; RHS, Right Hand Side; S, South; SE, South-East; SI, Système International; SW, South-West; TAS, Triangular Addressing Scheme; TGM, Triangular Geodesic Mesh; TVD, Total Variation Diminishing; VE, Vertex-edge (connectivity); VF, Vertex-face (connectivity); VV, Vertex-vertex (connectivity); W, West; WENO, Weighted Essentially Non-Oscillatory; WENO-AO, Weighted Essentially Non-Oscillatory—Adaptive Order.

Availability of data and materials
Data used to produce the figures in this paper are available from the authors on request.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
VF co-developed the mathematical models, implemented them in a C++ code, and wrote most of the paper. DSB co-developed the mathematical models and designed the Fortran version of the code. SG wrote parts of the Fortran code and performed test simulations. KFG developed numerical quadrature algorithms used by the code. All authors read and approved the final manuscript.

Author details
[1]Space Science Department, University of Alabama in Huntsville, Huntsville, USA. [2]Physics and ACMS Departments, University of Notre Dame, Notre Dame, USA. [3]Korea Astronomy and Space Science Institute, Daejeon, Republic of Korea. [4]Mathematics Department, Howard University, Washington, USA.

Publisher's Note

Received: 5 September 2019 Accepted: 17 March 2020
Published online: 27 March 2020

References

Balsara, D.S.: Multidimensional HLLE Riemann solver: application to Euler and magnetohydrodynamic flows. J. Comput. Phys. **229**, 1970–1993 (2010). https://doi.org/10.1016/j.jcp.2009.11.018

Balsara, D.S.: A two-dimensional HLLC Riemann solver for conservation laws: application to Euler and magnetohydrodynamic flows. J. Comput. Phys. **231**, 7476–7503 (2012). https://doi.org/10.1016/j.jcp.2011.12.025

Balsara, D.S.: Multidimensional Riemann problem with self-similar internal structure. Part I—application to hyperbolic conservation laws on structured meshes. J. Comput. Phys. **277**, 163–200 (2014). https://doi.org/10.1016/j.jcp.2014.07.053

Balsara, D.S., Dumbser, M.: Divergence-free MHD on unstructured meshes using high order finite volume schemes based on multidimensional Riemann solvers. J. Comput. Phys. **299**, 687–715 (2015a). https://doi.org/10.1016/j.jcp.2015.07.012

Balsara, D.S., Dumbser, M.: Multidimensional Riemann problem with self-similar internal structure. Part II—application to hyperbolic conservation laws on unstructured meshes. J. Comput. Phys. **287**, 269–292 (2015b). https://doi.org/10.1016/j.jcp.2014.11.004

Balsara, D.S., Florinski, V., Garain, S., Subramanian, S., Gurski, K.F.: Efficient, divergence-free, high order MHD on 3D spherical meshes with optimal geodesic meshing. Mon. Not. R. Astron. Soc. **487**, 1283–1314 (2019). https://doi.org/10.1093/mnras/stz1263

Balsara, D.S., Garain, S., Florinski, V., Boscheri, W.: An efficient class of WENO schemes with adaptive order for unstructured meshes. J. Comput. Phys. **404**, 109062 (2020). https://doi.org/10.1016/j.jcp.2019.109062

Balsara, D.S., Garain, S., Shu, C.-W.: An efficient class of WENO schemes with adaptive order. J. Comput. Phys. **326**, 780–804 (2016). https://doi.org/10.1016/j.jcp.2016.09.009

Balsara, D.S., Rumpf, T., Dumbser, M., Munz, C.-D.: Efficient, high accuracy ADER-WENO schemes for hydrodynamics and divergence-free magnetohydrodynamics. J. Comput. Phys. **228**, 2480–2516 (2009). https://doi.org/10.1016/j.jcp.2008.12.003

Balsara, D.S., Shu, C.-W.: Monotonicity peserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. J. Comput. Phys. **160**, 405–452 (2000). https://doi.org/10.1006/jcph.2000.6443

Balsara, D.S., Spicer, D.S.: A staggered mesh algorithm using high order Godunov fluxes to ensure solenoidal magnetic fields in magnetohydrodynamic simulations. J. Comput. Phys. **149**, 270–292 (1999)

Batten, P., Clarke, N., Lambert, C., Causon, D.M.: On the choice of wavespeeds for the HLLC Riemann solver. SIAM J. Sci. Comput. **18**, 1553–1570 (1997)

Beckmann, J., Mhaskar, H.N., Prestin, J.: Quadrature formulas for integration of multivariate trigonometric polynomials on spherical triangles. GEM Int. J. Geomath. **3**, 119–138 (2012). https://doi.org/10.1007/s13137-012-0035-4

Bernard, P.-E., Remacle, J.-F., Comblen, R., Legat, V., Hillewaert, K.: High-order discontinuous Galerkin schemes on general 2D manifolds applied to the shallow water equations. J. Comput. Phys. **228**, 6514–6535 (2009). https://doi.org/10.1016/j.jcp.2009.05.046

Browning, G.L., Hack, J.J., Swarztrauber, P.N.: A comparison of three numerical methods for solving differential equations on the sphere. Mon. Weather Rev. **117**, 1058–1075 (1989)

Choblet, G., Čadek, O., Couturier, F., Dumoulin, C.: ŒDIPUS: a new tool to study the dynamics of planetary interiors. Geophys. J. Int. **170**, 9–30 (2007). https://doi.org/10.1111/j.1365-246X.2007.03419.x

DeVore, C.R.: Flux-corrected transport trchniques for multidimensional compressible magnetohydrodynamics. J. Comput. Phys. **92**, 142–160 (1991)

Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi tessellations: applications and algorithms. SIAM Rev. **41**, 637–676 (1999)

Du, Q., Gunzburger, M.D., Ju, L.: Voronoi-based finite volume methods, optimal Voronoi meshes, and PDEs on the sphere. Comput. Methods Appl. Mech. Eng. **192**, 3933–3957 (2003). https://doi.org/10.1016/S0045-7825(03)00394-3

Dumbser, M., Balsara, D.S.: A new efficient formulation of the HLLEM Riemann solver for general conservative and non-conservative hyperbolic systems. J. Comput. Phys. **304**, 275–319 (2016). https://doi.org/10.1016/j.jcp.2015.10.014

Dumbser, M., Balsara, D.S., Toro, E.F., Munz, C.-D.: A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J. Comput. Phys. **227**, 8209–8253 (2008). https://doi.org/10.1016/j.jcp.2008.05.025

Dumbser, M., Käser, M.: Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems. J. Comput. Phys. **221**, 693–723 (2007). https://doi.org/10.1016/j.jcp.2006.06.043

Dunavant, D.A.: High degree efficient symmetrical Gaussian quadrature rules for the triangle. Int. J. Numer. Methods Eng. **21**, 1129–1148 (1985)

Einfeldt, B.: On Godunov-type methods for gas dynamics. SIAM J. Numer. Anal. **25**, 294–318 (1988)

Evans, C.R., Hawley, J.F.: Simulation of magnetohydrodynamic flows—a constrained transport method. Astrophys. J. **332**, 659–677 (1988). https://doi.org/10.1086/166684

Feng, X., Yang, L., Xiang, C., Wu, S.T., Zhou, Y., Zhong, D.: Three-dimensional solar wind modeling from the Sun to Earth by a SIP-CESE MHD model with a six-component grid. Astrophys. J. **723**, 300–319 (2010). https://doi.org/10.1088/0004-637X/723/1/300
Feng, X., Zhou, Y., Wu, S.T.: A novel numerical implementation for solar wind modeling by the modified conservation element/solution wlement method. Astrophys. J. **655**, 1110–1126 (2007). https://doi.org/10.1086/510121
Florinski, V., Guo, X., Balsara, D.S., Meyer, C.: Magnetohydrodynamic modeling of solar system processes on geodesic grids. Astrophys. J. Suppl. Ser. **205**, 19 (2013). https://doi.org/10.1088/0067-0049/205/2/19
Friedrich, O.: Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. J. Comput. Phys. **144**, 194–212 (1998)
Giraldo, F.X.: Lagrange–Galerkin methods on spherical geodesic grids. J. Comput. Phys. **136**, 197–213 (1997)
Gorski, K.M., Hivon, E., Banday, A.J., Wandelt, B.D., Hansen, F.K., Reinecke, M., Bartelmann, M.: HEALPix: a framework for high-resolution discretization and fast analysis of data distributed on the sphere. Astrophys. J. **622**, 759–771 (2005). https://doi.org/10.1086/427976
Guo, X., Florinski, V.: Galactic cosmic-ray intensity modulation by corotating interaction region stream interfaces at 1 AU. Astrophys. J. **826**, 65 (2016). https://doi.org/10.3847/0004-637X/826/1/65
Guo, X., Florinski, V., Wang, C.: Effects of anomalous cosmic rays on the structure of the outer heliosphere. Astrophys. J. **859**, 157 (2018). https://doi.org/10.3847/1538-4357/aabf42
Gurski, K.F.: An HLLC-type approximate Riemann solver for ideal magnetohydrodynamics. SIAM J. Sci. Comput. **25**, 2165–2187 (2004). https://doi.org/10.1137/S1064827502407962
Harten, A., Lax, P.D., van Leer, B.: On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. SIAM Rev. **25**, 35–61 (1983)
Harten, A., Osher, S.: Uniformly high-order accurate nonoscillatory schemes. I. SIAM J. Numer. Anal. **2**, 279–309 (1987)
Heikes, R., Randall, D.A.: Numerical integration of the shallow-water equations on a twisted icosahedral grid. Part I. Basic design and results of tests. Mon. Weather Rev. **123**, 1862–1880 (1995a)
Heikes, R., Randall, D.A.: Numerical integration of the shallow-water equations on a twisted icosahedral grid. Part II. A detailed description of the grid and an analysis of numerical accuracy. Mon. Weather Rev. **123**, 1881–1887 (1995b)
Ivan, L., De Sterck, H., Northrup, S.A., Groth, C.P.T.: Multi-dimensional finite-volume scheme for hyperbolic conservation laws on three-dimensional solution-adaptive cubed-sphere grids. J. Comput. Phys. **255**, 205–227 (2013). https://doi.org/10.1016/j.jcp.2013.08.008
Ivan, L., De Sterck, H., Susanto, A., Groth, C.P.T.: High-order central ENO finite-volume scheme for hyperbolic conservation laws on three-dimensional cubed-sphere grids. J. Comput. Phys. **282**, 157–182 (2015). https://doi.org/10.1016/j.jcp.2014.11.002
Jiang, G.-S., Shu, C.-W.: Efficient implementation of weighted ENO schemes. J. Comput. Phys. **126**, 202–228 (1995)
Kageyama, A., Sato, T.: "Yin–Yang grid": an overset grid in spherical geometry. Geochem. Geophys. Geosyst. **5**, Q09005 (2004). https://doi.org/10.1029/2004GC000734
Käser, M., Iske, A.: ADER schemes on adaptive triangular meshes for scalar conservation laws. J. Comput. Phys. **205**, 486–508 (2005). https://doi.org/10.1016/j.jcp.2004.11.015
Koldoba, A.V., Romanova, M.M., Ustyugova, G.V., Lovelace, R.V.E.: Three-dimensional magnetohydrodynamic simulations of accretion to an inclined rotator: the "cubed sphere" method. Astrophys. J. **576**, L53–L56 (2002)
Li, S.: An HLLC Riemann solver for magneto-hydrodynamics. J. Comput. Phys. **203**, 344–357 (2005). https://doi.org/10.1016/j.jcp.2004.08.020
Liu, X.-D., Osher, S., Chan, T.: Weighted essentially non-oscillatory schemes. J. Comput. Phys. **115**, 200–212 (1994)
Miura, H.: An upwind-biased conservative advection scheme for spherical hexagonal–pentagonal grids. Mon. Weather Rev. **135**, 4038–4044 (2007). https://doi.org/10.1175/2007MWR2101.1
Miyoshi, T., Kusano, K.: A multi-state HLL approximate Riemann solver for ideal magnetohydrodynamics. J. Comput. Phys. **208**, 315–344 (2005). https://doi.org/10.1016/j.jcp.2005.02.017
Nakamizo, A., Tanaka, T., Kubo, Y., Kamei, S., Shimazu, H., Shinagawa, H.: Development of the 3-D MHD model of the solar corona-solar wind combining system. J. Geophys. Res. **114**, A07109 (2009). https://doi.org/10.1029/2008JA013844
Ollivier-Gooch, C., Van Altena, M.: A high-order-accurate unstructured mesh finite-volume scheme for the advection-diffusion equation. J. Comput. Phys. **181**, 729–752 (2002). https://doi.org/10.1006/jcph.2002.7159
Peixoto, P.S., Barros, S.R.M.: Analysis of grid imprinting on geodesic spherical icosahedral grids. J. Comput. Phys. **237**, 61–78 (2013). https://doi.org/10.1016/j.jcp.2012.11.041
Phillips, N.A.: Numerical integration of the primitive equations on the hemisphere. Mon. Weather Rev. **87**, 333–345 (1959)
Pudykiewicz, J.A.: Numerical solution of the reaction-advection-diffusion equation on the sphere. J. Comput. Phys. **213**, 358–390 (2006). https://doi.org/10.1016/j.jcp.2005.08.021
Putman, W.M., Lin, S.-J.: Finite-volume transport on various cubed-sphere grids. J. Comput. Phys. **227**, 55–78 (2007). https://doi.org/10.1016/j.jcp.2007.07.022
Ronchi, C., Iacono, R., Paolucci, P.S.: The "cubed sphere": a new method for the solution of partial differential equations in spherical geometry. J. Comput. Phys. **124**, 93–114 (1996)
Ryu, D., Miniati, F., Jones, T.W., Frank, A.: A divergence-free upwind code for multidimensional magnetohydrodynamic flows. Astrophys. J. **509**, 244–255 (1998)
Sadourny, R., Arakawa, A., Mintz, Y.: Integration of the nondivergent barotropic vorticity equation with an icosahedral-hexagonal grid for the sphere. Mon. Weather Rev. **96**, 351–356 (1968)
Shu, C.-W., Osher, S.: Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**, 439–471 (1988). https://doi.org/10.1016/0021-9991(88)90177-5
Tomita, H., Tsugawa, M., Satoh, M., Goto, K.: Shallow water model on a modified icosahedral geodesic grid by using spring dynamics. J. Comput. Phys. **174**, 579–613 (2001). https://doi.org/10.1006/jcph.2001.6897
Ullrich, P.A., Taylor, M.A.: Arbitrary-order conservative and consistent remapping and a theory of linear maps: part I. Mon. Weather Rev. **143**, 2419–2440 (2015). https://doi.org/10.1175/MWR-D-14-00343.1
Usmanov, A.V., Goldstein, M.L., Matthaeus, W.H.: Three-dimensional magnetohydrodynamic modeling of the solar wind including pickup protons and turbulence transport. Astrophys. J. **754**, 40 (2012). https://doi.org/10.1088/0004-637X/754/1/40
Weller, H., Thuburn, J., Cotter, C.J.: Computational modes and grid imprinting on five quasi-uniform spherical C grids. Mon. Weather Rev. **140**, 2734–2755 (2012). https://doi.org/10.1175/MWR-D-11-00193.1
Zienkiewicz, O.C., Taylor, R.L., Zhu, J.Z.: The Finite Element Method: Its Basis and Fundamentals, vol. 1, 5th edn. Butterworth, Oxford (2013)