NUMERICAL MATHEMATICS: Theory, Methods and Applications
Numer. Math. Theor. Meth. Appl., Vol. **1**, No. 4, pp. 410-434 (2008)

# Hybrid Surface Mesh Adaptation for Climate Modeling

Ahmed Khamayseh[1,*], Valmor de Almeida[1] and Glen Hansen[2]

[1] *Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA.*
[2] *Multiphysics Methods Group, Idaho National Laboratory, Idaho Falls, ID 83415-3840, USA.*

Received 22 February, 2008; Accepted (in revised version) 30 June, 2008

**Abstract.** Solution-driven mesh adaptation is becoming quite popular for spatial error control in the numerical simulation of complex computational physics applications, such as climate modeling. Typically, spatial adaptation is achieved by element subdivision ($h$ adaptation) with a primary goal of resolving the local length scales of interest. A second, less-popular method of spatial adaptivity is called "mesh motion" ($r$ adaptation); the smooth repositioning of mesh node points aimed at resizing existing elements to capture the local length scales. This paper proposes an adaptation method based on a combination of both element subdivision and node point repositioning ($rh$ adaptation). By combining these two methods using the notion of a *mobility function*, the proposed approach seeks to increase the flexibility and extensibility of mesh motion algorithms while providing a somewhat smoother transition between refined regions than is produced by element subdivision alone. Further, in an attempt to support the requirements of a very general class of climate simulation applications, the proposed method is designed to accommodate unstructured, polygonal mesh topologies in addition to the most popular mesh types.

**AMS subject classifications**: 52B10, 65D18, 68U05, 68U07

**Key words**: surface mesh generation, mesh adaptation, mesh optimization, climate modeling.

## 1. Introduction

Mesh generation is an important consideration in the area of numerical simulation of physical phenomena. Indeed, the accuracy and convergence of approaches using mesh-based numerical methods are strongly dependent on the intrinsic characteristics of the mesh being used. The "quality" of a mesh is loosely termed as the degree in which a particular mesh supports a given simulation. For transient calculations, the mesh supporting the calculation must not only be of high quality initially, it also must effectively support the requirements of the dynamic simulation as it evolves. Generally, solution features will

*Corresponding author. *Email addresses:* khamaysehak@ornl.gov (A. Khamayseh), dealmeidav@ornl.gov (V. de Almeida), Glen.Hansen@inl.gov (G. Hansen)

 

develop and move across the domain; these features must be resolved as they propogate through the mesh.

One approach to mesh adaptation is to begin with an appropriate initial mesh, and sweep over this mesh in both space and time calculating an *error metric*. This error metric is chosen to represent some quantification of the error in the solution in each mesh element. As the sweep progresses, provided that this metric is above a certain value, the mesh element is subdivided to reduce its area (or volume) and hence the value of the error metric inside the refined portion of the original element.

This method of mesh element subdivision is commonly called *h* refinement, or *adaptive mesh refinement (AMR)*. An example of *h* refinement on an orographic map of the Himalayas mountains is shown in Fig. 1. In this example, triangular, quadrilateral, and polygonal meshes are adapted to better capture the Earth's orography field (the local average surface elevation). The upper-left inset of the figure shows a color intensity map of the orographic value of the Earth's surface, where blue indicates sea level and red indicates the highest elevation regions. The upper-right inset shows a quadrilateral mesh on the Earth's surface that is *h* refined using the intensity of the orographic value as a refinement metric. The lower-left inset shows an *h* refined triangle surface mesh, where the lower-right inset shows a refined polygonal mesh.

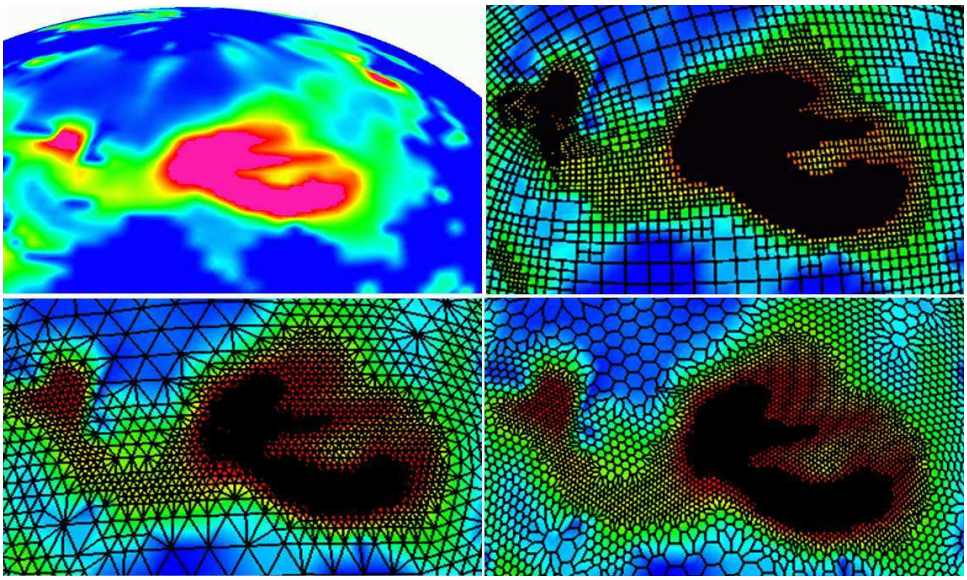

Figure 1: An example of *h*-refinement driven by an orographic scalar on a mesh of the Earth's surface. Upper-left figure shows elevation; blue is near sea-level and red indicates high elevation regions. The upper-right figure shows a refined quadrilateral mesh while the lower-left and lower-right figures show triangular and polyhedral meshes, respectively.

While the *h* refinement algorithm is a general approach that is effective on all element shapes and for all length scales, the method results in abrupt (*i.e.,* non-smooth) variations in element area or volume. These non-smooth transitions may impact local simulation accuracy at and near the transition elements. This limitation of element subdivision has to

the development of other methods to achieve spatial refinement, such as “mesh motion” or “mesh smoothing.” This mesh adaptation approach (also called $r$ adaptation) varies element size by moving existing mesh node points to new locations instead of subdividing elements.

Generally, $r$ adaptive methods also sweep over the mesh in both space and time calculating an error metric. In a given mesh element, if this computed error metric is above a particular value, the mesh node points are moved closer together to reduce the size of the mesh elements, which in turn reduces the value of the error metric. An example of $r$ adaptation is shown in Fig. 2. It is clear in the illustration that $r$ adaptation produces smoother results than subdivision. It is also clear that the degree to which elements may be resized is limited by distortion of the surrounding mesh elements. Thus, $r$ adaptation cannot capture arbitrarily small features without greatly distorting the mesh, which requires striking a compromise between spatial resolution and mesh quality.

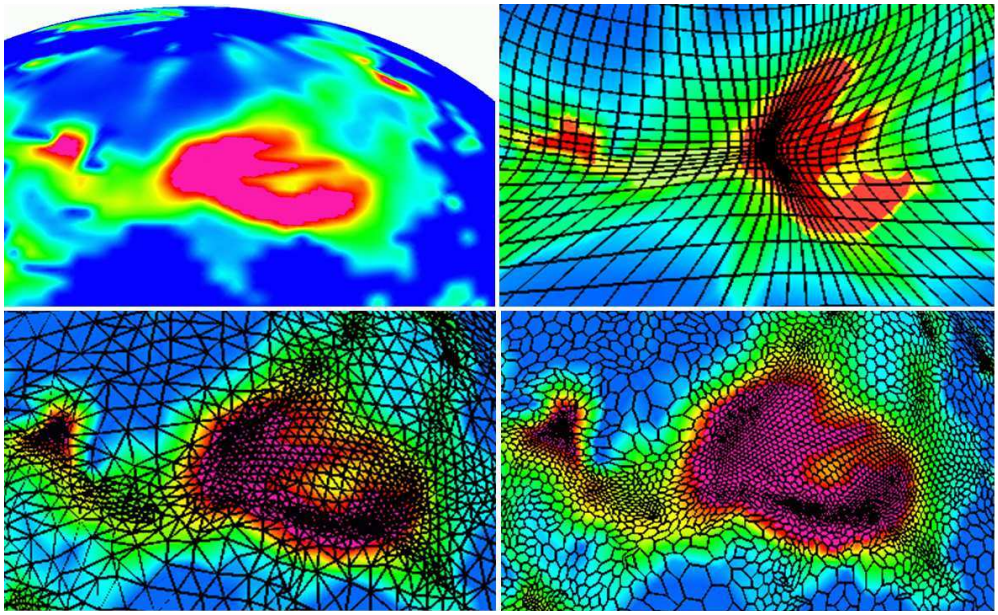

Figure 2: An example of $r$ adaptivity driven by an orographic scalar on a mesh of the Earth's surface. The figure to the upper left again indicates elevation, while the upper-right, lower-left, and lower-right diagrams show $r$ adaptation applied to quadrilateral, triangular, and polyhedral meshes, respectively.

An alternative to meshes that contain exclusively one element geometry is a polygonal hybrid mesh; *i.e.,* a mesh consisting of both quadrilateral and triangular elements (and possibly elements with five or more sides). A hybrid mesh attempts to combine the advantages of using a variety of element shapes to generate the mesh. It is usually easier to mesh a complex problem using different elements (both triangular and quadrilateral for example), than using quadrilatera exclusively. A polygonal mesh provides even more flexibility as it may contain elements of any shape, ranging from hexagons, pentagons, quads, to even simple triangles. In addition, polygonal/icosahedral meshes are often the best choice of solving symmetric computational problems, such as climate modeling on the

Earth's surface. These meshes have the property of producing symmetric, highly isotropic, higher order orthogonal meshes that provide quasi-uniform coverage of the sphere with small regional variation in the shape of elements without problematic mesh singularities at the poles [1, 2].

Mesh generation and adaptation for both structured and unstructured meshes is a popular area of research [3–6]. Research and methods development targeted at adaptive approaches supporting the generation and use of high-quality polygonal meshes is not as common. Approaches of surface mesh optimization have been demonstrated on meshes containing structured quadrilateral elements [7–10] and unstructured triangular elements [11–14]. Hansen *et.al.* [15, 16] demonstrate a mesh generation method that is effective for two- and three-dimensional problems, and structured and unstructured meshes that employ triangle/tetrahedral or quadrilateral/hexahedral elements. This method may support general polygonal meshes, although this is not demonstrated. Hansen and Zardecki [17] present a surface meshing algorithm that adapts to surface curvature on surface meshes containing quadrilateral or triangular elements. Again, polygonal mesh capability is not demonstrated, nor is adaptivity to solution data. Putman and Lin [18] apply several mesh generation methods including conformal mapping, elliptic, gnomonic, and spring-based methods to cube-sphere topologies applicable to global modeling and present results of atmospheric flow calculations on these meshes. Di *et.al.* [19] present an adaptive moving mesh method for calculations on a sphere. Both of these approaches are closely related to this work and would likely be quite effective for climate simulation applications. Neither study polygonal elements, nor consider *h* adaptation. The most effective approach for one's particular application would require further study of the advantages and disadvantages of each of these approaches on the application of interest.

## 2. Adaptive polygonal surface mesh optimization

The accuracy and convergence of computational solutions of mesh-based methods is quite dependent on the quality of the mesh used. This paper presents an algorithm for optimizing polygonal meshes comprised of elements of arbitrary polygonal shape. This proposed method allows mesh refinement/adaptation to be focused to areas of interest while globally equidistributing the nodes of the mesh. Additionally, the algorithm typically improves geometric quality metrics based on angle, length, and area.

Using a parametric representation, surface geometry may be defined in terms of a mapping $(x(u,v), y(u,v), z(u,v))$ from parametric space $(u,v)$ in $\mathbb{R}^2$ to physical space $(x,y,z)$ in $\mathbb{R}^3$. This mapping may be compactly denoted by the equation $\mathbf{x} = \boldsymbol{\chi}(\mathbf{u})$, where $\mathbf{x} = (x,y,z)$, and $\mathbf{u} = (u,v)$, as illustrated in Fig. 3.

The surface geometry mapping $\boldsymbol{\chi}$ may contain specific singularities, such as those that arise from mapping a line to a point during the parameterization of a cone, for example. However, if the $\boldsymbol{\chi}$ mapping is not one-to-one and onto, the mesh in physical space will often suffer from *folding* or *spill-over*, and thus would not have utility for computational problems.

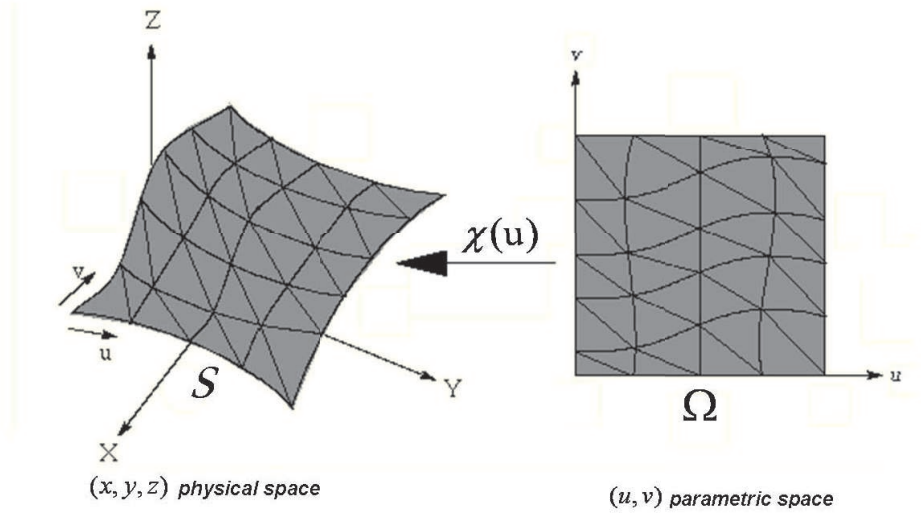


Figure 3: Pictoral showing a mapping from parametric $(u,v)$ space to surface in physical $(x,y,z)$ space.

## 2.1. Weighted gradient mesh optimization

The primary goal of the adaptive smoothing algorithm proposed here is to minimize the weighted gradient of the area of the mesh elements on the physical surface provided that the surface is defined in parametric terms. This goal is commonly referred to as an "equidistribution principle;" the notion of which was introduced by Brackbill and Saltzman [20, 21] as a guiding principle of variational mesh generation.

This minimization process is typically applied directly to the mesh resting on the physical surface as one desires the elements to contain roughly the same surface area. One complicating issue arises from the definition of the physical surface in parametric form. For such a surface, the minimization on the physical surface is inconvenient as it is difficult to guarantee conformality of the optimized mesh elements to the surface. Instead, an indirect formulation is adopted where the mesh element areas are adapted by posing the equidistribution algorithm in the parametric domain.

To develop this formulation, an *area function* is defined in the parametric domain $(u,v)$ that represents the variation in area of the elements of the mesh on the physical surface. This area function $\phi$ is required to be twice continuously differentiable. To smoothly equidistribute the area function over the mesh, a strategy of minimizing the $\mathscr{L}^2$ norm of the surface gradient of the area function is employed. This goal is equivalent to moving the mesh points to minimize

$$\int_{\mathscr{S}} \mu \|\nabla_{\mathscr{S}} \phi\|^2 \, dS, \tag{2.1}$$

where the integral applies over the surface area of the physical surface $\mathscr{S}$ and the *node mobility function* $\mu$ that will be discussed in Sec. 3.

Rather than minimizing (2.1) directly, an equivalent procedure is employed. This approach involves solving the Euler-Lagrange equation to minimize this functional, or find $\phi$ such that

$$\nabla_{\mathscr{S}} \cdot \left(\mu \nabla_{\mathscr{S}} \phi\right) = 0 \quad \text{in} \quad \mathscr{S}. \tag{2.2}$$

Given that the surface mapping $\boldsymbol{\chi} : \mathbf{u} \in \Omega \subset \mathbb{R}^2 \rightarrow \mathbf{x} \in \mathscr{S} \subset \mathbb{R}^3$ is twice continuously differentiable and that the Jacobian from the parametric space to the physical surface is non-vanishing in the region under consideration, the Jacobian $J$ of the surface mapping may be written as

$$J = \|\boldsymbol{\chi}_u \times \boldsymbol{\chi}_v\| = \sqrt{g_{11} g_{22} - g_{12}^2},$$

where

$$\begin{aligned} g_{11} &= \boldsymbol{\chi}_u \cdot \boldsymbol{\chi}_u = x_u^2 + y_u^2 + z_u^2, \\ g_{12} &= \boldsymbol{\chi}_u \cdot \boldsymbol{\chi}_v = x_u x_v + y_u y_v + z_u z_v, \\ g_{22} &= \boldsymbol{\chi}_v \cdot \boldsymbol{\chi}_v = x_v^2 + y_v^2 + z_v^2. \end{aligned}$$

Equation (2.2) is transformed from the physical surface to the parametric domain $(u, v)$ using coordinate transformation of the second-order differential equation for surfaces [4, 22]

$$\nabla_{\mathscr{S}} \cdot \left(\mu \nabla_{\mathscr{S}} \phi\right) = \frac{1}{J} \left\{ \frac{\partial}{\partial u} \frac{\mu g_{22} \phi_u - \mu g_{12} \phi_v}{J} + \frac{\partial}{\partial v} \frac{\mu g_{11} \phi_v - \mu g_{12} \phi_u}{J} \right\}.$$

Under this transformation, the parametric variables become the independent variables, resulting in a new form of the surface equation

$$\nabla_{\mathbf{u}} \cdot \Phi = 0 \quad \text{in} \quad \Omega, \tag{2.3}$$

where the vector function $\Phi$ is given by

$$\Phi = \frac{\mu}{J} \left( g_{22} \phi_u - g_{12} \phi_v, g_{11} \phi_v - g_{12} \phi_u \right).$$

In summary, solving (2.2) is equivalent to solving the partial differential equation (2.3) in the parametric domain, and the problem is effectively reduced to obtaining the function $\phi(\nabla_{\mathbf{u}} \boldsymbol{\chi})$ that provides the desired distribution of points on the physical surface.

There is a mathematical similarity (2.1) to computing the deformation of a hyperelastic surface. The parametric mapping $\boldsymbol{\chi}$ plays the role of a deformation mapping, and since $\phi$ only depends on the gradient of $\boldsymbol{\chi}$ and the position in the reference (parametric) domain, the function $\mu \|\nabla_{\mathscr{S}} \phi\|^2$ resembles a stored energy function. Given this analogy, the functional (2.1) is the strain energy associated to the deformation mapping $\boldsymbol{\chi}$. This approach has also been used to provide mesh adaptivity in three dimensions [23].

## 2.2. Derivation of the cell area function

The surface mesh in physical space consists of a multiply-connected network of piecewise planar arbitrary polygonal facets. Such a mesh will have a corresponding mesh in the parametric domain, and if either one of these meshes is valid, the transformed mesh is also valid. Further, $\phi$ is approximated by a set of continuous piecewise quadratic interpolants over the mesh cells, such that the discrete, cell-centered value of this area function at cell $i$ is given by

$$\phi_i(\mathbf{u}) = S_i \alpha_i(\mathbf{u}) + \sum_{j=1}^{n} S_j \beta_j(\mathbf{u}), \tag{2.4}$$

where $n$ is the number of vertices for the $i^{th}$ cell. The functions $\alpha_i$ and $\beta_j$ are quadratic basis functions that assume a value of 1 at the cell center and node $j$, and are equal to 0 at all the other nodes of the element in question. The area $S_j$ is evaluated differently depending on whether $j$ refers to a vertex or the cell centroid. If $j$ refers to the centroid, $S_j$ is the surface area of cell $j$. If $j$ is a vertex, then

$$S_j = \frac{1}{m} \sum_{i=1}^{m} S_i,$$

where $m$ is the number of cells incident to vertex $j$ and $S_i$ is the surface area of the $i^{th}$ incident cell at vertex $j$.

Alternatively, since the surface area element $\mathrm{d}S$ corresponds to the parametric area of element $\mathrm{d}A = \mathrm{d}u\,\mathrm{d}v$ in the plane given by

$$\mathrm{d}S = \| \boldsymbol{\chi}_u \,\mathrm{d}u \times \boldsymbol{\chi}_v \,\mathrm{d}v \| = J \,\mathrm{d}A,$$

for a given mesh cell $i$, the surface and the parametric areas are related by the equation

$$S_i = J_i A_i,$$

where $J_i = \frac{1}{A_i} \int J \,\mathrm{d}A$ is the averaged Jacobian which represents the ratio of physical to parametric cell areas. The surface area function in (2.4) may then be expressed as a Jacobian-weighted area interpolation in the parametric domain

$$\phi_i(\mathbf{u}) = J_i A_i \, \alpha_i(\mathbf{u}) + \sum_{j=1}^{n} J_j A_j \, \beta_j(\mathbf{u}),$$

where the parametric mesh cell areas $A_i$, $A_j$ are defined above, and the basis functions $\alpha_i$, $\beta_j$ are defined in (2.4).

## 2.3. Computational evaluation of the cell area function

To address arbitrary polygonal cells, a cell-centered finite volume discretization is used to solve (2.3) for $\phi$ in the parametric domain $\Omega$. Equation (2.3) may be rewritten in

integral form as

$$\int_{\Omega} \nabla_{\mathbf{u}} \cdot \Phi \, \mathrm{d}A = 0.$$

Using the divergence theorem, this is equivalent to

$$\int_{\partial\Omega} \Phi \cdot \mathrm{d}\mathbf{l} = 0,$$

where the vector $\mathbf{l} = l\hat{\mathbf{t}}$ is edge length $l$ times the *unit tangent vector* $\hat{\mathbf{t}}$. The vector $\hat{\mathbf{t}}$ is defined to be: (i) tangent to $\Omega$, (ii) orthogonal to $\partial\Omega$, and (iii) pointing *away* from $\Omega$. This equation is applicable to any cell or set of cells within the domain. In particular, for a given cell $C_i$ with *linear* edges, the above equation may be expressed as

$$\sum_{\text{edges } j} \int_{E_{ij}} \Phi \cdot \mathrm{d}\mathbf{l} = 0,$$

where $E_{ij}$ is the $j$th edge of current cell. The semi-discrete form of this integral equation is given by

$$\sum_{\text{edges } j} \Phi_{E_{ij}} \cdot \mathbf{l}_{ij} = 0, \tag{2.5}$$

where $\Phi_{E_{ij}} = \frac{1}{l_{ij}} \int_{E_{ij}} \Phi \cdot \mathrm{d}\mathbf{l}$ is the average value of $\Phi$ on the edge $E_{ij}$ with length $l_{ij}$. This function may be approximated on any edge using a cell-centered first-order Taylor series approximation

$$\Phi_{E_{ij}} \approx \frac{h_j \Phi_{C_i} + h_i \Phi_{C_j}}{h_i + h_j},$$

where $C_i$ and $C_j$ are the two cells sharing the edge, $E_{ij}$, $h_i$ and $h_j$ are the distances from the cell centers to the edge center, and $\Phi_{C_i}$ and $\Phi_{C_j}$ are the average values of $\Phi$ in the cells $C_i$ and $C_j$, as shown in Fig. 4.

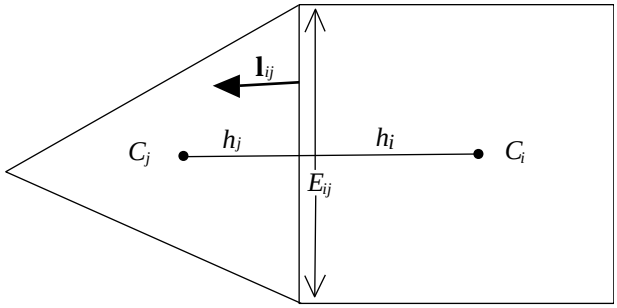


Figure 4: Definition of quantities on edge $E_{ij}$ of cell $C_i$ and that are shared by cell $C_j$.

The above approximation is only first-order accurate in the cell dimension, but higher-order extensions are possible. To calculate the vector function

$$\Phi_{C_i} = \left( \frac{\mu g_{22}\phi_u - \mu g_{12}\phi_v}{J}, \frac{\mu g_{11}\phi_v - \mu g_{12}\phi_u}{J} \right)_{C_i} \tag{2.6}$$

in the $i$th cell $C_i$, it is necessary to compute $\nabla_{\mathbf{u}}\phi_{C_i} = (\phi_u, \phi_v)_{C_i}$ and the quantities $\mu$, $J$, $g_{11}$, $g_{12}$, and $g_{22}$. Using the finite volume method, the average value of any function in a cell is assumed to be equivalent to the point value at the element centroid. As such, the quantities $\mu$, $J$, $g_{11}$, $g_{12}$, and $g_{22}$ in the numerical equation are evaluated analytically from the surface mapping $\boldsymbol{\chi}$ at the cell center. Using Gauss' theorem,

$$\begin{aligned} \nabla_{\mathbf{u}}\phi_{C_i} &= \frac{1}{A_i}\int_{C_i} \nabla_{\mathbf{u}}\phi \, dA \\ &= \frac{1}{A_i}\int_{\partial C_i} \phi \, \mathrm{d}\mathbf{l} \\ &= \frac{1}{A_i}\sum_{\text{edges } j} \phi_{E_{ij}} \mathbf{l}_{ij}, \end{aligned}$$

where $A_i$ is the area of cell $C_i$ and $\phi_{E_{ij}}$ is the average value of $\phi$ on the edge $E_{ij}$, which is approximated from the cell centered values by

$$\phi_{E_{ij}} = \frac{h_j\phi_{C_i} + h_i\phi_{C_j}}{h_i + h_j}.$$

Therefore,

$$\nabla_{\mathbf{u}}\phi_{c_i} = \frac{1}{A_i}\sum_{\text{edges } j} \frac{h_j\phi_{C_i} + h_i\phi_{C_j}}{h_i + h_j}\mathbf{l}_{ij}. \tag{2.7}$$

The final algorithm for solving (2.3) for $\phi$ may be formulated by solving (2.5) using the discretization shown in (2.6) and (2.7). The resultant algebraic system may be solved locally or globally. If the local approach is adopted as in this study, (2.3) is solved using Gauss-Seidel iteration in the subregion composed of all the cells $\{C_1, C_2, \ldots, C_m\}$ that are incident at the central node $\mathbf{u}_c$. Values of $\phi_{C_i}$ are obtained from (2.9) (*i.e.*, using the algorithm for $\phi_i^s$, where $i$ loops over the subregion cells in question). $\phi_{C_i}$ are the target areas for the new elements that are used to update the location of each node $\mathbf{u}_c$ and are calculated for each element $C_i$ on a node-by-node basis. The advantage of the local approach is that it supports user control over the degree of under-relaxation employed in the Gauss-Seidel iterative procedure; the user can select approriate values to guard against edge cross-over. The disadvantage of the local approach is that it is slow to propagate information to all nodes in the mesh. A global approach may be superior if mesh folding can be avoided, but this was not examined in this study.

## 2.4. Local mesh adaptivity

A fundamental step in optimizing an initial surface mesh is to pose an effective mechanism of sliding node points on the surface $\boldsymbol{\chi}$. This can be accomplished by selecting

a cluster of nodes $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\}$ and corresponding cell neighbors $\{C_1, C_2, \ldots, C_m\}$ surrounding the node $\mathbf{u}_c$ to be repositioned (*c.f.*, Fig. 5). Note that under the mapping $\boldsymbol{\chi}$, this cluster of parametric nodes corresponds to a unique set of physical nodes in $\mathbb{R}^3$. Let

$$\phi_i = J_i A_i = \frac{1}{2} J_i \|(\mathbf{u}_i - \mathbf{u}_c) \times (\mathbf{u}_{i+1} - \mathbf{u}_c)\|$$

be the physical area of the $C_i$th cell under $\boldsymbol{\chi}$. Here, $A_i$ is the area of $C_i$ in the parametric space and $J_i = \frac{1}{A_i} \int_{C_i} J \, \mathrm{d}A$ is the averaged Jacobian. By varying $\phi_i$ while keeping all cluster nodes fixed, a mechanism is obtained for moving $\mathbf{u}_c$ and therefore sliding the corresponding physical node $\mathbf{x}_c = \boldsymbol{\chi}(\mathbf{u}_c)$ on the surface. It is necessary to limit this sliding movement so as to keep $\mathbf{u}_c$ within the convex hull of the cell set.

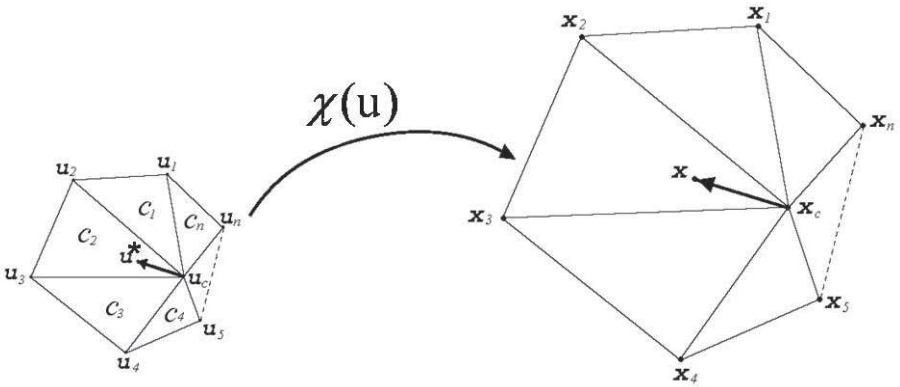


Figure 5: Node placement in parameteric space maps to a corresponding movement on the physical surface.

The method advocated here for relocating $\mathbf{u}_c$ is based on finding a new position $\mathbf{u}^*$ for the central node by enforcing new values of the physical area $\phi_i^*$ for the cells $C_i$ as follows. Let $\phi_i^* = J_i^* A_i^*$ be the desired area of the $i$th cell, where

$$\begin{aligned} A_i^*(\mathbf{u}) &= \frac{1}{2} \|(\mathbf{u}_i - \mathbf{u}^*) \times (\mathbf{u}_{i+1} - \mathbf{u}^*)\| \\ &= \frac{1}{2} \left(a_i u^* + b_i v^* + c_i\right), \end{aligned}$$

with coefficients

$$a_i = v_i - v_{i+1}, b_i = u_{i+1} - u_i, \text{ and } c_i = u_i v_{i+1} - u_{i+1} v_i.$$

One method of obtaining $\mathbf{u}^*$ within the polygon in the parametric domain involves the least-squares minimization of the functional

$$\mathscr{F} = \sum_{\text{cells } i} \left(\phi_i^* - \phi_i^s\right)^2,$$

where $\phi_i^s$ is the solution of the area function of the $i$th cell computed previously. In solving $\nabla\mathscr{F} = 0$ for $\mathbf{u}^*$'s components $u^*$ and $v^*$,

$$\begin{pmatrix} \sum_i J_i^2 a_i^2 & \sum_i J_i^2 a_i b_i \\ \sum_i J_i^2 a_i b_i & \sum_i J_i^2 b_i^2 \end{pmatrix} \begin{pmatrix} u^* \\ v^* \end{pmatrix} = \begin{pmatrix} \sum_i J_i a_i \phi_i^s \\ \sum_i J_i b_i \phi_i^s \end{pmatrix}, \tag{2.8}$$

the new parametric node location $\mathbf{u}^*$ is used to compute the updated position of the corresponding central vertex on the surface $\boldsymbol{\chi}(\mathbf{u}^*)$. For simplicity, (2.8) is calculated assuming that it represents a constant value Jacobian evaluated at the cell center. This is often a good assumption in practice as this minimization process is embedded within an outer loop that traverses all the nodes of the mesh.

The most recent value for cell area ($\phi_i^s$) within the cluster (2.8) should not vary significantly from the old value $\phi_i$. If the update is too severe, it is possible that the updated parametric position of the central node will fall outside the convex hull of the neighbor element set. One method to mitigate this problem is based on discarding or limiting any updates that lie outside of a bounded interval. For the orographic data sets considered in this study, limiting the magnitude of $\phi^s$ in the system (2.8) to

$$\phi_i^s = \max\{\phi_i^s, (1-\omega)\phi_i\}, \tag{2.9}$$

always accelerated algorithm convergence and did not result in mesh folding or spillover. In this expression, the node regularization scaling parameter $\omega$ is chosen to prevent the collapse or the inversion of cells as the mesh evolves. Experimentally, $\omega \in [\frac{1}{2}, 1)$ appears to be effective.

## 3. Adaptive hybrid node mobility

The cell area based mesh optimization procedure described above may be extended to take into account a combination of several geometric and solution criteria. In particular, the smoothing process might consider the curvature of the surface, solution values, and other metrics of the surface. This adaptive control factor, or node mobility function, governs the "strength" or degree of adaptation that is applied to the mesh.

### 3.1. Node mobility function

The smoothing mobility (weight) function $\mu$ in the minimization functional (2.1) may be generalized by making it a hybrid function that incorporates either additional measures of mesh element quality or metrics that drive solution adaptation on the surface mesh. For example, to implement smoothing based on area gradient equidistribution one can specify that $\mu = 1$ for all elements. To implement curvature weighted smoothing, where mesh points are concentrated in areas of large surface or solution curvature, $\mu$ should have the form

$$\mu \propto |K\sqrt{\phi}|,$$

where $K$ is the mean curvature. In the case of surface geometry, the curvature $K = K(\mathbf{u})$ of the surface $\boldsymbol{\chi}$ at $\mathbf{u}$ is given by

$$K = \frac{1}{2}\frac{g_{22}(\boldsymbol{\chi}_{uu}\cdot\boldsymbol{\chi}_u\times\boldsymbol{\chi}_v) - 2g_{12}(\boldsymbol{\chi}_{uv}\cdot\boldsymbol{\chi}_u\times\boldsymbol{\chi}_v) + g_{11}(\boldsymbol{\chi}_{vv}\cdot\boldsymbol{\chi}_u\times\boldsymbol{\chi}_v)}{\|\boldsymbol{\chi}_u\times\boldsymbol{\chi}_v\|^3},$$

where $\sqrt{\phi}$ (the square root of the area function) is used to non-dimensionalize the curvature weight function. In the case of adapting to the curvature of a piecewise solution, the curvature is computed using the method described in the next section.

For solution feature weighted smoothing, the mesh nodes should be concentrated toward areas of large gradient or curvature of the desired solution variable. For the case of gradient adaptation, $\mu$ should have the form

$$\mu \propto \|\nabla_{\mathscr{S}}\psi\sqrt{\phi}\|,$$

where $\nabla_{\mathscr{S}}\psi$ is the surface gradient of the non-dimensionalized solution variable to which the mesh should adapt.

In general, the overall node mobility function will typically be a linear combination of individual mobility functions, with the general form

$$\mu = \lambda_a + \lambda_\kappa\left(|K\sqrt{\phi}|\right) + \lambda_s\left(\|\nabla_{\mathscr{S}}\psi\,\sqrt{\phi}\|\right). \tag{3.1}$$

This function is used to specify the strength and degree of mesh node point motion and as a criteria for element subdivision as outlined in Sec. 4.3. The values of $\lambda \geq 0$ are experimentally determined; the optimal value of each parameter relative to the others is a function of the strength of the curvature of the surface geometry and the intensity of the gradient of the solution variable of interest. The three parameters must sum to one, $\lambda_a + \lambda_\kappa + \lambda_s = 1$. Given this relationship, one need only select a scaling of the curvature weight $0 \leq \lambda_\kappa \leq 1$ and a weight with which to scale the solution field $0 \leq \lambda_s \leq 1$; the remaining $\lambda_a$ may be calculated relative to the others

$$\lambda_a = 1 - \lambda_\kappa - \lambda_s.$$

## 3.2. Determination of the curvature of a piecewise solution

To calculate integrals involving curvature on a piecewise surface consisting of triangular facets requires the identity

$$\int_\Omega K\mathbf{n}\,dS = \int_{\partial\Omega}\mathbf{t}\,dl. \tag{3.2}$$

This identity states that the integral of the *sum of principal curvatures* $K$ times the unit normal vector $\mathbf{n}$ over a piece of surface $\Omega$ is equal to the line integral around the boundary with an integrand equal to the unit tangent vector $\mathbf{t}$ (see Sec. 2.3). An equivalent formula which avoids $\mathbf{t}$ is

$$\int_\Omega K\mathbf{n}\,dS = -\int_{\partial\Omega}\mathbf{n}\times d\mathbf{l}. \tag{3.3}$$

The form of (3.2) may be motivated physically; a force normal to a surface with magnitude equal to the surface curvature is equivalent to a *surface tension* force (of unit magnitude) applied to the boundary of the surface [24].

This result leads to a dilemma, as the relationship between the curvature of a piecewise planar surface with flat facets and that of the "actual" surface which the facets discretely represent is not readily defined in the general case. For example, if the unit sphere is discretized using triangular facets, one might reason that $K = 2$ could be employed in the above integrals. While this is quite true for the surface of a sphere, it is not true on the elements of the discretization of the sphere surface as the curvature is generally defined in terms of a *distribution* on the edges of the facets and zero elsewhere.

In the proposed approach, curvature is treated as a distribution, where the geometric object is an arbitrary piecewise linear surface (the fact that the discrete surface is an approximation to a sphere is not employed). Here, the finite element inner products are evaluated using $\int_{\text{nbd } i} K\mathbf{n}\alpha \, dS$. In the moving finite element method proposed by Kuprat and Miller [25, 26], a similar integral equation to (3.2) in the neighborhood $\Omega_i$ of point $i$ is evaluated, except that the integrand is multiplied by a "hat function" $\alpha$ which is unity at the $i^{\text{th}}$ node and decays to zero at its neighbors. The presence of $\alpha$ in the left hand term introduces a factor of $\frac{1}{2}$ in the right

$$\int_{\Omega_i} K\mathbf{n}\alpha \, dS = \frac{1}{2} \int_{\partial\Omega_i} \mathbf{t} \, dl. \tag{3.4}$$

To develop a discrete form of (3.4), assume that node $i$ is located at $\mathbf{x}_o$ and has neighbors $1, \ldots, m$ positioned at $\mathbf{x}_1, \ldots, \mathbf{x}_m$ (ordered in a counterclockwise stencil rotation). Define the quantities

$$\mathbf{r}_j \equiv \mathbf{x}_j - \mathbf{x}_o, \quad \mathbf{l}_j \equiv \mathbf{x}_{j+1} - \mathbf{x}_j, \quad \mathbf{a}_j \equiv \frac{1}{2}\mathbf{r}_j \times \mathbf{l}_j,$$
$$\mathbf{n}_j \equiv \frac{\mathbf{a}_j}{\|\mathbf{a}_j\|}, \quad \mathbf{t}_j \equiv -\frac{\mathbf{n}_j \times \mathbf{l}_j}{\|\mathbf{n}_j \times \mathbf{l}_j\|},$$

where $\mathbf{a}_j$ is the *area vector* of the $j^{\text{th}}$ triangle surrounding node $i$, which has length equal to the area of the triangle and a direction normal to the triangle. Given this notation, (3.4) is equivalent to

$$\int_{\text{nbd } i} K\mathbf{n}\alpha \, dS = \frac{1}{2} \sum_{j=1}^{m} \mathbf{t}_j \|\mathbf{l}_j\|.$$

Again, this derivation assumes that the underlying surface is unknown, thus an *exact* value for the curvature at point $i$ in the stencil is not known. However, one may calculate an integral average curvature about $i$ using

$$K_i \equiv \frac{\int_{\text{nbd } i} K\mathbf{n}\alpha \, dS \cdot \int_{\text{nbd } i} \mathbf{n}\alpha \, dS}{\int_{\text{nbd } i} \mathbf{n}\alpha \, dS \cdot \int_{\text{nbd } i} \mathbf{n}\alpha \, dS}.$$

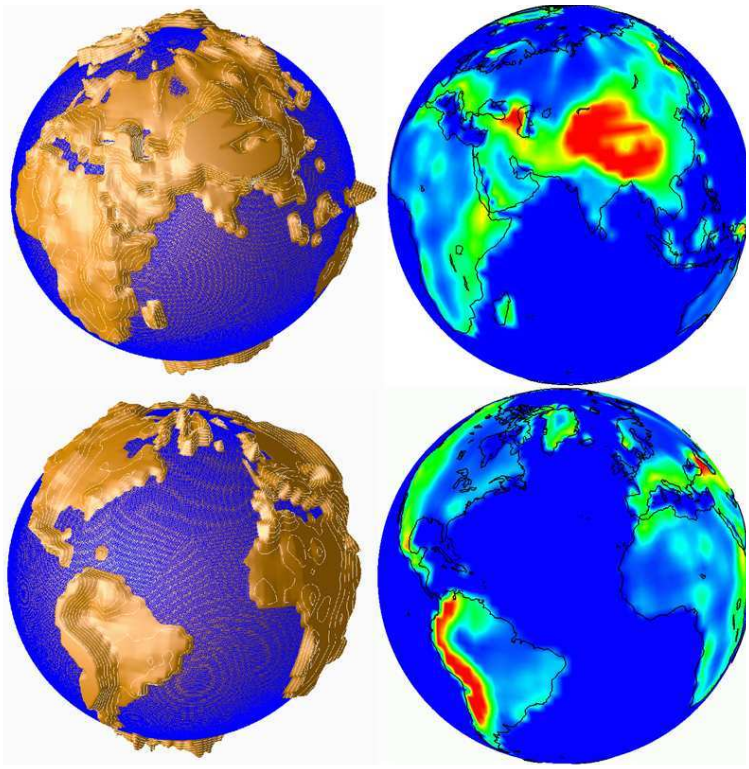

Figure 6: Color-coded plot of the earth's surface geopotential height illustrating high altitude regions (the Himalayas, Alps, Andes, and Rockies).

This may be written as

$$K_i \equiv \frac{\frac{1}{2}\sum_{j=1}^{m} \mathbf{t}_j \|\mathbf{l}_j\| \cdot \frac{1}{3}\sum_{j=1}^{m} \mathbf{a}_j}{\frac{1}{3}\sum_{j=1}^{m} \mathbf{a}_j \cdot \frac{1}{3}\sum_{j=1}^{m} \mathbf{a}_j}, \tag{3.5}$$

given the above definitions. This expression predicts the correct curvature for a small regular hexagon (collection of six triangles) consisting of six points surrounding a central node, all on the surface of a sphere. Unfortunately, as the hexagon is made irregular, the accuracy of (3.5) deteriorates. This degradation is often not of concern in a finite volume approximation, as the quality of the approximation of the inner products is typically more important than the precision of the mean curvature calculation.

## 4. Orography based adaptive meshing

Smooth adaptive mesh transformations are required for resolving orography (*i.e.,* the average height of land, measured in geopotential meters) and fine-scale processes in climate modeling. Orography plays an important role in determining the strength and location of the atmospheric jet streams that must be accurately predicted for detailed regional climate studies. Additionally, orography is an important consideration for the prediction of many key climatic dynamics, elements, and moisture physics; such as rainfall, snowfall,

and cloud cover. Climate variability is sensitive to orographic effects and can be resolved by the generation of finer meshes in regions of high altitude. Resolving orography effectively also allows a more accurate prediction of precipitation (to predict wetter or dryer seasons in particular regions). Lastly, resolved orography is important as orography defines the lower boundary for general circulation models.

In this section, an orography field illustrated in Fig. 6 will be used for the adaptation metric for the generation of various example meshes.

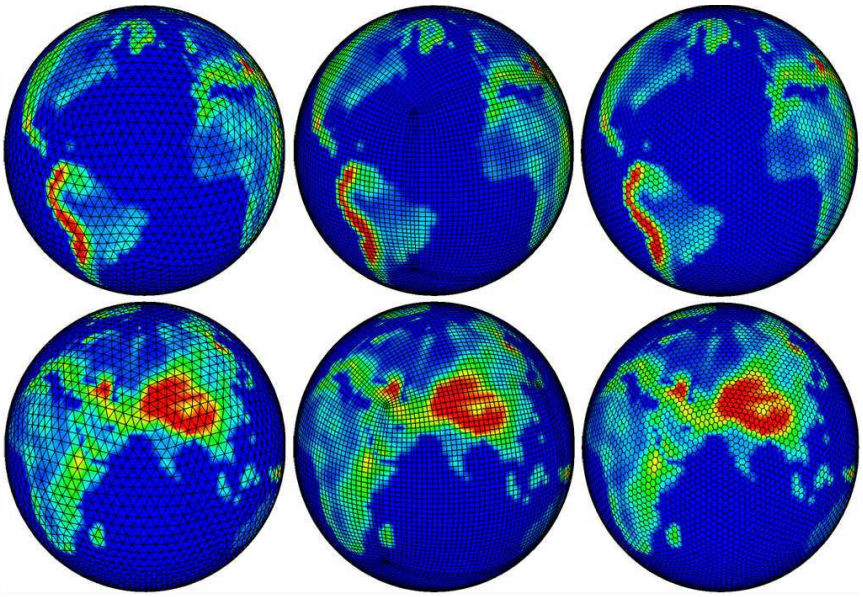

Figure 7: Illustration of triangle (left images), quadrilateral (center images), and polygonal/icosahedra surface meshes (right images) superimposed over the orographic data shown in Fig. 6.

## 4.1. Arbitrary polygonal surface mesh adaptation

This study proposes a surface mesh generation algorithm effective for general element shapes (*c.f.*, Fig. 7). Arbitrary polygonal meshes are of interest as they often combine the advantages of both structured and unstructured meshing approaches in that high degree polygonal/icosahedral meshes

- are often the best choice for solving symmetric computational problems (*e.g.,* no singular points are required to mesh a sphere),
- have the properties of producing symmetric higher order orthogonal meshes and do not introduce artificial geometric interfaces, and
- the geometry of the mesh and any problem symmetry may more effectively support the analytical and numerical methods used to solve the governing equations.

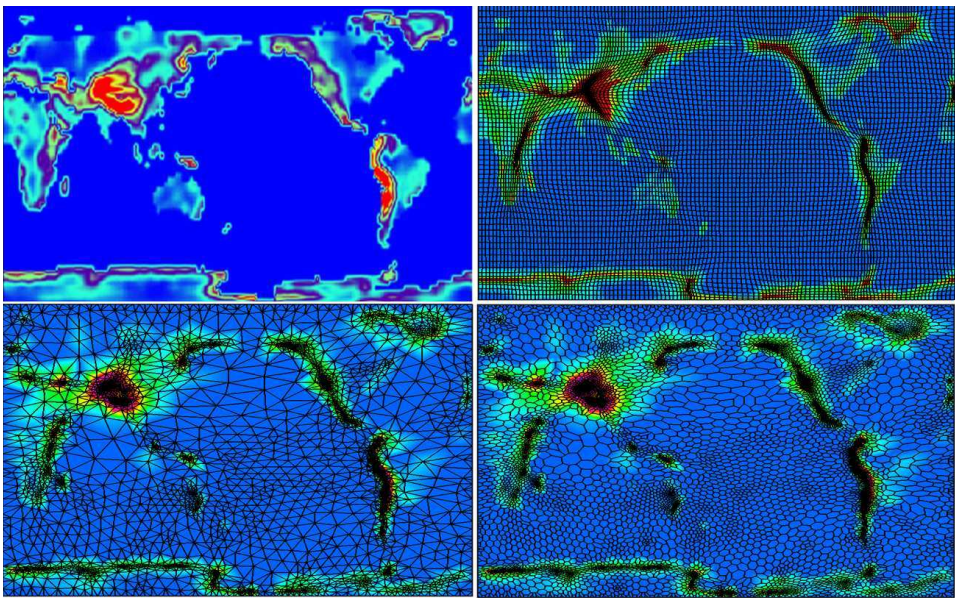

Figure 8: $r$-adaptive planar polygonal mesh generation shown on a planar projection of the above orographic data. In clockwise order from the upper-left, these figures depict the orographic data, quadrilateral element results, polygonal element results, and triangular mesh results. These results were created using $\lambda_\kappa = 0.2$, $\lambda_s = 0.3$, and $\lambda_a = 0.5$.

High degree polygonal elements are not a panacea, however. Polygons of low degree (triangles and quadrilatera), better support many geometric cases and problems:

- Layers of quadrilatera close to a boundary exhibit excellent orthogonality and clustering capabilities characteristic of structured mesh generation approaches.
- Quadrilatera more readily support the implementation of multi-grid convergence acceleration schemes, implicit methods, and may also result in memory savings.
- Unstructured triangular elements are well suited for element adaptation.

## 4.2. $r$ Adaptive polygonal surface mesh generation

Given the selection of a general, arbitrary polygonal meshing strategy, it is now necessary to adapt these meshes to the orographic field. Mesh motion, or $r$ adaptation, will be considered initially. This approach begins with the generation of a mesh with a prescribed node density on the sphere. The mesh nodes are then repositioned on the surface to more useful locations, obtained from the solution of the following equation system using the finite-volume method described in Sec. 2,

$$\frac{\partial}{\partial u}\frac{\mu g_{22}\phi_u - \mu g_{12}\phi_v}{J} + \frac{\partial}{\partial v}\frac{\mu g_{11}\phi_v - \mu g_{12}\phi_u}{J} = 0.$$

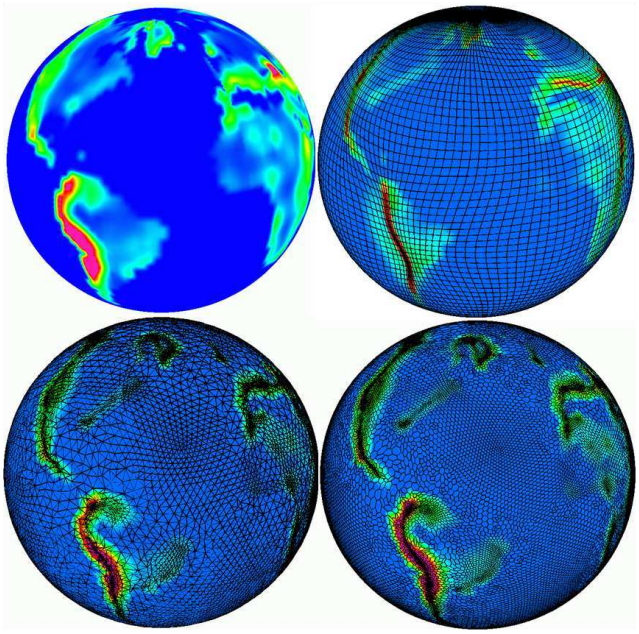

Figure 9: $r$-adapted polygonal surface mesh over the Andes and Rockies. In clockwise order from the upper-left, these figures depict the orographic data, quadrilateral element results, polygonal element results, and triangular mesh results. These results were created using $\lambda_\kappa = 0.2$, $\lambda_s = 0.3$, and $\lambda_a = 0.5$.

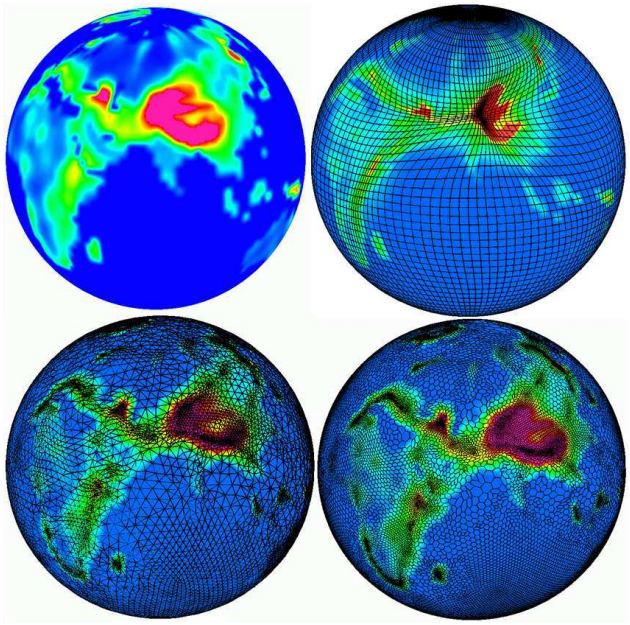

Figure 10: $r$-adapted polygonal surface mesh over the Alps and Himalayas. In clockwise order from the upper-left, these figures depict the orographic data, quadrilateral element results, polygonal element results, and triangular mesh results. These results were created using $\lambda_\kappa = 0.2$, $\lambda_s = 0.3$, and $\lambda_a = 0.5$.

Figures 8, 9, and 10 were generated using the mobility function (3.1), where $\lambda_\kappa = 0.2$ and $\lambda_s = 0.3$ ($\lambda_a = 0.5$), which illustrate the results of this solution procedure for meshes

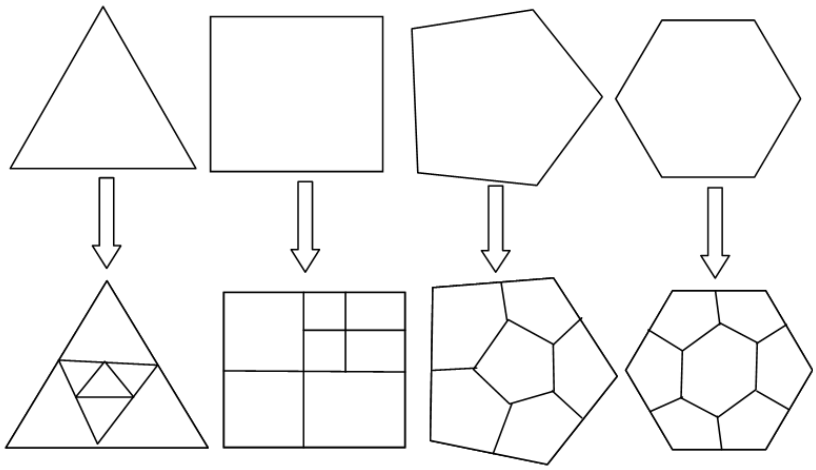

Figure 11: $h$-refinement structure of polygonal mesh cells. Local refinement using element subdivision results in the creation of "hanging nodes" along edges of elements at refinement level boundaries (see text).

employing various element shapes. From inspection of these diagrams, the mesh nodes appear to have been repositioned such that the elements are smaller in area over locations on the sphere with larger orography values corresponding to mountainous regions. From these results, the inherent smooth transition in element size provided by $r$ adaptation is apparent. It is also clear that spatial adaptation using this approach generally decreases element orthogonality to obtain adaptation, particularly in elements near the adapted regions. These results support the conclusion that $r$ refinement provides

- adaptation without increasing the total number of elements in the problem by using the existing elements in a more effective manner,
- adaptation at a cost in element geometric quality, and
- only a limited degree of node concentration.

### 4.3. $h$ Adaptive polygonal surface mesh generation

The second mesh adaptation method considered is termed the $h$ refinement method, or element subdivision. This method is based on a mesh cell subdivision strategy that creates smaller elements within each parent polygon, as shown in Fig. 11. To implement this approach, a mesh is initially generated using an element size too large to support the actual computation; this mesh may only resolve larger geometric features contained in the domain. Depending on a selected refinement criteria, the mesh elements are then subdivided until a mesh density function is less than a prescribed tolerance or convergence metric value. When $h$ adaptation is used, "hanging nodes" are created along edges where subdivided elements are adjacent to non-subdivided ones, as seen in Fig. 11. These nodes are placed mid-way between the endpoints of the edges.

In the case of $r$ adaptation, the node mobility function combines various solution and geometric data to determine the extent to which nodes were moved within the domain. In this case, it is possible to use similar mobility information to construct a mesh density function,

$$\rho = (\mu - \sigma),$$

where $\mu$ is obtained from (3.1) and $\sigma \geq 0$ is a user-specified refinement threshold. The $h$ adaptation algorithm proceeds by calculating $\rho$ in each element of the mesh and subdividing those elements in which $\rho > 0$. In this implementation, mesh coarsening was not attempted. The addition of mesh coarsening, although conceptually straightforward, would add significant complexity as only certain element combinations are good candidates for coarsening due to their edge relationship with each other (often only those elements that resulted from a previous refinement operation).

Figures 12 and 13 show the results of $h$ adaptation driven by orographic intensity. Values of $\lambda_a = 0.75$ and $\lambda_s = 0.25$ were used, and $\sigma$ was selected such that the elements in the open ocean were not subdivided. Spatial refinement provided by element subdivision is clear in all three figures; the quadrilateral, triangle, and polygonal meshes adapt to the orographic field over mountainous regions.

### 4.4. $rh$ Adaptive polygonal surface mesh generation

The final algorithm to be presented here is based on a combination of both $r$ and $h$ refinement. There are several strategies that might be examined to combine these methods:

1. Explicitly apply the above $h$ adaptation algorithm and follow this by one or more iterations of an $r$ refinement approach to smooth the result,
2. Explicitly interleave $h$ adaptation between iterations of $r$ until both algorithms have simultaneously converged,
3. Implicitly converge both $h$ and $r$ algorithms simultaneously, and/or
4. Select one of the above strategies and interleave it with selected topology operations (edge and face swapping) to further improve mesh quality.

This study employs the last approach. Fig. 14 illustrates the results of $rh$ refinement implemented using the following algorithm:

1. Apply $h$ refinement using values of $\lambda_a = 0.75$ and $\lambda_s = 0.25$.
2. Apply 10 iterations of $r$ refinement using $\lambda_\kappa = 0.2$ and $\lambda_s = 0.3$ ($\lambda_a = 0.5$).
3. Perform edge-swapping and deletion, employing the STRIPACK code [27].
4. Repeat steps 1–3, above, three more times.

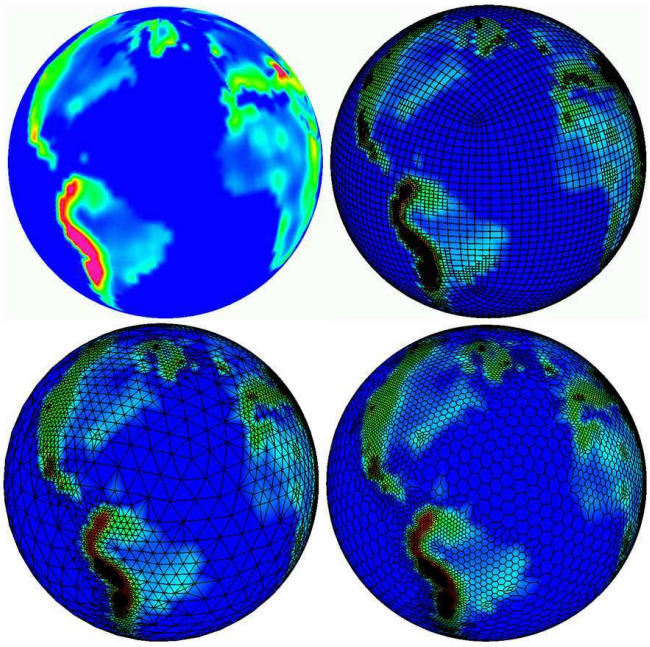

Figure 12: $h$ adapted polygonal surface mesh over the Andes and Rockies. In clockwise order from the upper-left, these figures depict the orographic data, quadrilateral element results, polygonal element results, and triangular mesh results. These results were created using $\lambda_s = 0.25$ and $\lambda_a = 0.75$.

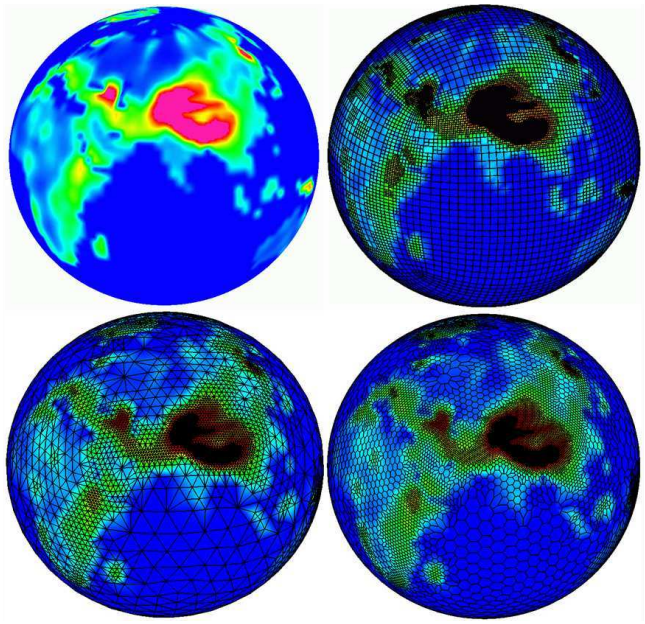

Figure 13: $h$ adapted polygonal surface mesh over the Alps and Himalayas. In clockwise order from the upper-left, these figures depict the orographic data, quadrilateral element results, polygonal element results, and triangular mesh results. These results were created using $\lambda_s = 0.25$ and $\lambda_a = 0.75$.

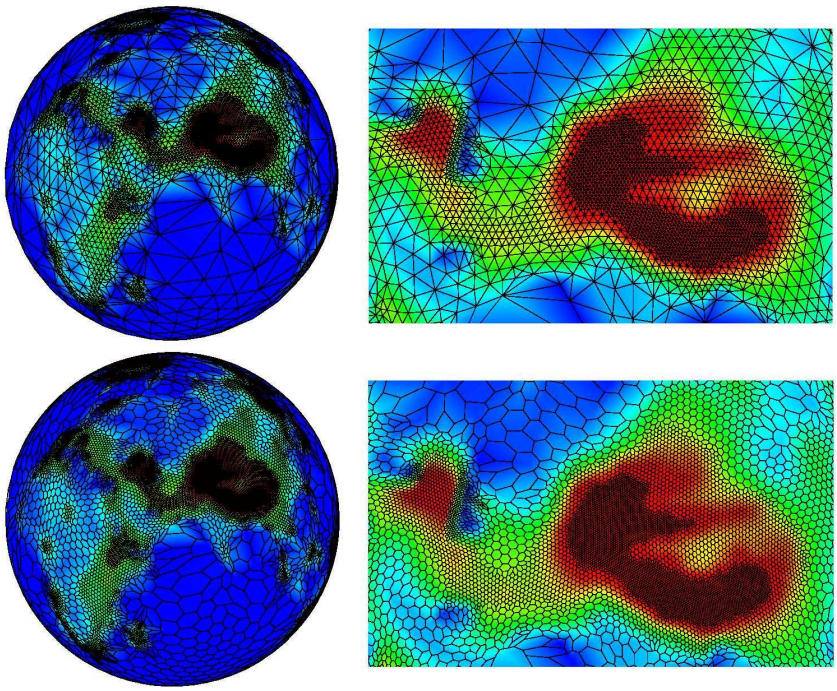

Figure 14: $rh$ adapted polygonal surface mesh over the Alps and Himalayas. The upper figures depict triangular mesh results, while the lower figures show results using polygonal elements. These results were created using $\lambda_a = 0.75$ and $\lambda_s = 0.25$ in the $h$ adaptation phase, and $\lambda_\kappa = 0.2$, $\lambda_s = 0.3$ and $\lambda_a = 0.5$ in the $r$ adaptation phase.

These results show that $r$ adaptation provides limited smoothing of the results of element subdivision. Further, at least visually, this result appears to combine the degree of spatial refinement available using $h$ refinement while smoothing the abruptness of the transition between levels of refinement somewhat. Edge topology operations were used to provide these results as $rh$ refinement, used alone, resulted in several elements with very large aspect ratios. Upon inspection, there are elements in these illustrations that possess aspect ratios approaching 5:1. Element aspect ratio is governed by the interaction of the edge swapping algorithm (if employed), $\lambda_\kappa$ and $\lambda_s$ in the $r$ adaptation algorithm, and to a lesser extent, $\sigma$ in the $h$ adaptation method. These results deliberately employ a strong $r$ adaptation weighting to motivate the degree of adaptation that can be achieved.

## 4.5. Orography field transfer

This study will now consider the application of $rh$ adaptation to a representative set of orography data, for use as a multiscale data representation application. Figure 15 shows a very dense orography field represented on a uniform planar mesh with two kilometer spatial resolution. The field data file size is two gigabytes and the file was obtained from `http://www.ngdc.noaa.gov/mgg/topo/globe.html`. The use of a uniform planar mesh to store orography data is relatively inefficient; in this case all elements have a characteristic length of two kilometers. This characteristic length is not small enough to store

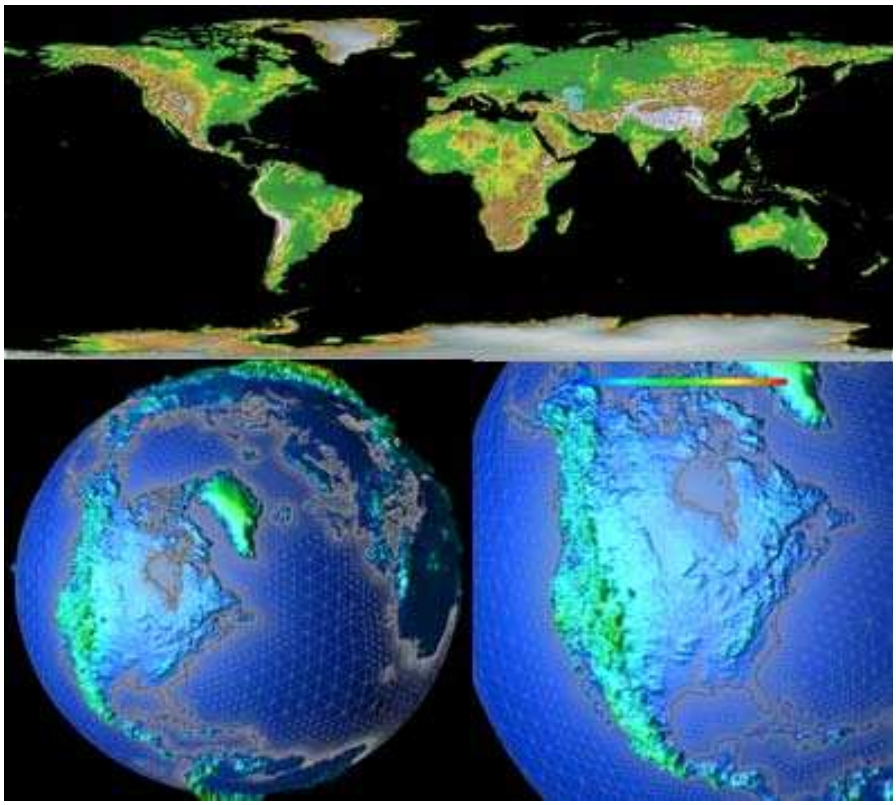

Figure 15: Coupled orography field transfer with $rh$ adaptivity. Planar orography field (upper-diagram), $rh$ adapted surface mesh (lower-left) and closeup view of the mesh (lower-right). Refined mesh has a similar information content to the source data, but is much coarser at sea level (50 kilometer elements) and finer in high elevation regions (1 kilometer elements), resulting in aggregate data set size reduction of 90%.

fine elevation detail in rapidly varying areas such as mountain ranges, but is excessively fine out in the open ocean. Orography data storage is a typical multiscale problem; adaptation is needed to store this information efficiently. This section outlines a more effective storage mechanism that is based on $rh$ adaptation.

The two kilometer element size of the source data is not sufficiently fine for this application. This study thus interpolates the source data to finer scales by using a combination of $h$ and $p$ adaptation ($p$ adaptation is based on the selection of higher order interpolating polynomials within an element to more accurately represent the intra-element solution). This $hp$ adaptive data fit strategy a least-squares method that employs spherical harmonic basis functions selected to give a reasonable fit of the source data. This $hp$ representation is used to generate the $rh$ mesh that will store the final result.

This procedure begins with the source two kilometer uniform planar mesh and its corresponding orography field, and a coarse triangular mesh of 50 kilometer resolution. The algorithm presented in [28] is used to perform the spatial search to determine which elements of the source mesh intersect with elements of the coarse target mesh. The elevation is mapped onto the nodes of the coarse mesh using the least-squares method with spher-

ical harmonics. Next, the basis functions are used to query the value of the elevation at the barycenter of the triangles on the coarse mesh. This barycenter value is then compared with a linear interpolation of the values contained on the uniform mesh. If the difference is greater than a small value (0.1%), the element is split into three children and the process is repeated until no further subdivisions are possible given the tolerance. This result is then smoothed using the above $r$ adaptation procedure. The final mesh (lower-right image of Fig. 15) is quite coarse at sea level (50-kilometers) and fine in the high altitude regions (1-kilometers). The resulting data file was only a fraction (10%) of the original file size.

Other applications of this approach include field transformation and mapping between multiple meshes (physics data remapping).

## 5. Conclusions

The paper presents a development of algorithms and methods for the generation, optimization, and adaptation of polygonal meshes for climate modeling applications. These meshes provide more flexibility than traditional approaches as a polyhedral mesh may be composed of polygonal planar cell structures ranging from hexagons, pentagons, quads, to simple triangles.

Algorithms were presented that implement $r$, $h$, and $rh$ adaptation using orographic data on the sphere. Each algorithm provides the ability to adaptively refine a polyhedral mesh; the combined approach attempts to combine the benefits of both $r$ and $h$ refinement. These results also suggest that edge topology operations may be necessary for high quality $rh$ meshes.

The $rh$ method employed here is based on using a total of four outer iterations, each consisting of an $h$ refinement operation that subdivides each cell based on the value of a mesh density function that is related to the node mobility expression employed in the $r$ refinement to follow. The mobility function (perhaps using different values of experimentally-derived scaling parameters $\lambda$ than that used for $h$) control a set of ten iterations of $r$ refinement that follow the $h$ operation. Finally, each element in the mesh is examined for quality and edge topology operations are applied if needed to modify the local connectivity of the mesh to improve its quality. This process then is repeated three additional iterations.

This approach is an initial attempt at coupled adaptation for climate modeling. Future study of a tighter coupling between $r$ and $h$ adaptation is likely warranted, perhaps including an implicit, simultaneous solution approach. There are possibly both solution and quality advantages to such a method. The authors also plan to extend this research to include element order elevation ($p$ refinement) in the future.

**Acknowledgments** The submitted manuscript has been authored by a contractor of the U.S. Government under Contract Nos. DE-AC05-00OR22725 and DE-AC07-05ID14517 (INL/JOU-07-13452). Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

## References


[1] D.L. Williamson. Integration of the barotropic vorticity equation on a spherical geodesic grid. *Tellus*, 20(4):642–653, October 1968.

[2] R. Sadourny, A. Arakawa, and Y. Mintz. Integration of the nondivergent barotropic equation with an icosahedral hexagonal grid for the sphere. *Monthly Weather Rev.*, 96(6):351–356, June 1968.

[3] J. F. Thompson and N. P. Weatherill. Fundamental concepts and approaches. In J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors, *Handbook of Grid Generation*, Chapter 1, pages 1–30. CRC Press, Boca Raton, FL, 1999.

[4] P. Knupp and S. Steinberg. *Fundamentals of Grid Generation*. CRC Press, Boca Raton, FL, 1994.

[5] V. D. Liseikin. *Grid Generation Methods*. Springer, Berlin, Heidelberg, New York, 1999.

[6] H. Edelsbrunner, editor. *Geometry and Topology for Mesh Generation*. Cambridge University Press, 2001.

[7] R. E. Smith. Transfinite interpolation (TFI) generation systems. In J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors, *Handbook of Grid Generation*, Chapter 3, pages 1–15. CRC Press, Boca Raton, FL, 1999.

[8] A. Khamayseh and C. W. Mastin. Computational conformal mapping for surface grid generation. *J. Comput. Phys.*, 123:394–401, 1996.

[9] A. Khamayseh and A. Kuprat. Surface grid generation systems. In J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors, *Handbook of Grid Generation*, Chapter 9, pages 9.1–9.29. CRC Press, Boca Raton, FL, 1999.

[10] S. Spekreijse. Elliptic grid generation systems. In J. F. Thompson, B. K. Soni, and N. P. Weatherill, editors, *Handbook of Grid Generation*, Chapter 4, pages 4.1–4.47. CRC Press, Boca Raton, FL, 1999.

[11] Y. Kallinderis, A. Khawaja, and H. McMorris. Hybrid prismatic/tetrahedral grid generation for complex geometries. *AIAA Paper*, 95-0211, 1995.

[12] E. D'Azevedo. Optimal triangular mesh generation by coordinate transformation. *SIAM J. Sci. Statist. Comput.*, 12:755–786, 1991.

[13] T. J. Baker. Automatic mesh generation for complex three-dimensional regions using a constrained launay triangulation. *Engrg. Comput.*, 5(3–4):161–175, 1989.

[14] D. L. Marcum and N. P. Weatherill. Unstructured grid generation using iterative point insertion and local reconnection. *AIAA J.*, 33(9):1619–1625, 1995.

[15] G. Hansen, A. Zardecki, D. Greening, and R. Bos. A finite element method for unstructured grid smoothing. *J. Comput. Phys.*, 194(2):611–631, 2004.

[16] G. Hansen, A. Zardecki, D. Greening, and R. Bos. A finite element method for three-dimensional unstructured grid smoothing. *J. Comput. Phys.*, 202(1):281–297, 2005.

[17] G. Hansen and A. Zardecki. Unstructured surface mesh adaptation using the Laplace-Beltrami target metric approach. *J. Comput. Phys.*, 225(1):165–182, 2007.

[18] W. M. Putman and S.-J. Lin. Finite-volume transport on various cubed-sphere grids. *J. Comput. Phys.*, 227:55–78, 2007.

[19] Y. Di, R. Li, T. Tang, and P. W. Zhang. Moving mesh methods for singular problems on a sphere using perturbed harmonic mappings. *SIAM J. Sci. Comput.*, 28:1490–1508, 2006.

[20] J. U. Brackbill. An adaptive grid with directional control. *J. Comput. Phys.*, 108:38–50, 1993.

[21] J. Saltzman. Variational methods for generating meshes on surfaces in three dimensions. *J. Comput. Phys.*, 63:1–19, 1986.

[22] Z. U. A. Warsi. Mathematics of space and surface grid generation. In J. F. Thompson, B. K.

Soni, and N. P. Weatherill, editors, *Handbook of Grid Generation*, Chapter 2, pages 2.1–2.46. CRC Press, Boca Raton, FL, 1999.

[23] V. F. de Almeida. Domain deformation mapping: Application to variational mesh generation. *SIAM J. Sci. Comput.*, 20(4):1252–1275, 1999.

[24] M. Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 4. Publish or Perish Press, Berkeley, CA, 2000.

[25] A. Kuprat. Modeling microstructure evolution using gradient-weighted moving finite elements. *SIAM J. Sci. Comput.*, 22:535–560, 2000.

[26] K. Miller. A geometrical-mechanical interpretation of gradient-weighted moving finite elements. *SIAM J. Num. Anal.*, 34:67–90, 1997.

[27] R. J. Renka. Algorithm 772: STRIPACK: Delaunay triangulation and Voronoi diagram on the surface of a sphere. *ACM Trans. Math. Software*, 23(3):416–434, 1997.

[28] A. Khamayseh and G. Hansen. Use of the spatial kD-tree in computational physics applications. *Commun. Comput. Phys.*, 2:545–576, 2007.