Journal of Computational Physics **149,** 114–127 (1999)
Article ID jcph.1998.6146, available online at http://www.idealibrary.com on IDEAL®

A Scheme for Generating Unstructured Grids on Spheres with Application to Parallel Computation

Giri Chukkapalli,*[,1] Steve R. Karpik,† and C. Ross Ethier*[,2]

*Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Ontario, Canada and †Ontario Science Centre, Toronto, Ontario, Canada
E-mail: ethier@mie.utoronto.ca

Received June 16, 1998; revised October 22, 1998

Numerical solution of differential equations on the surface of the sphere requires grid generation. Examples include numerical simulations of mantle convection, weather, and climate. Because of their ability to offer local resolution at modest computational cost, unstructured grids are attractive in this context. However, unstructured grids suffer from drawbacks such as high computational overhead and inefficient generation schemes. Here we present a scheme for generating unstructured grids on the surface of the sphere that overcomes these limitations. We also show how the scheme can be easily used to allow efficient domain decomposition for parallel computations. The surface of the sphere is covered with a spherical spiral, which is used to provide an underlying structure for the grid. The spiral is populated by nodes, which are then connected using an advancing front technique to generate near-equilateral spherical triangular elements. Methods for producing local grid refinement by adjusting the pitch of the spherical spiral are discussed, as is the extension of the method to the case of coupled pressure–velocity solvers. The same general idea of a spherical spiral also serves as the starting point for an algorithm to subdivide the grid into subdomains for parallel computation. The resulting unstructured grids are generally of very high quality: in uniform grids, 99.4% of the elements have areas between 90 and 107% of the mean element area, and 99.8% of the edges have lengths between 84 and 132% of the mean edge length. The quality of the grids increases with mesh density. Partitioning of the nodes and elements produces well-balanced and compact subdomains, with a maximum load imbalance that is small and rises gradually with number of subdomains. The proposed scheme produces grids that combine the benefits of an unstructured mesh with the structure conferred by the underlying spherical spiral. For example, this underlying structure greatly

[1] Present address: San Diego Supercomputing Centre, San Diego, CA.

[2] Supported in part by an NSERC E.W.R. Steacie Fellowship Grant (CRE).

0021-9991/99 $30.00

facilitates tasks such as element searching. This scheme is an attractive alternative for generating unstructured triangular grids on the sphere.

Key Words: grid generation; finite element mesh; parallel computation; sphere; numerical weather algorithm; domain decomposition.

INTRODUCTION

Numerical simulations of weather and climate require grid generation on the surface of the sphere. Although the computational effort spent in grid generation is relatively small, the grid influences every aspect of the subsequent computation, and the grid generation step is therefore very important. An extensive analysis of the relative merits and demerits of different grid generation schemes on the sphere can be found in Willliamson [1]. Here we briefly review the two main types of grids currently used for solving partial differential equations (PDEs) on a sphere, namely longitude–latitude grids and quasi-uniform grids.

Longitude–latitude grids are structured grids, with grid lines aligning with the spherical coordinates. These grids have the advantages of being fairly simple to generate and of being amenable to use with spectral or finite difference schemes for solution of the governing PDEs. They suffer from severe crowding of grid points near the poles. This implies excess computational expense without any gain of accuracy and in certain schemes can lead to time step limitations and/or the need for filtering schemes to eliminate high frequency waves near poles. Grid crowding can be avoided by using reduced grids at extreme latitudes, but this results in hanging nodes and pseudo-reflections at regions of abrupt grid density change. Kelly and Williams [2] found that forecast accuracy improves markedly by use of smoothly varying grids.

Quasi-uniform grids are unstructured grids obtained by subdividing uniform tesselations of the sphere. Because a sphere can be divided into at most 20 identical elements, higher resolution grids based on such subdivisions must be quasi-uniform. The most popular way to generate quasi-uniform grids is to project the vertices of an icosahedron inscribing the sphere and to then bisect the edges of the resulting triangles. Quasi-uniform grids are convenient to use with finite element based solution schemes for the governing PDEs (e.g., Cullen and Hall [3]), but they can also be used for finite difference discretizations (e.g., Williamson [4]). Quasi-uniform grids avoid poleward crowding and related problems, and are attractive due to renewed interest in finite element spatial discretization schemes that can handle such grids efficiently (e.g., Heikes and Randall [5, 6]). However, quasi-uniform grids have disadvantages, including complex data structures, inability to generate the required resolution, and inflexibility in generating variable resolution grids.

The purpose of this manuscript is to describe a new scheme for generating variable-resolution, unstructured, triangular grids on the sphere that overcome many of the above-mentioned problems. The basic concept involves using a spherical spiral to produce grids that have a simple underlying structure, even though the grids are “unstructured” in the finite-element sense. This simple underlying structure can be exploited in several ways to produce good quality grids that are well suited to both serial and parallel computations.

METHODS

Our scheme for generation of unstructured grids is based on the use of a spherical spiral, i.e. a spiral which lies on the surface of a sphere [7]. Spherical spirals can be characterized

by the orientation of their axis and by their pitch, defined as the number of revolutions they make on the sphere, n. Spherical spirals aligned with the north–south axis are described by

$$\phi = f(\theta), \quad -\pi/2 \le \phi \le \pi/2, \quad 0 \le \theta \le 2n\pi, \tag{1}$$

where f is any continuous, positive, monotonic function, and θ and ϕ are the longitude and latitude coordinates, respectively.

Node Placement

The first step in the process is to generate a nodal distribution along a spherical spiral. The nodes are placed on the spiral starting at one end and moving toward the other end. The distance between consecutive nodes, e, is chosen so that triangular elements formed by joining nodes from consecutive revolutions of the spiral will be approximately equilateral. Denoting the latitudinal distance between two subsequent turns of the spiral by $h(\theta, \phi)$ and assuming a small triangular element, we therefore require

$$e(\theta, \phi) = 2h(\theta, \phi)/\sqrt{3}. \tag{2}$$

The quantity e is computed by noting that the distance ds between two points on the unit sphere separated by $d\theta$ and $d\phi$ is given by

$$ds = \sqrt{\cos^2\phi + \left(\frac{d\phi}{d\theta}\right)^2}\, d\theta \tag{3}$$

so that the distance s_{12} between two points $P_1(\theta_1)$ and $P_2(\theta_2)$ lying on the spiral is

$$s_{12} = \int_{\theta_1}^{\theta_2} \sqrt{\cos^2 f(\theta) + f'^2}\, d\theta. \tag{4}$$

Thus, given a function $f(\theta)$ with an associated inter-turn spacing distribution $h(\theta, \phi)$, and a point located at $(\theta_1,\ f(\theta_1))$, the angular location θ_2 of the subsequent point is given implicitly by

$$h(\theta_1,\ f(\theta_1)) = \frac{\sqrt{3}}{2} \int_{\theta_1}^{\theta_2} \sqrt{\cos^2 f(\theta) + f'^2}\, d\theta \tag{5}$$

which can be evaluated by numerical methods.

Uniform Spiral

A special case arises if the function f is linear, in which case the spiral is said to be uniform and has the governing equation

$$\phi = \tfrac{1}{2}(\theta/n - \pi). \tag{6}$$

Such a spiral has uniform pitch, and on the unit sphere $h(\theta, \phi)$ is therefore constant and equal to π/n. Equation (4) takes the simplified form

$$s_{12} = \sqrt{1 + 4n^2}[E(\phi_2, m) - E(\phi_1, m)], \tag{7}$$

where E is the incomplete elliptic integral of the second kind with parameter $m = 4n^2/(4n^2+1)$ and ϕ_i is ϕ evaluated at θ_i, $i = 1, 2$. For $n \gg 1$, which is usually the situation in practice and for ϕ_i away from the poles, Eq. (7) simplifies to

$$s_{12} = 2n[\sin(\phi_2) - \sin(\phi_1)] \tag{8}$$

which can be combined with Eq. (2) to give a closed-form expression for the location θ_2 for a given θ_1. Also, for $n \gg 1$, the total number of nodal points on the spiral, N, is given by the approximate relation

$$N = \frac{2\sqrt{3}}{\pi} n^2. \tag{9}$$

Using this relation, the pitch of a uniform spiral can be chosen *a priori* so as to give a desired number of nodal points.

Nonuniform Spiral

A high-quality variable resolution mesh can be generated from a nonuniform spiral as follows. A function $f(\theta)$ is specified that creates a tighter spiral near one of the poles, which implies high node density near that pole. For example, by increasing the pitch along the spiral axis, a higher nodal density is achieved toward the second spiral pole. Typically, a constant, high pitch value is used within a high density "window" centred on the second spiral pole, and the pitch value is gradually reduced outside this window to a lower value. This results in a smoothly graded grid with local refinement around the second spiral pole. In order to generate a higher nodal density in an arbitrary latitude–longitude window, the spiral axis is rotated such that one of the spiral poles resides at the centre of the window.

Triangular Element Generation

The next step of the procedure is to create a triangular unstructured finite element mesh using the nodal points generated above. Rather than using Delaunay triangulation for this purpose, we employ a moving front technique that exploits the underlying linear structure of the nodal points lying on the spherical spiral. Triangulation is started from one pole of the spiral by joining nodes 1 and 2 with a geodesic to form the edge 1–2. This "active" edge is then used to seed the following algorithm (Fig. 1):

Given an active edge with endpoints P_1 and P_2:

- Form a test spherical triangle from points P_1, P_2, and $P_1 + 1$.
- Form a second test spherical triangle from points P_1, P_2, and $P_2 + 1$.
- Choose the test triangle that is closest to equilateral, and reject the other one. If the first test triangle is chosen, the new active edge becomes P_2–$P_1 + 1$; otherwise, the active edge becomes P_1–$P_2 + 1$.

Repeat until the second pole of the spiral is reached.

Vector-Scalar Placement on the Grid

An important special situation arises when the PDEs to be solved on the sphere are derived from the incompressible Navier-Stokes equations, e.g. the shallow water equations.

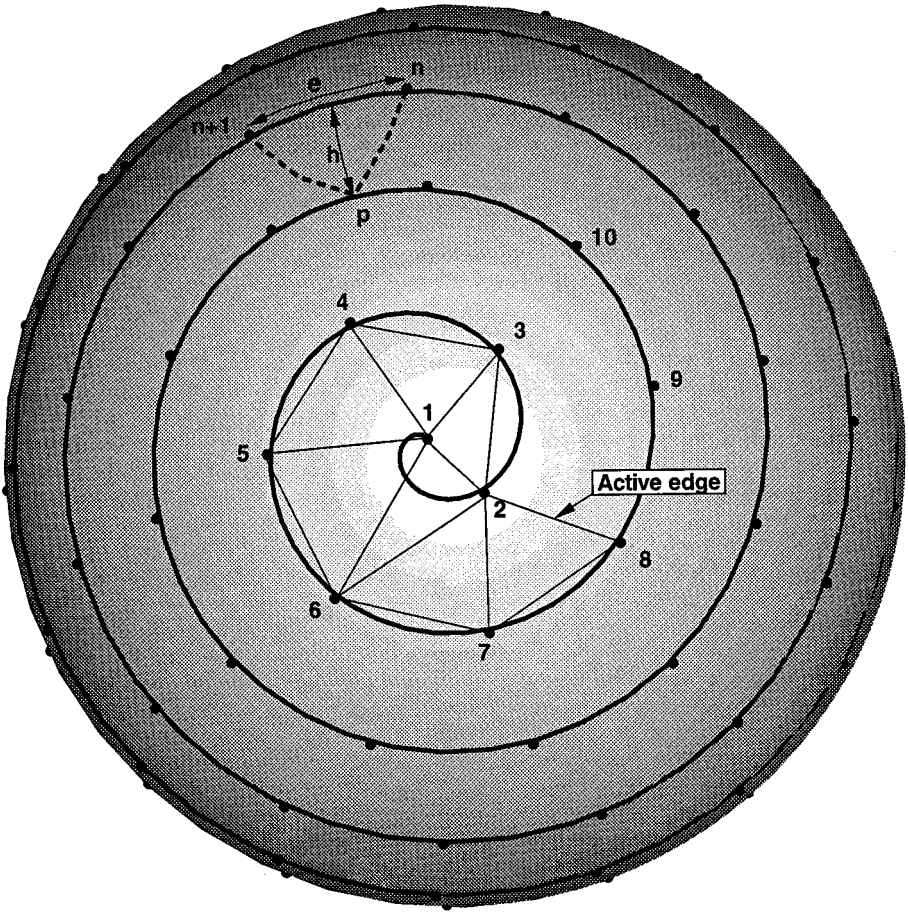

FIG. 1. Partially constructed triangular mesh showing the node placement procedure and the triangulation procedure. Triangulation procedure (upper portion of diagram); suppose that node n has been placed and node $n+1$ is about to be placed. The placement of node $n+1$ is governed by the distance e, which is computed from Eqs. (2)–(5) for the triangle defined by nodes n, $n+1$, and p. Triangulation procedure (centre of diagram): in this schematic, nodes 1 to 10 lie on the spiral, with node 1 coinciding with the spiral pole. As shown, the active edge is 2–8, and the two test triangles are 283 and 289. Triangle 289 will be accepted because it is closest to equilateral, and the active edge will advance to 2–9.

For numerical reasons, the pressure unknowns and the velocity component unknowns must usually be defined on different grids in such situations (e.g., Arakawa and Lamb [8]). In the finite element context, this requirement is known formally as the Brezzi–Bubuska (BB) condition [9]. Choosing the velocity basis functions to be polynomials of one order higher than the pressure, basis functions typically satisfies the BB condition, e.g. a linear approximation space for pressure, coupled with a quadratic approximation space for velocity. However, the calculation of inner products of the fields and their derivatives becomes cumbersome when higher order basis functions are employed on spherical grids.

An attractive way of avoiding this problem is to use the so-called pseudo P_2–P_1 element, first introduced by Bercovier and Pironneau [9]. Such elements, which satisfy the BB condition, approximate both the velocity and pressure fields using linear basis functions. However, the velocity basis functions are defined on elements formed by joining

the midpoints of the sides of the pressure elements with geodesics. Thus, every pressure element contains four velocity elements. When the pseudo P_2–P_1 element is used, the pressure grid is a subset of the velocity grid; thus additional positional coordinates need not be stored or calculated at every time step. Another important advantage of such elements, in the case of semi-Lagrangian treatment of the advection terms, is that expensive departure point calculations need not be performed for pressure and velocity grid points separately.

If pseudo P_2–P_1 elements are used for spatial discretization, the fill pattern of the matrices arising from a spatial discretization of a governing equation on the surface of the sphere will depend on the node numbering system used. One approach is to number pressure nodes "naturally," i.e. in the order in which they appear on the spiral. Velocity nodes can be numbered in a similar fashion, with the added complexity that the node number for each P_1 element midside node is "inserted" into the numbering list following the first corner node attached to that midside node. This scheme results in banded matrices with a bandwidth approximately equal to the number of nodes required to complete one wrap of the spiral.

Parallel Computation Issues

For many large-scale numerical problems, it is desirable to carry out the computations in parallel. A number of partitioning schemes for unstructured grids have been proposed in the literature [10, 11], but many of them suffer from drawbacks of one type or another. For example, most popular partitioning schemes for unstructured grids are based on recursive bisection algorithms. These are restricted to partitioning the domain such that the number of subdomains is a power of two, and thus they are not sufficiently flexible in some situations. Furthermore, some partitioning algorithms (e.g., spectral bisection) are relatively expensive. Finally, existing algorithms balance loads well between processors, but do not necessarily minimize interprocessor communication. Here we show how our grid generation scheme is consistent with an efficient partitioning of the computational space that overcomes these problems. This partitioning scheme is based on a data decomposition paradigm. For simplicity, we assume that the amount of data per node and per element is approximately uniform across the computational domain, so that data partitioning is equivalent to partitioning the total number of nodes and elements in the physical domain. A suitable partitioning scheme should have the following attributes:

1. It should be simple and economical to implement.
2. It should be flexible in generating the required number of partitions, including cases where the number of partitions is large.
3. It should produce a balanced load across all the processors. Without *a priori* knowledge of the computational load per node, we simply assume that balancing the number of grid points across all processors will produce a balanced load. Failure to produce balanced loads will give nonoptimal speedup on parallel architectures, particularly on machines with relatively weak processors.
4. It should minimize the inter-subdomain communications. In many applications, this implies that the subdomains should be compact, and we have taken this as a requirement for our partitioning scheme.

A suitable algorithm that uses the underlying spiral structure of our nonuniform grids to partition the spherical surface grid into P subdomains is as follows:

• Generate a spherical spiral (of suitable pitch) that contains P nodes distributed approximately uniformly on the surface of the sphere. These nodes will lie approximately at the centre of each subdomain and will be referred to as "processor nodes."

• For each processor node, identify and store the closest node from the numerical grid. These closest grid points will be referred to as "seed nodes" and will be used to start the following form of the greedy algorithm for accumulation of neighbouring grid points.

• For each seed node, use node-neighbour information from the connectivity table to attach all available immediate neighbour nodes to the corresponding subdomain. This creates a "shell" around each seed node.

• Repeat this process using each newly acquired node within the shell to form a second shell surrounding the seed node.

• Continue forming shells in this manner until N/P nodes are attached to the subdomain, or until no more available immediate neighbour nodes remain.

• Carry out a sweep over the entire domain to assign any remaining nodes to the adjacent subdomain with the smallest number of nodes. This is necessary for two reasons. First, the distribution of the seed nodes on the surface of the sphere can never be perfectly uniform, resulting in unacquired nodes around some subdomains. Second, some subdomains reach their limit of N/P nodes before acquiring all their neighbours. The second reason appears to be the main source of unevenness in the partitions generated using this scheme.

• Partition elements into subdomains according to which subdomain owns the majority of their nodes. Elements with three nodes belong to three different subdomains are allotted to the subdomain that has the lowest number of elements.

For cases in which the computational domain contains both a velocity and a pressure grid, the partitioning is done on the pressure grid, and the velocity grid is then partitioned. For iso P_2–P_1 elements, this is a well-defined process since the velocity grids are contained inside the pressure grids.

RESULTS

Grids Based on Uniform Spirals

A uniform spiral with pitch $n = 10$ is shown in Fig. 2, and the resulting pressure and velocity grids are shown in Fig. 3. It can be seen from Fig. 3 that the vast majority of the spherical triangular elements are well-formed (i.e., approximately equilateral). Near the second pole of the spiral grid, where meshing comes to completion, some of the elements are slightly deformed because the last node on the spiral does not usually coincide with the second spiral pole (see centre of Figs. 3b and 3d). However, these elements are not grossly deformed, and our experience is that in practice such meshes work well without the need for mesh smoothing near the second pole. Further, the deformation that occurs near the second pole diminishes as the pitch of the spiral (and thus the node density) increases.

Quantitative measures of unstructured grid quality include the distribution of elemental areas, edge lengths, and aspect ratios. Here we define the aspect ratio to be the length of the longest side divided by the altitude from this longest side. The "optimal" aspect ratio is approximately 1.1547 for small spherical triangles, which is a lower bound for the actual aspect ratios. Figure 4 shows the distribution of the above three quantities for a grid based on a uniform spiral with pitch $n = 32$. Overall, the distributions show that the resulting mesh is of good quality: 99.4% of the elements had areas between 90 and 107% of the mean element

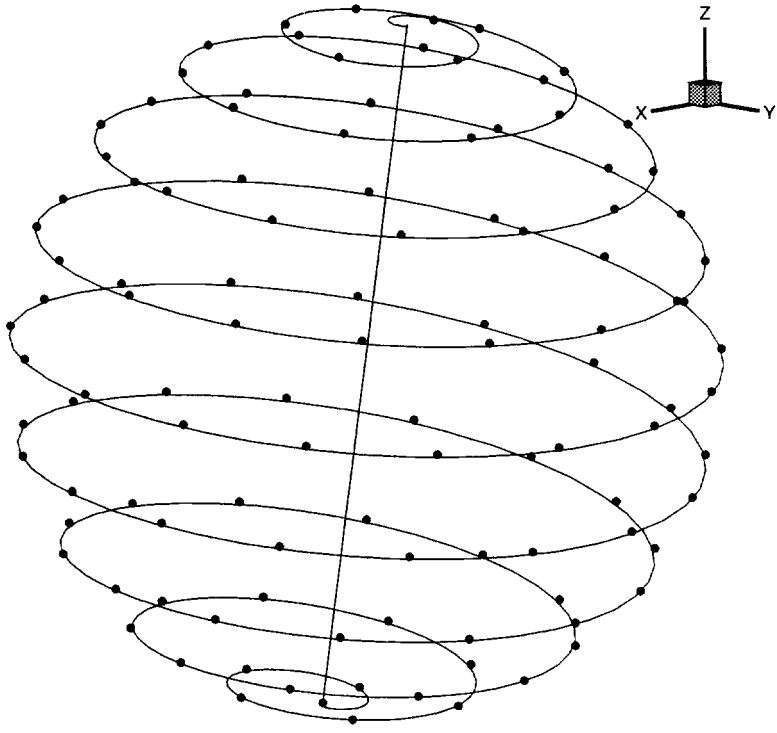

FIG. 2. A uniform spherical spiral with pitch $n = 10$. The solid line shows the spiral axis, which was inclined in the z-y plane by 10° to the polar axis. Superimposed on the spiral are nodes (black circles) generated by the procedure described in the text. This spiral was used to generate the meshes shown in Fig. 3.

area; 99.8% of the edges had lengths between 84 and 132% of the mean edge length; and 99.9% of the elements had aspect ratios less than 2.1. There were 12 elements (out of 8728) that had aspect ratios greater than 2.1. These elements were derived from subdivision of three pressure elements near the poles, and had a maximum aspect ratio of 2.54. Overall, these mesh statistics are very favourable and confirm the excellent quality of the grid.

Grids Based on Nonuniform Spirals

We present one nonuniform grid to illustrate the potential of the method. Figure 5 shows a grid with a refined zone within a circular window of arc 50° centred at (53°N, 103°W). Inside this window, the underlying spherical spiral was specified to be uniform with pitch $n = 22$. Outside this window, the effective local pitch was gradually reduced by a fixed increment of approximately -0.0015 each time a grid point was added, resulting in a final pitch at the second spiral pole of $n = 10$. This gradual reduction in pitch produces a mesh with smoothly varying element sizes, so that pseudo-reflections are avoided and solution accuracy is maintained [3].

Grid Partitioning for Parallel Computations

We illustrate the performance of the grid partitioning procedure by showing statistics for several partitioned grids. Figure 6 shows a polar view of a grid with inter-subdomain

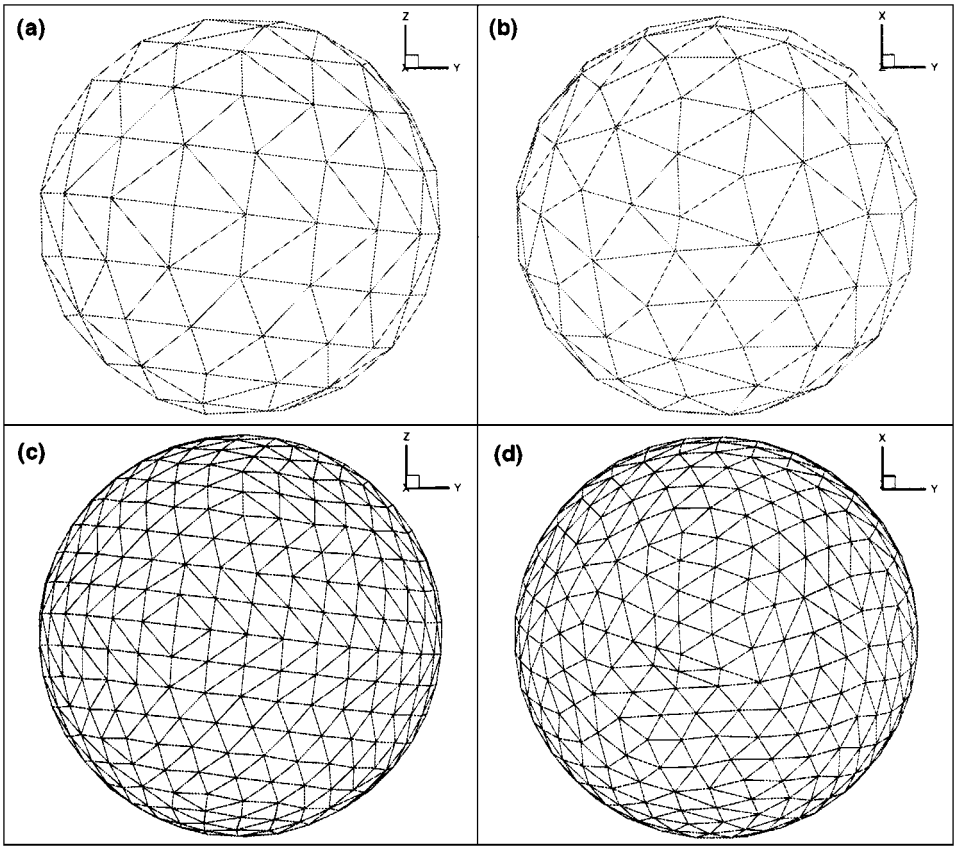

FIG. 3. Unstructured pressure and velocity grids based on a uniform spherical spiral with pitch $n = 10$ and the pseudo P_2–P_1 triangular element: panel (a) equatorial orthographic view of pressure grid; panel (b) polar orthographic view of pressure grid; panel (c) equatorial orthographic view of corresponding velocity grid; panel (d) polar orthographic view of corresponding velocity grid.

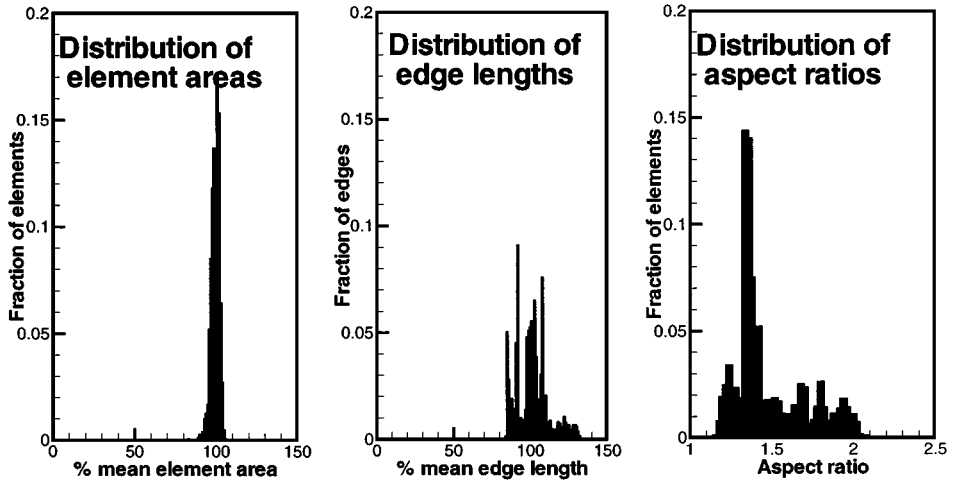

FIG. 4. Distribution of elemental areas, edge lengths, and elemental aspect ratios for the velocity mesh generated with an underlying uniform spiral of pitch $n = 32$. This mesh contains 8728 velocity elements (6546 edges). The bin size was 1% for the left panels and 0.02 for the right panel.

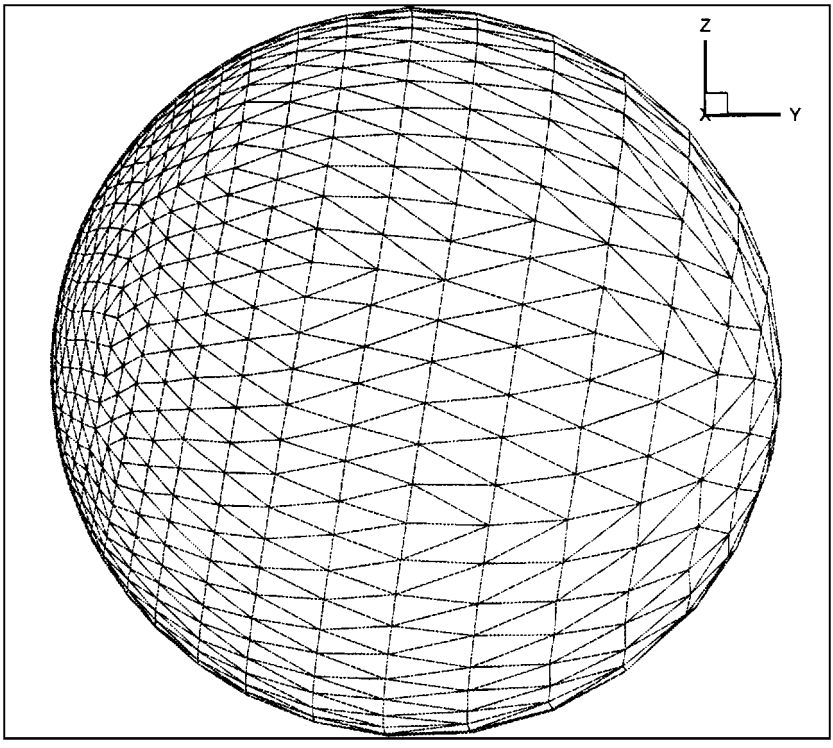

FIG. 5. Spiral geodesic grid with a local high resolution window over North America for regional forecast purposes. This grid contains 207 pressure nodes, 410 pressure elements, 822 velocity nodes, and 1640 velocity elements. See text for detailed discussion of how grid was generated.

boundaries outlined. It can be seen that the proposed algorithm generates reasonably compact subdomains, resulting in small inter-subdomain boundary lengths. Figure 7 quantifies the distribution of nodes and elements within each subdomain of a uniform mesh divided into 20 subdomains. Although the load is not precisely balanced between subdomains by this partitioning algorithm, it can be seen that a near-equal subdivision is obtained.

In order to further quantify how well the partitioning scheme distributes nodes and elements into subdomains, we define a load imbalance parameter, L_{u}, by

$$L_{\mathrm{u}} = 100(N_{\max} - N_I)/N_I \tag{10}$$

where N_{total} is the total number of nodes or elements, $N_I = N_{\mathrm{total}}/P$ is the ideal distribution of nodes or elements per subdomain, and $N_{\max}$ is the maximum number of nodes or elements actually allotted to any subdomain. Figure 8 shows how this parameter varies with the number of subdomains. The load imbalance increases with the number of subdomains, as expected. However, even for up to 38 processors the maximum imbalance is only 6%, which is acceptably small.

DISCUSSION AND CONCLUSIONS

Generation of appropriate grids on spheres is a nontrivial problem. Such grids are required in meteorological dynamics simulations, as well as in other nonmeteorological computations

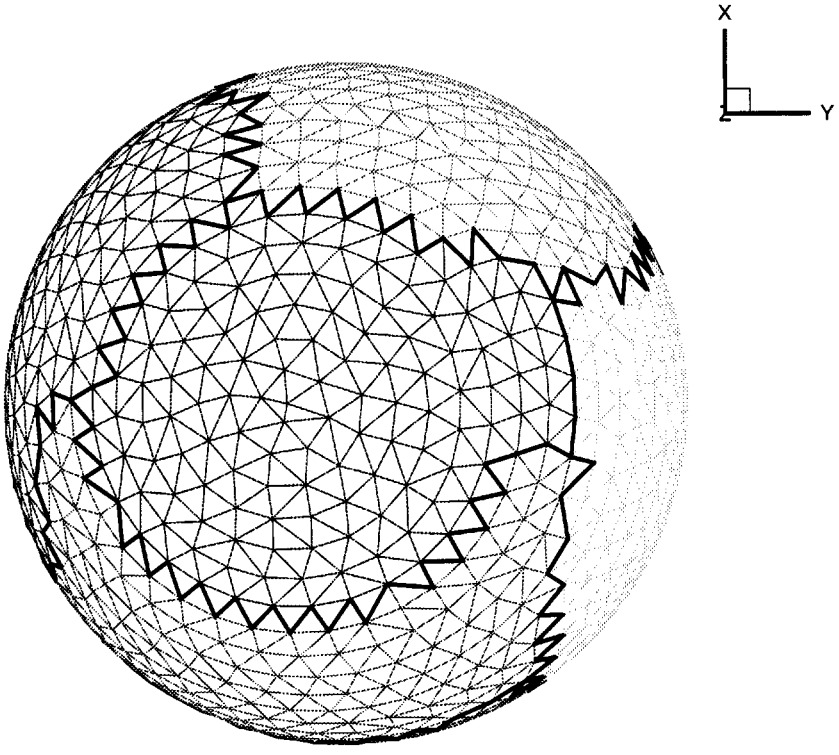

FIG. 6. Partitioning of a spherical spiral grid of pitch 32 into 8 subdomains. Heavy lines indicate subdomain boundaries.

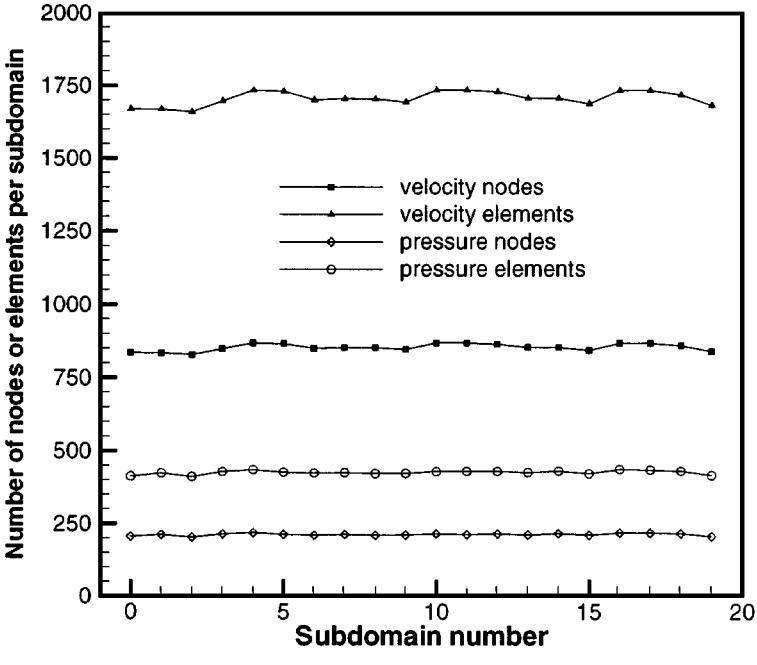

FIG. 7. Typical distribution of nodes and elements resulting from the mesh partitioning strategy described in the text. The original mesh was generated using a uniform spiral with pitch $n = 64$, and was subdivided into 20 subdomains.

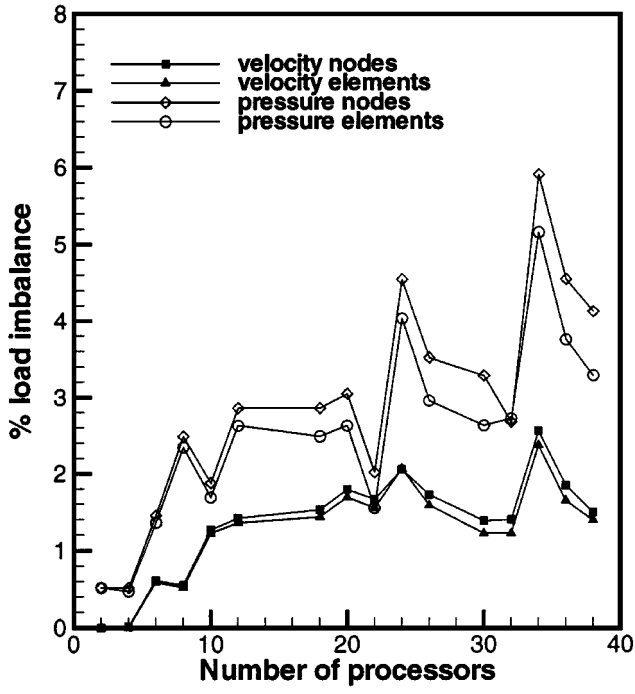

FIG. 8. The load imbalance parameter, L_u, vs the number of subdomains. See text for definition of load imbalance parameter. The original mesh was generated using a uniform spiral with pitch $n = 64$, giving 4270 pressure nodes, 17,074 velocity nodes, 8536 pressure elements, and 34,144 velocity elements.

in which the domain is the surface of a sphere. Here we have presented a scheme for the generation of unstructured triangular meshes suitable for use in finite-element based computations on the surface of a sphere. The scheme produces meshes that have a number of advantages, as listed below.

1. Grids of excellent quality are produced, as judged by the presence of near-equilateral triangles and uniform element sizes (for uniform grids). The quality of the grids has been confirmed by numerical tests in which the shallow water equations were solved on spherical domains [12]. The results showed that no unphysical reflections or grid related inaccuracies were present.

2. It is easy to produce locally refined grids, such as those required for regional weather forecasting. Producing local refinement in a window that does not coincide with a computational pole requires only a simple solid body rotation of the spiral axis. This rotation does not affect the governing equations or calculations near the computational poles. This is a distinct advantage over locally resolved structured grids, where the computational poles and the governing equations must be transformed so that the computational equator passes through the centre of the high resolution window [13]. Such a transformation has the disadvantage of making the Coriolis term dependent on both computational longitude and latitude.

3. Because grid points are not necessary at the poles, polar singularities can be avoided by slightly offsetting the spiral axis from the polar axis.

4. Although grids generated with this scheme are unstructured, they have an underlying spiral structure that can be exploited. For example, these grids were used in a semi-Lagrangian-based algorithm for solving the shallow water equations on the sphere [12].

Semi-Lagrangian algorithms require repeated solution of the element searching problem in which the element containing a point known only by its physical coordinates must be found. For a fully unstructured mesh, this portion of the algorithm can be very computationally intensive. However, the underlying spiral allows an efficient searching procedure to be constructed.

5. Because geodesics are used to form edges, each element is a spherical triangle. Since the domain is spherical, this implies that the continuous and discrete computational domains are identical, and the computational representation of the domain is exact no matter how coarse the grid is.

In addition to using a spherical spiral to generate an unstructured grid on the sphere, we have presented a technique that uses a second spherical spiral to efficiently decompose a grid into subdomains for parallel computation. The proposed decomposition scheme works well, as judged by the division of nodes and elements into compact subdomains of approximately equal size. Although there was a trend towards modest load imbalance as the number of subdomains increased, this was not judged to be severe. The proposed partitioning scheme also has the advantages that it can be done in parallel, is simple and economical to implement, and imposes no restrictions on the number of subdomains that can be generated.

Although we have not done so, there are a number of possible extensions to this grid-generation scheme. For example, the unstructured grids could form the basis of a semi-structured grid for three-dimensional spherical domains. One way of doing this would be to create prism-shaped finite elements by extending the radial vectors joining the nodes of the two-dimensional grid with the centre of the sphere. A second extension would be to use unstructured grids as generated above as a starting point for adaptive grid refinement.

REFERENCES

1. D. L. Williamson, Review of numerical approaches for modeling global transport, in *Air Pollution Modeling and Its Application IX*, edited by H. van Dop and G. Kallos (Plenum, New York, 1992), p. 377.
2. R. G. Kelly and R. T. Williams, *A Finite Element Prediction Model with Variable Element Sizes*, Naval Postgraduate School Report NPS-63Wu76101 (1976).
3. M. J. P. Cullen and C. D. Hall, Forecasting and general circulation results from finite element models, *Quart. J. R. Meteorol. Soc.* **105**, 571 (1979).
4. D. L. Williamson, Difference approximations for fluid flow on a sphere, in *Numerical Methods Used in Atmospheric Models, Vol. II* (JOC, World Meteorological Office, Geneva, Switzerland, 1979), p. 55.
5. R. Heikes and D. A. Randall, Numerical integration of shallow-water equations on a twisted icosahedral grid. Part I. Basic design and results of tests, *Mon. Weather Rev.* **123**, 1862 (1995).
6. R. Heikes and D. A. Randall, Numerical integration of 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**, 1880 (1995).
7. R. Dixon, Green spirals, in *Spiral Symmetry*, edited by I. Hargittai and C. A. Pickover (World Scientific, Singapore, 1992), p. 353.
8. A. Arakawa, V. R. Lamb, and J. Chang, Computational design of the basic dynamical processes of the UCLA general circulation model, in *General Circulation Models of the Atmosphere, Vol. 17* (Academic Press, New York, 1977), p. 173.
9. R. Glowinski and O. Pironneau, Finite element methods for Navier–Stokes equations, in *Annu. Rev. Fluid Mech., Vol. 24*, edited by J. L. Lumley, M. van Dyke, and H. L. Reed (Annual Reviews, Inc, Palo Alto, CA, 1992), p. 167.

10. H. D. Simon, Partitioning of unstructured problems for parallel processsing, *Comput. Systems Eng.* **2/3**, 135 (1991).
11. S. T. Barnard, PMRSB: Parallel multilevel recursive spectral bisection, in *Proceedings, of the 1995 ACM/IEEE Supercomputing Conference, San Diego, CA, December 1995.*
12. G. Chukkapalli, Weather and climate numerical algorithms: An efficient, parallel solution scheme for the shallow water equations, Ph.D. thesis, Department of Mechanical Engineering, University of Toronto, 1998.
13. J. Cote, M. Roch, A. Staniforth, and L. Fillion, A variable resolution semi-Lagrangian finite element global model of the shallow water equations, *Mon. Weather Rev.* **121**, 231 (1993).