INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids **28**: 761–788 (1998)

A BOUNDARY CONFORMING STRUCTURED GRID FOR GLOBAL OCEAN CIRCULATION STUDIES[1]

WILLIAM S. RUSSELL[a,*] AND PETER R. EISEMAN[b]

[a] *Department of Applied Physics, Columbia University and NASA Goddard Institute for Space Studies, 2880 Broadway, New York, NY 10025-7819, USA*
[b] *Program Development Corporation, 300 Hamilton Avenue, Suite 409, White Plains, NY 10601, USA*

SUMMARY

A boundary conforming two-dimensional structured grid for the irregular domain of the world's ocean is generated numerically using differential equation techniques. It is calculated using block structured methods which allow the inclusion of all major bodies of water including seas and basins, and which preserve slope continuity of the co-ordinate lines across the global domain. The block structure is coupled with an innovative *blown-up cube* model of the Earth which permits all areas of the global ocean to be modeled with the same resolution, eliminating problems associated with polar singularities. The grid is generated on the curved surface of the Earth (rather than the longitude–latitude plane) by employing the Beltrami operator instead of the standard Laplacian operator. Application of the grid to a steady state heat conduction problem shows the relative computational accuracy and the potential to resolve the complex, smaller scale oceanographic phenomena of great importance to global circulation studies.

KEY WORDS: numerical grid generation; block structure; ocean circulation; finite difference equations; boundary currents

1. INTRODUCTION

The oceans, with their irregular coastlines and bathymetry, present a unique challenge to numerical modeling. Global ocean general circulation models (OGCMs), which numerically solve the Navier–Stokes equations, were pioneered by Bryan [1] of the Geophysical Fluids Dynamics Laboratory. The Bryan model, updated for vector machines by the late Cox [2] is a widely used OGCM for climate studies. The model uses a simple Cartesian longitude–latitude grid. Resolutions of the OGCM range from 5° longitude × 4° latitude [3] to 0.5 × 0.5° [4].

There are several problems with a Cartesian longitude–latitude grid. The coastlines are represented as zigzag or sawtooth boundaries, which can introduce unrealistic convergences or divergences when boundary conditions are applied. Transforming the real world domain into rectangular cells of either land or ocean often results in narrow flow-through regions either

* Correspondence to: Columbia University and NASA Goddard Institute for Space Studies, 2880 Broadway, New York, NY 10025-7819, USA. Tel: +1 212 6785592; Fax: +1 212 6785552; E-mail: bill@haggis.giss.nasa.gov; or wsr1@columbia.edu
[1] Subject classifications: 65N50, mesh generation and refinement; 65C20, models, numerical methods; 86A05, oceanography.

CCC 0271–2091/98/150761–29$17.50

Received 27 February 1995
Revised 13 May 1997

being cut off, since cells containing these ocean flow-though regions are mostly land, or included only if they are as wide as a grid spacing. An example of this latter problem is Drakes Passage below South America, which is arbitrarily enlarged in low resolution OGCMs but crucial to the circulation of the world's oceans. Thin strips of land such as sections of Central America and Indonesia introduce the opposite problem, whereby independent oceans are unrealistically connected. In order to accommodate these difficulties, cells are often changed from land to ocean (to correct flow-through) and vice-versa (to cut off independent oceans) leading to unfair representation of the area ratio of land versus ocean in a single cell, and highly inaccurate locations of coastal boundaries.

Other major difficulties with a longitude–latitude grid also exist: sufficient resolution for strong regional currents such as the western boundary intensification [5,6] is difficult to accomplish without wasting resolution in the relatively quiescent ocean interior; grid cell areas are non-uniform, resulting in excessive resolution near the poles.

Although a considerable number of numerical simulations have been performed on local domains around the globe, relatively few have been done with numerically generated grids. Stretched grids have been generated for coastal modeling [7] and composite grids have been developed for irregular ocean basins [8]. Other non-uniform mesh procedures such as nested grids [9] and patching techniques [10] have been examined. The most common irregular grid ocean simulation is probably the analysis of the North Atlantic circulation. Haidvogel *et al.* [11] presented an orthogonal curvilinear grid for analyzing the dynamics of the coastal transition zone, and it was Haidvogel's contribution in Adams *et al.* [12] and Fukumori *et al.* [13] which presented similar conformal grids for analyzing wind driven circulation in the North Atlantic. A number of ongoing research projects are trying to marry grid generation to global ocean circulation simulations. For example, stretched grids are being tested at Los Alamos National Laboratory (J. Dukowicz, personal communication), as are composite grids at Sandia National Laboratory [14] and hexagonal grids at Colorado State (D. Randall, personal communication). However, most currently existing OGCMs use simple Cartesian longitude–latitude grids (or some variation thereof) which give rise to the aforementioned problems.

In an attempt to circumvent these problems, an irregular, boundary conforming, two-dimensional structured grid is developed for global ocean simulations (with the intention of eventually coupling it to an atmospheric GCM) in an effort to show that at least the ocean component of a coupled GCM can be run more accurately with an ability to increase resolution when required, to alleviate the wasted high resolution and various numerical difficulties at the poles, and to use exact coastal values at each of the fixed grid point boundaries. This last point cannot be overemphasized: boundary currents are extremely important and can never be modeled accurately in low resolution models if sawtooth boundaries are employed. The boundary conforming grid utilizes exact coastal locations and, moreover, generates grid lines which follow the contours of the boundary allowing more accurate simulation of regions such as the Gulf Stream and the Kuroshio Current.

This paper presents a new global grid which will be substituted for the Cartesian longitude–latitude grid currently used in the Bryan–Cox ocean circulation model [1,15,16]. Section 2 describes how the new grid can be numerically generated using differential equation techniques, specifically, elliptic partial differential equations (PDE's). A block structure approach [17,18] is introduced, which allows highly complex two-dimensional (and three-dimensional) domains to be divided into subregions (often referred to as *blocks*), each of which admits an independent local co-ordinate system. The blocks are connected after each iteration. A method to treat *floating* boundaries and resulting *weak singularities* is also described in Section 2. Section 3 follows the development of the boundary conforming numerically generated grid and

introduces a *blown-up cube* model of the Earth, which allows all ocean domains to be modeled with the same resolution. The governing equations for spherical surface grid generation are also presented. The grid is tested by solving a heat conduction problem in both local and global domains and the process by which this simulation is done using an irregular structured grid is described in Section 4. Section 5 discusses possible improvements to the grid and applications to future global ocean circulation studies.

In order to motivate the following work and subsequent articles describing the development of this new global ocean model, Plate 1 is introduced as an example of the capability of an irregular grid approach to ocean modeling. The governing equations for ocean circulation have been transformed to computational space and the dynamics have been solved using a streamfunction formulation. This figure shows the flow resulting from an annual mean wind stress forced at the surface along with a specified temperature and salinity field. The model has 16 layers and a variable bathymetry. This new ocean model, the resulting solutions and comparisons with Cartesian model solutions will be described in subsequent articles. Here, the focus is on the generation of an irregular grid which eliminates the disadvantages of Cartesian grids cited above.

2. FUNDAMENTALS

An arbitrary concave region can be mapped onto a rectangle. If the generation of a boundary conforming grid within this arbitrary region is required (meaning that one co-ordinate will be constant along each segment of the physical boundary curve) differential equation methods can be used, exploiting the properties of the solution of the grid generating equation in producing the mesh. The grids presented herein are generated using elliptic systems, specifically, Laplace's equation. They have various favorable properties, such as inherent smoothness, non-propagation of boundary slope discontinuities into the field, and non-overlap of grid lines for any configuration. Poisson's equation includes a forcing term which allows the positioning of the grid points in physical space to be controlled and manipulated with ease. The Laplace and Poisson equations have been extensively used for this purpose and are well documented [19,20].

Consider the arbitrary physical region in Cartesian co-ordinates x_i, $\{i = 1, 2, 3\}$ to be mapped to computational curvilinear co-ordinates ξ^i, $\{i = 1, 2, 3\}$. The procedure transforms from the physical domain to the computational domain, where the mapping is controlled by a partial differential equation. This mapping is constructed by specifying the desired grid points x_i, $\{i = 1, 2, 3\}$ on the boundary of the physical domain.

Partial derivatives with respect to Cartesian co-ordinates x_i are related to partial derivatives with respect to curvilinear co-ordinates ξ^i by

$$A_{x_i} = \sum_{j=1}^{3} A_{\xi^j} (\mathbf{a}^j)_i \,, \tag{1a}$$

or equivalently

$$A_{\xi^i} = \sum_{j=1}^{3} A_{x_j} (\mathbf{a}_i)_j \,, \tag{1b}$$

where $\mathbf{a}_i = \mathbf{r}_{\xi^i}$ $(\mathbf{r} = (x_1, x_2, x_3))$ is the covariant base vector and $\mathbf{a}^i = \nabla \xi^i$ is the contravariant base vector of the curvilinear co-ordinate system. General arc length increments depend on the covariant metric tensors given by

$$g_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j = g_{ji}. \tag{2}$$

The Jacobian $\sqrt{g}$ of the transformation is given by the triple scalar product

$$\sqrt{g} = [\det|g_{ij}|]^{1/2} = \mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3). \tag{3}$$

The contravariant base vectors may be written in terms of the covariant base vectors as

$$\mathbf{a}^i = \nabla \xi^i = \frac{1}{\sqrt{g}} (\mathbf{a}_j \times \mathbf{a}_k) \tag{4}$$

with (i, j, k) cyclic. Contravariant metric tensors are expressed as

$$g^{ij} = \mathbf{a}^i \cdot \mathbf{a}^j = g^{ji}, \tag{5}$$

and are related to covariant metric tensors by the expression

$$g^{il} = \frac{1}{g} (g_{jm} g_{kn} - g_{jn} g_{km}), \tag{6}$$

where $i = 1, 2, 3$; $l = 1, 2, 3$ with (i, j, k) and (l, m, n) both cyclic.

The grids discussed in this paper are generated using elliptic equations. The distribution of points on the interior of the physical domain is determined by the general Poisson type system

$$\nabla^2 \xi^i = P^i \quad i = 1, 2, 3, \tag{7}$$

where P^i controls the spacing and orientation of the co-ordinate lines. This system must be transformed to computational space by interchanging the roles of the dependent and independent variables. Using the above relations, the non-conservative form of the Laplacian operator can be written

$$\nabla^2 A = \sum_{i=1}^{3} \sum_{j=1}^{3} \mathbf{a}^i \cdot \mathbf{a}^j A_{\xi^i \xi^j} + \sum_{i=1}^{3} \sum_{j=1}^{3} \mathbf{a}^i \cdot (\mathbf{a}^j)_{\xi^i} A_{\xi^j}. \tag{8}$$

Since $\nabla^2 \xi^l = \nabla \cdot (\nabla \xi^l) = \nabla \cdot \mathbf{a}^l = \Sigma_{i=1}^{3} \mathbf{a}^i \cdot (\mathbf{a}^l)_{\xi^i}$, replacing A with $\mathbf{r} = (x_1, x_2, x_3)$, and using Equations (5) and (7) gives

$$\sum_{i=1}^{3} \sum_{j=1}^{3} g^{ij} \mathbf{r}_{\xi^i \xi^j} + \sum_{k=1}^{3} P^k \mathbf{r}_{\xi^k} = \nabla^2 \mathbf{r} = \mathbf{0}, \tag{9}$$

which, in practice, is the elliptic system of equations which is solved for x_1, x_2, x_3 in terms of the curvilinear co-ordinates ξ^i.

In two dimensions, x_3 is the direction of invariance and let $\xi^3 = x_3$. Henceforth, when referring to two dimensions, the notation will be $x = x_1$, $y = x_2$, $\xi = \xi^1$, $\eta = \xi^2$. It follows that for plane grid generation the distribution of points on the interior is then determined by solving

$$\xi_{xx} + \xi_{yy} = P(\xi, \eta), \tag{10a}$$

$$\eta_{xx} + \eta_{yy} = Q(\xi, \eta), \tag{10b}$$

where $P = P^1$ and $Q = P^2$. This yields the following elliptic system:

$$g_{22} x_{\xi\xi} - 2 g_{12} x_{\xi\eta} + g_{11} x_{\eta\eta} = -g(P x_\xi + Q x_\eta), \tag{11a}$$

$$g_{22} y_{\xi\xi} - 2 g_{12} y_{\xi\eta} + g_{11} y_{\eta\eta} = -g(P y_\xi + Q y_\eta), \tag{11b}$$

where

$$g_{11} = x_\xi^2 + y_\xi^2, \tag{12a}$$

$$g_{12} = x_\xi x_\eta + y_\xi y_\eta, \tag{12b}$$

$$g_{22} = x_\eta^2 + y_\eta^2, \tag{12c}$$

$$\sqrt{g} = x_\xi y_\eta - x_\eta y_\xi. \tag{13}$$

Using centered finite difference approximations for these derivatives allows the use of a simple SOR iteration procedure to solve for all interior (x, y) values.

Our arbitrary region can be regarded as a single block. Consider now a highly irregular physical region with islands and convex boundaries. This domain can be segmented into subregions, each bounded by four generally curved sides, within each of which an independent co-ordinate system can be prescribed. The global co-ordinate system encompassing all subregions (i.e. the entire physical domain) is formed by combining all subregions and linking all block boundaries. A simple example of block structure is given in Figure 1(a). The physical domain has been divided into two simple blocks. Each is mapped onto an associated rectangular block in computational space. The common boundary between the two regions is unphysical in the sense that, unlike a fixed boundary segment, it is free to move like other co-ordinate lines. Consequently, it is termed a floating boundary and if treated correctly cannot be differentiated between other interior co-ordinate lines composing the resulting physical grid. The grid should be completely continuous and smooth.

After finite difference approximations are substituted for the derivative terms in Equations (11), (12) and (13), the resulting iteration procedure at index '*ij*' requires a nine-point stencil. Therefore, adjacent indices to '*ij*' are used (eight in all). For floating boundary values, this is handled by adding an imaginary layer around each block (see computational domain in Figure 1(a)). If any portion of a boundary is floating, then the exterior layer to this portion of the boundary is an interior layer of the adjacent block. That is, the values lying on the exterior layer have image values which lie within the adjacent block. Hence, before each sweep through a block, the exterior layer values are updated with their associated image values and then the iteration is performed for all interior points of the block and all boundary values which are prescribed as floating. For example, in Figure 1(b), when calculating values in block 1 on the floating boundary, say at index $(i, j) = (6, 3)$, the nine-point stencil includes values on the additional layer $i = 7$, $j = 2, 3, 4$ which have image values corresponding to $i = 2$, $j = 1, 2, 3$ on block 2. Once all floating boundary values are updated, their associated image values on the adjacent block boundary are updated immediately. For example, once the values at (6, 3) are computed for block 1, the values at (1, 2) on block 2 are also updated. This procedure is followed for each block for multiple block regions until every block has been swept through once. This constitutes one iteration.

The idea here is to generate a boundary conforming, macroscopically irregular, but locally structured grid. A structured grid is one which has a definitive order. In most cases the order comes from a premapped Cartesian grid, which leads to a physical space grid which has four sides to every cell. When three or more of the macroscopic blocks meet at one point (at the interface between subregions of a composite curvilinear co-ordinate system), a *weak singularity* results. These points commonly arise when geometrically complicated physical regions are involved and special attention is required for the finite difference representation of derivatives. A weak singularity may be recognized in physical space as an interior point which is a common vertex to a non-standard number of cells. (Standard interior grid points in two dimensions have four immediate neighbors.) There are many kinds of weak singularities [17] but we consider only four denoted I–IV. Type I occurs when three corner points meet, type II occurs

at the intersection of an edge and two corners, type III at the intersection of two edges and a corner, and type IV at three edge points. Each is illustrated in Figure 2. Type II is indistinguishable from an interior grid point. However, since three blocks meet at this point, a type II weak singularity must be treated as such because once this value is updated from a sweep through one block, its image point on two other blocks must be updated.

The usual finite difference representations for derivatives break down at weak singularities. However, if the transformed equations and difference approximations are rephrased in terms of a local co-ordinate system, this problem can be circumvented. Hence, a local co-ordinate system $(\bar{\xi}, \bar{\eta})$ is introduced for each type of singularity. The local system is chosen so as to orient and label only the surrounding points needed in the difference expressions. Therefore, it specifies the image values for the points on the exterior layer directly surrounding the weak singularity. A local co-ordinate system relative to all three blocks could be specified to aid convergence (i.e. update the weak singularity three times per iteration) but it is not necessary. Of course, for evolution simulations, the prognostic variables at this location should be updated only once per iteration (using only one block) to prevent lagging or time step violations.

An algorithm was formulated to determine automatically the grid points (and their associated blocks) required of a finite difference update of a weak singularity. Thus, given the location and type of each weak singularity, the local co-ordinate system is automatically determined. This saves a great deal of time when designing or modifying block structure.

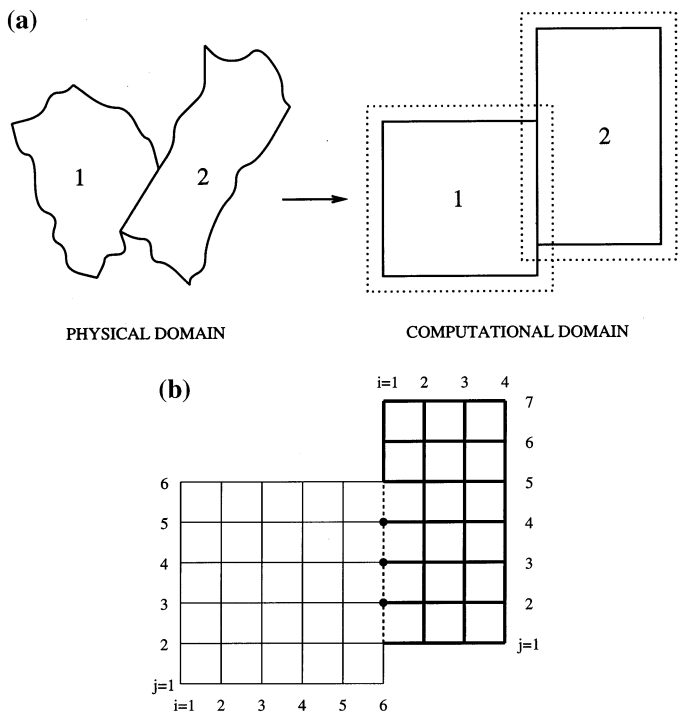

Figure 1. Schematics of block structure. (a) Two blocks in the physical (x, y) domain are mapped onto the computational (ξ, η) domain. An extra layer of points is included around each computational block to allow floating boundaries to be updated with neighboring values from adjacent blocks. (b) Example of indexing and floating boundaries for two adjacent blocks.

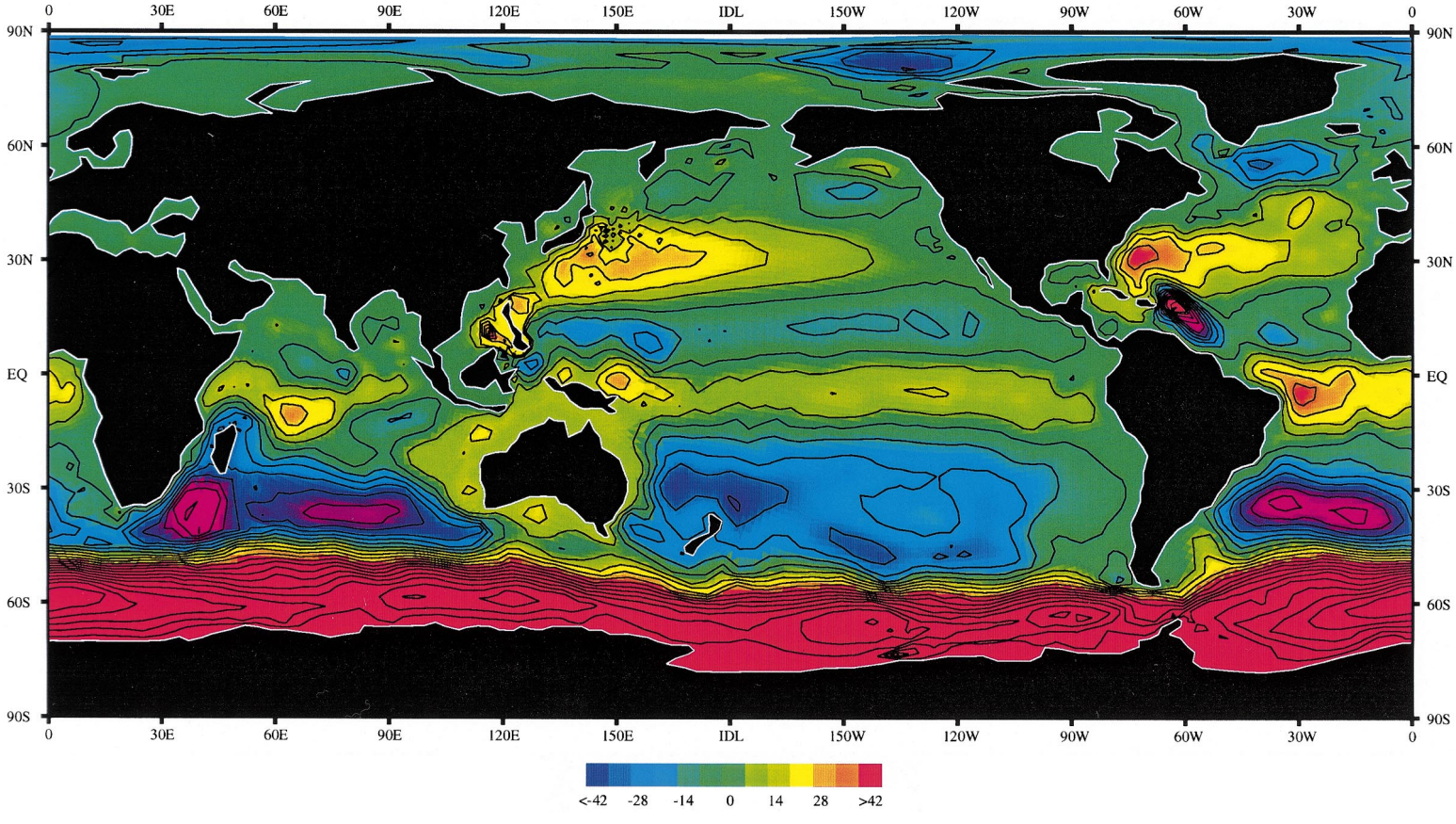

Plate 1. Steady state horizontal streamfunction distribution (Svedrup) for a new global ocean circulation model using the global grid in Figure 10 with 16 layers in the vertical. A constant annual mean wind stress is imposed at the ocean surface and an annual mean temperature and salinity distribution is specified at each grid point. Only the velocity field is predicted. The new ocean model developed to produce these results is described in a separate paper.

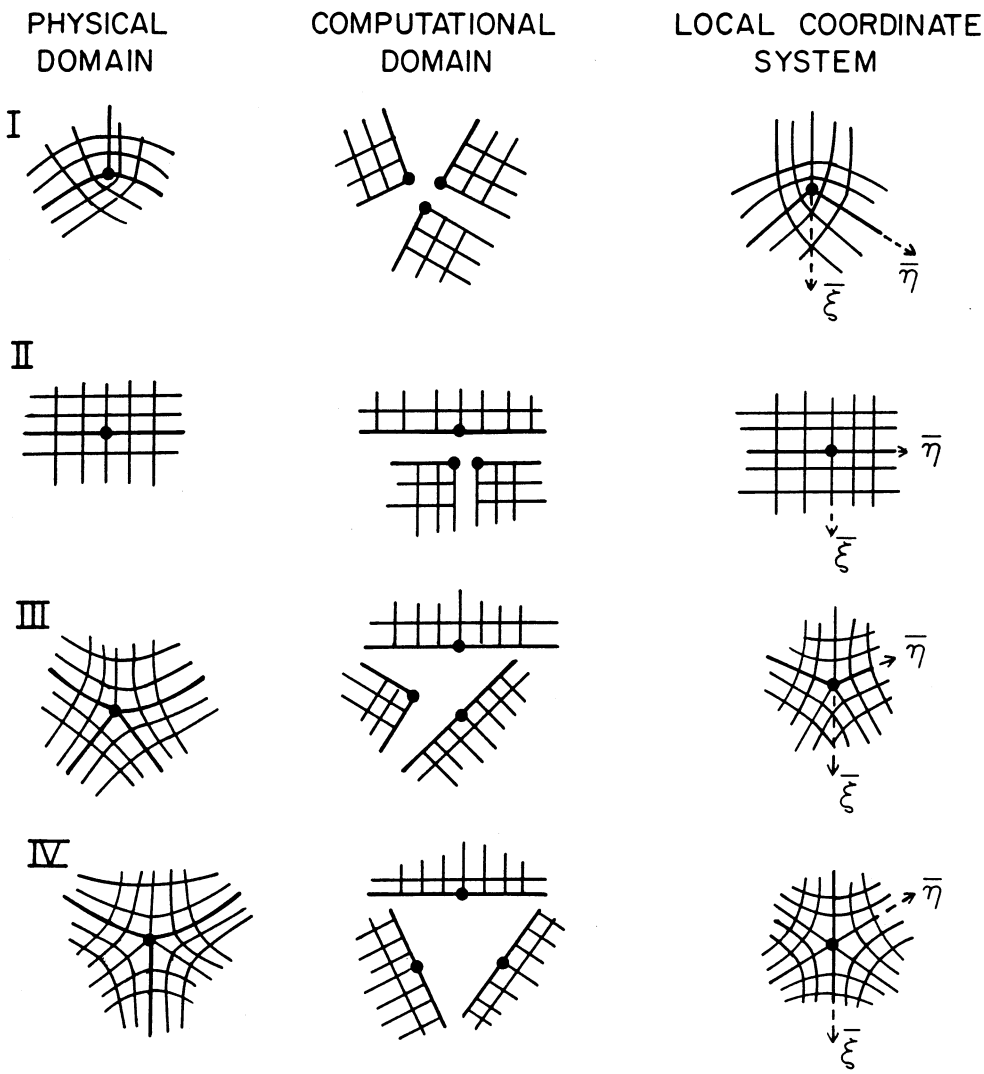

Figure 2. Diagram of the four types of weak singularities. Each singularity occurs when three blocks intersect. Type I occurs when three corner points meet. Type II occurs at the intersection of an edge and two corners (indistinguishable from a regular interior point). Type III occurs when two edges and a corner join. Type IV occurs for three block edges.

Heretofore, input data sets (boundary value locations) had to contain information describing the local co-ordinate system for every single weak singularity. These data sets had to be created manually and would often contain errors among the thousands of integer values required for such a calculation.

It is advantageous to locate weak singularities over the ocean (interior grid points) rather than at coastal values (boundary grid points). When they are not free to move around, grid lines tend to stretch away from the fixed point yielding long, thin, and sometimes concave cells. Badly placed weak singularities induce folding which must be avoided at all costs. Designating weak singularities as floating points allows them to move freely, preserving the smooth

variation in cell areas both across a block itself and across adjacent blocks. Other advantages include easy generation of grids with orthogonal boundaries. For simulation purposes, coastal values are associated with unique blocks, allowing simple application of Neumann boundary conditions.

3. DEVELOPMENT OF THE GLOBAL OCEAN GRID

Most currently existing global ocean circulation models use Cartesian grids which impose sawtooth boundaries at land–ocean interfaces. A typical example of this kind of grid is shown in Figure 3. This is a 4×5 resolution grid meaning 4° grid spacing in latitude and 5° grid spacing in longitude. The 4×5 grid has been chosen for display here because it is a common grid used for ocean simulations. The vectorized form of the Bryan–Cox 4×5 OGCM consequently uses 3312 grid points; scalar versions skip internal land points resulting in approximately 2300 grid values. Therefore, it is the aim of this research to show that if 3312 grid points are used, the grid points can be better distributed using a boundary conforming grid to allow more accurate modeling of oceanographic physical phenomena.

3.1. Plane grid generation

In order to overcome the polar singularity problem the idea of working in longitude–latitude space must be eliminated. The globe can be considered as a *blown-up cube* where a cube is placed inside the sphere with the eight corner points fixed to the sphere. Then if this cube were blown up like a balloon, at some point it could form a sphere. Inversely, if all values on the sphere were projected towards its origin they would intersect the cube at unique points. Hence, each of the six faces of the cube would represent a unique surface portion of the sphere. The values to be mapped, of course, are coastal locations representing the boundaries of our ocean domain. The advantage of this approach is that each face represents an equally shaped surface area and so the North Pole (Arctic Ocean) can be modeled with the same resolution as say the Indian Ocean. The grid can be generated on each (plane) face of the cube, then projected back onto the sphere. Herein lie three problems. Weak singularities are predetermined by the location of the corner points of the cube and subsequent orientation inside the sphere. Secondly, but more importantly, cells of equal area generated on the plane represent surface elements of varying area on the sphere. (Those lying at the edge of the face have smaller respective surface elements than those lying at the center.) Finally, there is an obvious discontinuity (for the three local co-ordinate systems) at each corner of the cube. These three problems can be alleviated by generating the grid on the associated spherical surface patches instead of the planes (discussed hereinafter). Nevertheless, since the governing equations for numerically generating a grid on a plane are quite straightforward and well-documented, it is a natural intermediate step in developing the code for surface grid generation to formulate the block structure on the six cube faces. Consequently, grids can be generated for each face independently, with floating boundary values for ocean–ocean interfaces but fixed values at the cube edges (where there is an obvious discontinuity in local co-ordinate systems). Each face has a local physical (x, y) co-ordinate system imposed with origin at the center of the face. Digital coastal values $\{x_b, y_b\}$ in these local (x, y) co-ordinates are created and used as the boundary value data sets for each cube face. The block structure is shown in Figure 4. An attempt has been made to minimize the number of blocks, but to include all important seas and bays and flow-through regions. Blocks are designed to produce smooth grids with

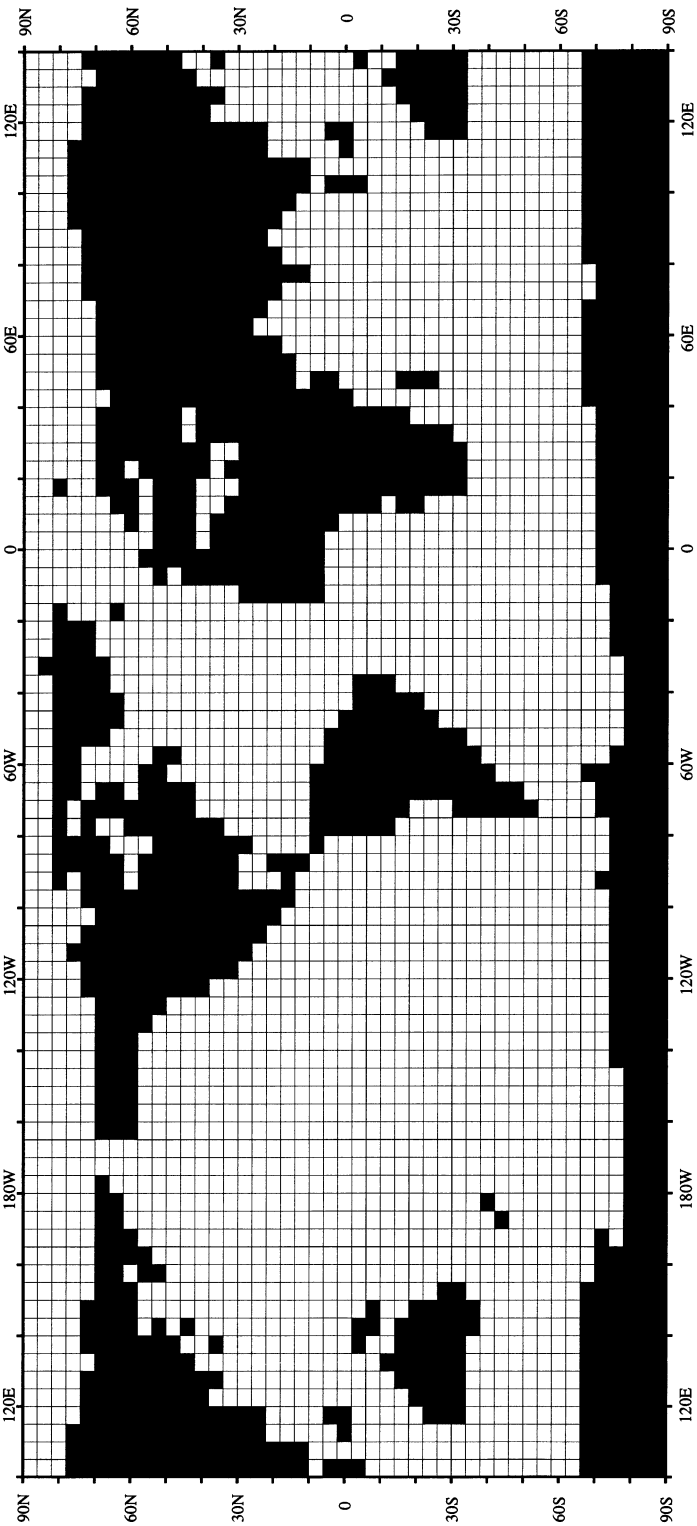

Figure 3. Typical 4×5 longitude–latitude grid used in some global ocean calculations. This grid has a resolution of 4° and 5° in the latitudinal and longitudinal directions, respectively. Other typical grids are 8×10 and $2 \times 2\frac{1}{2}$.

minimum variation in area across adjacent cells, bearing in mind the inherent smoothing effects of the Laplacian operator and its tendency to pack grid lines around convex boundaries and to pull them away from concave boundaries.

For each block, the Laplace equation is used to generate the grid. The equations to be solved for x and y, subject to the boundary conditions given by the digital boundary value data sets $\{x_b, y_b\}$, are therefore

$$g_{22}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{11}x_{\eta\eta} = 0, \tag{14a}$$

$$g_{22}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{11}y_{\eta\eta} = 0, \tag{14b}$$

where the covariant metric tensors are given by Equation (12). For all analyses and results presented in this paper, the clustering functions P and Q in Equation (11) are set to zero. Hence the grids presented herein are termed Laplacian grids.

The parametric (ξ, η) domain has unit square cells within each of the blocks. Equations (14) are finite differenced using simple second-order approximations and solutions for x and y at each grid point in (ξ, η) space are calculated using SOR. The initial guess for the grid is found using transfinite interpolation and using a tolerance value of $10^{-5}\ a_e$ (where a_e is the Earth's radius): convergence is when all grid point locations change by less than $10^{-5}\ a_e$ from one iteration to the next. Approximately 250 iterations are required for convergence. The resulting grids are shown in Figure 5. Notice the continuity and smoothness properties across the floating boundaries of adjacent blocks. Also, the difference in cell areas is fairly constant across all domains.

In Figure 5 floating boundaries are employed across adjacent blocks on the same face. If the edges of the cube are now also permitted to float, the face shapes become distorted (from a square) but the grid lines become continuous and smooth from one face to the next. Figure 6 illustrates this continuity and smoothness of the grid across cube edges and shows how the grids on the cube faces are related. Due to the aforementioned discontinuities present at the corners of the cube, the corners of each face can never be floated with complete success. Due to the unphysical nature of the cube and the ability of the surface generation method to overcome this hurdle, the problem of corner discontinuities is disregarded and we proceed to numerical grid generation on a spherical surface.

3.2. Surface grid generation

The grid generation system described above is for two-dimensional Cartesian co-ordinates. It is beneficial to consider two-dimensional curvilinear co-ordinates on the spherical surface. The problem is to generate a two-dimensional grid on a sphere with the implicit third dimension kept constant on the surface. The prescribed boundaries between blocks now become bounding curves lying in physical space on the sphere. The problem is similar to that described for the plane, but now the curvature of the spherical surface must be included in the differential equation system which generates the grid. The block structure seen in Figure 7 is obtained if the domains and the block structure remain unchanged (Figure 4), but all boundary values are projected back onto the surface of the sphere. With the block structure complete and the floating boundary and weak singularity relationships already calculated for the cube, the only problem remaining is the derivation of the governing equations for x and y on the surface. For a more comprehensive derivation of general surface grid generation see Thompson and Warsi [17] or Warsi [21].

For the sake of simplicity, we begin with the system equivalent to (7)

$$\Delta_2 \xi = 0, \tag{15a}$$

$$\Delta_2 \eta = 0, \tag{15b}$$

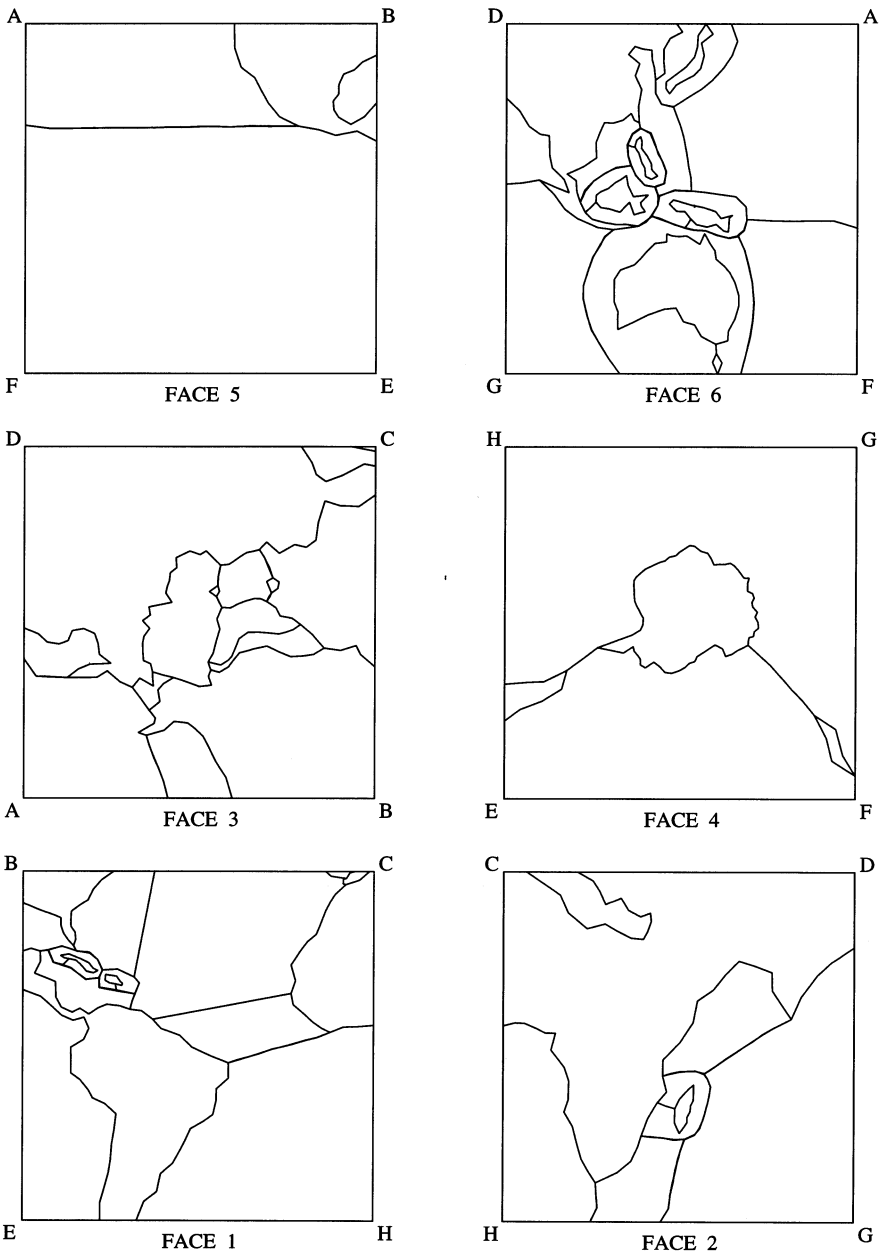

Figure 4. Block structure for the six faces of the cube. For each face, the number of the block is indicated. Blocks are designed to produce smooth grids with minimal change in area across adjacent cells. The corners of the cube are ascribed letters to aid in identifying common corners.

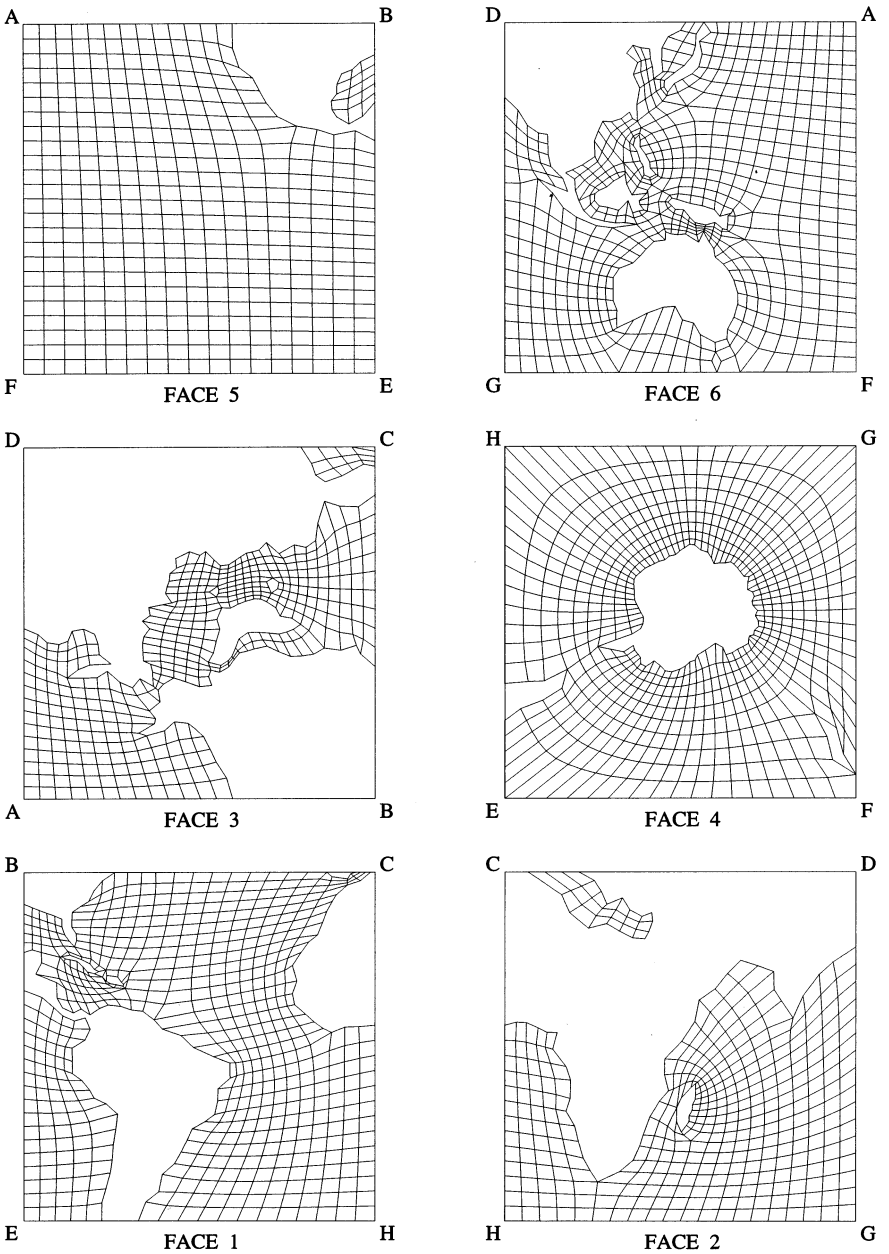

Figure 5. Converged grids for the six plane faces of the cube. The edges of the cube have been fixed, but the floating boundaries between blocks on the same face are smooth and are indistinguishable from interior co-ordinate lines.

where Δ_2 is the second kind of Beltrami operator (sometimes called the Laplace–Beltrami operator). ξ and η now represent space curves generated on a surface rather than a plane. Note that for surface generation, the Beltrami operator has simply replaced the Laplacian operator, which is a special case of the more general Beltrami operator. The Beltrami operator includes the effects of surface curvature. Inverting dependent and independent variables yields

Figure 6. Converged grids for the six plane faces of the cube (pieced together to illustrate continuity and smoothness) with all ocean–ocean boundaries now permitted to float. The corner points of each face of the cube remain fixed due to the difficulty in dealing with discontinuities in local co-ordinate systems.

$$g_{22}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{11}x_{\eta\eta} = n^{(1)}R, \tag{16a}$$

$$g_{22}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{11}y_{\eta\eta} = n^{(2)}R, \tag{16b}$$

where the metrics g_{22}, g_{12} and g_{11} now satisfy

$$g_{22} = x_\eta^2 + y_\eta^2 + z_\eta^2, \tag{17a}$$

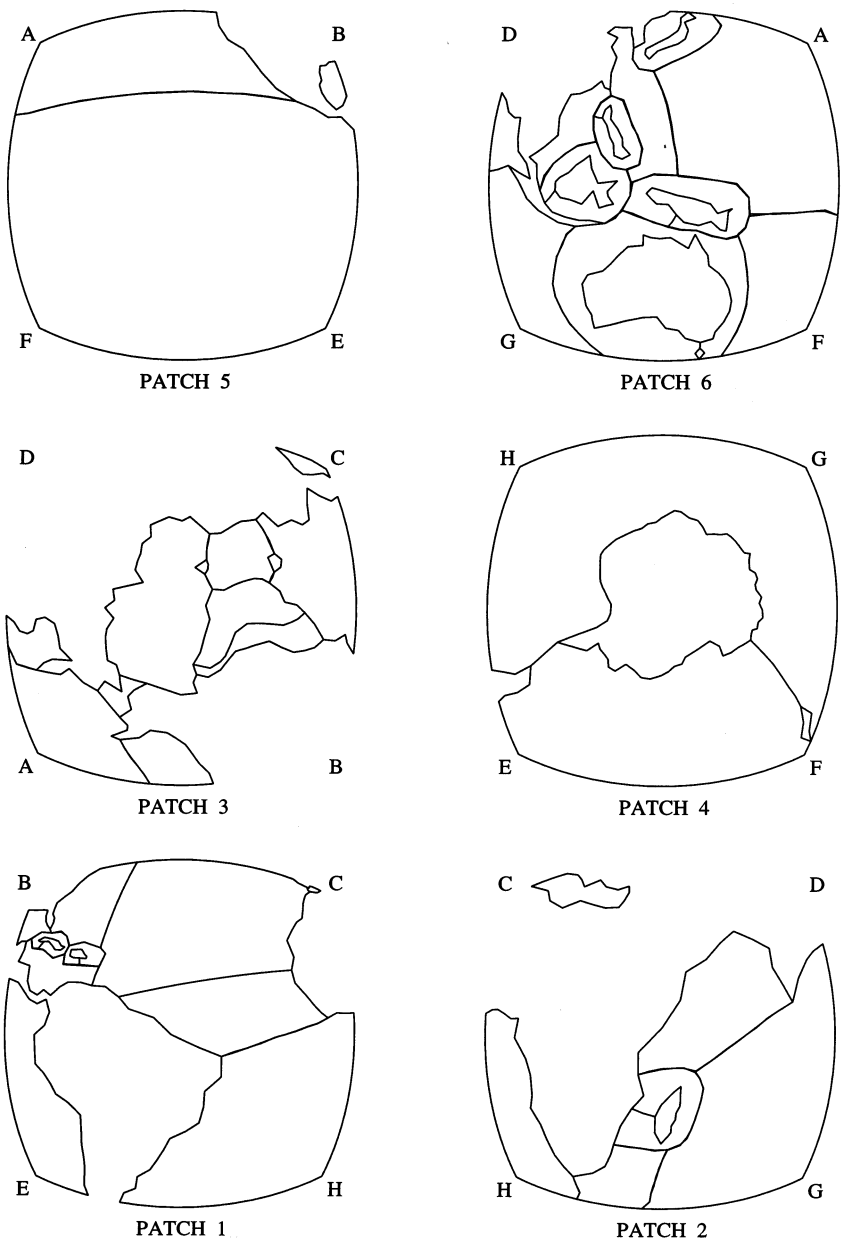

Figure 7. Block structure for the curved surfaces (on the sphere) equivalent to those for the cube seen in Figure 4.

$$g_{12} = x_\xi x_\eta + y_\xi y_\eta + z_\xi z_\eta, \tag{17b}$$

$$g_{11} = x_\xi^2 + y_\xi^2 + z_\xi^2, \tag{17c}$$

and $n^{(1)} = x/a_e$, $n^{(2)} = y/a_e$ are the normal components to the surface, and a_e is the radius of the Earth. $R = (g_{22}g_{11} - g_{12}^2)(k_I + k_{II})$ where $k_I + k_{II}$ is twice the mean curvature of the surface. Therefore, because the surface is of the form $z = f(x, y) = (a_e^2 - x^2 - y^2)^{1/2}$, it can be deduced from elementary differential geometry that $k_I + k_{II} = -2/a_e$. Equations then become

$$g_{22}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{11}x_{\eta\eta} = -2gx/f^2, \tag{18a}$$

$$g_{22}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{11}y_{\eta\eta} = -2gy/f^2, \tag{18b}$$

since $G = g_{22}g_{11} - g_{12}^2$ can be reduced to ga_e^2/f^2, where $\sqrt{g}$ is the Jacobian given in Equation (3). The new metrics satisfy

$$g_{22} = \left(1 + \frac{x^2}{f^2}\right)x_\eta^2 + \left(1 + \frac{y^2}{f^2}\right)y_\eta^2 + \frac{2xy}{f^2}x_\eta y_\eta, \tag{19a}$$

$$g_{12} = \left(1 + \frac{x^2}{f^2}\right)x_\xi x_\eta + \left(1 + \frac{y^2}{f^2}\right)y_\xi y_\eta + \frac{xy}{f^2}(x_\xi y_\eta + x_\eta y_\xi), \tag{19b}$$

$$g_{11} = \left(1 + \frac{x^2}{f^2}\right)x_\xi^2 + \left(1 + \frac{y^2}{f^2}\right)y_\xi^2 + \frac{2xy}{f^2}x_\xi y_\xi, \tag{19c}$$

where $f > 0$ for all six local patches. The blocks are constructed so that for any patch, the converged grid (for the block structure of that patch) never crosses the inherent $z = 0$ plane. In other words, the grids for each patch never wrap around to the opposite side of the spheroidal projection.

The finite difference formulation for this surface grid generation case is rather more complicated than that for the plane. The $[g_{22}]_{ij}$, $[g_{12}]_{ij}$ and $[g_{11}]_{ij}$ terms now involve f_{ij}, x_{ij} and y_{ij} as well as their derivatives. f_{ij} is also a function of x_{ij} and y_{ij} and we have an additional term on the right-hand side of Equation (18). The finite difference forms of (18) are algebraically non-linear in the unknowns x_{ij}^{n+1} and y_{ij}^{n+1} due to the appearance of unknown quantities at the next iteration level $(n+1)$ in the coefficients. Consequently, it is advantageous to linearize (19).

Newton's linearization procedure (also called quasi-linearization) has the advantage of an enhanced convergence rate. If the difference δ_x between two successive iterates of x_{ij} is assumed to be small, then by setting $x_{ij}^{n+1} = x_{ij}^n + \delta_x$ one can easily linearize powers of x_{ij}^{n+1}. For example, $(x_{ij}^{n+1})^2 = (x_{ij}^n)^2 + 2x_{ij}^{n+1}x_{ij}^n - 2(x_{ij}^n)^2 + \delta_x^2 \approx 2x_{ij}^n x_{ij}^{n+1} - (x_{ij}^n)^2$, which is linear in x_{ij}^{n+1}. Applying these expansions to powers of both x_{ij}^{n+1} and y_{ij}^{n+1} a linear system of equations can be solved with an inherent second-order convergence rate.

Increased convergence rate is important when a large proportion of the overall computational simulation time is spent on the numerical generation of the grid (for example, in three-dimensional, high resolution grids or adaptive gridding schemes), but for the work presented in this paper it is sufficient to employ a simpler, more common approach which ensures convergence. Linearization by evaluating all coefficients at the nth level is known as lagging and provides a consistent representation. Formally, however, it is no better than first-order-accurate.

Using centered differences, the metrics are expressed as

$$[g_{22}]_{ij}^n = [(a_e^2 - (y_{ij}^n)^2)(x_{ij+1}^n - x_{ij-1}^n)^2 + (a_e^2 - (x_{ij}^n)^2)(y_{ij+1}^n - y_{ij-1}^n)^2 + 2x_{ij}^n y_{ij}^n (x_{ij+1}^n - x_{ij-1}^n)(y_{ij+1}^n - y_{ij-1}^n)]/(2f_{ij}^n)^2, \tag{20a}$$

$$[g_{12}]_{ij}^n = [(a_e^2 - (y_{ij}^n)^2)(x_{i+1j}^n - x_{i-1j}^n)(x_{ij+1}^n - x_{ij-1}^n) + (a_e^2 - (x_{ij}^n)^2)(y_{i+1j}^n - y_{i-1j}^n)(y_{ij+1}^n - y_{ij-1}^n) + x_{ij}^n y_{ij}^n[(x_{i+1j}^n - x_{i-1j}^n)(y_{ij+1}^n - y_{ij-1}^n) + (x_{ij+1}^n - x_{ij-1}^n)(y_{i+1j}^n - y_{i-1j}^n)]]/(2f_{ij}^n)^2, \tag{20b}$$

$$[g_{11}]_{ij}^n = [(a_e^2 - (y_{ij}^n)^2)(x_{i+1j}^n - x_{i-1j}^n)^2 + (a_e^2 - (x_{ij}^n)^2)(y_{i+1j}^n - y_{i-1j}^n)^2 + 2x_{ij}^n y_{ij}^n (x_{i+1j}^n - x_{i-1j}^n)(y_{i+1j}^n - y_{i-1j}^n)]/(2f_{ij}^n)^2, \tag{20c}$$

where $(f_{ij}^n)^2 = a_e^2 - (x_{ij}^n)^2 - (y_{ij}^n)^2$. The iteration scheme which ensures convergence is then

$$x_{ij}^{n+1} = \left[(\sqrt{g_{ij}^n}/f_{ij}^n)^2 x_{ij}^n + \frac{1}{2}[g_{22}]_{ij}^n (x_{i+1j}^n + x_{i-1j}^n) + \frac{1}{2}[g_{11}]_{ij}^n (x_{ij+1}^n + x_{ij-1}^n) - \frac{1}{4}[g_{12}]_{ij}^n (x_{i+1j+1}^n + x_{i-1j-1}^n - x_{i+1j-1}^n - x_{i-1j+1}^n) \right] \Big/ ([g_{22}]_{ij}^n + [g_{11}]_{ij}^n), \tag{21a}$$

$$y_{ij}^{n+1} = \left[(\sqrt{g_{ij}^n}/f_{ij}^n)^2 y_{ij}^n + \frac{1}{2}[g_{22}]_{ij}^n (y_{i+1j}^n + x_{i-1j}^n) + \frac{1}{2}[g_{11}]_{ij}^n (y_{ij+1}^n + y_{ij-1}^n) - \frac{1}{4}[g_{12}]_{ij}^n (y_{i+1j+1}^n + y_{i-1j-1}^n - y_{i+1j-1}^n - y_{i-1j+1}^n) \right] \Big/ ([g_{22}]_{ij}^n + [g_{11}]_{ij}^n), \tag{21b}$$

where $\sqrt{g_{ij}^n} = (1/4)[(x_{i+1j}^n - x_{i-1j}^n)(y_{ij+1}^n - y_{ij-1}^n) - (x_{ij+1}^n - x_{ij-1}^n)(y_{i+1j}^n - y_{i-1j}^n)]$.

Each surface patch is assigned a local (x, y) co-ordinate system. Floating boundaries across adjacent patches, as well as patch corner points, are treated easily. When updating values from one patch to the next, the local (x, y) co-ordinate is mapped to spherical polars, then mapped back to the local co-ordinate system on the adjacent patch. There are no discontinuities at the corners. The converged grids are presented in Figure 8. There is complete continuity and smoothness across floating boundaries on the same patch and across adjacent patches. The corners points are completely hidden unless they are weak singularities. The spacing of grid points along fixed boundaries has been chosen so that the total number of cells generated is approximately equal to the number of cells for a 4×5 Cartesian simulation. In addition, cell areas are fairly uniform. Overall, it is a satisfactory grid; however, one or two of its features can be improved.

Reconsider the six patches formed by the blown-up cube. The patch edges are distorted because of the floating boundaries. There are no constraints on using six patches of equal area. In fact, on re-examination of Figure 8, it can be concluded that as long as the local (x, y) values of any one patch do not extend beyond the edge of the circle (i.e. the implicit third dimension z does not transverse the $z = 0$ plane) the block structure for any one patch can be completely restructured so that all islands and continents are surrounded by *O-type* grids. Also, one is not forced into locating weak singularities into regions where they are obviously not needed (such as the one present in the center of the North Pacific). Hence, the blocks and therefore patches are entirely restructured with the following goals: shift all weak singularities off the boundaries, that is, try to minimize the number of fixed land–ocean boundary locations where two or more blocks meet; add more grid points to areas of more important oceanographic significance; attempt to model all major bodies of water as well as flow-through regions.

The new block structure is illustrated in Figure 9 and the resulting grids are shown in Figure 10. This grid contains almost 5700 cells. It is therefore equivalent to a Cartesian grid of closer to 3×4 resolution. The tolerance value for convergence is kept at $10^{-5}\, a_e$ and roughly 600

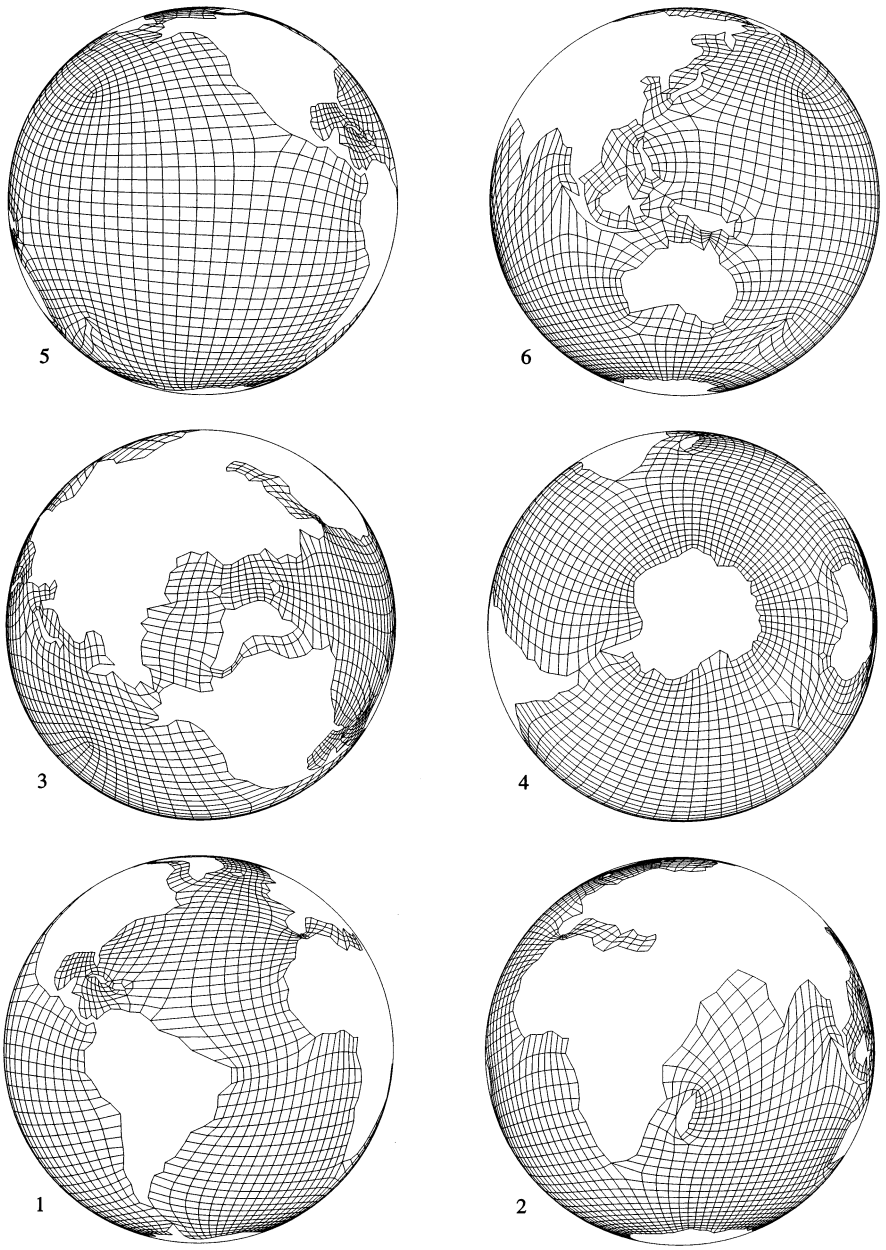

Figure 8. Converged grids from the surface grid generation technique. The origin of each patch lies at the center of each projection.

iterations are required to produce the grid in Figure 10. However, since the equations for surface grid generation are more complicated than those for the plane, the expressions for x_{ij} and y_{ij} at each iteration take longer to calculate, resulting in increased computer time: approximately 1 min on an IBM RISC6000 560 Workstation. Since the typical spin-up time for a global ocean calculation of 1000 years requires 100 hours of CPU time on this IBM, the

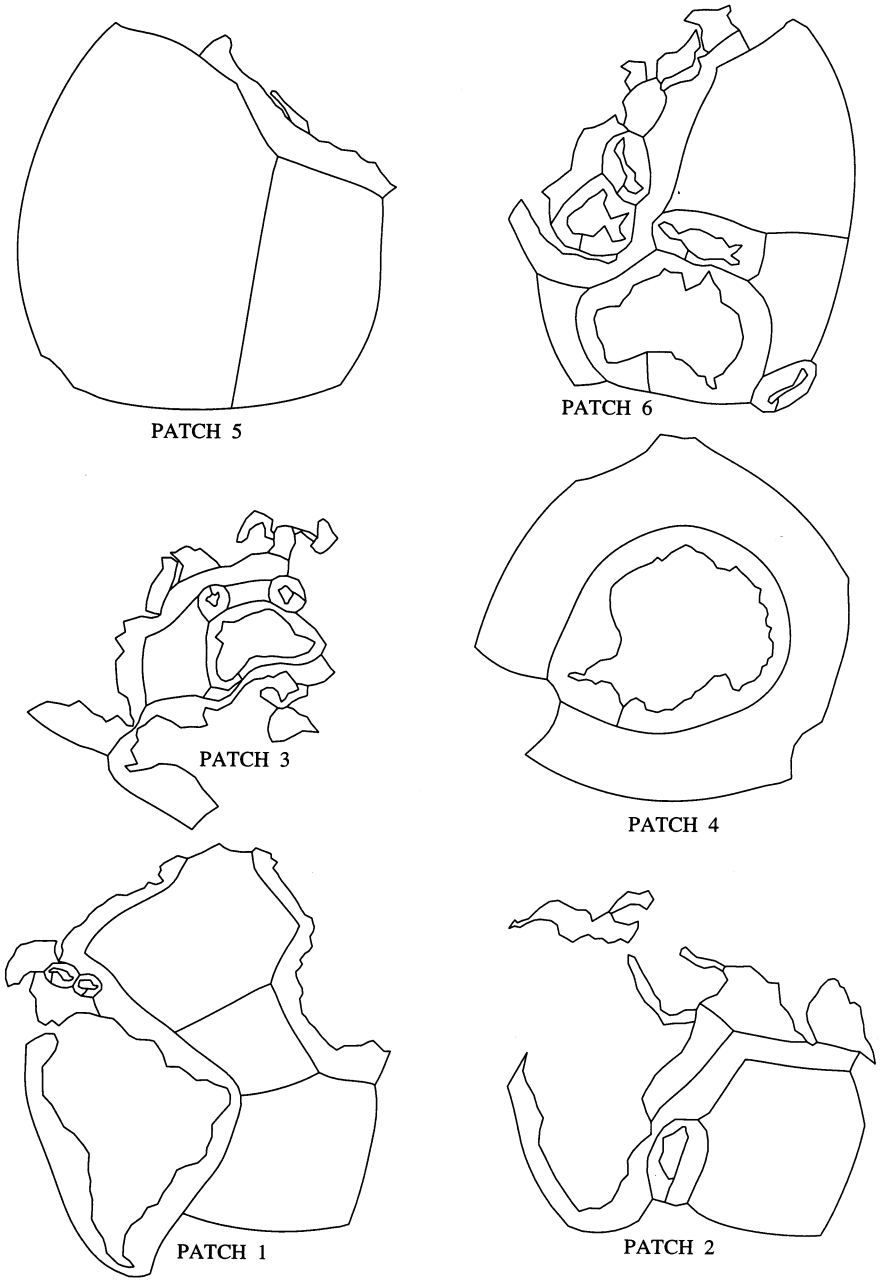

Figure 9. New, more advanced block structure for the surface grid generation method. *O-type* grids are used where possible for a number of reasons: grid lines can be added and subtracted without affecting the weak singularities and image values for the floating boundaries; fixed boundary values are associated with a unique block (better for simulation purposes); it allows a smoother grid (weak singularities are free to move about); better foundations for implementation of orthogonal boundaries.

time needed to generate the grid is negligible. So, what has been achieved for this negligible computer time? It is instructive to conclude this section by including the equivalent projections for the original 4×5 Cartesian longitude–latitude grid presented in Figure 3. The previously

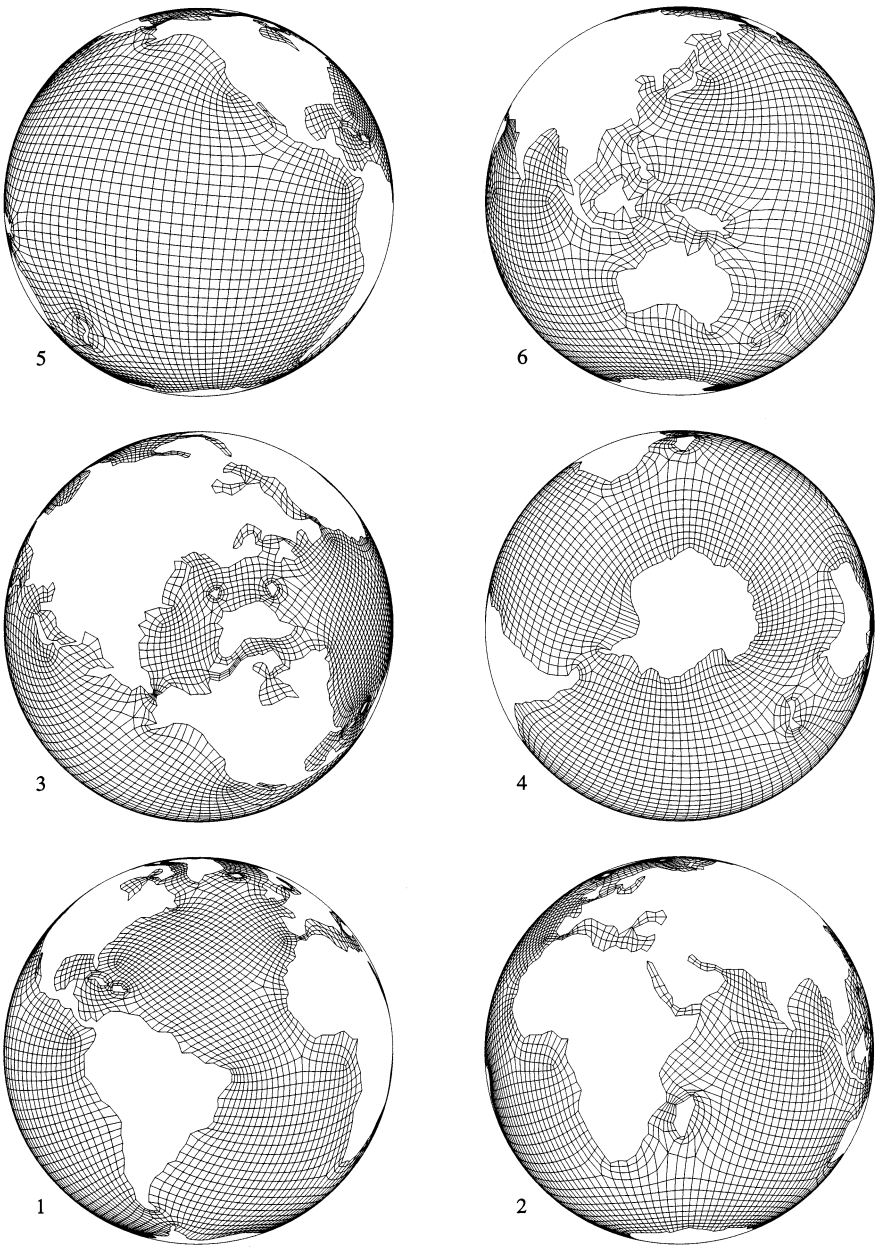

Figure 10. The converged, higher resolution grids produced from the new block structure presented in Figure 9. Although there seem to be more weak singularities than the grid in Figure 8 there are, in fact, approximately the same number but they now lie over the ocean instead of at coastal locations. They are therefore further away from boundary currents (which are more important from an oceanographic standpoint) and consequently less likely to affect the global simulations adversely.

mentioned advantages of the numerically generated boundary conforming grid now become clear on comparison of Figures 10 and 11.

4. STEADY STATE HEAT CONDUCTION SIMULATION

Ultimately, grids have been developed to be implemented in the Bryan–Cox global ocean circulation model. This requires a three-dimensional grid. In order to test the two-dimensional grid and to determine whether the metrics and derivatives are being calculated correctly, a simple temperature conduction problem is formulated on the grid shown in Figure 8. The grid is generated in local (x, y) co-ordinates on the sphere surface which has a one-to-one correspondence with global (X, Y, Z) Cartesian co-ordinates.

Consider the heat conduction equation in three-dimensional Cartesian co-ordinates

$$T_t = -k[T_{XX} + T_{YY} + T_{ZZ}]. \tag{22}$$

This has the corresponding equation on the surface of a sphere, given by

$$T_t = -k\left[\frac{1}{a_e^2 \sin\phi}(\sin\phi T_\phi)_\phi + \frac{1}{a_e^2 \sin^2\phi} T_{\lambda\lambda}\right], \tag{23}$$

where $\lambda \in [0, 2\pi]$, $\phi \in [0, \pi]$, (λ, longitude and ϕ, co-latitude measured from north pole). However, this form of the equation is not necessary since we have from Section 2, the base vectors relating Cartesian and curvilinear co-ordinates. Then, from (8) we have

$$\nabla^2 T = \sum_{i=1}^{2}\sum_{j=1}^{2} \mathbf{a}^i \cdot \mathbf{a}^j T_{\xi^i \xi^j} + \sum_{i=1}^{2}\sum_{j=1}^{2} \mathbf{a}^i \cdot (\mathbf{a}^j)_{\xi^i} T_{\xi^j}, \tag{24}$$

where the sum is to two because ξ^3 is constant. Note also that $\mathbf{a}^3 = \mathbf{a}_3$. The relationship between global (X, Y, Z) co-ordinates on the *sphere surface* and local (x, y) co-ordinates for each patch is simple. The resulting equation for the heat conduction then becomes

$$T_t = -\frac{k}{a^2}[c_1 T_{\xi\xi} + 2c_2 T_{\xi\eta} + c_3 T_{\eta\eta} + c_4 T_\xi + c_5 T_\eta], \tag{25}$$

where

$$c_1 = \mathbf{a}^1 \cdot \mathbf{a}^1 = g^{11} = g_{22}/g, \tag{26a}$$

$$c_2 = \mathbf{a}^1 \cdot \mathbf{a}^2 = g^{12} = -g_{12}/g, \tag{26b}$$

$$c_3 = \mathbf{a}^2 \cdot \mathbf{a}^2 = g^{22} = g_{11}/g, \tag{26c}$$

$$c_4 = \mathbf{a}^1 \cdot (\mathbf{a}^1)_\xi + \mathbf{a}^2 \cdot (\mathbf{a}^1)_\eta, \tag{26d}$$

$$c_5 = \mathbf{a}^1 \cdot (\mathbf{a}^2)_\xi + \mathbf{a}^2 \cdot (\mathbf{a}^2)_\eta, \tag{26e}$$

where g_{22}, g_{12} and g_{11} are given by Equations (10a), (10b) and (10c), respectively.

These equations were programmed and tested for a simple rectangular region in Cartesian (λ, ϕ) space, as shown in Figure 12. Exact theoretical solutions (derived by separation of variables) can be obtained for the steady state problem by specifying simple boundary conditions. Consider the domain $\lambda \in [-5\pi/6, -\pi/3]$, $\phi \in [\pi/18, \pi/2]$ and the boundary conditions

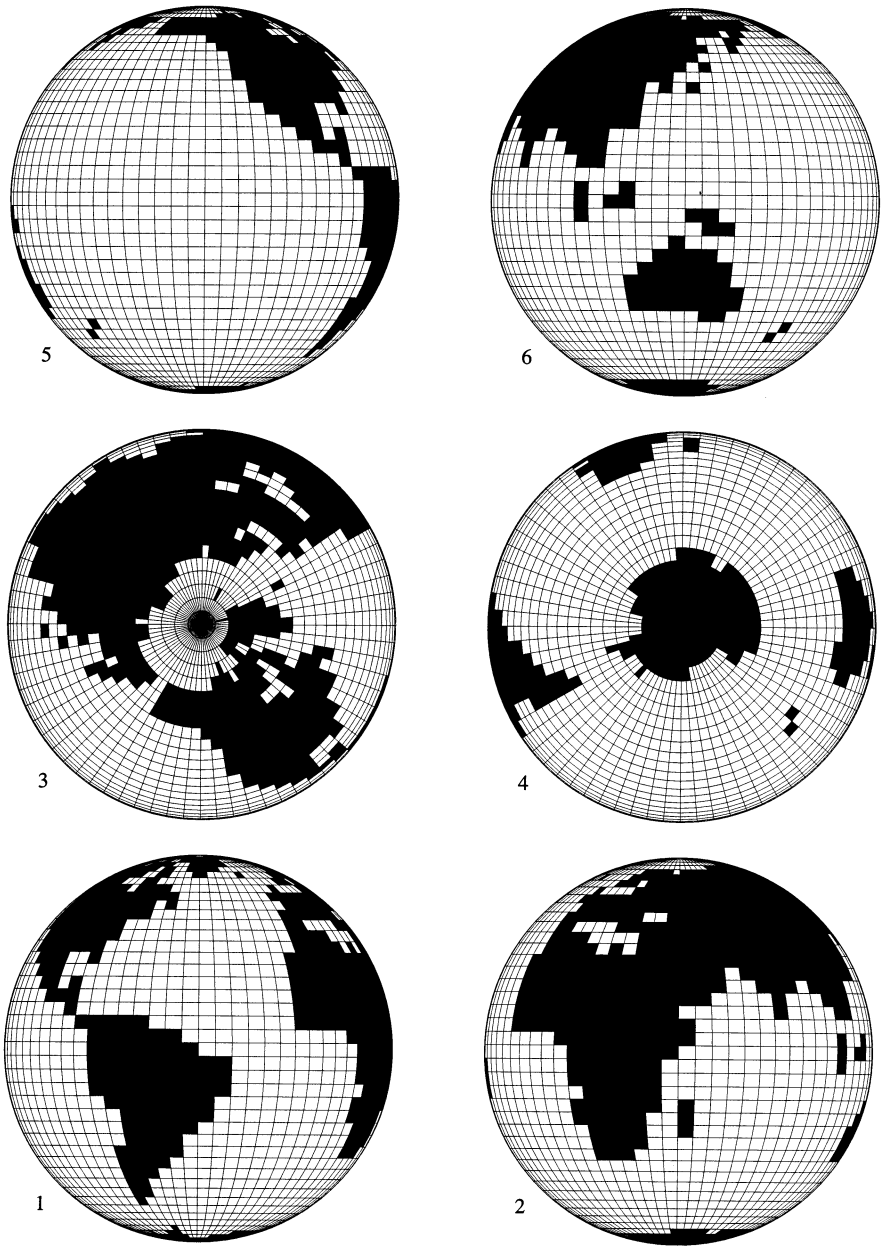

Figure 11. Equivalent projection to those presented in Figures 8 and 10 of the standard Cartesian 4×5 longitude–latitude grid seen in Figure 3. This projection of the standard grid illustrates wasted resolution at the North Pole and that boundaries are not only sawtooth, but often quite inaccurate. In addition, many flow-through regions are blocked off and islands and bays are often eliminated completely and therefore excluded in ocean simulations.

$$T(-5\pi/6, \phi) = 0, \tag{27a}$$

$$T(-\pi/3, \phi) = 0, \tag{27b}$$

$$T(\lambda, \pi/2) = 0, \tag{27c}$$

$$T(\lambda, \pi/18) = -2\ \sin(\pi/3 - 2\lambda). \tag{27d}$$

The exact steady state solution to Equation (24) can be expressed as

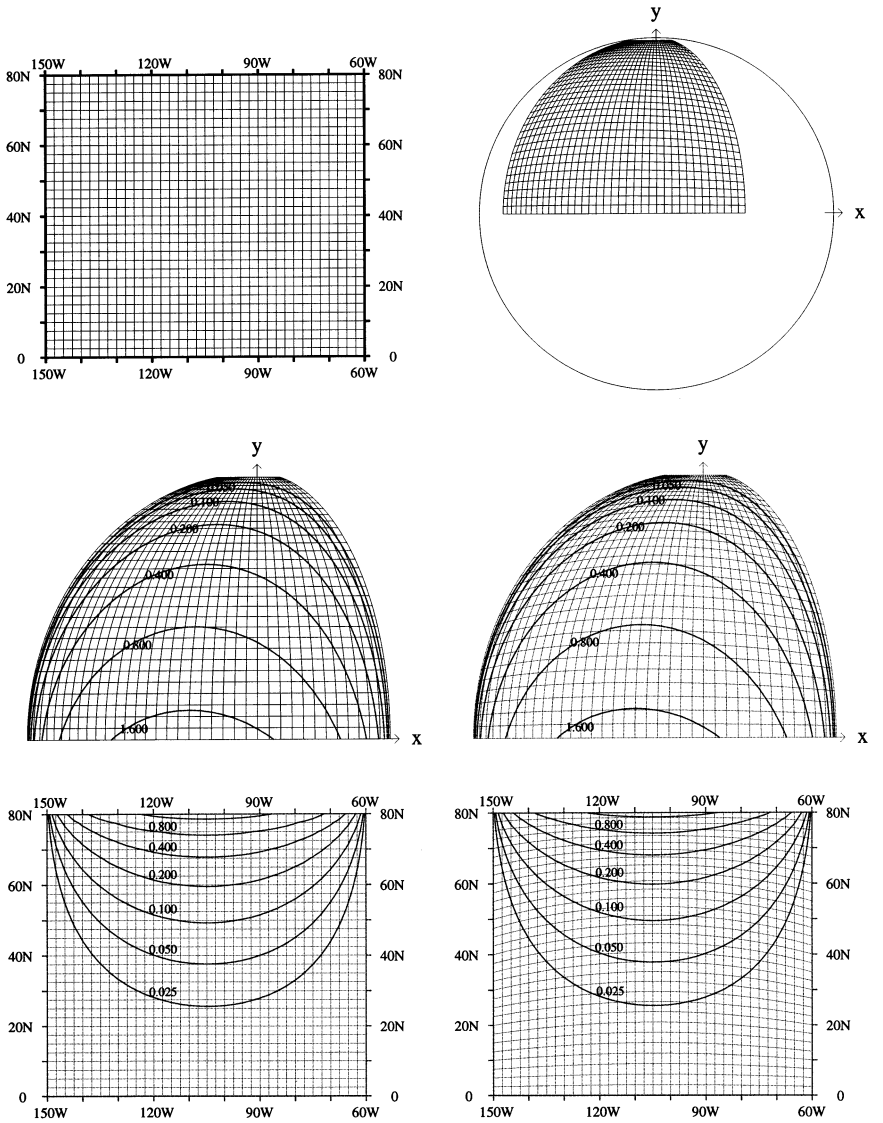

Figure 12. Two grids used for a simple temperature conduction problem for the domain $\lambda \in [-5\pi/6, -\pi/3]$, $\phi \in [\pi/18, \pi/2]$ with boundary conditions for the middle figures given by $T(\lambda, 0) = -2\ \sin(\pi/3 - 2\lambda)$; $T = 0$ elsewhere. For the bottom plots, the boundary conditions are $T(\lambda, \pi/18) = -2\ \sin(\pi/3 - 2\lambda)$ elsewhere. The standard Cartesian grid is shown top left (with longitude and latitude labels). A numerically generated surface grid for the same domain in shown top right. The middle figures show the two grids mapped onto the sphere with a non-zero lower boundary condition. Isotherms are plotted for various values of temperature (0.025, 0.05, 0.1, 0.2, 0.4, 0.8, 1.6; only some are labeled). The lower figures show the two grids mapped to the (λ, ϕ) plane with a non-zero upper boundary condition. For both problems the different grids yield equally good results.

$$T(\lambda, \phi) = 2D \sin(\pi/3 - 2\lambda) \frac{\cos \phi}{\sin^2 \phi}, \tag{28}$$

and the corresponding problem in local (x, y) space has the steady state solution

$$T(x, y) = \frac{D_1 a_e y}{(a_e^2 - y^2)^2} [2x(a_e^2 - x^2 - y^2)^{1/2} - \sqrt{3}(a_e^2 - 2x^2 - y^2)], \tag{29}$$

where

$$D_1 = \frac{-\sin^2(\pi/18)}{\cos(\pi/18)}. \tag{30}$$

Using the $(M \times N) = (37 \times 33)$ meshes shown in Figure 12, the contours are plotted for the associated (λ, ϕ) and (x, y) projections. Both grids yield accurate solutions and, on the contour plots, are indistinguishable from the exact solutions. The relative (2-norm) and absolute (1-norm) errors can be expressed as

$$e_{\text{rel}} = \frac{\sum_{i=2}^{M-1} \sum_{j=2}^{N-1} (T_{ij}^{\text{c}} - T_{ij}^{\text{e}})^2}{\sum_{i=2}^{M-1} \sum_{j=2}^{N-1} (T_{ij}^{\text{e}})^2}, \tag{31}$$

and

$$e_{\text{abs}} = \frac{\sum_{i=2}^{M-1} \sum_{j=2}^{N-1} |T_{ij}^{\text{c}} - T_{ij}^{\text{e}}|}{(M-2)(N-2)}, \tag{32}$$

where T^{c} represents the computed value of T and T^{e} is the exact value. For the Cartesian (λ, ϕ) grid, the errors are $e_{\text{rel}} = 4.464 \times 10^{-4}$ and $e_{\text{abs}} = 4.249 \times 10^{-3}$. For the irregular grid, $e_{\text{rel}} = 3.007 \times 10^{-4}$ and $e_{\text{abs}} = 4.413 \times 10^{-3}$. The relative errors are slightly better for the irregular grid because co-ordinate lines are attracted to the convex top boundary. This is typical for Laplacian grids. The gradient from this non-zero boundary is resolved just slightly better. The exact values at the grid points, compared with corresponding grid points on the Cartesian (λ, ϕ) grid are slightly greater. Consequently, the absolute errors for the irregular grid are slightly higher.

If the non-zero boundary is now switched to be the lower boundary (in effect, a more realistic physical condition), the steady state solutions can be expressed as

$$T(\lambda, \phi) = -2D_2 \sin(\pi/3 - 2\lambda) \left[\frac{\sin^2(\phi)}{D_2(1 - \cos(\phi))^2} + \frac{4 \cos \phi}{\sin^2 \phi} \right], \tag{33}$$

$$T(x, y) = \frac{(4D_2 a y - (a_e + y)^2)}{(a_e^2 - y^2)^2} [2x(a_e^2 - x^2 - y^2)^{1/2} - \sqrt{3}(a_e^2 - 2x^2 - y^2)], \tag{34}$$

where

$$D_2 = -\frac{\cos^4(\pi/36)}{\cos(\pi/18)}. \tag{35}$$

The errors for the Cartesian (λ, ϕ) grid for this problem are $e_{\text{rel}} = 1.280 \times 10^{-4}$ and $e_{\text{abs}} = 5.465 \times 10^{-3}$ and for the irregular grid, $e_{\text{rel}} = 1.321 \times 10^{-4}$ and $e_{\text{abs}} = 4.997 \times 10^{-3}$. Hence,

the absolute error for the irregular grid is slightly better for this problem and the relative error slightly worse (which is to be expected due to the aforementioned convex boundary). Note that the relative errors are lower for the second test case. Exact values for the second case, throughout the grid, are generally greater compared with the first case because the non-zero boundary is physically longer.

These simple test cases confirm that the governing equations (24) being solved in physical and computational space are consistent with (23) in (λ, ϕ) space and that the metric and derivative terms at each grid point (26) are being computed successfully. The irregular structured grids and the standard Cartesian grids perform equally well for the simple rectangular domain. One method may be slightly better given certain boundary conditions. However, given complicated regions with complex boundary conditions, carefully generated, irregular grids can yield significantly more accurate results.

In order to test the global grid and to check whether the temperature values are being updated and matched correctly across successive blocks, a simple temperature distribution $T(\lambda, \phi) = 1 - 2|\phi|/\pi$ is specified at each coastal value. Thus, boundary temperature values are unity at the equator and zero at the poles. The resulting steady state temperature contours are shown in Figure 13 using the same projections seen in Figure 8. The simulations produce temperature distributions and contours similar to those for a Cartesian grid simulation and the global grid can now be employed for more complicated simulations related to the circulation of the world's oceans.

5. DISCUSSION

An irregular, boundary conforming, structured grid for global ocean simulations has been generated numerically, using differential equation techniques. It was produced by employing block structured techniques which allow many of the smaller seas and basins to be included or eliminated with ease, and which preserve the slope continuity of the co-ordinate lines across different ocean regions. The block structure has been coupled with an innovative *spherical patching* approach which permits all areas of the global ocean to be modeled with essentially the same resolution everywhere. The problems associated with polar singularities have been overcome and traded for a number of floating weak singularities. This grid does not inherently produce high resolution in polar regions where it is not needed. In fact, due to the smoothing nature of the Laplace system, the cell areas are fairly uniform. This is also a consequence of the extension to surface grid generation from generation on a plane. One of the goals of this research was to produce a nearly uniform grid and leave the manipulation of grid points and resolution to clustering functions P and Q; i.e., a desire to minimize area variation across neighboring cells by manipulating the block structure and varying the location of the coastal boundary points. This way, P and Q forcing terms need not be too complex in order to produce the ultimately desired resolution in various regions of the ocean.

The current grid in use for simulations (Figure 8) has approximately the same number of cells as a standard 4×5 grid. It is expected that for the two-dimensional wind-driven ocean circulation problem (the barotropic or stream function calculation [22]) the numerical grid will give more accurate results. In addition, with the more uniform distribution of grid points, some of the physical oceanography characteristics such as western boundary intensification [5,6] can be solved to a much higher degree than the standard Cartesian grid.

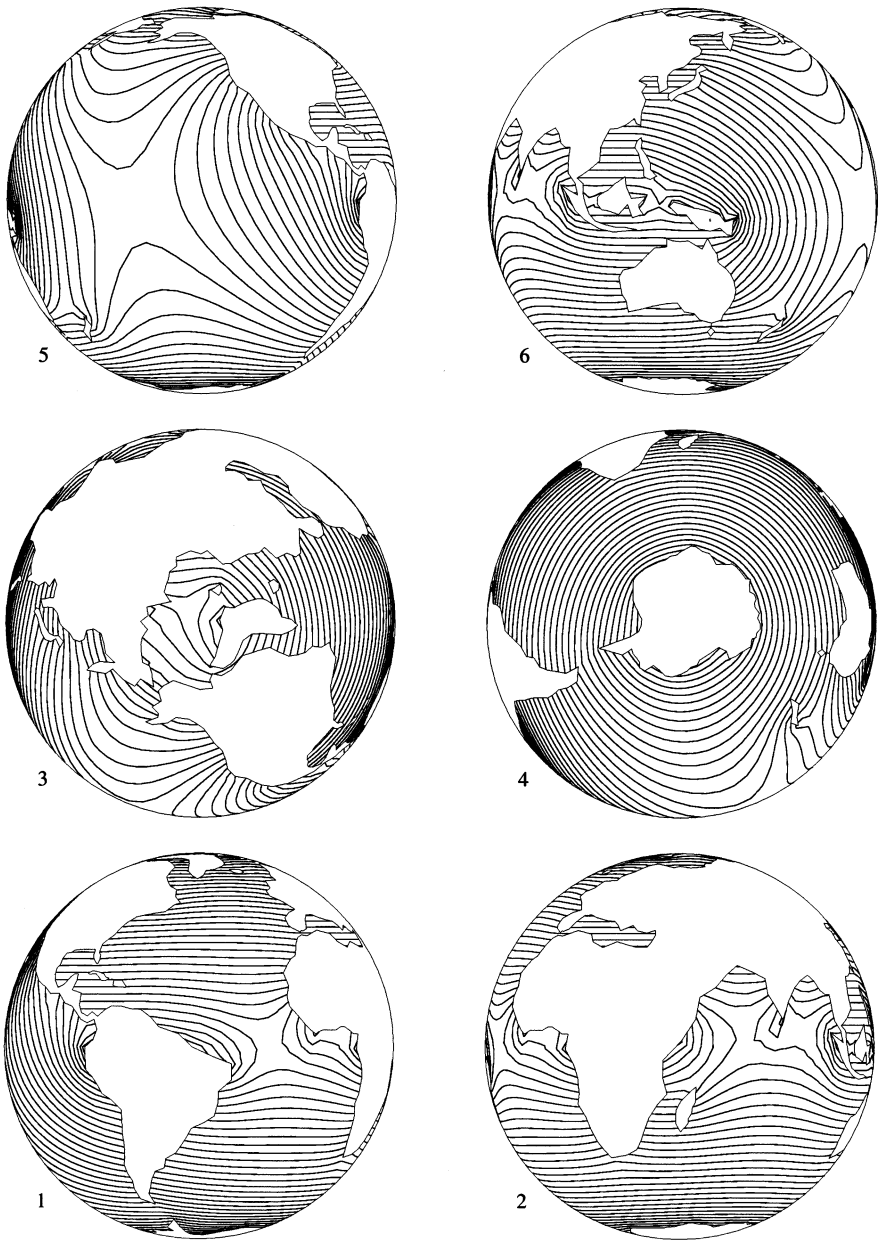

Figure 13. Steady state isotherms for the heat conduction problem. The grid used is shown in Figure 8. The Dirichlet boundary condition is $T = 1 - 2|\phi|/\pi$ (unity at the equator, zero at the poles) at all coastal locations and the initial conditions are $T = 0$ for all ocean values.

This is currently under investigation. It is also expected that because of the boundary conforming nature of the grid, ocean boundary currents will be more accurately modeled due to boundary values being prescribed at exact coastal locations (instead of to the nearest few degrees), and to derivative boundary conditions being applied more precisely. The new,

higher resolution grid will then be tested to determine how much accuracy is gained by using more grid points, what significant effects are produced by moving the weak singularities off the boundaries, and whether the inclusion of the less prominent bodies of water has an effect on the global stream function distribution.

This work represents the first step in the direction of a new global ocean simulation. The next phase involves manipulation of the grid points using the clustering functions P and Q. These functions P and Q essentially allow the grid points to be placed anywhere. This implies that characteristics of the Kuroshio, Gulf Stream, East Australian, Agulhas and Brazil currents may be evaluated with even more accuracy. Moreover, important dynamical regions such as the equator (which acts as a wave duct) may be integrated into the grid. It must be stressed here that with 3500 grid points it is impossible to incorporate the general characteristics of all oceanic features into the grid, nor is it possible to resolve physics on all scales. However, it *is* possible to develop different grids to address different oceanic circulation problems.

From an ocean modeling standpoint it appears that major advances have been made in this study, since up to now, high resolution in one region of the ocean implied high resolution everywhere else. Although regional ocean domains have been analyzed with irregular grids [11] global ocean domains have not, especially not with spherical surface grids. Irregular grids generated on the (λ, ϕ) plane defeat the purpose because they produce inherent high resolution at the poles where it is obviously not required. Another improvement would be to enforce orthogonality at the boundaries to allow simple numerical application of the various boundary conditions. Block structure can be improved by adding blocks. Since our simulation will soon be extended to three dimensions, which will increase the number of blocks dramatically, the focus is more towards innovative block structure.

Extension to three dimensions can be accomplished in many ways. The initial formulation here will be to adopt vertical grid lines for depth and to use grid lines at a constant depth in the horizontal. This simplifies the pressure and density calculations significantly. More complex approaches include using different ocean outlines at specific depths, vertical grid lines with accurate bathymetry but smoothly varying grid box depths, and what is eventually hoped for, a full three-dimensional boundary conforming, numerically generated, structured grid.

It would also be desirable to formulate an algorithm which could automatically produce boundary values for the ocean domains when block structure is modified. Moreover, when simulating standard resolutions ($4 \times 5, 2 \times 2\frac{1}{2}, 1 \times 1$, etc.) spacing between grid point locations along land–ocean interface boundaries should be a function of both curvature and length. Concave boundaries generally require more grid points than convex boundaries and as concavity increases (greater curvature) a greater number of grid points is required to produce a smooth, accurate, acceptable, numerically generated grid. As these higher resolutions are simulated, the boundaries must be modified to accommodate additional islands and flow-through regions. Only those islands deemed to play a significant role in oceanic circulation will be included.

The grid generation techniques employed in this study are naturally amenable to parallel implementation. Each block can be assigned to a different processor and information from adjacent boundaries can be passed to associated processors. Not only is it possible to make the generation of the grid parallel, but the physical simulations of the ocean could also be implemented on MPPs. Accordingly, it is imperative that information relating adjacent blocks is kept as fundamental as possible. Furthermore, since the addition of blocks is

expected to increase accuracy at the cost of computational efficiency on scalar machines, the scalability of this method on MPPs looks promising.

A global ocean simulation with a fully three-dimensional, irregular, structured grid is now on the horizon. Once this has been achieved, a strong possibility would be to develop an adaptive grid to allow thermohaline circulation effects to be revealed, salinity to be transported accurately and various boundary currents to be resolved. Our ultimate goal is to formulate a fully coupled atmosphere–ocean model which utilizes a three-dimensional, boundary conforming, irregular, structured, adaptive grid.

ACKNOWLEDGMENTS

We wish to acknowledge the support of the NASA EOS-IDS program and the NASA Climate Modeling Program. We also wish to thank Inez Fung for useful discussions and suggestions during the course of this work. The comments of the reviewers were also very helpful.

REFERENCES

1. K. Bryan, 'A numerical method for the study of the circulation of the world ocean', *J. Comput. Phys.*, **4**, 347 (1969).
2. M.D. Cox, 'A primitive equation, 3-dimensional model of the ocean', *GFDL Technical Report 1*, (1984).
3. W.M. Washington and G.A. Meelh, 'Climate sensitivity due to increased CO_2: experiments with a coupled atmosphere and ocean general circulation model', *Climate Dyn.*, **4**, 1 (1989).
4. A.J. Semtner and R.M. Chervin, 'Ocean general circulation from a global eddy resolving model', *J. Geophys. Res.*, **97**, 5493 (1992).
5. H. Stommel and A.B. Arons, 'On the abyssal circulation of the world ocean—I. Stationary planetary flow patterns on a sphere', *Deep Sea Res.*, **6**, 140 (1960).
6. H. Stommel and A.B. Arons, 'On the abyssal circulation of the world ocean—II. An idealized model of the circulation pattern and amplitude in oceanic basins, *Deep Sea Res.*, **6**, 217 (1960).
7. A.F. Blumberg and G.L. Mellor, 'A description of a three-dimensional coastal ocean circulation model', in R. Heaps (ed.), *Three Dimensional Coastal Ocean Models*, American Geophysical Union, Washington DC, 1990, p. 1.
8. Y.F. Xie, G.L. Browning and C.G. Chesshire, 'The composite grid method for the reduced system for the shallow-water equations', *SIAM J. Sci. Comput.*, submitted (1993).
9. M. Ciment, 'Stable difference schemes with uneven mesh spacings', *Math. Comput.*, **25**, 219 (1971).
10. S.P. Spekreijse, J.W. Boerstoel, P.L. Vitagliano and J.L. Kuyvenhoven, 'Domain modeling and grid generation for multi-block structured grids with application to aerodynamic and hydrodynamic configurations', in R.E. Smith (ed.), *Software Systems for Surface Modeling and Grid Generation*, NASA-CP-3143, 207 (1992).
11. D.B. Haidvogel, A. Beckmann and K.S. Hedstrom, 'Dynamical simulations of filament formation and evolution in the coastal transition zone', *J. Geophys. Res.*, **96**, 15017 (1991).
12. J. Adams, R. Garcia, B. Gross, J. Hack, D.B. Haidvogel and V. Pizzo, 'Application of multigrid software in the atmospheric sciences', *Mon. Weather Rev.*, **120**, 1447 (1991).
13. I. Fukumori, J. Benveniste, C. Wunsch and D.B. Haidvogel, 'Assimilation of sea surface topography into an ocean circulation model using a steady-state smoother', *J. Phys. Oceanogr.*, **23**, 1831 (1993).
14. D.W. Barnette, 'Progress report on high-performance high-resolution simulations of coastal and basin-scale ocean simulations', Parallel Computational Sciences Department, *Sandia National Laboratories*, (1994).
15. K. Bryan and M.D. Cox, 'An approximate equation of state for numerical models of ocean circulation', *J. Phys. Oceanogr.*, **2**, 510 (1972).
16. A.J. Semtner, 'An oceanic general circulation model with bottom topography', *Numerical Simulation of Weather and Climate Technical Report 9*, Department of Meteorology, UCLA (1974).
17. J.F. Thompson, Z.U.A. Warsi and C.W. Mastin, *Numerical Grid Generation: Foundations and Applications*, North-Holland, New York, 1985.
18. Z.U.A. Warsi, 'Basic differential models for coordinate generation', J.F. Thompson (ed.), *Numerical Grid Generation*, North-Holland, New York, 1982, p. 41.
19. J.F. Thompson, F.C. Thames and C.W. Mastin, 'Automatic numerical generation of body-fitted curvilinear coordinate systems for fields containing any number of arbitrary two-dimensional bodies', *J. Comput. Phys.*, **15**, 299 (1974).

20. J.F. Thompson and Z.U.A. Warsi, 'Boundary-fitted coordinate systems for numerical solution of partial differential equations—a review', *J. Comput. Phys.* **47**, 1 (1982).
21. Z.U.A. Warsi, 'A note on the mathematical formulation of the problem of numerical coordinate generation', *Q. Appl. Math.*, **41**, 221 (1983).
22. H. Stommel, 'The westward intensification of wind-driven ocean currents', *Trans.: Am. Geophys. Union*, **29**, 202 (1948).

