A Fast Dynamic Grid Adaption Scheme for Meteorological Flows

BRIAN H. FIEDLER AND R. JEFFREY TRAPP

School of Meteorology, University of Oklahoma, Norman, Oklahoma

(Manuscript received 5 October 1992, in final form 27 April 1993)

ABSTRACT

The continuous dynamic grid adaption (CDGA) technique is applied to a compressible, three-dimensional model of a rising thermal. The computational cost, *per grid point* per time step, of using CDGA instead of a fixed, uniform Cartesian grid is about 53% of the total cost of the model with CDGA. The use of general curvilinear coordinates contributes 11.7% to this total, calculating and moving the grid 6.1%, and continually updating the transformation relations 20.7%. Costs due to calculations that involve the gridpoint velocities (as well as some substantial unexplained costs) contribute the remaining 14.5%. A simple way to limit the cost of calculating the grid is presented. The grid is adapted by solving an elliptic equation for gridpoint coordinates on a coarse grid and then interpolating the full finite-difference grid. In our application, the additional costs per grid point of CDGA are shown to be easily offset by the savings resulting from the reduction in the required number of grid points. In the simulation of the thermal, we are able to reduce costs by a factor of 3, as compared with those of a companion model with a fixed, uniform Cartesian grid.

1. Introduction

The technique of continuous dynamic grid adaption or CDGA (Dietachmayer and Droegemeier 1992; Dietachmayer 1992) has recently been introduced to the meteorological community as a viable means of increasing the accuracy of numerical meteorological models. This paper adds to this recent work by contributing a significantly more efficient algorithm for adapting the grid in both two and three dimensions. For demonstrative purposes, we apply CDGA to a model of a rising thermal.

We will not repeat a broad review and introduction here but we will reiterate a few points about CDGA that are sometimes misunderstood. In CDGA, the grid points move smoothly and continuously in time in order to adapt to moving features requiring enhanced resolution. The grid movement is naturally incorporated into the advection term. At no time in the integration do the model fields have to be interpolated onto new grids. When viewed in physical space, the grid is skew and nonuniform; however, a transformation to logical coordinates renders the grid orthogonal and uniform. All the familiar finite-difference approximations for uniform, orthogonal grids can thus be used in CDGA. The calculation of grid velocities and grid transformation relations are the only additional requirements for a CDGA code.

Corresponding author address: Dr. Brian H. Fiedler, School of Meteorology, University of Oklahoma, Energy Center, 100 East Boyd, Room 1310, Norman, OK 73019-0470.

In three dimensions, the computational cost, in CPU time, for continually calculating the grid is about 6% of the total cost of the CDGA model. In the two-dimensional model of Dietachmayer (1992), this cost could have been as great as 40% of his CDGA model. The guiding principal behind our new grid adaption scheme is that the grid need not vary on scales as small as those in the flow: the gridpoint coordinates are found by first rapidly solving an elliptic equation for a coarse grid and then interpolating the coordinates of the finite-difference grid. Grid velocities are then computed to adjust the current finite-difference grid toward the newly computed grid. The smoothly varying grid is efficient to solve for, minimizes one source of discretization error, and need not be updated as often as a more elaborate grid.

One indirect cost of continuously moving the grid is that new transformation relations must be calculated at each time step. This cost is 20.7% of the total. Computations that involve the velocity of the grid points require about 14.5% of the total. (There may be some other substantial hidden costs contributing to the last number that our experiments did not elucidate.) The cost of using a general curvilinear coordinate system, a cost that would be incurred even if the grid were static, is 11.7% of the total. Thus, the cost *per grid point* per time step, of using CDGA instead of a fixed, uniform Cartesian grid is 53% of the total cost of the model with CDGA. (The 53% includes the 6% mentioned in the previous paragraph.) The benefits of CDGA are that the number of grid points required for a simulation can be reduced and therefore both the maximum memory usage and total CPU time for an

integration can be reduced. In our simulation of a thermal with CDGA, we are able to reduce the number of grid points by a factor of 8, and save a factor of 6.2 in maximum memory usage and a factor of 3 in total CPU time, as compared with a companion fixed-grid model.

2. Numerical model

The model solves the following equations for compressible, buoyant convection:

$$\left(\frac{\partial}{\partial t} + \mathbf{u}\cdot\nabla\right)\mathbf{u} = -\frac{1}{\rho}\nabla p + (1 - e^{-2t})b\hat{\mathbf{e}}_3 + \nu\nabla^2\mathbf{u}, \tag{1}$$

$$\frac{\partial \rho}{\partial t} = -\nabla\cdot(\rho\mathbf{u}), \tag{2}$$

$$\left(\frac{\partial}{\partial t} + \mathbf{u}\cdot\nabla\right)b = \kappa\nabla^2 b, \tag{3}$$

and

$$\rho = 1 + \frac{p}{c^2}, \tag{4}$$

where ν, κ, and c are the dynamic viscosity, thermal diffusivity, and speed of sound, respectively—all of which are assumed constant. The model variables are considered to be dimensionless. The factor $(1 - e^{-2t})$ reduces the amplitude of acoustic waves that are generated at initialization.

The necessary background for applying adaptive grids to convection–diffusion equations can be found in Thompson et al. (1985). The relations essential for the transformation of (1)–(3) from physical space to logical space are summarized in the Appendix.

A great savings in the usage of computer memory is obtained when the finite-difference approximations are applied to the nonconservative form of the transformed equations instead of the conservative form. With the nonconservative form, the transformation relations can be discarded after all the tendencies are calculated at a grid point; the transformation relations are never differenced and need not be stored in arrays. Applying (A11)–(A13) to (1)–(3) gives the transformed, nonconservative model equations

$$\frac{\partial u_i}{\partial t} = -\sum_{j=1}^{3} U^j \frac{\partial u_i}{\partial \xi^j} - \frac{\hat{\mathbf{e}}_i}{\rho}\cdot\sum_{j=1}^{3}\mathbf{a}^j\frac{\partial p}{\partial \xi^j} + (1 - e^{-2t})\hat{\mathbf{e}}_i\cdot\hat{\mathbf{e}}_3 b + \nu\nabla^2 u_i, \tag{5}$$

$$\frac{\partial p}{\partial t} = \sum_{j=1}^{3}\dot{\mathbf{x}}\cdot\mathbf{a}^j\frac{\partial p}{\partial \xi^j} - c^2\left[\sum_{j=1}^{3}\mathbf{a}^j\cdot\frac{\partial}{\partial \xi^j}(\rho\mathbf{u})\right] \tag{6}$$

and

$$\frac{\partial b}{\partial t} = -\sum_{j=1}^{3} U^j\frac{\partial b}{\partial \xi^j} + \kappa\nabla^2 b. \tag{7}$$

The finite-difference approximations to (5)–(7) are solved on a lattice in which p is staggered from both b and $\mathbf{u}$; the u nodes occupy the boundary of the domain (Fig. 1). A fifth-order upwind scheme (Dietachmayer 1992) is applied to all advective terms. An example of the fifth-order upwind scheme is

$$U^1\frac{\partial A(\xi^1)}{\partial \xi^1} = \frac{U^1}{60h}\times[-3A(\xi^1 + 2h) + 30A(\xi^1 + h) + 20A(\xi^1) - 60A(\xi^1 - h) + 15A(\xi^1 - 2h) - 2A(\xi^1 - 3h)] \tag{8}$$

for $U^1 > 0$, and

$$U^1\frac{\partial A(\xi^1)}{\partial \xi^1} = -\frac{U^1}{60h}\times[-2A(\xi^1 + 3h) + 15A(\xi^1 + 2h) - 60A(\xi^1 + h) + 20A(\xi^1) + 30A(\xi^1 - h) - 3A(\xi^1 - 2h)] \tag{9}$$

for $U^1 < 0$. In (8) and (9) we have implied $A(\xi^1) \equiv A(\xi^1, \xi^2, \xi^3, t)$. The grid increment h between consecutive u nodes or p nodes is chosen to be

$$h \equiv \Delta\xi^1 = \Delta\xi^2 = \Delta\xi^3 = 2. \tag{10}$$

The pressure gradient is calculated to second order:

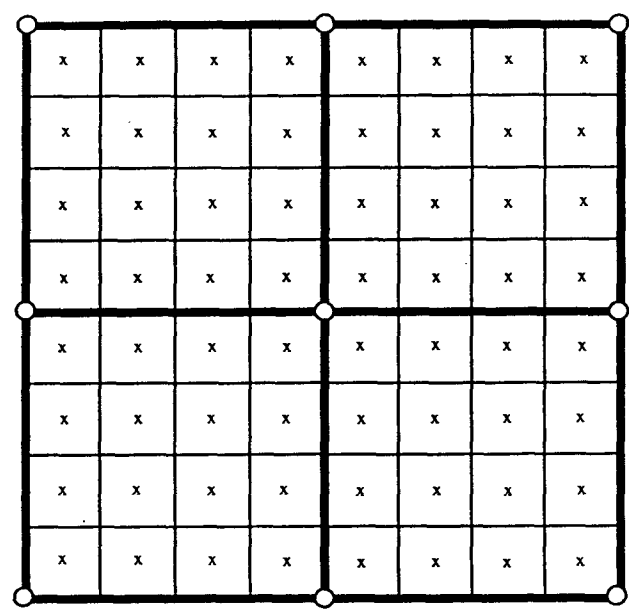

FIG. 1. The grids of the two-dimensional model. The u nodes of the full grid are connected by thin lines, the p nodes are indicated by "×", and the coarse grid used for grid adaption is connected by thick lines.

$$\frac{\partial p}{\partial x_i} \approx \hat{\mathbf{e}}_i \cdot [\mathbf{a}^1 \overline{\delta_{\xi^1}(p)}^{\xi^2\xi^3} + \mathbf{a}^2 \overline{\delta_{\xi^2}(p)}^{\xi^1\xi^3} + \mathbf{a}^3 \overline{\delta_{\xi^3}(p)}^{\xi^2\xi^1}]. \quad (11)$$

The pressure tendency is also calculated to second order:

$$\frac{\partial p}{\partial t} \approx (\dot{\mathbf{x}}_p \cdot \mathbf{a}^1)\delta_{\xi^1}(p) + (\dot{\mathbf{x}}_p \cdot \mathbf{a}^2)\delta_{\xi^2}(p) + (\dot{\mathbf{x}}_p \cdot \mathbf{a}^3)\delta_{\xi^3}(p) - c^2[\mathbf{a}^1 \cdot \overline{\delta_{\xi^1}(\rho \mathbf{u})}^{\xi^2\xi^3} + \mathbf{a}^2 \cdot \overline{\delta_{\xi^2}(\rho \mathbf{u})}^{\xi^1\xi^3} + \mathbf{a}^3 \cdot \overline{\delta_{\xi^3}(\rho \mathbf{u})}^{\xi^1\xi^2}] \quad (12)$$

where $\dot{\mathbf{x}}_p$ is interpolated from the u nodes to p nodes with

$$\dot{\mathbf{x}}_p \equiv \bar{\dot{\mathbf{x}}}^{\xi^1\xi^2\xi^3}. \quad (13)$$

The time step required for numerical stability is limited by the acoustic wave speed; we use

$$\Delta t = \frac{0.3[\min(g^{1/2})]^{1/3}}{c}. \quad (14)$$

The Jacobian $g^{1/2}$ represents the physical volume of the smallest grid cells. If, in physical space, the smallest grid cells are cubic, then (14) would be the form of the conventional Courant–Friedrichs–Lewy condition. However, since the smallest grid interval is sometimes associated with grid cells that are not cubic, we found we had to use a factor no greater than 0.3 in (14) in order to ensure stability; if we considered only uniform, fixed-grid applications, then the factor could have been as large as 0.6.

The conventional third-order Adams–Bashforth scheme (e.g., Durran 1991) uses a constant value of time step Δt. Consider the generic equation for forecasting a variable at a grid point:

$$\frac{dq(t)}{dt} = f(t). \quad (15)$$

Because the minimum grid spacing and hence the time-step limit continually change, we use a third-order scheme for variable Δt, which is easily shown to be

$$q(t + \Delta t) \approx q(t) + \Delta t \times [Af(t) + Bf(t - \alpha\Delta t) + Cf(t - \beta\Delta t)], \quad (16)$$

where

$$C = \frac{2 + 3\alpha}{6\beta(\beta - \alpha)}, \quad (17)$$

$$B = -\frac{1 + 2\beta C}{2\alpha}, \quad (18)$$

and

$$A = 1 - B - C. \quad (19)$$

3. Grid adaption scheme

Our grid adaption scheme is based on an equidistribution principle (Dietachmayer and Droegemeier 1992; Dietachmayer 1992; Thompson et al. 1985) that concentrates grid points where a user-defined, positive weight function W is relatively large. Here, we use this principle to model the buoyancy field more accurately by increasing the local resolution both at and near the region where b is significant. The details of the CGDA scheme are as follows: First a function $\hat{W}$ is computed on the coarse grid by averaging the values of b from the full grid that fall within the grid cell surrounding a grid point of the coarse grid. The resultant function is then filtered for N_f iterations using

$$\hat{W}^{n+1}_{i,j,k} = \frac{1}{2}\hat{W}^{n}_{i,j,k} + \frac{1}{12} \times (\hat{W}^{n}_{i-1,j,k} + \hat{W}^{n}_{i+1,j,k} + \hat{W}^{n}_{i,j-1,k} + \hat{W}^{n}_{i,j+1,k} + \hat{W}^{n}_{i,j,k-1} + \hat{W}^{n}_{i,j,k+1}), \quad (20)$$

then normalized to fall between 0 and 1, and then "clipped" with

$$\tilde{W} = \begin{cases} \hat{W}, & \hat{W} < \gamma \\ \gamma, & \hat{W} \geq \gamma, \end{cases} \quad (21)$$

where $0 < \gamma \leq 1$. The final form of the weight function is obtained from

$$W = 1 + \frac{S}{\gamma}\tilde{W}, \quad (22)$$

where S is a positive constant.

The coarse gridpoint coordinates are then generated with a simple iterative solution to

$$\sum_{i=1}^{3}\sum_{j=1}^{3} g^{ij}\frac{\partial}{\partial \xi^i}\frac{\partial \mathbf{x}}{\partial \xi^j} + \sum_{k=1}^{3} g^{kk}P_k\frac{\partial \mathbf{x}}{\partial \xi^k} = 0, \quad (23)$$

where

$$P_k = \frac{1}{W}\frac{\partial W}{\partial \xi^k} \quad (24)$$

and P_k and g^{ij} are calculated using second-order centered differences on the coarse grid. Orthogonality of the coordinate lines at the boundary is imposed by using symmetry conditions at the boundary. An adequate solution of (23) typically requires about $4k_c$ iterations, where k_c is the number of coarse grid points in the ξ^3 direction. The gridpoint coordinates of the full grid are then interpolated from the coarse grid using tricubic interpolation (bicubic for the two-dimensional model) (Press et al. 1985), again imposing orthogonality at the boundary. The final step in the grid adaption process is the calculation of gridpoint velocities that steer the coordinates of the current grid toward those of the newly computed grid. The gridpoint velocities are calculated as the differences between the gridpoint coordinates of the newly computed grid and the current grid, divided by the expected time increment to the next grid calculation. No smoothing of the gridpoint velocities is implemented. The grid is calculated only every N_g time steps.

Our CDGA approach will be shown in the next section to have two salient characteristics. The first is computational efficiency. The second is that the resultant grid is smooth and hence minimizes a source of discretization error. This smooth grid also promotes efficiency in that it allows the grid to be recalculated less frequently than a more elaborate grid that places the high resolution only where it is immediately needed.

4. Application to a thermal

The model is applied to a rising thermal in a neutrally stratified environment. Both two-dimensional and three-dimensional simulations are presented. The model domain is $0 \leq x \leq 0.5$, $0 \leq y \leq 0.5$, and $0 \leq z \leq 1.0$ in three dimensions, and $0 \leq x \leq 0.5$ and $0 \leq z \leq 1.0$ in two dimensions. At the boundaries $z = 0$ and $z = 1$, we impose $b = 0$. The buoyancy is assumed symmetric, or even, about the x and y boundaries; hence, only one-quarter (one-half in two dimensions) of the thermal is explicitly simulated. Free-slip conditions on the velocity are enforced on all the boundaries. In all integrations we use $c = 1$, $N_f = 8$, $S = 7$, $\gamma = 0.3$, and $N_g = 20$.

a. *Two-dimensional results*

For the two-dimensional integrations we use $\nu = 5 \times 10^{-5}$, $\kappa = 5 \times 10^{-5}$. The thermal is initiated at $t = 0$ with a disk of buoyancy of radius 0.1:

$$b(x, z, t = 0) = \begin{cases} \cos\{5\pi[x^2 + (z - 0.15)^2]^{1/2}\}, \\ \quad [x^2 + (z - 0.15)^2]^{1/2} < 0.1 \\ 0, \quad [x^2 + (z - 0.15)^2]^{1/2} \geq 0.1. \end{cases} \tag{25}$$

The full grid is composed of 49×97 u nodes and the coarse grid is composed of 13×25 grid points. An important model parameter is

$$N_c \equiv \frac{k_n - 1}{k_c - 1}. \tag{26}$$

where k_n is the number of u nodes in the vertical direction and k_c is the number of grid points on the coarse grid in the vertical direction. In this case $N_c = 4$, which is also the situation depicted in Fig. 1. Figure 2 shows the buoyancy and grid at the initial and final times, $t = 0$ and $t = 4$.

Our method of constructing a weight function successfully concentrates grid points in the vicinity of significant b. Although a substantial velocity field (not shown) exists outside the high-resolution grid, sacrificing resolution in this part of the velocity field did not cause significant errors in the evolution of the buoyancy field. The effectiveness of CDGA in improving the modeled buoyancy is demonstrated by comparing the CDGA integration with integrations from a uniform, Cartesian, fixed-grid model with 49×97, 97×193, and 145×289 u nodes (Fig. 3). By simply inspecting qualitative features of the thermal, we conclude that the CDGA integration with 49×97 u nodes is far superior to that of the fixed-grid integration with the same number of grid points and is comparable to the 97×193 fixed-grid solution, which has four times the number of grid points. The 97×193 fixed-grid solution is in turn nearly identical to the 145×289 fixed-grid integration, which leads us to conclude that the 49×97 CDGA and 97×193 fixed-grid integrations have, for most practical purposes, converged to the "exact" solution.

Figure 4 shows results from the CDGA model as in Fig. 3, except with different values of N_c. Notice that

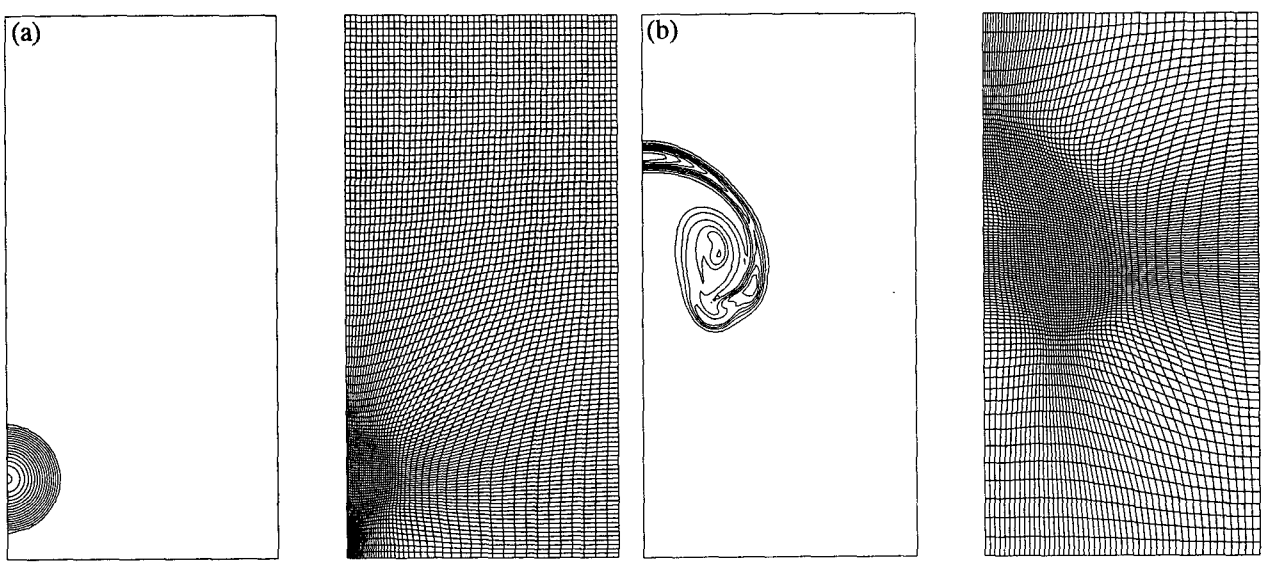

FIG. 2. Buoyancy b and u nodes of the full grid at (a) $t = 0$ and (b) $t = 4$ in the two-dimensional 49×97 CDGA model. The contour interval is 0.06. The maximum buoyancy values are 1.00 and 0.554 at $t = 0$ and $t = 4$, respectively. All the major extrema are maxima.

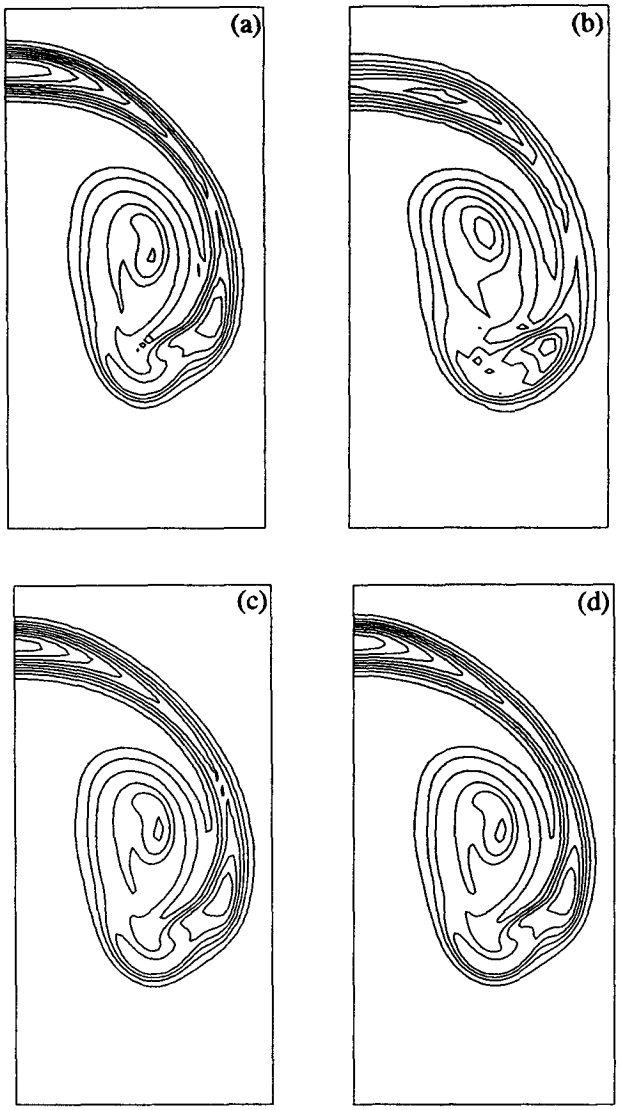

FIG. 3. Buoyancy b at $t = 4$ for (a) two-dimensional CDGA model solution of Fig. 2a and uniform, Cartesian, fixed-grid models with (b) 49 × 97, (c) 97 × 193, and (d) 145 × 289 grid points. The contour interval is 0.06. The maximum buoyancy values in (a), (b), (c), and (d) are 0.554, 0.418, 0.542, and 0.552, respectively. The portion of the domain shown is $0 \leq x \leq 0.25$ and $0.3 \leq z \leq 0.8$.

the high-resolution area of the grid in both the $N_c = 1$ case and $N_c = 2$ case (Figs. 4a and 4b) conform quite well to the significant buoyancy of the thermal. The weight function has preserved much of the detail in the buoyancy field, and this detail has been passed on to the coordinates of the grid. Although aesthetically pleasing, the grids in these cases led to less accurate solutions when compared with the exact solution of Fig. 3d. Perhaps too much weighting was applied to the maxima in the buoyancy field so that a significant portion of the buoyancy field may have, at certain times, fallen outside the high-resolution area. This fault possibly could be remedied by putting more effort into optimizing N_g, N_f, S, and γ for use with different values of N_c; here, we left these parameters the same in all integrations. The integrations in which $N_c = 6$ and $N_c = 8$ (Figs. 4c and 4d) also result in a degraded solution compared with that for $N_c = 4$. In these cases, the lack of detail in the weight function leads to an overly diffuse, high-resolution area of the grid. The thermal does indeed consistently fall within this area, but the associated grid spacing is larger than with $N_c = 4$.

The integrations with smaller values of N_c may also suffer greater discretization error arising from the second-order numerical approximations of the transformation relations. Consider a normalized difference in the base vectors calculated with the centered second-order and centered fourth-order approximations:

$$\frac{|\mathbf{a}^i(4\text{th}) - \mathbf{a}^i(2\text{d})|}{|\mathbf{a}^i(2\text{d})|}. \tag{27}$$

Figure 5 shows the maximum of this difference in the domain at $t = 4$ for the different values of N_c. This difference becomes greater as the resolution of the coarse grid approaches that of the full grid, or $N_c \to 1$. Therefore, as the grid becomes less smooth, second-order approximations to the transformation relations may become inadequate, necessitating the use of higher-order numerical approximations, which would contribute to a greater computational cost.

Besides the quality of the solution, the efficiency of the CDGA model also changes significantly with the resolution of the coarse grid. Using the performance analysis tool PERFTRACE, we find that for the $N_c = 4$ integration described earlier, 12.4% of the total time is used for the combined tasks of processing the weight function (0.3%), iteratively solving for the gridpoint coordinates (9.7%), and interpolating the coordinates from the coarse grid to the full grid (2.4%). Figure 6 shows that this percentage cost decreases significantly as the number of points in the coarse grid decreases.

The data presented in Figs. 4–6 argue against generating the gridpoint coordinates at the same resolution as that used to integrate the dynamical equations. The main reason for using a coarse grid for the grid adaption is the tremendous savings in CPU time implied by Fig. 6. This argument is buttressed by the fact that solution accuracy here is apparently superior with $N_c \geq 4$ grids. However, it is probably possible to improve the accuracy of the cases with $N_c = 1$ or $N_c = 2$ by filtering the weight function more, thus making the grid smoother. Nevertheless, it seems unlikely that the resulting accuracy would exceed that that results from the natural smoothness of the $N_c = 4$ grid.

b. *Three-dimensional results*

Having optimized our grid adaption scheme with two-dimensional integrations, we now turn to three-dimensional integrations. We do so primarily to assess the cost of CDGA in greater detail rather than to further improve the CDGA scheme. Due to constraints on

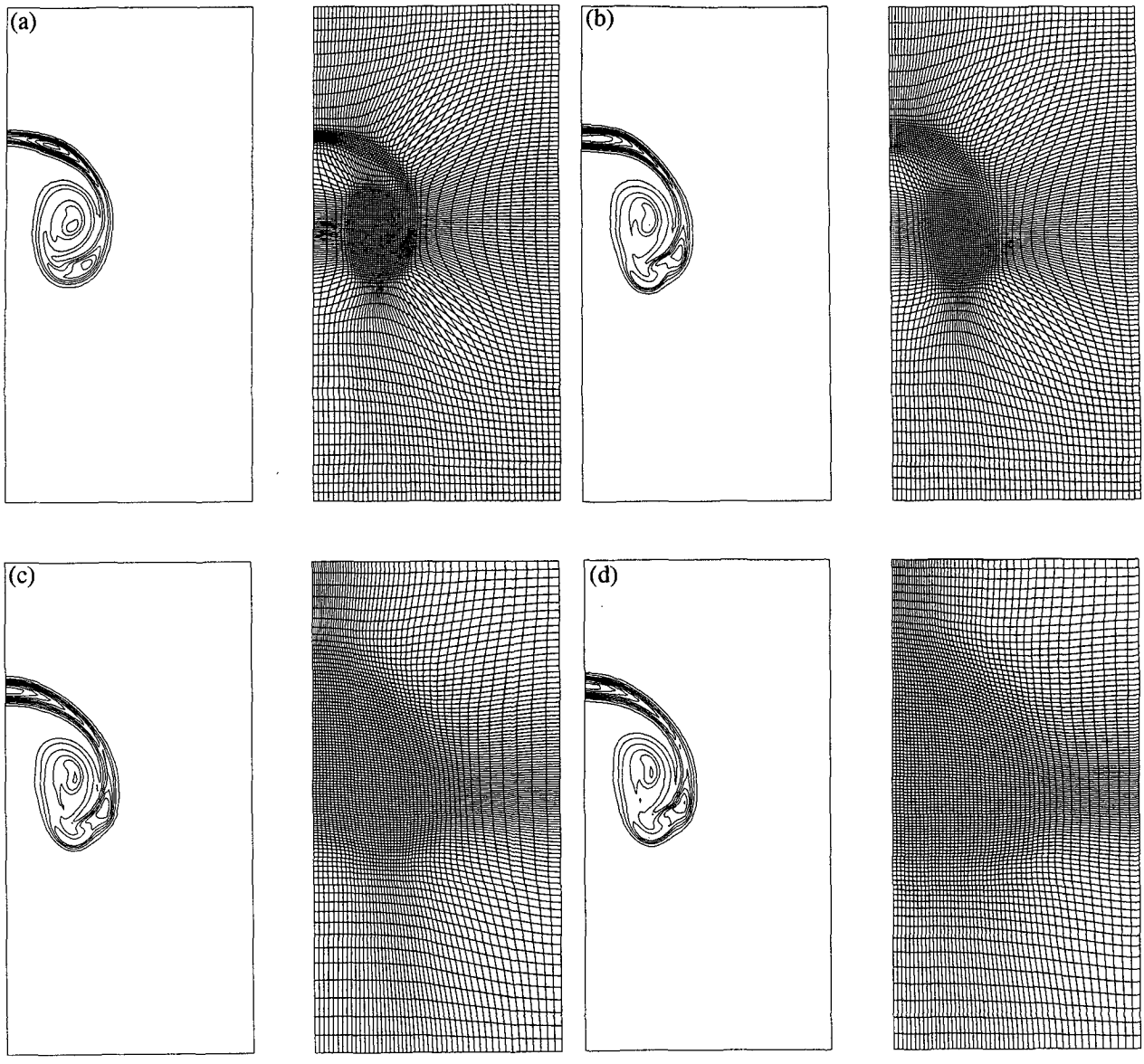

FIG. 4. Buoyancy b and u nodes of the full grid defined with (a) $N_c = 1$, (b) $N_c = 2$, (c) $N_c = 6$, and (d) $N_c = 8$ in two-dimensional 49 $\times$ 97 CDGA integrations at $t = 4$. The contour interval is 0.06. The maximum buoyancy values in (a), (b), (c), and (d) are 0.478, 0.562, 0.561, and 0.547. All the major extrema are maxima.

computational resources, we use less resolution than in the two-dimensional model, and use greater diffusion: $\nu = 10^{-4}$, $\kappa = 10^{-4}$. A three-dimensional thermal is initialized with an ellipsoid of buoyancy:

$$b(\mathbf{x}, t = 0) = \cos\left\{\frac{\pi}{2}\left[\frac{x^2}{0.2^2} + \frac{y^2}{0.1^2} + \frac{(z - 0.15)^2}{0.1^2}\right]^{1/2}\right\}, \quad (28)$$

when the argument of the cosine is less than $\pi/2$ and $b = 0$ otherwise. The CDGA integration with a full grid of $33 \times 33 \times 65$ u nodes and a coarse grid defined by $N_c = 4$ is shown in Fig. 7 at $t = 3$. In the three cross sections, the high-resolution region of the grid extends just outside the periphery of the significant buoyancy. Analogous to the foregoing two-dimensional cases, the weight function was successful in drawing grid points from all three directions toward the buoyancy.

The CDGA model has been compared with fixed-grid models. As seen in Fig. 8, the $33 \times 33 \times 65$ CDGA integration apparently agrees quite well with a $65 \times 65 \times 129$ uniform, Cartesian, fixed-grid integration. As could be expected, the $33 \times 33 \times 65$ CDGA integration is obviously superior to the $33 \times 33 \times 65$ fixed-grid integration.

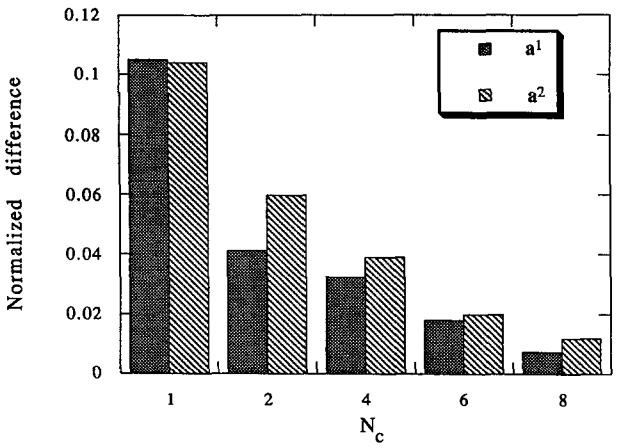

FIG. 5. Plot of the maximum value of Eq. (27) in the domain at $t = 4$ for various values of N_c.

Using PERFTRACE we can again obtain some of the costs associated with CDGA. The three-dimensional CDGA model updates approximately 820 000 gridpoint variables per second running at 127.6 Mflops on a single processor on the NCAR Cray-YMP, or 155.6 flops per variable per time step. In the CDGA model, the base vectors and Jacobian are calculated every iteration at each grid point on both the u and p nodes, consuming 18.5% of the total CPU time. The cost of processing the weight function (0.1%), generating the gridpoint coordinates (2.3%), and interpolating the coordinates from the coarse grid to the full grid (3.7%) amounts to 6.1% of the total. This grid calculation process is carried out every $N_g = 20$ iterations. Although the transformation of the diffusion term is in general quite complicated, the computational cost is comparatively small since the term need be evaluated only once about every 20–25 iterations; hence, diffusion requires only a few percent of the total time, 2.2% of which is devoted to the determination of (A14). Summing the *direct* costs involved in calculating the transformation relations and moving the grid, we find this amount to be 26.8% of the total.

Further *indirect* costs incurred with the application of CDGA in three dimensions are determined by comparison with a fixed-grid, Cartesian coordinate model and a fixed-grid, generalized curvilinear coordinate model. The fixed-grid, Cartesian coordinate model, which is simply a version of the CDGA model less the coordinate transformation and grid adaption routines, updates around 1 420 000 gridpoint variables per second running at 103.8 Mflops or 73.1 flops per gridpoint variable per time step. A fixed-grid, generalized curvilinear coordinate model (another version of the CDGA model less the grid adaption routines, in which the transformation relations are computed only once) updates around 1 175 000 gridpoint variables per second, runs at 107.4 Mflops, and so requires 91.3 flops per variable per time step. Thus, the overhead cost for the increased complexity of expressing the dynamical equations in generalized curvilinear coordinates is $(91.3 - 73.1)/155.6 = 11.7\%$ of the total CPU time, and the overhead cost for the calculations needed to generate and move the grid is $(155.6 - 91.3)/155.6 = 41.3\%$ of the total. When this last number is compared with the 26.8% that was accounted for using PERFTRACE, there remains 14.5% of the costs still unaccounted for. Some of this difference can be ascribed to calculations that involved the gridpoint velocity and that were not isolated by PERFTRACE. However, we feel that some of this unaccounted cost may result from subtleties in the compilation or something else yet to be explained. So, by directly comparing the CDGA model with the Cartesian, fixed-grid model, the total cost of CDGA is apparently 53% of the CDGA model.

For reference, we also include here some performance statistics for the two-dimensional experiments. In two dimensions, the fixed-grid Cartesian coordinate, fixed-grid curvilinear coordinate, and CDGA models require 55.5, 58.4, and 94.4 flops per variable per time step, respectively. Therefore, upon implementing CDGA in two dimensions, we incurred an overhead cost of 3.1% of the total CPU time for the increased complexity of expressing the dynamical equations in generalized curvilinear coordinates, and an overhead cost of 38.1% of the total for the calculations associated with moving the grid.

5. Summary

The demonstrations here indicate that three-dimensional CDGA is a viable technique for modeling moving, localized entities. In our application to a rising thermal, the cost of using CDGA, as measured in CPU

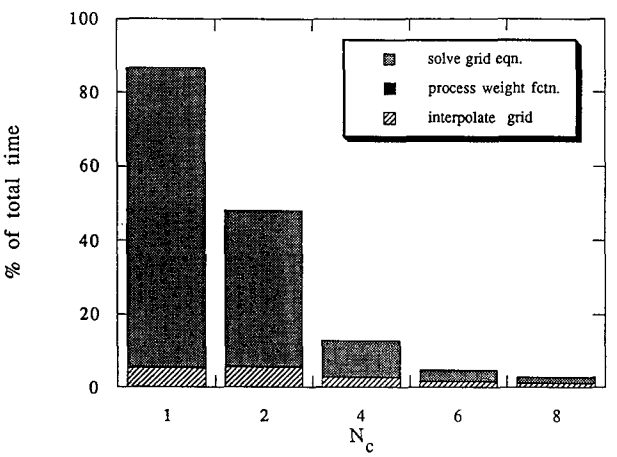

FIG. 6. Graph of the computational cost as a percent of the total cost of (i) processing the weight function, (ii) solving for the gridpoint coordinates, and (iii) interpolating the coordinates from the coarse grid to the full grid, for grids generated by the two-dimensional CDGA model and defined by different values of N_c.

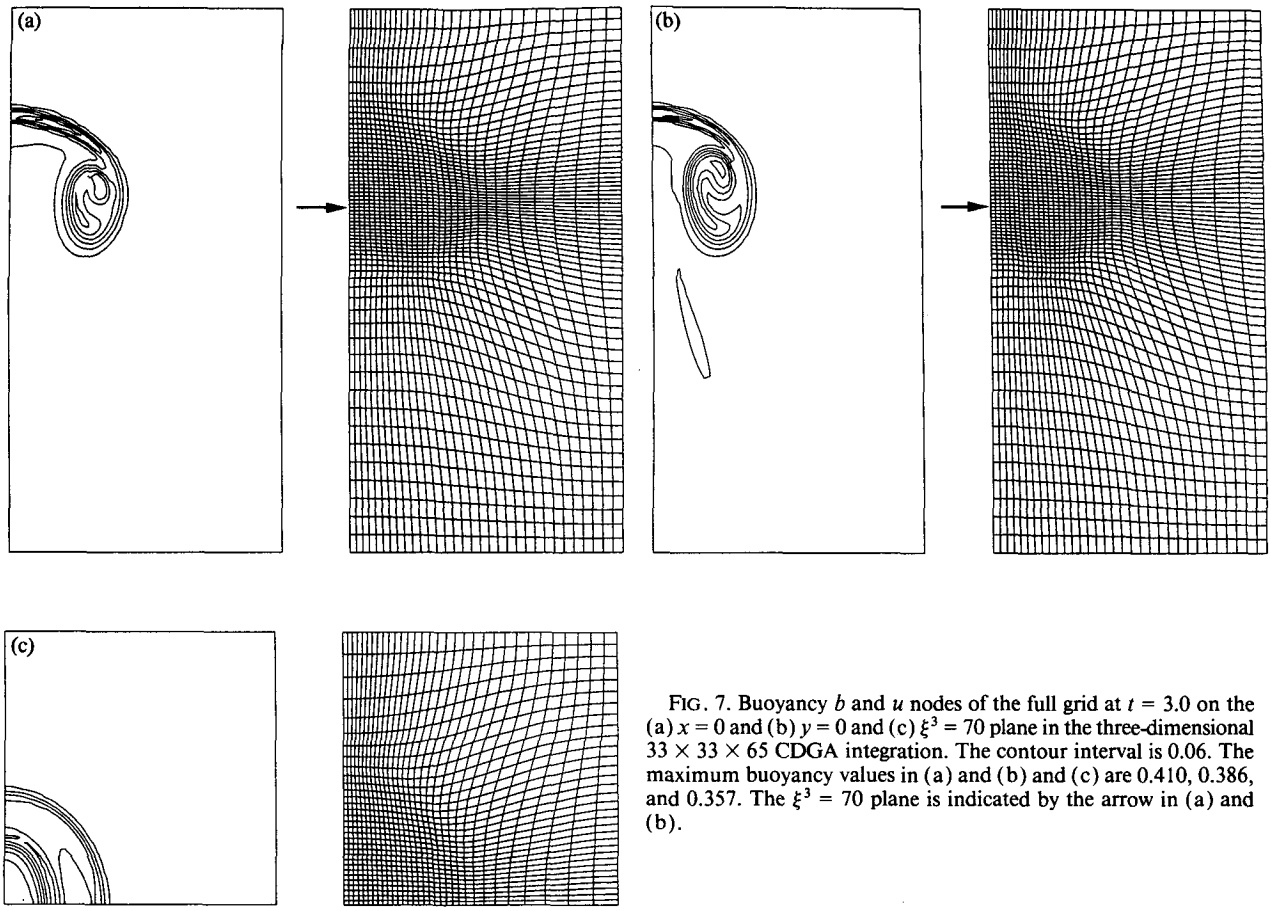

FIG. 7. Buoyancy b and u nodes of the full grid at $t = 3.0$ on the (a) $x = 0$ and (b) $y = 0$ and (c) $\xi^3 = 70$ plane in the three-dimensional $33 \times 33 \times 65$ CDGA integration. The contour interval is 0.06. The maximum buoyancy values in (a) and (b) and (c) are 0.410, 0.386, and 0.357. The $\xi^3 = 70$ plane is indicated by the arrow in (a) and (b).

time, *per gridpoint variable* per time step, is about half the total cost of the CDGA model. Thus, in a comparison between a CDGA model and a model employing a uniform, Cartesian fixed-grid in which both have the same minimum grid size and same time step, we would expect to save CPU time with the use of the CDGA model when it reduces the number of grid points by a factor of at least 2 from that in the fixed-grid model. The model for Fig. 8a had about one-eighth the number of grid points as that for Fig. 8c, and the total CPU time for the integration was reduced by a factor of about 3, yet nearly equivalent results were obtained. (We would have expected to reduce the cost by a factor of 4 had the time step been the same, but here the time step was slightly shorter in the CDGA simulation.) Furthermore, the CDGA model was able to reduce the required maximum memory by a factor of 6.4.

Without the ability to add grid points, CDGA will not be as versatile as other popular grid adaption techniques that use nested orthogonal grids (Skamarock 1989; Kurihara et al. 1979). For instance, consider a line of thunderstorms stretching across a model domain. CDGA would not offer much improvement in the resolution of the structure along the line, though it could be successful in resolving the structure across the line. Enhancing resolution along the line would require adding more grid points, not just shifting their position. However, if one were to include the ability to add grid points as needed in a CDGA codes, then one would obviate the need for working with nonorthogonal, stretched, moving grids. It would be simpler and more efficient to keep the grid uniform and orthogonal in various levels of nesting. Nested models also naturally allow for a discrete change in time step between nesting levels—CDGA allows for no such easy implementation. One should not become mesmerized by the smoothly varying grids and conclude that they necessarily offer a superior way to model smoothly varying fields. However, a general curvilinear coordinate system offers a clear advantage over a Cartesian one for solving partial differential equations in arbitrarily shaped regions where one desires a boundary-conforming coordinate system, as in Sharman et al. (1988). CDGA will combine nicely with such a coordinate system and adds little additional complexity to the code. Thus, nested models and CDGA both offer distinct advantages depending on the application.

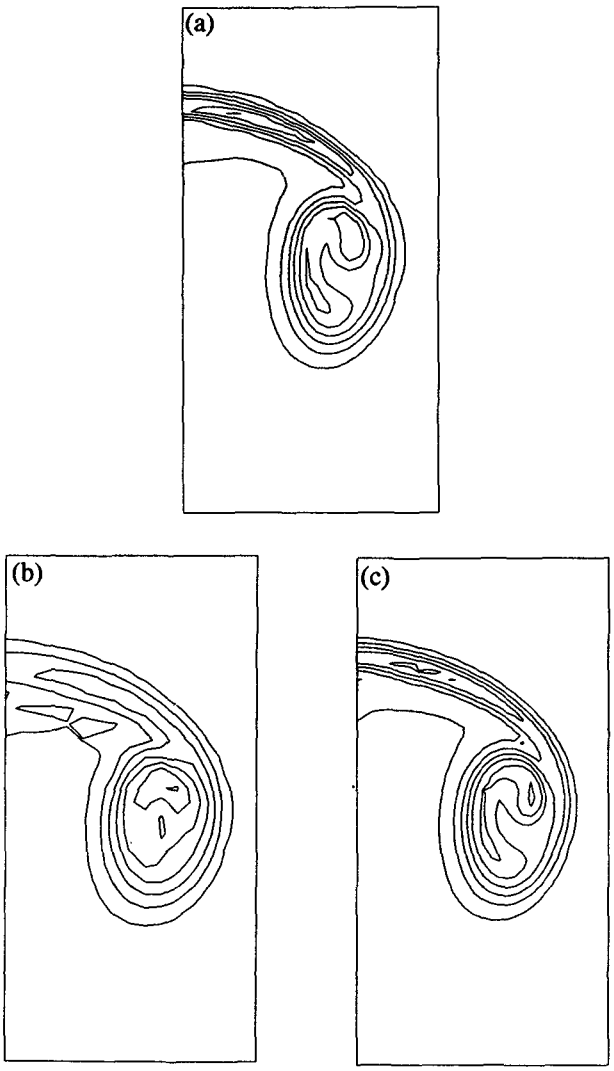

FIG. 8. Buoyancy b at $t = 3$ and $x = 0$ in the three-dimensional (a) $33 \times 33 \times 65$ CDGA integration and uniform, Cartesian, fixed-grid integrations with (b) $33 \times 33 \times 65$ and (c) $65 \times 65 \times 129$ u nodes. The contour interval is 0.06. The maximum buoyancy values in (a), (b), and (c) are 0.386, 0.396, and 0.404, respectively. The portion of the domain shown is $0 \leq x \leq 0.25$ and $0.4 \leq y \leq 0.9$. All the major extrema are maxima.

Acknowledgments. This research was supported by NSF Grant ATM-9002391. Computations were performed at the National Center for Atmospheric Research, which is sponsored by the National Science Foundation.

APPENDIX

The Transformation Relations

The physical coordinates are

$$\mathbf{x} = \sum_{i=1}^{3} x_i \hat{\mathbf{e}}_i, \tag{A1}$$

and the velocity is

$$\mathbf{u} = \sum_{i=1}^{3} u_i \hat{\mathbf{e}}_i. \tag{A2}$$

Finite-difference equations are developed on a grid that is uniform and orthogonal in logical coordinates ξ^i ($i = 1, 2, 3$). The equations transformed to logical coordinates make use of the three covariant base vectors

$$\mathbf{a}_i = \frac{\partial \mathbf{x}}{\partial \xi^i}, (i = 1, 2, 3), \tag{A3}$$

the Jacobian of the transformation

$$g^{1/2} = \mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3), \tag{A4}$$

and the three contravariant base vectors

$$\mathbf{a}^i = \frac{1}{g^{1/2}} \mathbf{a}_j \times \mathbf{a}_k, (i, j, k \text{ cyclic}). \tag{A5}$$

It is also convenient to define a contravariant velocity

$$U^i = \mathbf{a}^i \cdot (\mathbf{u} - \dot{\mathbf{x}}), \tag{A6}$$

where $\dot{\mathbf{x}}$ is the physical velocity of a grid point with fixed coordinates ξ^i.

The divergence theorem applied to a differential volume element bounded by the coordinate surfaces gives the *conservative forms* of the gradient, divergence, and Laplacian operators:

$$\nabla A = \frac{1}{g^{1/2}} \sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (g^{1/2} \mathbf{a}^i A), \tag{A7}$$

$$\nabla \cdot \mathbf{A} = \frac{1}{g^{1/2}} \sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (g^{1/2} \mathbf{a}^i \cdot \mathbf{A}) \tag{A8}$$

and

$$\nabla^2 A = \frac{1}{g^{1/2}} \sum_{i=1}^{3} \sum_{j=1}^{3} \frac{\partial}{\partial \xi^i} \left[\mathbf{a}^i \cdot \frac{\partial}{\partial \xi^j} (g^{1/2} \mathbf{a}^j A) \right]. \tag{A9}$$

Now, applying the fundamental metric identity

$$\sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (g^{1/2} \mathbf{a}^i) = 0 \tag{A10}$$

to (11)–(13) yields their *nonconservative* forms

$$\nabla A = \sum_{i=1}^{3} \mathbf{a}^i \frac{\partial A}{\partial \xi^i}, \tag{A11}$$

$$\nabla \cdot \mathbf{A} = \sum_{i=1}^{3} \mathbf{a}^i \cdot \frac{\partial \mathbf{A}}{\partial \xi^i}, \tag{A12}$$

and

$$\nabla^2 A = \sum_{i=1}^{3} \sum_{j=1}^{3} g^{ij} \frac{\partial}{\partial \xi^i} \left(\frac{\partial A}{\partial \xi^j} \right) + \sum_{j=1}^{3} (\nabla^2 \xi^j) \frac{\partial A}{\partial \xi^j}. \tag{A13}$$

In (A13) we use

$$\nabla^2\xi^l = -\sum_{i=1}^{3}\sum_{j=1}^{3} g^{ij}\mathbf{a}^l\cdot\frac{\partial}{\partial\xi^i}\left(\frac{\partial\mathbf{x}}{\partial\xi^j}\right)(l = 1, 2, 3), \quad \text{(A14)}$$

where the contravariant metric tensor

$$g^{il} = \frac{1}{g}(g_{jm}g_{kn} - g_{jn}g_{km})(i = 1, 2, 3;\ l = 1, 2, 3;$$

$$i, j, k \text{ cyclic};\ l, m, n \text{ cyclic}), \quad \text{(A15)}$$

is expressed in terms of the covariant metric tensor

$$g_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j \quad \text{(A16)}$$

and

$$g = \det|g_{ij}|. \quad \text{(A17)}$$

The covariant base vectors (A3) are computed with centered, second-order differences at each time step, on both the u nodes and p nodes. The grid is orthogonal at the boundaries, and symmetry conditions are used in the calculation of (A3) at the boundaries. The subsequent transformation relations (A4) and (A5) are also calculated every time step. Equations (A13) and (A14) are approximated with centered second-order differences on the u nodes and are evaluated only at the times when the diffusion term is calculated, which occurs about every 20 time steps.

REFERENCES

Dietachmayer, G. S., 1992: On the applications of continuous dynamic grid adaption techniques to meteorological modeling. Part II: Efficiency. *Mon. Wea. Rev.*, **120,** 1707–1722.

——, and K. K. Droegemeier, 1992: Application of continuous dynamic grid adaption techniques to meteorological modeling. Part I: Basic formulation and accuracy. *Mon. Wea. Rev.*, **120,** 1675–1706.

Durran, D. R., 1991: The third-order Adams–Bashforth method: An attractive alternative to leapfrog time differencing. *Mon. Wea. Rev.*, **119,** 702–720.

Kurihara, Y., G. J. Tripoli, and M. A. Bender, 1979: Design of a movable nested-mesh primitive equation model. *Mon. Wea. Rev.*, **107,** 239–249.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, 1986: *Numerical Recipes.* Cambridge University Press, 818 pp.

Sharman, R. D., T. L. Keller, and M. G. Wurtele, 1988: Incompressible and anelastic flow simulations on numerically generated grids. *Mon. Wea. Rev.*, **116,** 1124–1136.

Skamarock, W. C., J. Oliger, and R. L. Street, 1989: Adapative grid refinement for numerical weather prediction. *J. Comput. Phys.*, **80,** 27–60.

Thompson, J. F., Z. U. A. Warsi, and C. W. Mastin, 1985: *Numerical Grid Generation Foundations and Applications.* North-Holland, 483 pp.