JOURNAL OF COMPUTATIONAL PHYSICS **129,** 284–295 (1996)
ARTICLE NO. 0250

A Forward-in-Time Advection Scheme and Adaptive Multilevel Flow Solver for Nearly Incompressible Atmospheric Flow

David E. Stevens* and S. Bretherton†

Center for Computational Sciences and Engineering, Lawrence Berkeley National Laboratory, Berkeley, California 94720; and †Departments of Applied Mathematics and Atmospheric Sciences, University of Washington, Seattle, Washington 98195

Received April 17, 1995; revised March 14, 1996

This paper presents a new forward-in-time advection method for nearly incompressible flow, MU, and its application to an adaptive multilevel flow solver for atmospheric flows. MU is a modification of Leonard *et al.'s* UTOPIA scheme. MU, like UTOPIA, is based on third-order accurate semi-Lagrangian multidimensional upwinding for constant velocity flows. For varying velocity fields, MU is a second-order conservative method. MU has greater stability and accuracy than UTOPIA and naturally decomposes into a monotone low-order method and a higher-order accurate correction for use with flux limiting. Its stability and accuracy make it a computationally efficient alternative to current finite-difference advection methods. We present a fully second-order accurate flow solver for the anelastic equations, a prototypical low Mach number flow. The flow solver is based on MU which is used for both momentum and scalar transport equations. This flow solver can also be implemented with any forward-in-time advection scheme. The multilevel flow solver conserves discrete global integrals of advected quantities and includes adaptive mesh refinement. Its second-order accuracy is verified using a nonlinear energy conservation integral for the anelastic equations. For a typical geophysical problem in which the flow is most rapidly varying in a small part of the domain, the multilevel flow solver achieves global accuracy comparable to a uniform-resolution simulation for 10% of the computational cost. © 1996 Academic Press, Inc.

1. INTRODUCTION

Many geophysical flows, such as boundary layer turbulence, convection, and topographically forced flows, are nearly incompressible and nonhydrostatic. The simulation of such flows requires the accurate solution of scalar advection equations

$$\frac{\partial \psi}{\partial t} + \frac{1}{\rho} \nabla \cdot (\rho \mathbf{U} \psi) = R, \tag{1}$$

with source term R and velocity field $\mathbf{U}$ that obey a continuity equation of the form

$$\nabla \cdot (\rho \mathbf{U}) = 0, \tag{2}$$

where ρ is some approximation to the fluid density. In the Boussinesq approximation ρ is constant, while in the anelastic approximation ρ is a known function of height z. For many flows, a "multilevel" scheme that incorporates regions of higher grid refinement in time and space can be advantageous to track localized fine-scale flow features.

Advection schemes should ideally be efficient, highly accurate, preserve in discrete form the conservation laws governing the advected quantities, and be "monotone" (not introduce spurious overshoots and undershoots). Different types of advection schemes have evolved in response to these requirements. Centered "leapfrog" differencing in space and time, long popular for atmospheric applications, is simple and second-order accurate, but it is prone to large numerical overshoots and must be time-filtered for stability. Semi-Lagrangian methods [30, 32] have also been proven attractive for scalar advection, primarily due to their stability and accuracy properties. However, they are not conservative. "Forward-in-time" methods [13, 16, 17, 24, 29] which use only the most recent time level to advance to a new time level are more complex than leapfrog schemes (especially for solving the momentum equations subject to mass conservation), but they are more accurate and stable. These schemes are increasingly being used, both for complete flow solvers and hybrid solvers, where a leapfrog method is used for the momentum equations and a forward-in-time scheme is used for advecting scalars [4, 27].

The first goal of this paper is to present a more accurate and efficient forward-in-time advection scheme. Its superior accuracy and efficiency are due to improved third-order upwinding that is semi-Lagrangian for constant velocity flows. It is based on finding the fluxes through faces of a control volume that would be associated with the back-trajectories of all points within that control volume. The resulting algorithm is then generalized to a variable velocity flow by writing it in Eulerian flux form. The flux form ensures discrete conservation of advected scalars and can be flux corrected [10, 35] to maintain monotonicity. Several advection schemes using similar approaches have recently been proposed. Collela [13] developed a two-dimensional algorithm, CTU (corner transport upwind), by integrating the characteristics passing through a two-dimensional face. This algorithm was later extended by Saltzman [24] to three dimensions. Leonard *et. al.* [16] derived a three-dimensional method, UTOPIA (uniform third-order polynomial interpolation algorithm) by decomposing into flux form a semi-Lagrangian method that has additional multidimensional or transverse terms to eliminate third-order errors. UTOPIA is third-order accurate for constant velocity flow, but it has a reduced stability region. LeVeque [17] proposed a similar algorithm, interpreting the multidimensional terms as correction waves propagating transversely to the grid. Our proposed method, "modified UTOPIA," or MU, is an extension of UTOPIA that uses additional transverse correction terms for improved accuracy while retaining the superior stability of CTU.

A particularly attractive application of forward-in-time methods is adaptive multilevel flow solvers, in which the finite difference grid is refined in selected regions that may be adaptively changed in time to follow evolving features. Multilevel models with fixed, nonadaptive grids have been commonly used in meteorological applications during the past 15 years and recently have become adaptive [28]. Most such models are based on compressible fluid equations. For troposphere-deep convection or gravity waves, for which the Mach number is 0.1–0.2, compressible models are easier to implement and comparably efficient and accurate to models which are based on the anelastic or Boussinesq approximation. However, an anelastic model is much more efficient than a compressible model for modelling very low Mach number (0.01) flows such as atmospheric boundary layer turbulence. Because anelastic models are somewhat more difficult to implement, their popularity in the atmospheric sciences has largely been restricted to simulating the atmospheric boundary layer [19, 20, 25]. Most of these models have not incorporated the added complexity of multilevel refinement. An important exception is the model of Clark and Farley [12], perhaps the most broadly applied atmospheric model based on the anelastic equations. To date, no *adaptive* multilevel anelastic model has been routinely used for simulating atmospheric flows, despite the existence of many flows for which this type of model is advantageous and the existence of at least one such solver [1] that has been developed for other applications.

Hence, the second goal of this paper is to present a new anelastic flow solver for nearly incompressible fluid flow that incorporates adaptive multilevel refinement. It is patterned on [1], but it has different grid staggering and a simpler algorithm for maintaining mass continuity. The boundary conditions at the edge of refined grids and the algorithms for transferring fields between grids follow Clark and Farley [12]. Our solver is based around MU but can be used with any forward-in-time scheme.

This paper first derives MU and discusses its performance, followed by its extension to a multilevel flow solver. In Section 2, we present our forward-in-time advection method and analyze its monotonicity, accuracy, and stability properties. Comparisons of this method with other methods are also made. The advection method is generalized to a flexible flow solver for the anelastic equations and tested in Section 3. An adaptive multilevel flow solver and applications are shown in Section 4.

2. ADVECTION METHOD

Advection can be modelled in an Eulerian manner by determining the tendency of a gridpoint as the convergence of normalized fluxes through cell faces

$$\psi_p^{n+1} = \psi_p - \frac{1}{\rho_p}(F_e - F_w + F_n - F_s + F_t - F_b), \qquad (3)$$

where superscripts are dropped for values at t^n and the subscript letters indicate position in the compass point notation of the Appendix. This formulation ensures that, except for boundary fluxes, the integral of ψ over an arbitrary region is conserved. The normalized fluxes for this algorithm are computed by integrating the characteristics that impinge on a cell face over a timestep and dividing by the cell volume. For flow with constant velocity (u, v, w) and density ρ, the flux for the eastern cell face normal to the positive x direction is expressed as

$$\begin{aligned} F_e &= \frac{u}{\Delta x\,\Delta y\,\Delta z}\int_0^{\Delta t}\int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}}\int_{-\frac{\Delta z}{2}}^{\frac{\Delta z}{2}} \psi\left(\frac{\Delta x}{2}, y, z, t^n + \tau\right) dy\,dz\,d\tau \\ &= \frac{u}{\Delta x\,\Delta y\,\Delta z}\int_0^{\Delta t}\int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}}\int_{-\frac{\Delta z}{2}}^{\frac{\Delta z}{2}} \qquad (4) \\ &\quad \times \psi\left(\frac{\Delta x}{2} - u\tau, y - v\tau, z - w\tau, t^n\right) dy\,dz\,d\tau. \end{aligned}$$

We have expressed the flux as an integral that only involves ψ at time n. This approach is equivalent to that of Saltzman [24] and similar to that of LeVeque [17].

In our algorithm, ψ is approximated as the sum of a low-order representation ψ^{lo} and a correction term ψ^{c}. If used alone ψ^{lo} would give a monotonic, highly stable, but only first-order accurate advection scheme. Thus, flux-limiting, if desired, can be performed on ψ^{c}. The flux is correspondingly decomposed as

$$F_e = F_e^{\mathrm{lo}} + F_e^{\mathrm{c}}. \qquad (5)$$

Our low-order representation is a bilinear interpolation between upwind cell values in the transverse directions and is constant in the normal direction. For the case u, v, $w > 0$, this choice of upwinding yields

$$\psi_{\mathrm{lo}}(\mathbf{x}) = \psi_p + y\,\frac{\psi_p - \psi_s}{\Delta y} + z\,\frac{\psi_p - \psi_b}{\Delta z} + yz\,\frac{\psi_p - \psi_{sb} - \psi_s - \psi_b}{\Delta y\,\Delta z}. \tag{6}$$

When integrated in (4), this representation yields the flux

$$F_e^{\mathrm{lo}} = \nu_x \nu_p - \frac{\nu_x\nu_y}{2}(\psi_p - \psi_s) - \frac{\nu_x\nu_z}{2}(\psi_p - \psi_b) + \frac{\nu_x\nu_y\nu_z}{3}(\psi_p + \psi_{sb} - \psi_s - \psi_b), \tag{7}$$

where $(\nu_{\mathrm{x}}, \nu_{\mathrm{y}}, \nu_{\mathrm{z}})$ is the vector of Courant numbers $(u\Delta t/\Delta x,\ v\Delta t/\Delta y,\ w\Delta t/\Delta z)$. We will show that this flux is monotonic even for spatially varying flows and has the optimal stability of a nearest neighbor algorithm.

This method can be made second-order accurate by including a normal slope correction to our representation of ψ as in other methods [13, 17, 24, 29],

$$\psi^{\mathrm{c}}(x, y, z) = x\,\frac{\psi_e - \psi_p}{\Delta x}. \tag{8}$$

The corresponding corrective flux is

$$F_e^{\mathrm{c}} = \frac{\nu_x}{2}(1 - \nu_x)(\psi_e - \psi_p). \tag{9}$$

This correction was found to be unstable for three-dimensional flow directions, but it can be stabilized and improved by adding higher order terms. Our strategy for improving the corrective flux was to use the natural second-order expansion of ψ, modified by a corrective factor,

$$\begin{aligned}\psi(x,y,z) = {}& \psi_p + x\frac{\psi_e - \psi_w}{2\Delta x} + y\frac{\psi_n - \psi_s}{2\Delta y} + z\frac{\psi_t - \psi_b}{2\Delta z} \\ & + \frac{x^2}{2}\frac{\psi_e - 2\psi_p - \psi_w}{\Delta x^2} + \frac{y^2}{2}\frac{\psi_n - 2\psi_p - \psi_s}{\Delta x^2} \\ & + \frac{z^2}{2}\frac{\psi_t - 2\psi_p - \psi_b}{\Delta x^2} + xy\,\frac{\psi_e + \psi_s - \psi_{se} - \psi_p}{\Delta x\,\Delta y} \\ & + xz\,\frac{\psi_e + \psi_b - \psi_{be} - \psi_p}{\Delta x\,\Delta z} + yz\,\frac{\psi_p + \psi_{sb} - \psi_s - \psi_b}{\Delta y\,\Delta z} \\ & - \frac{1}{24}(\psi_e + \psi_n + \psi_t - 6\psi_p + \psi_w + \psi_s + \psi_b).\end{aligned} \tag{10}$$

The corrective factor is needed, since the time tendency of a cell average ψ_{cell} is different from that of the cell center,

$$\psi_t = (\psi_{\mathrm{cell}})_t - \frac{\Delta x^2}{24}\psi_{xxt} - \frac{\Delta y^2}{24}\psi_{yyt} - \frac{\Delta z^2}{24}\psi_{zzt}. \tag{11}$$

When integrated for constant velocity flows, this representation yields a third-order accurate flux which can be shown to be algebraically identical to UTOPIA, by use of a discrete version of the identity $(\psi_{yy})_x = (\psi_{xy})_y$,

$$\begin{aligned}&(\psi_n - 2\psi_p + \psi_s) - (\psi_{nw} - 2\psi_w + \psi_{sw}) \\ &\quad = (\psi_n + \psi_w - \psi_p - \psi_{nw}) - (\psi_p + \psi_{sw} - \psi_w - \psi_s).\end{aligned} \tag{12}$$

which states that the effect of the ψ_{yy} representation in the x direction flux is equivalent to the effect of the ψ_{xy} representation in the y direction flux. One can considerably simplify the algebraic form of the flux and the multidimensional stability of the method by adding higher order correction terms to (10). We added the (xyz, x^2y, x^2z, and x^2yz) terms in the Taylor series of $\psi(x, y, z)$ and used the natural discretizations of the corresponding derivatives ψ_{xyz}, ψ_{xxy}, ψ_{xxz}, and ψ_{xxyz}. The difference between this flux and the low order flux is the corrective flux

$$F_e^c = \frac{\nu_x}{2}(1 - \nu_x)(\hat{\psi}_e - \hat{\psi}_p) - \frac{\nu_x}{6}(1 - \nu_x^2)(\hat{\psi}_e - 2\hat{\psi}_p + \hat{\psi}_w), \tag{13}$$

where the $\hat{\psi}$ quantities are given by bilinear interpolation in y and z,

$$\begin{aligned}\hat{\psi}_p = {}& \psi_p(1 - \nu_y)(\nu_z) + \psi_b(\nu_y)(\nu_z) \\ & + \psi_{sb}(1 - \nu_y)(1 - \nu_z) + \psi_s(\nu_y)(1 - \nu_z).\end{aligned} \tag{14}$$

This corrective flux is equivalent to bilinearly interpolating the correction needed to cancel the leading second-order error terms of Lax–Wendroff. It will be shown that this method has the same stability as the low order method. In the Appendix, a simple technique for computing the low order and corrective fluxes for variable density and velocity is presented.

For constant velocity flow, the low-order method (Fig.1) can be viewed as a semi-Lagrangian method in which the value of the scalar at its departure point $(\mathbf{x}_o, t^n)$ is computing by trilinear interpolation between neighboring gridpoint values. Trilinear interpolation ensures that $\psi(\mathbf{x}_o, t^n)$ lies between the maximum and minimum ψ of surrounding gridpoints, guaranteeing monotonicity. For nondivergent variable-velocity flow, the low-order method can be written

$$\begin{aligned}\psi_p^{n+1} = {}& a_1\psi_p + a_2\psi_s + a_3\psi_w + a_4\psi_b + a_5\psi_{sw} + a_6\psi_{wb} \\ & + a_7\psi_{sw} + a_8\psi_{swb},\end{aligned} \tag{15}$$

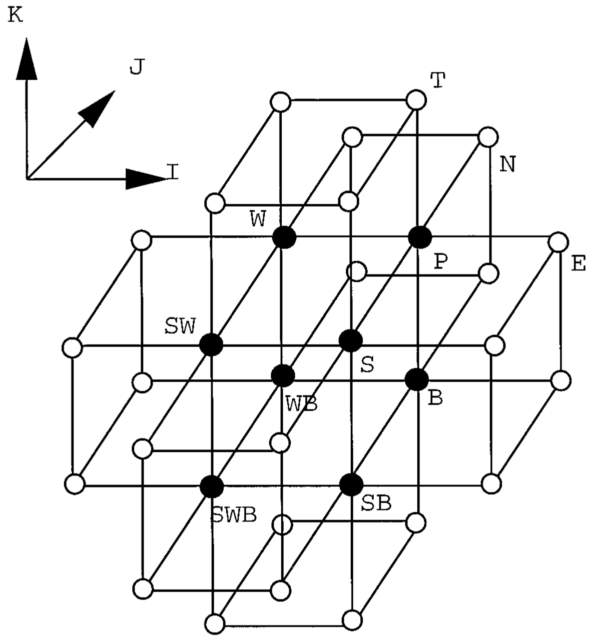

FIG. 1. The MU stencil for the case u, v, $w > 0$. The low order method uses trilinear interpolation about the eight labeled dark points which contain the departure point. The high order correction uses the additional points indicated by hollow circles.

where the weights a_i are positive. One can show for divergence-free velocities that

$$\sum_{k=1}^{8} a_k = 1 + \alpha h^2, \tag{16}$$

from which we deduce

$$|\psi_P^{n+1}| \le (1 + \alpha h^2)\max(|\psi_i|), \tag{17}$$

where α is identically zero for the constant velocity case and $h = \max(\Delta x, \Delta y, \Delta z)$. This inequality is the maximum principle observed by Saltzman [24] and implies that the variation of the solution remains bounded.

For constant velocity flow, the high-order method can be viewed as a semi-Lagrangian method with a multidimensional cubic interpolation stencil (Fig. 1) centered about $\mathbf{x}_o$. The cubic stencil consists of the eight dark points used in the low-order method plus the additional 24 hollow points surrounding them. This stencil is highly isotropic with respect to the center of the upstream cube. Correspondingly, MU is almost equally accurate for flow in an arbitrary direction. Furthermore, the cubic stencil has the same stability properties as the low-order method. By using cubic interpolation, the method has less dispersion than traditional second-order methods. Although this method is less dispersive, numerical undershoots and overshoots are still possible. This is corrected by using the Zalesak [29, 35] flux limiter. This flux limiter is a true multidimensional limiter that takes into account the total flux entering or leaving a cell. This is an improvement over applying one-dimensional flux limiters in the different normal directions. In a multidimensional flow, one-dimensional limiting cannot account for multiple fluxes acting in concert to cause numerical oscillations.

For advection at a constant velocity, the stability of this method can be analyzed using von Neumann analysis. This analysis measures the change in amplitude of a Fourier component

$$\begin{aligned}\psi^n_{\hat{\imath},\hat{\jmath},\hat{k}} &= \exp(i(\theta_x\hat{\imath} + \theta_y\hat{\jmath} + \theta_z\hat{k})),\\ i &= \sqrt{(-1)},\\ -\pi &\le (\theta_x, \theta_y, \theta_z) \le \pi,\end{aligned} \tag{18}$$

undergoing advection. The resulting field can be written in terms of a complex amplification factor $\hat{\psi}$ such that

$$\psi^{n+1}_{\hat{\imath},\hat{\jmath},\hat{k}} = \hat{\psi}(\nu_x, \nu_y, \nu_z, \theta_x, \theta_y, \theta_z)\exp(i(\theta_x\hat{\imath} + \theta_y\hat{\jmath} + \theta_z\hat{k})), \tag{19}$$

where $(\theta_x, \theta_y, \theta_z)$ are the wavenumbers allowed on the grid weighted by the grid spacing in each direction. The stability of any linear method for a given velocity can be found by looking at the maximum magnitude of $\hat{\psi}$ over all wavenumbers $(\theta_x, \theta_y, \theta_z)$,

$$S(\nu_x, \nu_y, \nu_z) = \max(|\hat{\psi}(\nu_x, \nu_y, \nu_z, \theta_x, \theta_y, \theta_z)|). \tag{20}$$

It is possible to derive a compact formula for the amplification factor of the low order method and thereby analyze its stability,

$$\hat{\psi} = (1 - \nu_x(1 - e^{-i\theta_x}))(1 - \nu_y(1 - e^{-i\theta_y}))(1 - \nu_z(1 - e^{-i\theta_z})). \tag{21}$$

Each of the terms in $\hat{\psi}$ are ≤ 1 in magnitude, unless ν_x, ν_y, or $\nu_z > 1$. This indicates that this method is stable for the cube, $0 \le \nu_x, \nu_y, \nu_z \le 1$. $\hat{\psi}$ has a magnitude that drops off steeply for nonzero wavenumbers. In general, it is not possible to derive such a useful analytical formula for S, even when $\hat{\psi}$ is known. However, knowing $\hat{\psi}$, an estimate of S can be found by numerically iterating over the complete range of wavenumbers $(\theta_x, \theta_y, \theta_z)$ for a given Courant number. The resulting S can then be plotted as a function of Courant number. A method is stable in the region bounded by the origin and the isosurface $S = 1$. The method with only the normal slope correction (Fig. 2b) is stable for all two-dimensional, but not all three-dimensional velocities inside the unit cube of Courant numbers. The stability region includes the two-dimensional $\nu_x - \nu_y$, $\nu_x - \nu_z$, and $\nu_y - \nu_z$ unit squares, but it excludes a large,

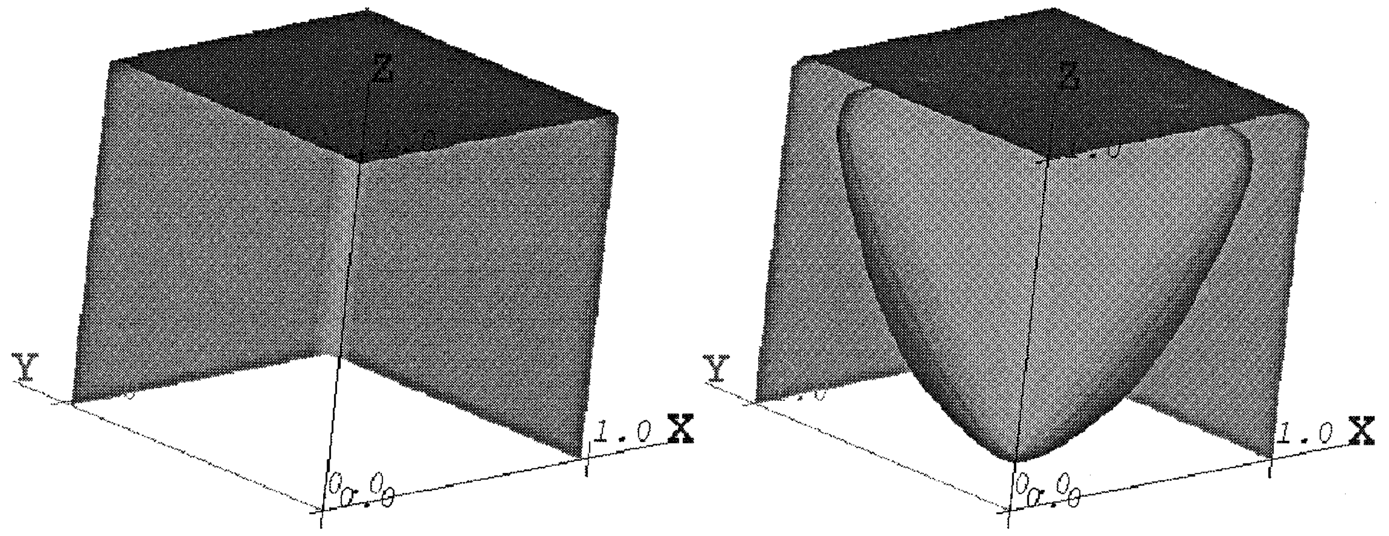

FIG. 2. The three-dimensional stability regions for the low order method with the third-order corrective flux (a) and with only the normal slope correction (b). The coordinate axes are Courant number in x, y, and z. The isosurfaces bound the region (including the origin) in which the method is stable. The low order method also has the stability region shown at the left.

roughly tetrahedral volume which includes very small Courant numbers in the (1, 1, 1) direction. The stability region for MU (Fig. 2a) is given by the cube, $0 \leq \nu_x, \nu_y, \nu_z \leq 1$. The stability region of the upwind, leapfrog, and UTOPIA methods is shown in Fig. 3 for comparison. These methods have smaller stability regions of the form $0 \leq \nu_x + \nu_y + \nu_z \leq 1$. This is a substantial advantage of our method for use in three-dimensional flow solvers.

We compared the accuracy of MU with existing methods by advecting both smooth and nonsmooth initial conditions in a spatially varying three-dimensional flow field. Our test flow was one full solid body rotation about a vector extending from the origin to the corner (1, 1, 1) of the unit cube with an angular velocity of $\Omega = 2\pi$. From this velocity field, we can compute analytical solutions for any choice of initial condition. The initial scalar concentrations were chosen to start and remain zero near the boundaries of the unit cube which bounded the numerical domain. Hence, the periodic boundary conditions applied to the scalar concentrations had negligible impact on the solution. The error after one rotation was measured using the 1-norm defined by

$$\|e\|_1 = \frac{1}{n^3} \sum_{i,j,k} |e_{i,j,k}|, \tag{22}$$

where $e_{i,j,k}$ was the error of the solution at gridpoint (i, j, k). The domain had n gridpoints in each direction and the sum was over all gridpoints in the domain. In our advection tests, we used a uniform spatial step $h = 1/n$ and timestep $\Delta t = h/12$ unless otherwise indicated. The maximum Courant number in all three coordinate directions, 0.26, was achieved at the vertices of the cube that were off the rotation axis. At these points, the timestep is 78% of the stability limit of all the methods considered, except for MU, for which the timestep is only 26% of the stability limit. Thus, we also tested the accuracy of MU when the timestep was tripled. Note that this test produces a particularly large disparity between the maximum stable timestep for MU and for the other methods, because of the flow direction and domain geometry. In general, the maximum

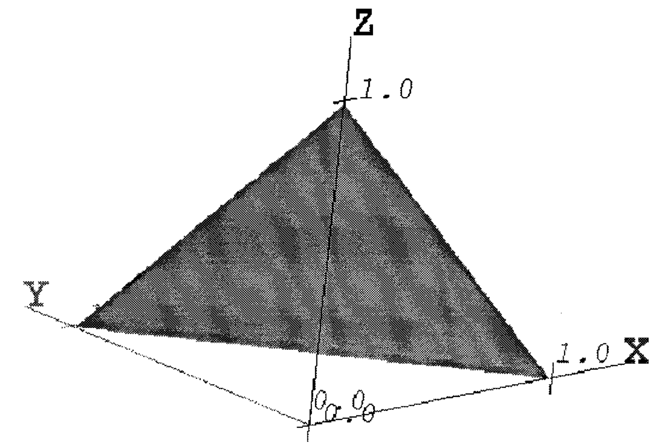

FIG. 3. The three-dimensional stability region for the MPDATA, UTOPIA, leapfrog, and upwind algorithms.

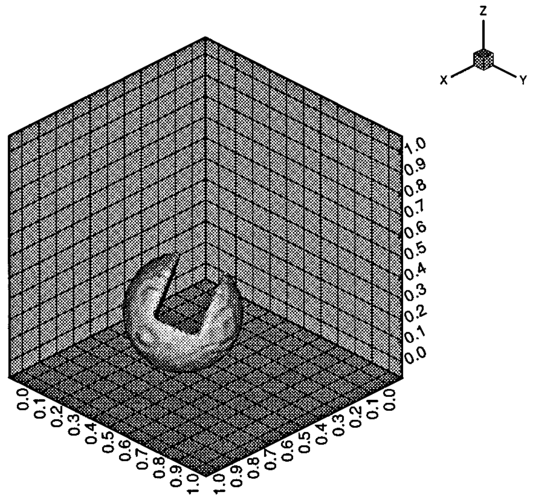

FIG. 4. The initial condition for the nonsmooth advection test.

stable timestep will be dependent on the flow direction and the stability limit of MU will be 1–3 times as large. We determined the relative performance of each method by comparing the logarithmic error convergence as a function of the mesh spacing h. A spherical Gaussian initial condition was used to verify the second-order convergence of the methods,

$$\psi(\mathbf{x}, 0) = \exp(-16((x - \tfrac{3}{4})^2 + (y - \tfrac{1}{4})^2 + (z - \tfrac{1}{4})^2)). \quad (23)$$

To test the performance on a nonsmooth field, we used a three-dimensional sphere of uniform concentration 1 centered at $(\frac{3}{4}, \frac{1}{4}, \frac{1}{4})$ with radius 0.25 with a rectangular wedge of width 0.20 and depth 0.42 removed in the $(-1, 1, 1)$ direction. This test (Fig. 4) is a three-dimensional analog to the familiar slotted disk test [29, 35].

The error convergence of MU was compared with popular advection methods currently used in small-scale atmospheric models: leapfrog [27], leapfrog-trapezoidal [35], UTOPIA, and Smolarkiewicz's MPDATA [29]. MPDATA (multidimensional positive definite advection transport algorithm) is a nonlinear predictor–corrector method that ensures that scalars cannot become negative as a result of advection errors. As scalar values approach zero, MPDATA becomes more diffusive and less accurate. We added a constant of 100 to the scalar values. This preserves the maximum possible accuracy of MPDATA, although it disables the positivity preservation and causes a loss of two significant digits in machine precision due to the need to subtract large and nearly equal quantities.

For smooth initial data, no flux correction or temporal filtering was used. All methods showed second-order convergence (Fig. 5a), except MPDATA, which was only 1.8-order accurate for this test due to the nonlinear advection. MU had the smallest overall error. UTOPIA had a 10% larger error, MPDATA had a fourfold larger error, and the leapfrog and leapfrog-trapezoidal methods had a sixfold larger error than MU at the highest grid resolution. We also see from Fig. 5a that MU's error does not increase when the timestep is tripled. For the leapfrog methods and MPDATA, on the other hand, the timestep used in the advection tests was approximately 78% of the stability limit and could not be appreciably increased without instabilities developing in corners of the domain. For UTOPIA, instabilities did not grow to large amplitude when the timestep was tripled. However, fine-scale variability in the solution was noticeable, and the error increased fourfold with the increased timestep. For nonsmooth initial data (Fig. 5b), MU and UTOPIA had similar errors. MPDATA had 15% larger errors, while the leapfrog and leapfrog-trapezoidal methods had 55% and 40% larger errors, respectively. Again, MU's errors are unaltered when the timestep is tripled, while the other methods become unstable. This is a substantial advantage for using MU in terms of computational efficiency.

We have shown that MU achieves superior accuracy with a larger timestep than the other methods. The utility of an advection scheme depends on the number of floating point operations (flops) and storage required to achieve a given global accuracy for a field that is advected over a fixed period of time. For MU, UTOPIA, and MPDATA, approximately 30 flops are required to evaluate the flux through each face, while for the leapfrog schemes, six flops (only 20% as much) are required. An additional time level of storage is required for the leapfrog methods. For the smooth advection tests with grid spacing h, a global error after one revolution of approximately Ah^2 was achieved using Bh^{-4} flops and Ch^{-3} storage, where A, B, and C are method-dependent constants. To reduce the error below ε requires $BA^2\varepsilon^{-2}$ flops and $CA^{3/2}\varepsilon^{-3/2}$ storage. Thus the computational effort is proportional to BA^2 and the storage is proportional to $CA^{3/2}$. In the advection test shown in Fig. 5a, A was six times as large for the leapfrog scheme as for MU. For the same timestep, B is only 20% as large for the leapfrog scheme, while C is at least as large. This lower B is partially compensated by the fact that the maximum stable timestep is 1–3 times larger for MU, depending on the advection direction. Assuming a timestep for MU that is $\sqrt{3}$ as large as for leapfrog, B is $\sqrt{3}/5 = 0.35$ as large for the leapfrog scheme. Hence, the computational effort and storage requirements for a given accuracy are $6^2\sqrt{3}/5 = 12$ and $6\sqrt{6} = 15$ times, respectively, as large as for MU. Similarly, from Fig. 5a, the computational effort is 2 (27) times as large for UTOPIA (MPDATA) as for MU, and the storage requirements are 1.15 (8) as large. For the nonsmooth advection tests, if we assume the global

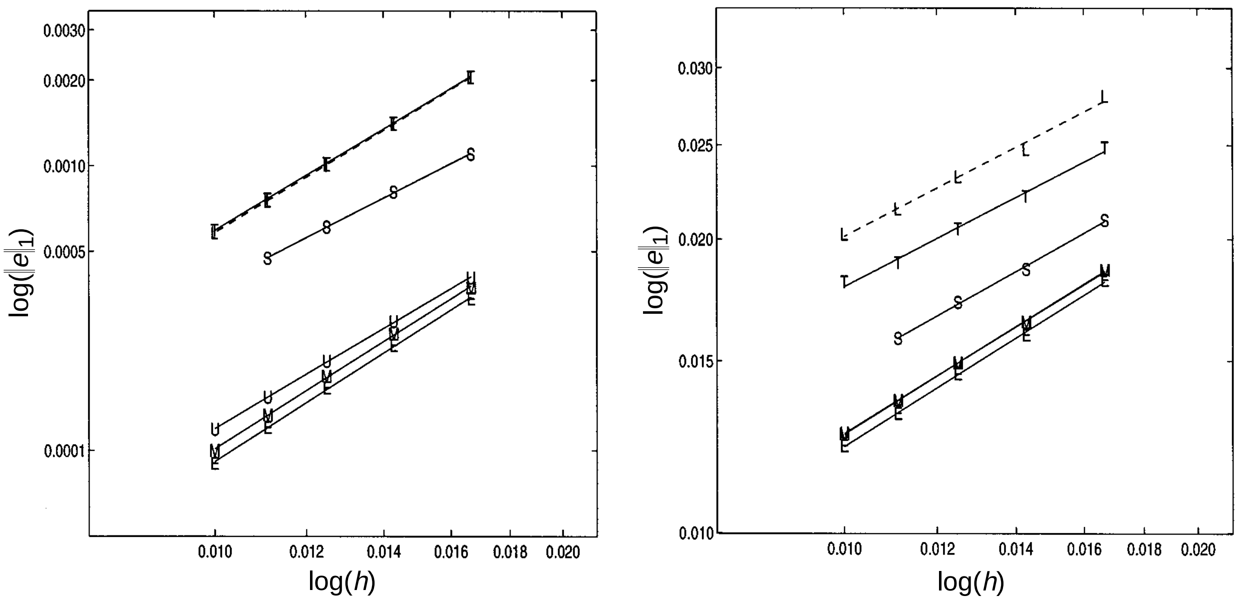

FIG. 5. Comparison of error convergence for flux-corrected Robert–Asselin filtered leapfrog (L), trapezoidal-leapfrog (T), MPDATA (S), UTOPIA (U), MU (M), and MU with threefold larger timestep (E). The plots show the 1-norm error after one rotation for smooth (a) and for nonsmooth (b) initial data.

error of all methods is approximately $Ah^{0.7}$, the computational effort scales as $BA^{5.7}$ and the storage requirements as $CA^{4.3}$. Furthermore, B is similar for the leapfrog and MU methods. After flux correction and time-filtering, the leapfrog methods required 60–80% as long per gridpoint per timestep as MU on a DEC Alpha-series workstation. This is compensated by MU's larger allowable timestep. Figure 5b suggests that the computational effort required by UTOPIA, MPDATA, and leapfrog schemes are respectively 1.8, 4, and 8 times as large and the storage requirements are 1, 2, and 8 times as large. Considering advection of both smooth and nonsmooth initial data, MU was computationally more efficient than the other schemes tested.

3. INCOMPRESSIBLE FLOW SOLVER

The simulation of nonhydrostatic atmospheric flows requires a general framework for the inclusion of physical processes involving source terms. This section presents an algorithm for integrating the inviscid dry anelastic equations [21],

$$\frac{\partial u_t}{\partial t} + \frac{1}{\overline{\rho}} \nabla \cdot (\overline{\rho} \mathbf{U} u_i) = -\frac{\partial \phi}{\partial x_i} + \delta_{i3} g \theta^*,$$
$$\frac{\partial \theta}{\partial t} + \frac{1}{\overline{\rho}} \nabla \cdot (\overline{\rho} \mathbf{U} \theta) = 0, \tag{24}$$
$$\nabla \cdot \overline{\rho} \mathbf{U} = 0,$$

where the scaled pressure perturbation $\phi = p'/\overline{\rho}$ is often referred to as the Exner function, $\overline{\rho}$ is the density associated with an isentropic base state of uniform potential temperature $\overline{\theta}$, and θ^* is the scaled perturbation $\theta^* = (\theta - \overline{\theta})/\overline{\theta}$. In (24), there are two source terms, pressure gradients and buoyancy. In our flow solver, source terms are incorporated using the second-order accurate method of Smolarkiewicz and Pudykiewicz [30] and Smolarkiewicz and Margolin [31]. This treatment, semi-Lagrangian in spirit, is a form of Strang splitting [34] and is easily generalized to handle additional sources and advection equations for other quantities such as moisture.

The anelastic equations are a coupled system of the form (1) that can be represented using Stoke's theorem as

$$\psi(\mathbf{x}, t^{n+1}) = \psi(\mathbf{x}_o, t^n) + \int_T R dt, \tag{25}$$

where T is the trajectory of a fluid parcel. This equation is approximated by advecting the quantity $\psi + \Delta t R/2$ with MU and using the integration rule

$$\psi(\mathbf{x}, t^{n+1}) = A\left(\psi(\mathbf{x}_o, t^n) + \frac{\Delta t}{2} R(\mathbf{x}, t^n), \boldsymbol{\nu}^{n+1/2}\right) + \frac{\Delta t}{2} R(\mathbf{x}, t^{n+1}), \tag{26}$$

where we denote MU as the advection operator A. This rule eliminates the need to compute $R(\mathbf{x}_o, t^n)$ and $\psi(\mathbf{x}_o, t^n)$ separately. Symbolically, we can integrate the anelastic equations (24) via

$$\theta^{*n+1} = A(\theta^{*n}, \boldsymbol{\nu}^{n+1/2})$$
$$u_i^{n+1} = A\left(u_i^n + \frac{\Delta t}{2}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i}\right), \boldsymbol{\nu}^{n+1/2}\right) + \frac{\Delta t}{2}\left(\delta_{i3}B^{n+1} - \frac{\partial \phi^{n+1}}{\partial x_i}\right). \tag{27}$$

This requires the evaluation of Courant numbers $\boldsymbol{\nu}^{n+1/2}$ and "pressures" $\phi^{n+1/2}$ and ϕ^{n+1}. The pressure can be interpreted as the potential or projection [11] that adjusts the velocities to preserve mass continuity. We insist that the velocities at $t^{n+1/2}$ and t^{n+1} both satisfy mass continuity, requiring two Poisson solves per timestep. The full solution of (27) proceeds as follows:

1. Calculate the momentum forcing at t^n. If this is the start of the simulation, a Poisson solve is necessary.

2. Use the momentum equation to compute a midpoint velocity at $t^{n+1/2}$, accurate to first order. The Courant numbers and buoyancy have been replaced by their values at t^n. However, for stability it is necessary to exactly preserve discrete mass continuity, so we require a Poisson solve for $\phi^{n+1/2}$ to ensure that $u_{\mathrm{i}}^{n+1/2}$ is nondivergent:

$$u_i^{n+1/2} = A\left(u_i^n + \frac{\Delta t}{4}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i}\right), \boldsymbol{\nu}^n\right) + \frac{\Delta t}{4}\left(\delta_{i3}B^n - \frac{\partial \phi^{n+1/2}}{\partial x_i}\right). \tag{28}$$

3. Compute the advection of θ^* using the midpoint velocities, where $\boldsymbol{\nu}^{n+1/2}$ is the vector of Courant numbers associated with $u_i^{n+1/2}$:

$$\theta^{*n+1} = A(\theta^{*n}, \boldsymbol{\nu}^{n+1/2}). \tag{29}$$

4. Advance the velocities using the modified trapezoidal rule. This requires a Poisson solve for ϕ^{n+1} to ensure that u_i^{n+1} is nondivergent:

$$u_i^{n+1} = A\left(u_i^n + \frac{\Delta t}{2}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i}\right), \boldsymbol{\nu}^{n+1/2}\right) + \frac{\Delta t}{2}\left(\delta_{i3}B^{n+1} - \frac{\partial \phi^{n+1}}{\partial x_i}\right). \tag{30}$$

This algorithm extrapolates velocities to $t^{n+1/2}$ by reusing the momentum equation in a manner similar to the algorithm of Bell and Marcus [5]. This method differs from theirs in that the quantities $\phi^{n+1/2}$ and $\boldsymbol{\nu}^{n+1/2}$ used in advection are the only values calculated at $t^{n+1/2}$. Another method presented by Smolarkiewicz and Margolin [31] for evaluating $\boldsymbol{\nu}^{n+1/2}$ is via the extrapolation $\boldsymbol{\nu}^{n+1/2} = 1.5\boldsymbol{\nu}^n - 0.5\boldsymbol{\nu}^{n-1}$. This eliminates the expense of the first Poisson solve at the expense of limiting the Courant number to less than $\frac{1}{2}$.

Maintaining a discrete analogue to mass conservation was a strong priority in the design of the solver. It uses the MAC (marker and cell) formulation of Harlow and Welch [15], where the velocities are staggered one half gridpoint in the normal direction. This formulation, also known as the Arakawa C-grid [2], was chosen because natural discretizations of pressure gradient and divergence in (24) lead to the standard seven-point stencil for the three-dimensional Laplacian. If nonstaggered grids for velocity [5, 11] and centered differences for pressure gradients and divergence are used, adjacent gridpoints can decouple. An alternative approach [1, 26, 36] is to use nonstaggered velocities, but to interpolate them to the MAC grid at half-time levels.

The inviscid anelastic equations with uniform stratification have a locally conserved energy integral

$$E = \int_\Omega \bar{\rho}\left(\frac{|\mathbf{U}|^2}{2} + \frac{(g\theta''/N\bar{\theta})^2}{2}\right) dV, \tag{31}$$

where $N = \sqrt{g\partial\theta'/\theta\partial z}$ is the Brunt–Väisälä frequency and θ is expressed as the sum of a base state, uniform stratification, and dynamic perturbation:

$$\theta = \bar{\theta} + \theta'(z) + \theta''(\mathbf{x}). \tag{32}$$

When integrated over a domain Ω with rigid or periodic boundary conditions, E is conserved. A bubble collapse experiment similar to that of Orlanski [22] and Clark and Farley [12] was performed to test the energy conservation properties of the flow solver. The flow solver was initialized with a potential temperature perturbation in the center of a domain of width 4 km in each direction and Brunt–Väisälä frequency $N = 0.01\ \mathrm{s}^{-1}$. The perturbation was of the form

$$\theta'' = A \exp\left(-3\left(\frac{(x-x_o)^2 + (y-y_o)^2 + (z-z_o)^2}{R_0^2}\right)^{1/2}\right) \tag{33}$$

with $A = 1$ K and $R_0 = 600$ m. The total energy oscillates between potential and kinetic energy as gravity waves are radiated throughout the domain. Rigid top and bottom and periodic lateral boundary conditions are applied, so the degree to which the total energy remains constant is an indication of the accuracy of the simulation. For these

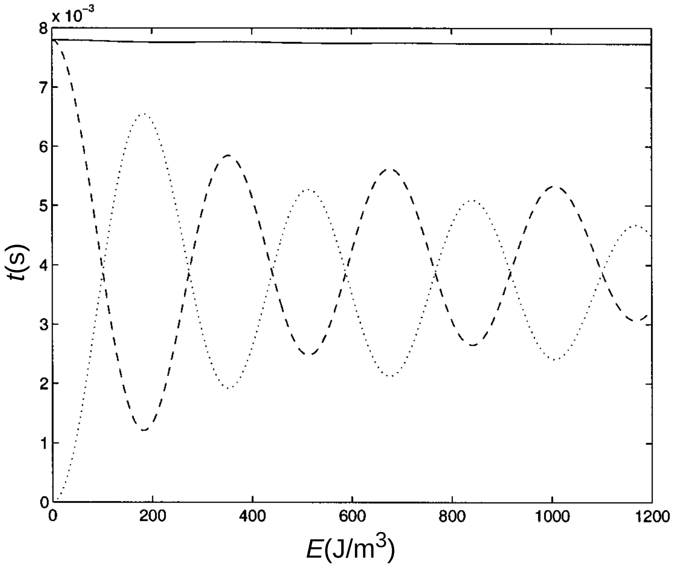

FIG. 6. Conservation of energy for a 100^3 uniform domain. The curves are total energy (solid), potential energy (dashed), and kinetic energy (dotted).

simulations, flux-correction was not used, since the scalar field was sufficiently well resolved by the grid that flux-correction does not improve the solution. The average energies for a 20-min simulation of a 100^3 domain are plotted in Fig. 6. The total energy of the simulation varied by only 1% despite large energy transfers between kinetic and potential energy. The second-order convergence of the flow solver can be verified by comparing the accuracy of the energy conservation at several resolutions. This is similar to the kinetic energy conservation test of Bell and Marcus [5]. Total energy conservation was found to converge quadratically with increased resolution as shown in Fig. 7.

4. MULTILEVEL FLOW SOLVER

Modern finite-difference algorithms are generally formulated in conservation form, since a discrete analogue to a conservation law obeyed by a fluid helps to ensure that even in regions of rapid change, the numerical scheme will not produce unphysical results. We used this idea as the guiding design principle in constructing our multilevel method. A cell-centered discretization is a natural selection for implementing this principle. The value at each gridpoint represents the average of a physical quantity over a rectangular cell and the integral of ψ over a domain Ω is given by a sum over uniform rectangular cells dV_i,

$$\int_\Omega \psi \, dV = \sum_{i \in \Omega} \psi_i \, dV_i. \tag{34}$$

A natural way of implementing refinement is to subdivide cells by an integer refinement ratio. This allows for easy adaptation of uniform-grid algorithms because no special cells are created at grid boundaries. Our multilevel method uses refinement in time as well as in space. This is necessary for both stability and accuracy, since we are using an explicit algorithm for advection. The Courant number of the flow remains constant when we refine temporally and spatially by the same ratio.

Our flow solver uses conservation of the global integral of ψ to determine the discretization of advected scalars. On the interior of fine domains, we satisfy this constraint by prescribing coarse grid values with volume averaging,

$$\psi_c = \frac{1}{n^3} \sum_{i \in \Omega_c} \psi_{fi}, \tag{35}$$

where Ω_c is the coarse grid cell containing ψ_c and n is the integer refinement ratio. Conservation in the presence of internal interfaces, is enforced by maintaining a relationship between the coarse and fine fluxes at the grid interface $\partial\Omega$,

$$F^n = \frac{1}{n^3} \sum_{m=0}^{n-1} \sum_{i \in \partial\Omega} f_i^{n+m}. \tag{36}$$

Here F^n is the coarse grid flux and f_i^{n+m} are the fine grid fluxes. We have found that by using the appropriate coarse grid information as image points for the fine domain that (36) is approximately preserved. It can be enforced exactly via refluxing (Berger [6]) which is the addition of a corrective update to F^n.

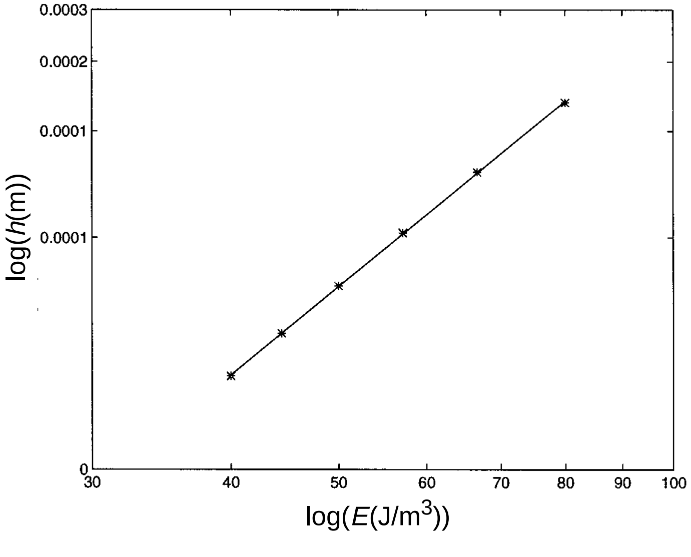

FIG. 7. Convergence of total energy for a sequence of uniform simulations with mesh spacing h = 40.0, 44.4, 50, 57.1, 66.7, and 80.0 m.

Discrete mass conservation was the fundamental motivation for the multilevel discretization of the velocities. For a multilevel method, this requires both a method for velocity restriction of fine to coarse velocities and a method for interpolating coarse velocities as boundary conditions for fine internal domains. The MAC grid staggering provides a simple mass conserving formula for restriction. For a face $\partial\Omega$ of a coarse cell, this formula is given by,

$$F = \frac{1}{n^2} \sum_{i \in \partial\Omega} f_i. \tag{37}$$

Here f_i are the fine grid mass fluxes that overlie the coarse grid mass flux F. This ensures that if mass is conserved within the fine domain, restriction will preserve this property in the underlying coarse domain. Matching coarse and fine grid mass fluxes at internal boundaries prescribes a Neumann boundary condition on the inner grid pressure. To satisfy (37) at internal boundaries, we interpolate coarse velocities, following Clark and Farley [12].

Adaptive refinement has been incorporated into the multilevel method. The user can choose to refine the base grid, either in fixed regions or using an adaptive refinement algorithm developed by Berger and Rigoutsos [9]. The composite grid is adjusted at a frequency determined by an estimate of the maximum Courant number to keep high-gradient regions within refined regions and to eliminate the need to maintain a large buffer zone in adjacent lower-gradient regions. Cells are marked as being underresolved by a user-specified criterion. The marked cells are clustered into rectangular subdomains by a recursive process that is repeated until the ratio of marked to total refined cells exceeds a desired percentage. This is an important consideration for three-dimensional simulations where a refinement ratio of n will create n^3 fine cells for every coarse cell refined.

As an example of the efficiency gain one can achieve using adaptive multilevel refinement, consider the "smooth" advection test of Section 2, in which a three-dimensional Gaussian hump of maximum amplitude 1 is rotated one full revolution around a diagonal axis. Because the hump is localized, refinement can be concentrated around the hump. Points were marked for refinement if they were within a six gridpoint wide buffer zone of scalar concentrations exceeding 0.01, and the composite grid was adjusted every six timesteps. A single level of refinement was used with a refinement ratio of two. Figure 8 shows the grid configuration for the initial condition of a 60^3 base domain. Figure 9 compares the 1-norm error after a full revolution between adaptively refined simulations and uniform grid simulations of varying resolutions. Each refined simulation is plotted at the fine mesh grid spacing h. We found that the coarsest refined simulation had a lower error than the highest resolution (100^3) uniform simulation. The error convergence of the refined simulations implies they are second-order accurate and have no more error than a uniform grid simulation with the same resolution as the fine grid. This figure illustrates the economy of computational resources that can be achieved using local refinement. The coarsest multilevel simulation required only 10% of the computation and 20% of the storage of an equivalent uniform simulation.

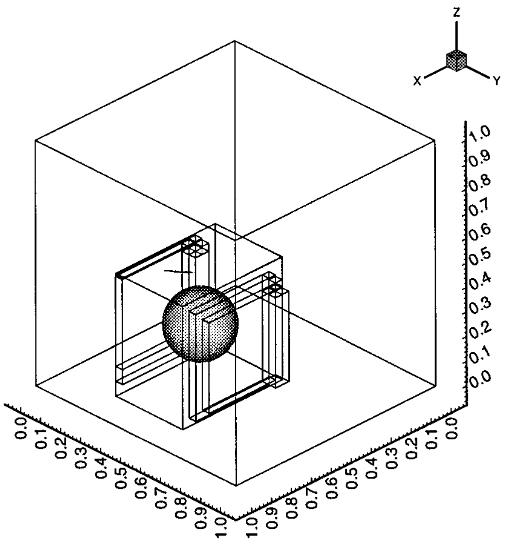

FIG. 8. The initial condition for the smooth multilevel advection test. The isosurface is the $\psi = 0.01$ contour.

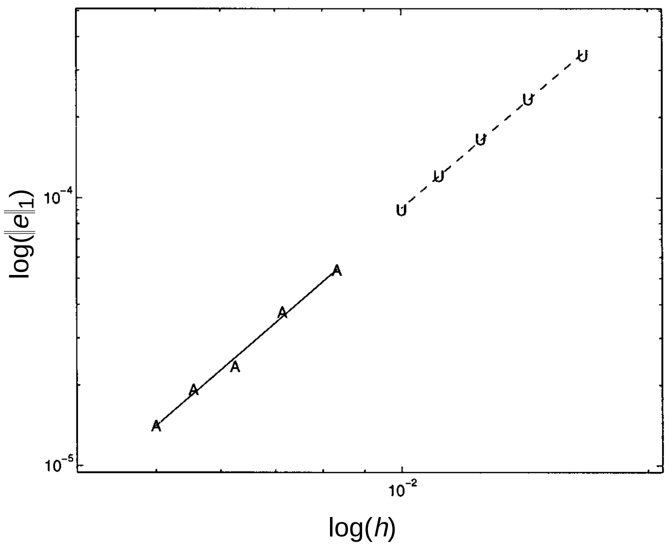

FIG. 9. Comparison of error convergence for the smooth advection test for a sequence of uniform resolution simulations (U) with resolution ranging from 60^3 to 100^3 and a sequence of adaptive simulations (A) with base domain resolutions ranging from 60^3 to 100^3.

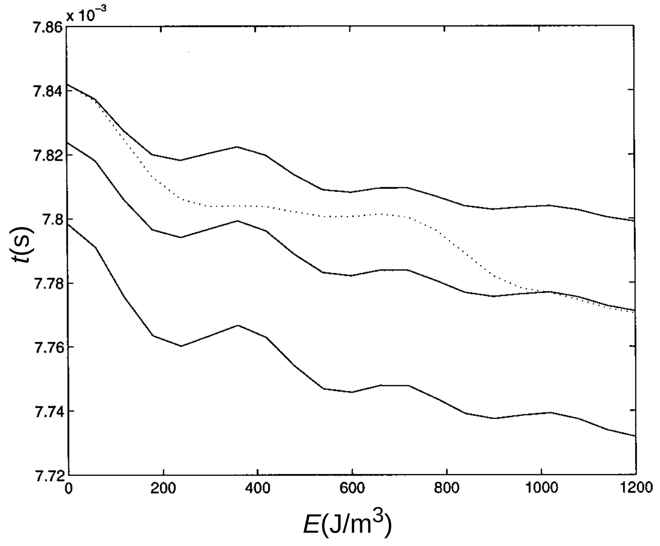

FIG. 10. Comparison of energy conservation between refined and uniform simulations. The solid curves are uniform 80^3, 90^3, and 100^3 simulations with increasing resolution upwards. The dotted curve is a refined 50^3 simulation.

The multilevel performance of the entire flow solver was tested by repeating the earlier bubble collapse experiment. We used a fine 36^3 domain covering the initial location of the bubble, where gradients are sharpest, nested within a coarse 50^3 domain with a refinement ratio of two. We compared the accuracy of energy conservation with uniform domains of 80^3, 90^3, and 100^3 gridpoints. The total energy evolution of these simulations is shown in Fig. 10. The small initial differences in total energy reflect the accuracy with which the initial disturbance is represented for different grid spacings. During the simulation, the change in the total energy (and, by inference, the global error) in the nested simulation is comparable to the 80^3 uniform-grid simulation. The 80^3 simulation required tenfold more flops and storage than the nested simulation. As we will show in a forthcoming paper, the errors in energy conservation can be further reduced to the level of the 100^3 uniform simulation by using a composite pressure solver over the full grid at every fine-grid timestep in place of the Neumann solver used in this work. However, this approximately doubles the computation required for the simulation. By using nested grid simulations with different coarse-grid spacings h and the same refinement ratio, we found that the energy conservation error is proportional to $h^{1.7}$; i.e., with almost second-order accuracy. This exponent increases to 1.9 if the composite pressure solver is used.

5. CONCLUSION

We have developed a forward-in-time multidimensional advection method, MU, that compares favorably with other advection methods. It is conservative and fully second-order accurate, including the treatment of source terms. Like other forward-in-time methods, it can be made monotonic using flux correction. This is particularly convenient for MU, since the scheme is implemented by adding corrective fluxes to a multidimensional monotonic low-order scheme. A substantial advantage over other forward-in-time schemes is its larger multidimensional stability region and third-order upwinding. Advection comparisons have shown that MU allows a one to threefold increase in the timestep, while retaining a considerable gain in accuracy.

This paper presents and evaluates an adaptive multilevel flow solver for the anelastic equations, a prototypical low Mach number flow. The solver is based on MU, but could easily be used with another forward-in-time advection scheme. There are no special startup procedures, half timesteps, or temporal filters to control computational modes. The solver is fully second-order accurate, and is discretized on a staggered MAC grid. It uses the same advection scheme for momenta as for scalars and requires only one representation of the solution to be stored. These are substantial advantages over earlier centered, differenced, and hybrid schemes. This paper demonstrates that this solver can often reduce by an order of magnitude the computational work and storage required to achieve a given accuracy. We are currently applying our solver to a variety of atmospheric flows that benefit from multilevel refinement, such as boundary layer cumulus convection and turbulent entrainment through highly stratified temperature inversions which commonly form above cloud-topped boundary layers.

APPENDIX: UPWINDING LOGIC FOR MU

The x direction flux for MU can be implemented using the upwinded locations

$$\begin{aligned} i^* &= i + \mathrm{nint}(1/2 - \mathrm{sign}(1/2, \nu_x)) \\ j^* &= j + \mathrm{nint}(1/2 - \mathrm{sign}(1/2, \nu_y)) \\ k^* &= k + \mathrm{nint}(1/2 - \mathrm{sign}(1/2, \nu_z)), \end{aligned} \tag{38}$$

where nint( ) and sign( ) are the FORTRAN nearest integer and sign functions. These locations determine the upwinded finite-differences used. The point $\psi_{i^*,j,k}$ corresponds to the upwinded value ψ_p and the compass point notation (ψ_e, ψ_w, ψ_n, ψ_s, ψ_t, ψ_b) of Section 2 corresponds to the points

$$(\psi_{i^*+1,j,k}, \psi_{i^*-1,j,k}, \psi_{i^*,j+1,k}, \psi_{i^*,j-1,k}, \psi_{i^*,j,k+1}, \psi_{i^*,j,k-1}). \tag{39}$$

The flux is computed by first evaluating the altered Courant numbers,

$$\begin{aligned}\mu_y &= (\nu_y)(j+1-j^*) + (1+\nu_y)(j^*-j)\\ \mu_z &= (\nu_z)(k+1-k^*) + (1+\nu_z)(k^*-k),\end{aligned} \tag{40}$$

and the bilinear interpolations

$$\begin{aligned}\hat{\psi}_e &= (1-\mu_y)(1-\mu_z)\psi_{i^*+1,j^*,k^*} + (\mu_y)(1-\mu_z)\psi_{i^*+1,j^*-1,k^*}\\ &\quad + (1-\mu_y)(\mu_z)\psi_{i^*+1,j^*,k^*-1} + (\mu_y)(\mu_z)\psi_{i^*+1,j^*-1,k-1},\\ \hat{\psi}_p &= (1-\mu_y)(1-\mu_z)\psi_{i^*,j^*,k^*} + (\mu_y)(1-\mu_z)\psi_{i^*,j^*-1,k^*}\\ &\quad + (1-\mu_y)(\mu_z)\psi_{i^*,j^*,k^*-1} + (\mu_y)(\mu_z)\psi_{i^*,j^*-1,k-1},\\ \hat{\psi}_w &= (1-\mu_y)(1-\mu_z)\psi_{i^*-1,j^*,k^*} + (\mu_y)(1-\mu_z)\psi_{i^*-1,j^*-1,k^*}\\ &\quad + (1-\mu_y)(\mu_z)\psi_{i^*-1,j^*,k^*-1} + (\mu_y)(\mu_z)\psi_{i^*-1,j^*-1,k-1}.\end{aligned} \tag{41}$$

Once these quantities have been evaluated, the low order flux is given by

$$\begin{aligned}F^{\mathrm{lo}}_{i,j,k} &= \nu_x\psi_{i^*,j,k} - \frac{\nu_x\nu_y}{2}(\psi_{i^*,j^*,k} - \psi_{i^*,j^*-1,k})\\ &\quad - \frac{\nu_x\nu_z}{2}(\psi_{i^*,j,k^*} - \psi_{i^*,j,k^*-1})\\ &\quad + \frac{\nu_x\nu_y\nu_z}{3}(\psi_{i^*,j,k^*} + \psi_{i^*,j^*-1,k^*-1} - \psi_{i^*,j^*,k^*-1} - \psi_{i^*,j^*-1,k^*})\end{aligned} \tag{42}$$

and the corrective flux by

$$\begin{aligned}F^{\mathrm{c}}_{i,j,k} &= \frac{|\nu_x|(1-|\nu_x|)}{2}((\hat{\psi}_e - \hat{\psi}_p)(i+1-i^*)\\ &\quad + (\hat{\psi}_p - \hat{\psi}_w)(i^*-i))\\ &\quad - \frac{\nu_x(1-\nu_x^2)}{6}(\hat{\psi}_e - 2\hat{\psi}_p + \hat{\psi}_w).\end{aligned} \tag{43}$$

ACKNOWLEDGMENTS

This research was supported by NSF Grants ATM8858846 and ATM9216645. A portion of this work was done as part of the doctoral dissertation [33].

REFERENCES

1. A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. L. Welcome, LBNL-39075, *J. Comput. Phys.,* submitted.
2. A. Arakawa, *J. Comput. Phys.* **31,** 119 (1966).
3. R. A. Assalin, *Mon. Weather Rev.* **100,** 487 (1972).
4. K. D. Beheng, *Atmos. Res.* **34,** Part II, 193 (1994).
5. J. P. Bell and D. L. Marcus, *J. Comput. Phys.* **100,** 334 (1989).
6. M. Berger and J. Oliger, *J. Comput. Phys.* **53,** 484 (1984).
7. M. Berger, *SIAM J. Numer. Anal.* **24,** 967 (1987).
8. M. Berger and P Colella, *J. Comput. Phys.* **82,** 64 (1989).
9. M. Berger and I. Rigoutsos, *IEEE Trans. Systems Man Cybern.* **21,** 1278 (1991).
10. J. P. Boris and D. L. Book, *J. Comput. Phys.* **11,** 38 (1973).
11. A. J. Chorin, *Math. Comput.* **23,** 341 (1969).
12. T. L. Clark and W. R. Farley, *J. Atmos. Sci.* **41,** 329 (1984).
13. P. Colella, *J. Comput. Phys.* **87,** 171 (1990).
14. D. R. Durran and J. B. Klemp, *J. Atmos. Sci.* **111** (1983).
15. F. H. Harlow and J. E. Welch, *Phys. Fluids* **8,** 2182 (1965).
16. B. P. Leonard, M. K. MacVean, and A. P. Loch, NASA *Technical Memorandum 1060555,* ICOMP-*93-05. NASA Lewis*, 1993.
17. R. J. LeVeque, *SIAM J. Numer. Anal.*, submitted.
18. J. LightHill, *Waves in Fluids* (Cambridge Univ. Press, London, 1978).
19. Mason, P. J., *Bound.-Layer Meteorol.* **32,** 281 (1985).
20. Moeng, C.-H., *J. Atmos. Sci.* **43,** 2886 (1986).
21. Y. Ogura and N. Phillips, *J. Atmos. Sci.* **19,** 173 (1962).
22. L. Orlanski, *J. Comput. Phys.* **21,** 251 (1978).
23. A. J. Robert, *J. Meteorol. Soc. Japan* **44,** 237 (1966).
24. J. Saltzman, *J. Comput. Phys.* **115,** 153 (1994).
25. A. P. Siebesma and J. W. M. Cuijpers, *J. Atmos. Sci.* **52,** 650 (1995).
26. T. M. Shih, C. H. Tan, and B. C. Hwang, *Int. J. Numer. Methods Fluids* **9,** 193 (1989).
27. W. C. Skamarock and J. B. Klemp, *Mon. Weather Rev.* **120,** 2109 (1992).
28. W. C. Skamarock and J. B. Klemp, *Mon. Weather Rev.* (1994).
29. P. K. Smolarkiewicz and T. L. Clark, *J. Comput. Phys.* **67,** 396 (1986).
30. P. K. Smolarkiewicz, and J. A. Pudykiewicz, *J. Atmos. Sci.* **49,** 2082 (1992).
31. P. K. Smolarkiewicz, and L. G. Margolin, *Mon. Weather Rev.* **121,** 1847 (1993).
32. Staniforth and Cotes, *Mon. Weather Rev.* **119,** 2206 (1991).
33. D. E. Stevens, Ph.D. thesis, University of Washington, 1994.
34. G. Strang, *SIAM J. Numer. Anal.* **5,** 506 (1968).
35. S. T. Zalesak, *J. Comput. Phys.* **31,** 335 (1979).
36. Y. Zhang, R. L. Street, and J. R. Koseff, *J. Comput. Phys.* **114,** 18 (1994).