Block-Structured Adaptive Grids on the Sphere: Advection Experiments

CHRISTIANE JABLONOWSKI, MICHAEL HERZOG,* AND JOYCE E. PENNER

Department of Atmospheric, Oceanic, and Space Sciences, University of Michigan, Ann Arbor, Michigan

ROBERT C. OEHMKE AND QUENTIN F. STOUT

Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan

BRAM VAN LEER AND KENNETH G. POWELL

Department of Aerospace Engineering, University of Michigan, Ann Arbor, Michigan

(Manuscript received 23 June 2005, in final form 21 January 2006)

ABSTRACT

A spherical 2D adaptive mesh refinement (AMR) technique is applied to the so-called Lin–Rood advection algorithm, which is built upon a conservative and oscillation-free finite-volume discretization in flux form. The AMR design is based on two modules: a block-structured data layout and a spherical AMR grid library for parallel computer architectures. The latter defines and manages the adaptive blocks in spherical geometry, provides user interfaces for interpolation routines, and supports the communication and load-balancing aspects for parallel applications. The adaptive grid simulations are guided by user-defined adaptation criteria. Both statically and dynamically adaptive setups that start from a regular block-structured latitude–longitude grid are supported. All blocks are logically rectangular, self-similar, and independent data units that are split into four in the event of refinement requests, thereby doubling the horizontal resolution. Grid coarsenings reverse this refinement principle. Refinement and coarsening levels are constrained so that there is a uniform 2:1 mesh ratio at all fine–coarse-grid interfaces. The adaptive advection model is tested using three standard advection tests with increasing complexity. These include the transport of a cosine bell around the sphere, the advection of a slotted cylinder, and a smooth deformational flow that describes the roll-up of two vortices. The latter two examples exhibit very sharp edges and gradients that challenge not only the numerical scheme but also the AMR approach. The adaptive simulations show that all features of interest are reliably detected and tracked with high-resolution grids. These are steered by either a threshold- or gradient-based adaptation criterion that depends on the characteristics of the advected tracer field. The additional resolution clearly helps preserve the shape and amplitude of the transported tracer while saving computing resources in comparison to uniform-grid model runs.

1. Introduction

Adaptive mesh refinement (AMR) techniques provide an attractive framework for atmospheric flows since they allow improved spatial resolutions in limited regions without requiring a fine-grid resolution throughout the entire model domain. The model regions at high resolution are kept at a minimum and can be individually tailored toward the atmospheric flow conditions. A solution-adaptive grid is a virtual necessity for resolving a problem with different length scales. To avoid, for example, underresolving high-gradient regions in the problem, or conversely, overresolving low-gradient regions at the expense of more critical regions, solution adaptation is a powerful tool saving several orders of magnitude in computing resources for many problems (Gombosi et al. 2004).

Climate and weather models, or generally speaking

* Current affiliation: Geophysical Fluid Dynamics Laboratory, Princeton, New Jersey.

Corresponding author address: Dr. Christiane Jablonowski, University of Michigan, Department of Atmospheric, Oceanic, and Space Sciences, 2455 Hayward St., Ann Arbor, MI 48109.
E-mail: cjablono@umich.edu

computational fluid dynamics codes, are among the many applications that are characterized by multiscale phenomena and their resulting interactions. But although today's atmospheric general circulation models (GCMs), and in particular weather prediction codes, are already capable of uniformly resolving horizontal scales of order 25 km (Temperton 2004), the atmospheric motions of interest span many more scales than those captured in a fixed-resolution model run. As an example, the resolution of convective motions and cloud dynamics may require resolutions of order 1 km or even finer mesh sizes (Bryan et al. 2003). The widely varying spatial and temporal scales, in addition to the nonlinearity of the dynamical system, raise an interesting and challenging modeling problem. Solving such a problem more efficiently and accurately requires variable resolution.

Today, AMR techniques are rarely applied to atmospheric flow simulations. More commonly, two alternative nonuniform-grid approaches are utilized. These are the widely used nested and stretched grid techniques, both of which can be implemented in a statically or dynamically adaptive way [see also Fox-Rabinovitz et al. (1997) for an overview]. The fundamental differences between AMR and the nested or stretched grid strategies lie in their flexibilities to adapt readily to arbitrary flow situations. While AMR varies the number of grid points as demanded by the adaptation criterion and evolving flow features, the total number of grid points in nested or stretched meshes stays constant during the simulation. They may therefore be considered global remapping approaches that, in case of dynamic remappings, move the fine-resolution regions at the expense of other coarsened model areas. Recent examples of dynamic grid deformations include Iselin et al. (2002) and Prusa and Smolarkiewicz (2003). Iselin et al. (2002) applied flow-dependent weighting functions to steer the varying grid resolution whereas Prusa and Smolarkiewicz (2003) used a priori information about the evolving flow field for their remapping strategy. AMR, on the other hand, adds and removes grid points locally without affecting the resolution in distant model domains and, most importantly, does not require a priori knowledge of future refinement regions. Both aspects are a strength of the AMR design. However, despite the differences between the three nonuniform-grid paradigms they all have one aspect in common. The varying resolution can cause artificial reflections and refractions of waves due to incompatible mechanisms at fine–coarse-grid interfaces. As an example, a traveling wave may undergo false reflections or aliasing when propagating from the fine grid to the coarse domain. Therefore, special attention needs to be paid to the fine–coarse-grid interface conditions. Here, mass-conserving interpolation and flux-matching mechanisms that foster the smooth transport of the advected feature across varying grid resolutions are employed.

Introduced in this paper is a 2D adaptive grid technique on the sphere, which is built upon a block-structured data layout and an AMR grid library for parallel computer architectures (Oehmke and Stout 2001; Oehmke 2004). For conciseness, the discussion is focused on an adaptive advection problem, although the underlying principles are readily applicable to nonlinear model setups (Jablonowski 2004; Jablonowski et al. 2004). These are further addressed in a future paper that illustrates the AMR technique for 2D shallow water and 3D primitive equation models. In general, atmospheric dynamics on all scales is dominated by the advection process. A precise numerical solution of the advection problem is therefore fundamentally important to the overall accuracy of atmospheric flow solvers and tracer transport schemes. To date, various dynamically adaptive advection codes have been presented in the atmospheric science literature. These include the 2D passive advection algorithms by Behrens (1996) and Behrens et al. (2000) who formulated an adaptive grid triangulation method in the *x*–*y* plane. Kessler (1999) implemented a finite-element advection technique and evaluated different refinement criteria for the adaptive transport process. Another AMR advection study by Stevens and Bretherton (1996) concentrated on numerical aspects of adaptive multilevel solvers, and Tomlin et al. (1997) investigated adaptive gridding options for modeling chemical transports with multiscale sources. In particular, the analysis of Tomlin et al. (1997) was focused on the interactions among emission plumes and the ambient air. Similar emission scenarios were also discussed by Odman et al. (1997), Sarma et al. (1999), and Srivastava et al. (2000) in their adaptive air quality studies. In addition, Bacon et al. (2000) and Boybeyi et al. (2001) introduced the adaptive regional weather and tracer transport model OMEGA (Operational Multiscale Environmental Model with Grid Adaptivity), which is based on unstructured, triangulated meshes with rotated Cartesian coordinates. The triangulations can statically or dynamically be adapted to user-defined regions or features of interest. This operational modeling system has mainly been designed for real-time aerosol and gas hazard predictions.

Most recently and most relevant to the study presented here, Hubbard and Nikiforakis (2003) described the design of an adaptive 3D passive advection code for tracer transport problems on the sphere. They utilized an extended version of the publicly available Berger–Oliger AMR grid library (Berger and Oliger 1984) which was originally designed for logically rectangular block-data approaches in Cartesian coordinates. To date the Berger–Oliger AMR approach has also been used multiple times in the context of limited-area or regional atmospheric modeling in Cartesian geometry. Examples include Skamarock et al. (1989) and Skamarock and Klemp (1993) who investigated adaptive meshes for regional weather prediction applications. Furthermore, the movable nested grids for cyclone track predictions by Fulton (1997, 2001) as well as the adaptive ocean model by Blayo and Debreu (1999) are based on the Berger–Oliger AMR design. The main differences between Berger and Oliger (1984), Hubbard and Nikiforakis (2003), and the AMR approach discussed here are the underlying space–time refinement paradigms, the native coordinate systems and grid hierarchies, the AMR user interfaces, and most importantly, the software engineering aspects like the parallel computing support for modern distributed-memory hardware architectures. This parallel computing support is not provided by the Berger–Oliger AMR library but will become fundamentally important for more complex fully nonlinear AMR applications, especially in 3D. Therefore, the AMR library proposed here has built-in parallel communication and dynamic load-balancing mechanisms.

The paper is organized as follows. In section 2 the basic AMR design principles are explained. This section includes a discussion of the block-data grid configuration in spherical geometry, the refinement and coarsening strategies, as well as the interpolation, averaging, and numerical flux-matching mechanisms at fine–coarse-grid interfaces. In addition, the functionality of the AMR grid library is described. Section 3 reviews the so-called Lin–Rood finite-volume advection algorithm (Lin and Rood 1996, referred to as LR96 hereafter), which serves as an example application for the adaptive mesh approach. The adaptive transport code is tested using three standard advection examples that increase in complexity. In particular, these are the transport of a cosine bell around the sphere at various rotation angles, the advection of a slotted cylinder, and a smooth deformational flow (cyclogenesis) problem. The model setups, error measures, test definitions, and the results of the adaptive advection simulations are presented in section 4. Section 5 summarizes the findings and outlines the future potential of AMR for nonlinear atmospheric flow problems.

2. AMR design on the sphere

To date, dynamically adaptive meshes in atmospheric modeling have mostly been applied to limited-area models in Cartesian geometry. Cartesian grids are well suited for AMR techniques since the physical locations of neighboring grid points are uniquely defined by their coordinate positions. This is in contrast to global grids in spherical geometry, especially if regular latitude–longitude grids with converging meridians are selected as discussed here. The main differences occur at the pole points due to the singularity of the grid and its coordinate system. The identification of neighbors across the poles, together with the cross-polar sign reversal of the velocity components in spherical coordinates (see also Hubbard and Nikiforakis 2003), adds extra complexity to the AMR approach on the sphere. As a consequence, the AMR design in spherical coordinates needs special provisions for polar regions that take a 180° shift in longitudinal direction for cross-polar neighbors into account. This is further discussed in section 2b. In addition, the periodicity of the spherical grid in longitudinal direction needs to be supported by the AMR library.

a. Block-structured adaptive grids

The AMR approach is based on a 2D block-structured data configuration in spherical coordinates that only requires minimal changes to the preexisting Lin–Rood transport algorithm (LR96). The concept of the block-data structure is displayed in Fig. 1, which shows an orthographic projection of the earth with a blocked latitude–longitude grid. Each self-similar block is logically rectangular and comprises a constant number of $N_x \times N_y \times$ grid cells in longitudinal and latitudinal direction. Here, $N_x = 9$ and $N_y = 6$ grid points per block are selected with $B_x \times B_y = 8 \times 6$ blocks on the entire sphere. These parameters correspond to a 5° × 5° uniform mesh resolution. The computational grid can then be viewed as a collection of individual blocks that are independent data units. Here the block-data principle is solely applied to the horizontal directions so that the whole vertical column is contained in a block in case of 3D model configurations. Other block-data approaches, as described in Stout et al. (1997), MacNeice et al. (2000), and Hubbard and Nikiforakis (2003), employ a 3D strategy that includes a block distribution in the vertical direction as well.

The block-data structure is well suited for adaptive mesh applications. The basic AMR principle is illustrated in Fig. 2. Starting from an initial mesh at constant resolution with, for example, 3×3 cells per block, a *parent* block is divided into four *children* in the event of

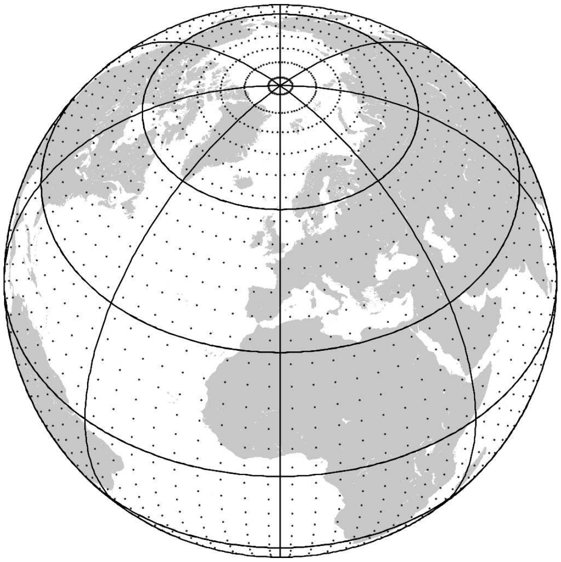


FIG. 1. Distribution of grid points and blocks over the sphere in an orthographic projection centered at 45°N, 0°. The resolution is 5° × 5° (nonadapted case).

refinement requests. Each child becomes an independent new block with the same number of grid cells in each dimension, thereby doubling the horizontal resolution in the region of interest. Coarsening, on the other hand, reverses the refinement principle. Then four children are coalesced into a single self-similar parent block, which reduces the grid resolution in each direction by a factor of 2. Both the refinement and coarsening steps are mass-conservative. In the present AMR setup, neighboring blocks can only differ by one refinement level, guaranteeing a uniform 2:1 mesh resolution at fine–coarse-grid interfaces. As a result, continuously cascading refinement regions are created that provide a desired buffer zone around the blocks at the finest nesting level (see also section 4b).

Each block is surrounded by ghost cell regions that share the information along adjacent block interfaces. This makes each block independent of its neighbors since the solution technique can now be individually applied to each block. The ghost cell information ensures that the requirements for the numerical stencils are satisfied. The advection algorithm then loops over all available blocks on the sphere before a communication step with ghost cell exchanges becomes necessary. The number of required ghost cells highly depends on the numerical scheme. In the LR96 advection scheme the piecewise parabolic method (PPM) (Colella and Woodward 1984) is chosen. As a consequence, three ghost cells in each horizontal direction are needed. Note that all ghost regions are at the same resolution as the inner domain of the block. There are two types of interfaces in the adaptive grid configuration as illustrated in Fig. 3. If the adjacent blocks are at the same refinement level (Fig. 3a) the neighboring information can easily be exchanged since the data locations overlap. The ghost cell data are then assigned the appropriate solution values of the neighboring block, which is indicated by the gray-shaded areas. If, on the other hand, the resolution changes between adjacent blocks (Fig. 3b), averaging and interpolation routines are invoked at fine–coarse mesh boundaries. Here, an interpolation method is chosen that is based on the PPM approach. This conservative and monotonic remapping

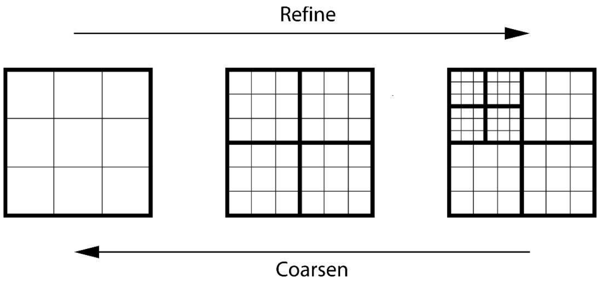


FIG. 2. Refinement and coarsening principles with two refinement levels.

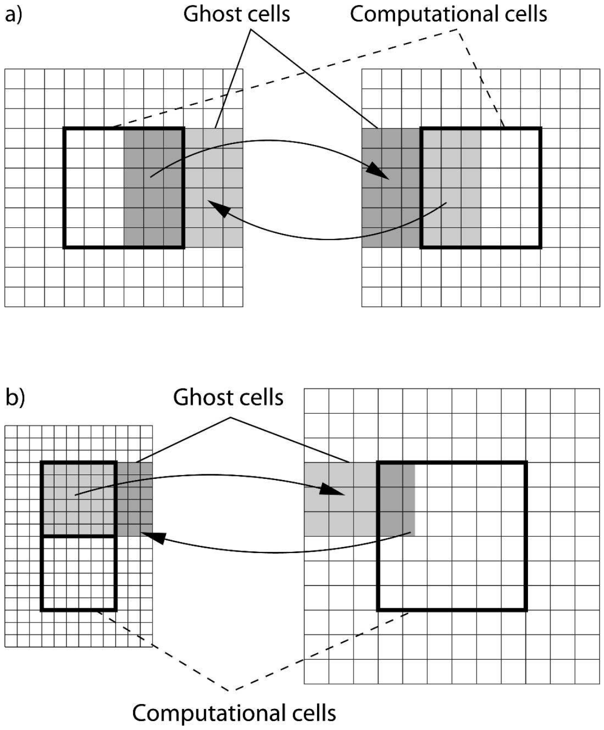


FIG. 3. Ghost cell updates for blocks at (a) the same and (b) different refinement levels. The shading indicates the overlap regions of the ghost cells. In this example 6 × 6 grid points per block with three ghost cells are chosen.

technique matches the order of accuracy of the underlying LR96 advection algorithm.

It is important to note that the adapted blocks do not overlay each other in the AMR approach presented here. Instead, each block is assigned a unique surface patch on the sphere and, as a consequence, any coarse-grid information is no longer accessible in refined regions. This is in contrast to the AMR block-data design by Berger and Oliger (1984). In particular, the Berger–Oliger AMR method is built upon a hierarchy of nested grids that are individually sized and, furthermore, all actively used during the simulation. The solution is then concurrently computed on all blocks at all refinement levels, and in a second step, the coarse resolution data are overwritten wherever the fine-resolution nests overlap. This approach adds overhead to the AMR simulation but, on the other hand, allows a Richardson-type estimation of the local truncation error. Such an adaptation criterion was for example used by Skamarock (1989). In the study presented here, a flow-based refinement criterion is examined instead. Also note that the Berger–Oliger library allows the refinement factor between neighboring blocks to be any positive integer number whereas a constant 2:1 ratio is chosen for the AMR approach here. This guarantees rather accurate inflow and outflow conditions at the fine–coarse-grid interfaces, but on the other hand, limits the flexibility of the adapted grid. For 3D applications, the Berger–Oliger library also supports refinements in the vertical direction. Another difference between the AMR approaches lies in their time-stepping procedures. The Berger–Oliger AMR design provides smaller time steps at fine resolutions and as a result, the solvers are subcycled in the nested domains. This requires temporal interpolations of the coarse boundary data during the subcycling steps, which are not applied in the AMR approach here. Instead, the time step is held constant at all refinement levels, which provides instantaneous updates of the boundary information. Only spatial interpolations are invoked in the ghost regions at fine–coarse-grid interfaces. However, there are also limitations to such an approach. The chosen time step must be numerically stable on the finest grid in an adapted model run. Therefore, overhead is added to the coarse-resolution regions that do not require a short time step for stability reasons. These pros and cons of the two time-stepping techniques need to be further examined for future AMR applications.

In general, the Berger–Oliger AMR design is based on Cartesian meshes. Thus, special extensions for polar regions were introduced by Hubbard and Nikiforakis (2003) when customizing the approach for global grids on the sphere. By contrast, the AMR grid library applied here is specifically designed for spherical coordinate systems. Therefore, it has built-in pole point provisions, provides user interfaces for application-specific interpolation and averaging routines, and most importantly, supports the AMR approach on today's parallel computing architectures. An overview of the AMR library is given in the following section.

b. Overview of the AMR grid library

The adaptive blocks are managed by a spherical adaptive grid library for distributed memory architectures (Oehmke and Stout 2001; Oehmke 2004). The library functions can be accessed via Fortran90 subroutine calls. In brief, the characteristics and functions of the library are listed as follows.

1) *Sphere*. The library provides functions for the creation of a sphere that is built upon a block-structured latitude–longitude grid configuration. In addition, reduced spherical grids are supported that widen the zonal grid intervals in polar regions. In both cases, the size of the blocks as well as the number of blocks on the sphere are user defined and determine the initial, and thereby coarsest, resolution of the adaptive simulation. Overall, the library maintains and initializes the geometric information and allows the reinitialization of the geometry data after adaptations have occurred. Here, each block covers a unique surface patch on the sphere.
2) *Bookkeeping*. The library maintains the adjacency information for all blocks at arbitrary refinement levels. These comprise the cross-polar neighbors for blocks at the pole points, which are shifted by 180° in longitudinal direction. A schematic view of the neighboring blocks is shown in Fig. 4. In this constant resolution example, each block has eight neighbors, which includes the neighboring blocks in the cross directions. The communication in the cross direction is provided to allow for a wide range of user-selected algorithms. In the transport example discussed in this paper, the cross directions are needed for the cross derivatives of the interpolation algorithm (see below) as well as the advective-form inner operators of the Lin–Rood advection scheme (Lin and Rood 1996). In a more general setting with varying neighboring resolutions, the number of neighbors lies between 6 and 12.
3) *Iterators*. Iterators enable the user to loop over all assigned blocks on a given processor. They can be viewed as pointers that pick out the next independent block index for application-specific calculations. Note that consecutive blocks in the data

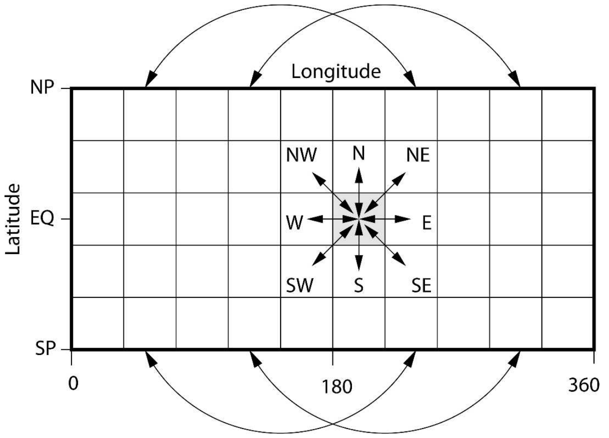


FIG. 4. Information exchange amongst nearest neighbors. Blocks at the poles (NP or SP) have neighbors across the poles that are 180° shifted in longitudinal direction. The letters N, E, S, and W symbolize the orientation of neighboring blocks.

structure may lie at arbitrary positions on the sphere.

4) *Communication*. The library provides transfer functions that manage the ghost cell updates on parallel computer architectures. The communication is either based on the message passing interface (MPI) library or memory copies. The former is invoked for neighboring blocks on different processors whereas memory copies speed up the ghost cell exchange for neighbors on the same processor. The transfer module utilizes send-and-receive buffers and accesses the block-adjacency information during the ghost cell exchange. The details of the parallel communication are hidden from the user. Instead, user-friendly communication functions are provided.
5) *Load-balancing*. The load-balancing module redistributes the blocks among the processors during the adaptive model run. Currently, the dynamic load-balancing strategy is guided by the total number of blocks. Here, the goal is to assign an approximately equal number of blocks to each processor. For the future, a space-filling-curve load-balancing approach is planned that takes data locality aspects into account (Dennis 2003; Behrens and Zimmermann 2000). Note that the parallel performance, especially of complex adaptive 3D GCM applications, strongly depends on an efficient domain decomposition and load-balancing strategy.
6) *Adaptation module*. The library manages the coarsenings and refinements of the blocks based on adaptation flags. These flags are set via library functions during a model run and guided by user-defined adaptation criteria. The adaptation module redefines the block connectivity after adaptations occurred and enforces the constraint that adjacent blocks can differ by no more than one refinement level.
7) *User interface*. User-defined subroutines need to be provided that specify the algorithms for split-and-join and ghost cell operations. These routines include the interpolation and averaging procedures for the initialization of new blocks and the data exchange algorithms for neighboring blocks at both identical and varying resolutions.

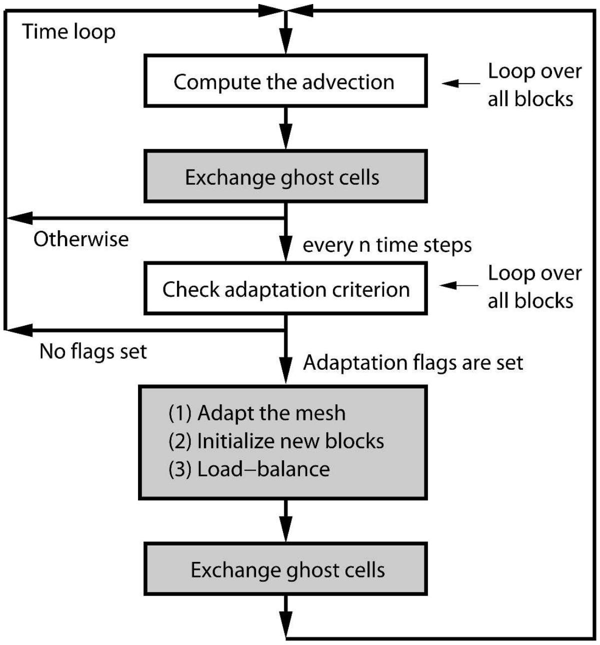


FIG. 5. Program flow with adaptive mesh functionality. AMR library calls are shaded in gray. The user-defined parameter *n* denotes the number of time steps before the adaptation criterion is checked.

Adaptive blocks are well suited for distributed-memory computing concepts. Since ghosted blocks are self-sufficient they can be readily distributed among many processors. During the simulation each processor then iterates over its assigned blocks to solve the model equations on a block-by-block basis. When ghost cell exchanges are required all processors reach a synchronization point and execute a parallel exchange of ghost cell information. A schematic view of the program flow is presented in Fig. 5, which shows the time loop of the adaptive grid application discussed in this paper. In particular, an adaptation cycle of $n = 1$ is chosen that checks the adaptation criterion at every time step. Refinements are triggered if the refinement flag for one or more grid points in a block is set. Coarsenings, on the other hand, are slightly harder to activate. They not only require all four children to set the coarsening flag but also must ensure a uniform 2:1 mesh ratio at all fine–coarse-grid boundaries after an adaptation step. This is checked by the adaptation module of the AMR library. Note that the gray-shaded areas in Fig. 5 indicate the AMR library calls that are added to the advection model. In general, such an AMR principle is universally applicable to any flow solver that can utilize block-structured data setups on the sphere.

c. Fine–coarse-grid interfaces

In an adaptive model run, neighboring blocks at different resolutions require the use of interpolation and averaging routines to update the ghost cell regions at every time step. In addition, blocks need to be reinitialized after adaptation requests. An overview of these AMR user routines for the ghost cell exchange is presented below. In addition, a flux-matching strategy at fine–coarse-grid interfaces is discussed that ensures mass conservation in an adapted application.

1) Interpolation method

A suitable interpolation technique for the ghost cell update in an AMR application is required to be (a) monotonic, (b) conservative, and (c) consistent with the selected transport algorithm. These three design principles lead to a natural 2D extension of the underlying 1D finite-volume PPM algorithm as applied in LR96. Details of PPM's 1D reconstruct–evolve–average principle can be found in Carpenter et al. (1990) and Colella and Woodward (1984) who also show schematic diagrams of the PPM strategy. In short, each cell holds cell-averaged model quantities that are pieced together in either latitudinal or longitudinal direction to construct a parabolic reconstruction of the selected field. This is also called a "subgrid distribution," which can be made monotonic. Here, the 1D subgrid distributions serve as building blocks for a PPM-like reconstruction algorithm in 2D.

To use the PPM approach for the interpolation of a conservative scalar h in an adaptive application, two steps become necessary. First, a monotonic subgrid distribution $h(x, y)$ is computed that is based on the coarse-grid data. Then the subgrid distribution is integrated over the nested fine-grid regions, which guarantees the conservative mapping of the coarse-grid information.

For the 2D extension of the PPM algorithm, a biparabolic function needs to be defined. As shown by Rančić (1992), a full 2D extension of the PPM scheme requires the calculation of nine coefficients, which leads to a rather computationally expensive method. Therefore, a quasi-biparabolic approach with six components is derived that modifies the scheme proposed by Nair and Machenhauer (2002) with five coefficients. Here, a directionally bias-free cross-term is added to the Nair and Machenhauer (2002) algorithm that helps smooth the subgrid distribution near sharp edges. A similar mixed derivative term was also introduced by van Leer (1985) in an alternative definition of a 2D biparabolic subgrid function. The piecewise parabolic subgrid distribution $h(x, y)$ for a discrete cell-averaged scalar field $\overline{h}$ at the cell center (i, j) is given by

$$h(x, y) = \overline{h} + \delta a^x x + b^x\left(\frac{1}{12} - x^2\right) + \delta a^y y + b^y\left(\frac{1}{12} - y^2\right) + \frac{1}{2}(c^{xy} + c^{yx})xy. \qquad (1)$$

The indices (i, j) are dropped for conciseness. Here, δa^x, δa^y indicate the slopes, and b^x, b^y indicate the curvature terms of the parabola in longitudinal and latitudinal direction, respectively. These 1D coefficients are defined in Carpenter et al. (1990) who used the notation δa for δa^x and δa^y, a_6 for b^x and b^y, and $\langle a \rangle$ for $\overline{h}$. All coefficients are monotonized as in LR96's monotonic Flux-Form Semi-Lagrangian (FFSL-3) scheme. The cross-term consists of the two components c^{xy} and c^{yx} that are averaged to avoid a directional bias.

In particular, c^{xy} and c^{yx} at a cell center (i, j) are determined by

$$c^{xy} = \frac{1}{2}(\Delta a^x_{i,j+1} - \Delta a^x_{i,j-1}), \qquad (2)$$

$$c^{yx} = \frac{1}{2}(\Delta a^y_{i+1,j} - \Delta a^y_{i-1,j}), \qquad (3)$$

where Δa^x and Δa^y are the centered-difference slopes

$$\Delta a^x_{i,j} = \frac{1}{2}(\overline{h}_{i+1,j} - \overline{h}_{i-1,j}), \qquad (4)$$

$$\Delta a^y_{i,j} = \frac{1}{2}(\overline{h}_{i,j+1} - \overline{h}_{i,j-1}), \qquad (5)$$

which are further monotonized via the van Leer (1977) monotonized-central (MC) slope limiter. This limiter is given by

$$\begin{aligned}\Delta a^x_{i,j} &= \min(|\Delta a^x_{i,j}|, 2|h_{i+1,j} - h_{i,j}|, \\ &\quad 2|h_{i,j} - h_{i-1,j}|)\,\mathrm{sgn}(\Delta a^x_{i,j}) \quad \text{if} \\ &\quad (h_{i+1,j} - h_{i,j})(h_{i,j} - h_{i-1,j}) > 0 \\ &= 0 \quad \text{otherwise.}\end{aligned} \qquad (6)$$

The MC slope limiter is also applied to Δa^y with respect to the y direction and both cross components c^{xy} and c^{yx}. Furthermore, the cell-averaged value $\overline{h}$ is defined as

$$\overline{h} = \int_{-1/2}^{1/2}\int_{-1/2}^{1/2} h(x, y)\, dx\, dy, \tag{7}$$

where x, y are the local normalized coordinates in each grid cell with $x, y \in [-1/2, 1/2]$. For the spherical coordinate system (λ, φ) they are given as in Nair and Machenhauer (2002):

$$x = \frac{\lambda - \lambda_i}{\Delta\lambda_i} - \frac{1}{2}, \tag{8}$$

$$y = \frac{\mu - \mu_j}{\Delta\mu_j} - \frac{1}{2}, \tag{9}$$

with $\Delta\lambda_i = (\lambda_{i+1} - \lambda_i)$, $\Delta\mu_j = (\mu_{j+1} - \mu_j)$ and $\mu_j = \sin\varphi_j$ [for the derivation see Nair and Machenhauer (2002)]. It is important to point out that the μ coordinates are no longer equidistant and become increasingly compressed in polar regions. Nevertheless, the parameters of the parabola in Eq. (1) are computed in the equidistant (λ, φ) grid point space. This approximation of the $h\ (x, y) = h\ [x(\lambda), y(\mu)]$ distribution is motivated by Veldman and Verstappen (1998) who discussed the advection–diffusion problem for nonuniform meshes. They found that equidistant estimates of the derivatives maintain the skew-symmetry of the problem. As an alternative, a more complex nonequidistant formulation for the parameters of the parabola (Colella and Woodward 1984) can also be used.

In a second step, the coarse-grid data are remapped via analytic integrals. If assuming that the fine-grid region lies within the lower and upper limits $x \in [x_l, x_u]$, $y \in [y_l, y_u]$ inside a coarse-grid cell, the new $\overline{h}^r$ value for the refined mesh is determined by

$$\begin{aligned}\overline{h}^r &= \frac{1}{\Delta x \Delta y}\int_{y_l}^{y_u}\int_{x_l}^{x_u} h(x,y)\, dx\, dy\\ &= \overline{h} + \frac{\delta a^x}{2}(x_u + x_l) + b^x\left[\frac{1}{12} - \frac{1}{3}(x_u^2 + x_u x_l + x_l^2)\right]\\ &\quad + \frac{\delta a^y}{2}(y_u + y_l) + b^y\left[\frac{1}{12} - \frac{1}{3}(y_u^2 + y_u y_l + y_l^2)\right]\\ &\quad + \frac{c^{xy} + c^{yx}}{8}(x_u + x_l)(y_u + y_l),\end{aligned} \tag{10}$$

with $\Delta x = (x_u - x_l)$ and $\Delta y = (y_u - y_l)$. The limits of the integral are also schematically shown in Fig. 6 for the lower-right refined cell h_4^r. Here, the coarse-grid symbol h^c corresponds to $\overline{h}$ in Eq. (10). Note that in practice, the position of y_u in Fig. 6 is slightly off-centered due to the nonequidistant μ distribution. As an aside, in this specific configuration with equidistant normalized x coordinates $(x_u^2 + x_u x_l + x_l^2) = 1/4$ holds in each of the four refined grid boxes. Therefore, the b^x expression (third term on the right-hand side) in Eq. (10) is identical to zero.

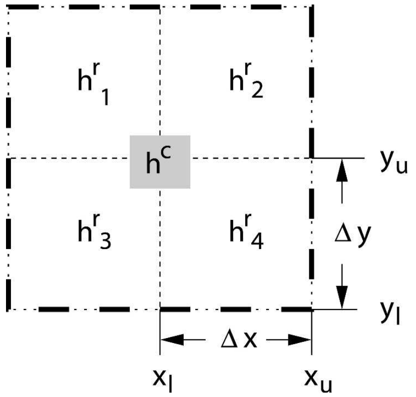


FIG. 6. Location of the scalar cell-mean variable h on the coarse and refined grid (the overbar is omitted). The coarse cell is dashed with gray-shaded symbol h^c. The four refined cells are denoted with h^r (dotted contours). For cell h_4^r, the upper (u) and lower (l) limits of the integral that span the latitudinal and longitudinal distances Δy and Δx in the normalized coordinate system are shown.

2) Averaging

Averaging routines need to be invoked for join-operations in newly coarsened model domains and during fine-to-coarse ghost cell transfers. For the cell-centered variables as shown in Fig. 6, this averaging step involves the four fine-resolution grid cells (dotted contours) that are entirely contained in the corresponding coarse-grid domain (dashed contour). Then the coarse-grid cell mean h^c can be determined by the weighted average

$$h^c = \frac{1}{A^c}\sum_{i=1}^{4} h_i^r A_i^r, \tag{11}$$

where A^c, A^r stand for the area of the coarse and refined spherical surface patches, and h_i^r symbolizes the fine-grid cell means in the ith grid box. This averaging strategy is mass-conservative.

3) Flux updates

In the LR96 transport algorithm, neighboring grid cells exchange flux information across cell edges to prognosticate the time evolution of the advected quantity (details in section 3). However, at a fine–coarse-grid interface the numerical fluxes on the coarse grid are not consistent with the accumulated fluxes on the fine grid. The differences result, for example, from the approximations of the subgrid distributions in the two domains. Therefore, flux corrections at fine–coarse-grid interfaces are imperative to ensure global mass conservation. Because of the constant time step at all refinement levels, the numerical fluxes are readily available at any given time. Neither temporal interpolations nor flux accumulations at the interfaces are required. By contrast, both techniques are essential for AMR approaches with subcycled time steps in refined domains (Berger and Colella 1989). Here, it is assumed that the fine-resolution fluxes are more accurate than the corresponding flux in the coarse area. Therefore, the fine-grid fluxes are averaged and consequently override the coarse-grid flux at the interface. This flux-matching strategy is schematically displayed in Fig. 7. The figure shows the locations of the coarse- and fine-grid fluxes in latitudinal (G) and longitudinal (F) direction at the mesh boundaries. The arrows point from the two fine-grid fluxes to the coarse neighboring flux that is replaced at every time step. In this 2D design, the flux-matching condition becomes

$$f^c = \frac{1}{A^c}(f_1^r A_1^r + f_2^r A_2^r), \tag{12}$$

where f symbolizes the fluxes F or G. As before, the superscripts c and r stand for the coarse and refined grid and A represents the area of a spherical surface patch. One of these surface weights is symbolically indicated by the dashed grid box for the lower-left cell. Note that each flux is considered a new cell center for the computation of the area weights that vary spatially in contrast to equivalent weights in a Cartesian setup. This flux update algorithm ensures mass conservation up to machine precision.

3. Review of the finite-volume advection algorithm

The AMR advection algorithm is built upon the LR96 finite-volume scheme, which utilizes advanced oscillation-free numerical approaches to solving the transport equation in conservation form. Details of the algorithm in spherical geometry are provided in Lin and Rood (1997) who applied the finite-volume concepts to the shallow water framework. Here, only a brief overview of the relevant transport equation in the shallow water system is given.

The model equation for the adaptive advection experiments is the conservation law

$$\frac{\partial}{\partial t}h + \nabla \cdot (h\mathbf{V}) = 0 \tag{13}$$

for the free surface height h of the shallow water system. Here, $\mathbf{V} = (u, v)^{\mathrm{T}}$ is the two-dimensional vector velocity, and $\nabla\,\cdot$ represents the horizontal divergence operator. The corresponding integral form of the conservation law can be derived when integrating Eq. (13) over the control volume Ω and time t (see also LeVeque 2002):

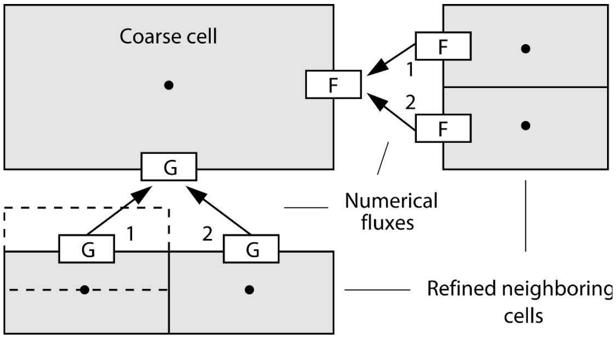


FIG. 7. Flux updates at fine–coarse model interfaces; G and F symbolize the numerical fluxes in latitudinal and longitudinal direction. Fine-grid fluxes are averaged and override the coarse-grid flux at the interface. The dashed box indicates the area weight of G in box 1.

$$\int_{t_n}^{t_{n+1}}\int_{\Omega}\left(\frac{\partial}{\partial t}h\right)d\Omega\,dt + \int_{\Omega}\int_{t_n}^{t_{n+1}}\nabla\cdot(h\mathbf{V})\,dt\,d\Omega = 0, \tag{14}$$

where n denotes the discrete time level, and $\Delta t = t_{n+1} - t_n$ stands for the duration of a time step. In the following derivation, only two-dimensional control volumes with surface areas A_Ω are considered. Eq. (14) can be rewritten as

$$\int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\overline{h}\right)dt + \frac{\Delta t}{A_\Omega}\int_{\Omega}\nabla\cdot\mathbf{F}\,d\Omega = 0, \tag{15}$$

where the overbar ($\overline{\ }$) symbolizes the spatial average over the area A_Ω and $\mathbf{F} = (1/\Delta t)\int_{t_n}^{t_{n+1}} h\mathbf{V}\,dt$ indicates the temporal average of the flux vector. Applying the Gauss divergence theorem yields

$$\int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\overline{h}\right)dt + \frac{\Delta t}{A_\Omega}\oint_{\partial\Omega}\mathbf{F}\cdot\hat{\mathbf{n}}\,dl = 0 \tag{16}$$

in which $\hat{\mathbf{n}}$ is an outward-pointing unit normal vector to the boundary $\partial\Omega$ of the control volume and dl is an infinitesimal line segment along the contour. Thus, the discrete representation of the conservation law becomes

$$\overline{h}^{n+1} = \overline{h}^{n} - \frac{\Delta t}{A_\Omega}\sum_{i=1}^{4}\mathbf{F}_i\cdot\hat{\mathbf{n}}_i l_i, \tag{17}$$

where the sum comprises the four line segments with lengths l_i that surround a rectangular 2D region; $\mathbf{F}_i$ symbolizes the time-averaged flux vectors at the cell interfaces, and $\hat{\mathbf{n}}_i$ indicates the unit normal vectors to the ith contour line. Assuming an orthogonal x–y control volume with surface area $A_{i,j} = \Delta x_{i,j}\,\Delta y_{i,j}$ and corresponding 1D numerical fluxes F and G in x and y direction, Eq. (17) is equivalent to

$$\overline{h}^{n+1}_{i,j} = \overline{h}^{n}_{i,j} - \frac{\Delta t}{A_{i,j}}[\Delta y_{i+(1/2),j}F_{i+(1/2),j} - \Delta y_{i-(1/2),j}F_{i-(1/2),j} + \Delta x_{i,j+(1/2)}G_{i,j+(1/2)} - \Delta x_{i,j-(1/2)}G_{i,j-(1/2)}] \tag{18}$$

in which the indices i, j define the gridpoint position of the cell center, and the half index represents the boundaries of the grid box. In such a Cartesian setup, the lengths of the line segments $\Delta x_{i,j} = [x_{i+(1/2),j} - x_{i-(1/2),j}]$ and $\Delta y_{i,j} = [y_{i,j+(1/2)} - y_{i,j-(1/2)}]$ are independent of the y and x direction, respectively. This is in contrast to grids in spherical geometry where $\Delta x_{i,j}$, $\Delta y_{i,j}$ and the surface area $A_{i,j}$ in Eq. (18) are substituted with the corresponding representation in spherical space:

$$\Delta x_{i,j} = a\cos\varphi_j \Delta\lambda_i, \tag{19}$$

$$\Delta y_{i,j} = a\Delta\varphi_j, \tag{20}$$

$$A_{i,j} = a^2 \int_{\lambda_{i-(1/2)}}^{\lambda_{i+(1/2)}} \int_{\varphi_{j-(1/2)}}^{\varphi_{j+(1/2)}} \cos\varphi\, d\varphi\, d\lambda = a^2[\sin\varphi_{j+(1/2)} - \sin\varphi_{j-(1/2)}]\Delta\lambda_i. \tag{21}$$

Here, $a = 6.371229 \times 10^6$ m represents the earth's radius and $\Delta\lambda_i = [\lambda_{i+(1/2)} - \lambda_{i-(1/2)}]$ and $\Delta\varphi_i = [\varphi_{j+(1/2)} - \varphi_{j-(1/2)}]$ denote the longitudinal (λ) and latitudinal (φ) grid spacings measured in radians. In the Lin–Rood advection algorithm though (Lin and Rood 1997), a slightly different flux-differencing approach has been chosen that corresponds to Eq. (18) if $A_{i,j}$ is replaced with the approximation $A_{i,j} = a^2 \cos\varphi_j\, \Delta\lambda_i\, \Delta\varphi_j$. As a result, the discrete conservation law for the cell-averaged free surface height in spherical coordinates becomes

$$\overline{h}^{n+1}_{i,j} = \overline{h}^{n}_{i,j} + F_{\text{net}} + G_{\text{net}} \tag{22}$$

with

$$F_{\text{net}} = -\frac{\Delta t}{a\cos\varphi_j \Delta\lambda_i}[F_{i+(1/2),j} - F_{i-(1/2),j}], \tag{23}$$

$$G_{\text{net}} = -\frac{\Delta t}{a\cos\varphi_j \Delta\varphi_j}[\cos\varphi_{j+(1/2)}G_{i,j+(1/2)} - \cos\varphi_{j-(1/2)}G_{i,j-(1/2)}]. \tag{24}$$

Here, G_{net} and F_{net} denote the net fluxes through the interfaces in latitudinal and longitudinal direction that solely determine the rate of change of the spatially averaged scalar field $\overline{h}$. This notation corresponds exactly to the Eqs. (4)–(5) in Lin and Rood (1997) when defining the time-averaged fluxes as $\mathcal{X}$ and $\mathcal{Y}$ and omitting the subscript "net" in the preceding equations. Note that Eq. (22) still contains a directional bias when operating splitting methods are applied. Therefore, the following directional-bias free 2D scheme is employed:

$$\overline{h}^{n+1}_{i,j} = \overline{h}^{n}_{i,j} + F_{\text{net}}\left[\overline{h}^{n}_{i,j} + \frac{1}{2}g(\overline{h}^{n}_{i,j})\right] + G_{\text{net}}\left[\overline{h}^{n}_{i,j} + \frac{1}{2}f(\overline{h}^{n}_{i,j})\right], \tag{25}$$

where g and f symbolize advection operators in latitudinal and longitudinal direction, respectively. These advective operators avoid a deformational error (details in LR96). In particular, LR96 combined a first-order Euler upwind scheme in advective form (inner operator) with higher-order finite-volume methods in flux form (outer operator). Both a second-order van Leer–type flux algorithm (van Leer 1974, 1977) and the third-order PPM scheme were implemented for the flux calculations that are both monotonic, upstream-biased, and fully conservative. For the adaptive advection experiments discussed here, the PPM approach has been chosen for the fluxes F and G. Details of the flux computations and monotonicity constraints are provided in Carpenter et al. (1990), Colella and Woodward (1984), and LR96 (with monotonicity constraint FFSL-3). Note that the scheme is in fact not strictly monotonic in two dimensions. As explained in LR96, very minor violations of the monotonicity constraint can be observed near sharp edges due to the application of purely 1D monotonicity operators. The time-stepping scheme is explicit and stable for zonal and meridional Courant numbers [Courant–Friedrichs–Lewy (CFL) criterion] $|\text{CFL}| < 1$. This restriction arises since the semi-Lagrangian extension of the LR96 algorithm is not utilized in the AMR model experiments to keep the width of the ghost cell regions small. Note that the variables u, v, and h for the advection experiment are staggered as in the Arakawa C grid with originally (prior to the AMR extension) two cell centers at the poles. For the adaptive mesh implementations though, this base C-grid configuration needed to be shifted by half a grid length in latitudinal direction. Such a shift places velocity points at the poles and avoids fixed polar mass centers that would inhibit the flexible AMR principle.

TABLE 1. Refinement levels and corresponding global grid resolutions on the sphere for a 5° × 5° base grid.

Refinement level	Resolution $\Delta\lambda$, $\Delta\varphi$	Max No. of grid points lat × lon (full grid)	Resolution Δx (km) at different positions φ			
			Equator	60°N/S	75°N/S	Near pole
0	5°	36 × 72	556.0	278.0	143.9	24.26
1	2.5°	72 × 144	278.0	139.0	72.0	6.06
2	1.25°	144 × 288	139.0	69.5	36.0	1.52
3	0.625°	288 × 576	69.5	34.8	18.0	0.38
4	0.3125°	576 × 1152	34.8	17.4	9.0	0.09

4. Results

The advection algorithm is one of the fundamental building blocks of atmospheric flow simulations. It is therefore imperative to evaluate its performance not only in the adaptive model runs but also in the non-adapted model setups. This allows comparisons of the AMR approach to both analytic reference solutions and uniform-grid model experiments. This section discusses results from three standard advection tests with increasing complexity. These are the transport of a cosine bell, the advection of a slotted cylinder and a smooth deformational flow that describes the roll-up of two vortices.

Different adaptation criteria are assessed. In general, each criterion is compared to a problem-dependent threshold value. If one or more grid points within a block exceed this user-determined limit the block is flagged for refinement. If, on the other hand, the grid points in an adapted block no longer meet the adaptation criterion the coarsening flag in the corresponding block is set. The refinement criterion is examined at each time step, which is determined dynamically during the simulation to match a chosen |CFL| = 0.95 limit. All adaptations occur consecutively until either the user-defined maximum refinement level or the initial resolution is reached. The mesh cannot be coarsened further than the initial layout. Here, the block configuration $B_x = 8$, $B_y = 6$, $N_x = 9$, $N_y = 6$ (as introduced in Fig. 1) is chosen for all test cases with a maximum refinement level between 0 and 4. Such a setup corresponds to the uniform $\Delta\lambda$, $\Delta\varphi$ grid spacings 5°, 2.5°, 1.25°, 0.625°, and 0.3125° at the individual nesting levels. They are further explained in Table 1, which gives an overview of the corresponding resolutions in spherical coordinates and physical space. The effect of the converging meridians in the latitude–longitude grid can clearly be seen at all refinement levels. This significantly reduces the physical grid spacings in polar regions and, as a consequence, the maximum allowable time step for stable computations.

All three passive advection tests are driven by prescribed wind speeds. These are reinitialized analytically whenever adaptations occur during the course of the simulation. In addition, the initial geopotential height field is reinitialized analytically if initial adaptations are requested. This is the case for the cosine bell and slotted cylinder test scenarios. During the model runs though, the geopotential height field in newly adapted blocks is initialized via the conservative averaging algorithm and the PPM-based interpolation method as discussed earlier.

a. *Error statistics*

The performance of the advection tests is quantitatively measured using the standard normalized l_1, l_2, l_∞ error norms and the height error $h_{\max}$. They are defined as in Williamson et al. (1992). For the adaptive grid runs, the following approximation to the global integral of the scalar field h is adopted:

$$I(h) = \frac{1}{4\pi}\int_0^{2\pi}\int_{-\pi/2}^{\pi/2} h(\lambda, \varphi)\cos\varphi \, d\varphi \, d\lambda \qquad (26)$$

$$\approx \frac{1}{A_{\text{sp}}}\sum_{m=1}^{N_b}\sum_{j=1}^{N_y}\sum_{i=1}^{N_x} h_{i,j,m}A_{i,j,m}. \qquad (27)$$

In particular, the integral is replaced by three nested sums that loop over the total number of blocks N_b and all grid cells. As before, N_y and N_x denote the number of latitudinal and longitudinal grid points per block and m stands for the block index. The spherical area weights $A_{i,j,m}$ are defined as in Eq. (21) and take the particular resolution $\Delta\lambda$, $\Delta\varphi$ in the mth block into account. The normalization factor $A_{\text{sp}} = 4\pi a^2$ indicates the surface area of the whole sphere. In a parallel computing setup, each processor then loops over its assigned blocks and computes a partial sum, which is collectively communicated over the network to determine the global value.

b. *Adaptive advection experiments*

1) TRANSPORT OF A COSINE BELL

The first test of the 2D adaptive finite-volume advection model is the passive transport of a cosine bell around the sphere (Williamson et al. 1992; shallow water test case 1). The advecting wind field is given by

$$u = u_0(\cos\alpha \cos\varphi + \sin\alpha \cos\lambda \sin\varphi), \tag{28}$$

$$v = -u_0 \sin\alpha \sin\lambda, \tag{29}$$

where u and v stand for the nondivergent zonal and meridional velocities with maximum wind speed $u_0 = 2\pi a/(12 \text{ days}) \approx 38.61 \text{ m s}^{-1}$. Various flow orientation angles α to the equator can be chosen. Here, $\alpha = 0°$, $\alpha = 45°$, and $\alpha = 90°$ are tested that define the advection of the cosine bell along the equator, at a 45° angle through the Tropics and midlatitudes as well as the transport across the poles.

The initial distribution of the free surface height h is defined as

$$h(\lambda, \varphi) = \begin{cases} h_0/2[1 + \cos(\pi r/R)] & \text{if} \quad r < R \\ 0 & \text{if} \quad r \geq R \end{cases}, \tag{30}$$

with radius $R = a/3$ and the peak amplitude $h_0 = 1000$ m. Here, r denotes the great circle distance

$$r = a \arccos[\sin\varphi_c \sin\varphi + \cos\varphi_c \cos\varphi \cos(\lambda - \lambda_c)] \tag{31}$$

between a position (λ, φ) and the center of the cosine bell, initially set to $(\lambda_c, \varphi_c) = (3\pi/2, 0)$. For convenience, pointwise values instead of cell averages are used for the initialization and validation. In general, the numerical scheme is expected to translate the cosine bell without any change of shape. The bell then reaches its initial position after one full revolution at the end of model day 12.

The refinement criterion for the adaptive transport of the cosine bell is a simple threshold criterion that assesses the value of the geopotential height at each grid point. In particular, if the geopotential height exceeds the user-determined limit $h \geq 53$ m the block is flagged for refinement. This sensitive value corresponds to approximately 5% of the initial peak with $\max(h) \approx 1000$ m. The threshold is chosen since the adapted blocks now tightly bind the cosine bell, which is also illustrated in Fig. 8. This keeps the total number of adapted blocks in the given example small while covering most of the traveling feature with the adaptive block structure. Any smaller threshold would trigger a wider refinement area.

Figure 8 explains the basic adaptation principle and gives insight into the load-balancing strategy on parallel computing platforms. The left column shows four snapshots of the adaptive $\alpha = 90°$ simulation at day 0, 1, 3, and 4 with three refinement levels. The adapted blocks reliably track the geopotential height field of the cosine bell as indicated by the overlaid block distribution. It can clearly be seen that the cosine bell passes over the North Pole without distortions or noise. The pole point itself with its converging grid lines is spread out in the chosen equidistant cylindrical map projection. This emphasizes the numerous grid boxes in polar regions that need to be refined for transport processes at high latitudes. They severely restrict the maximum allowable time step that obeys the $|\text{CFL}| = 0.95$ condition in polar regions. For example, in this simulation with three refinement levels and a maximum zonal wind speed of $u \approx 38.61 \text{ m s}^{-1}$ at the poles, the adaptive time step depends greatly on the position of the cosine bell. It varies between $\Delta t \approx 597.3$ s in equatorial regions and $\Delta t \approx 9.3$ s at very high latitudes. The right column in Fig. 8 displays the corresponding distribution of the adaptive blocks in a parallel execution mode. Here four processors are used and are indicated by the gray shadings. The current load-balancing strategy assumes an equal workload per block and therefore, assigns an approximately equal number of blocks to each processor. If needed, new blocks are readily redistributed after adaptations occurred. Here, the redistribution of the blocks does not take data locality issues into account. For the future, a space-filling-curve load-balancing approach (Dennis 2003; Behrens and Zimmermann 2000) is planned that improves the data locality and thereby is expected to reduce the communication across the parallel network. Note that the parallel speedups that can be achieved with the AMR library highly depend on several factors. Among them are the size of the blocks, the ratio between compute cells and ghost cells, the workload per block, and the frequency of the ghost cell updates due to the algorithmic design, the data locality of the blocks, and the performance of the network. These issues will be thoroughly discussed in a different publication.

The cosine bell is advected once around the sphere and reaches its initial position after 12 days. Then the solution can be compared to the initial conditions that serve as the true reference field. A closer examination of the final states at refinement level 1 (2.5°) and different rotation parameters α is illustrated in Fig. 9, which also presents the overlaid true solution with dotted contours. Here it is shown that the cosine bell undergoes a stretching in the flow direction which is typically observed for monotonic finite-volume advection algorithms (LR96; Nair and Machenhauer 2002; Hubbard and Nikiforakis 2003). The effect is clearly visible at this relatively coarse resolution and diminishes significantly with decreasing grid sizes (Fig. 10). As pointed out by Nair and Machenhauer (2002) the deg-

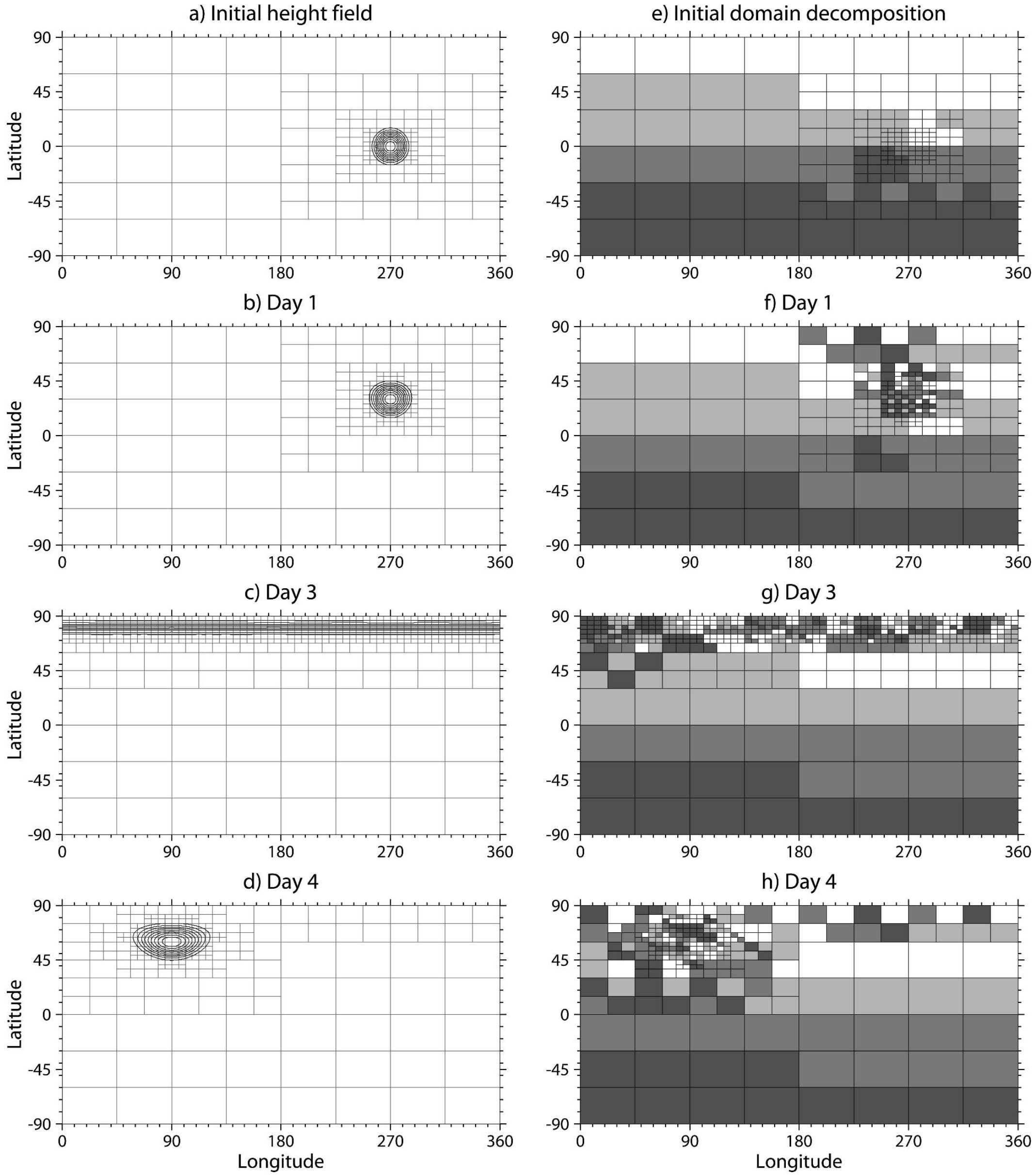


FIG. 8. Snapshots of the adaptive cosine bell advection test ($\alpha = 90°$) with three refinement levels (0.625°) at (from top to bottom) days 0, 1, 3, and 4. (left) The geopotential height field of the cosine bell with overlaid adapted blocks. The contour intervals are 100 m, and the zero contour line is omitted. (right) The distribution of the adapted blocks among four parallel processors as indicated by the gray shadings.

radation of the shape is caused by the monotonicity constraint that translates into slightly less accurate error norms in comparison to, for example, positive definite methods. Nevertheless, in practice the advantages of the monotonicity-preserving advection algorithm outweigh the slight decrease in accuracy [see also comparisons in van Leer (1977)].

Figure 10 confirms that the shape of the cosine bell after one revolution is well preserved at high resolutions. The figure depicts a sequence of model runs with increasing number of refinement levels at rotation angle $\alpha = 45°$. This transport direction represents the "worst case" scenario for the advection algorithm with underlying operator splitting approach. The results at the highest refinement levels 3 and 4 are almost indistinguishable from each other. It is also interesting to note that the cosine bells at the higher resolutions (Figs. 10c–e) no longer show the small phase error, which is visible in the coarser runs (Figs. 10a–b). The solutions then resemble the true solution very closely, which is

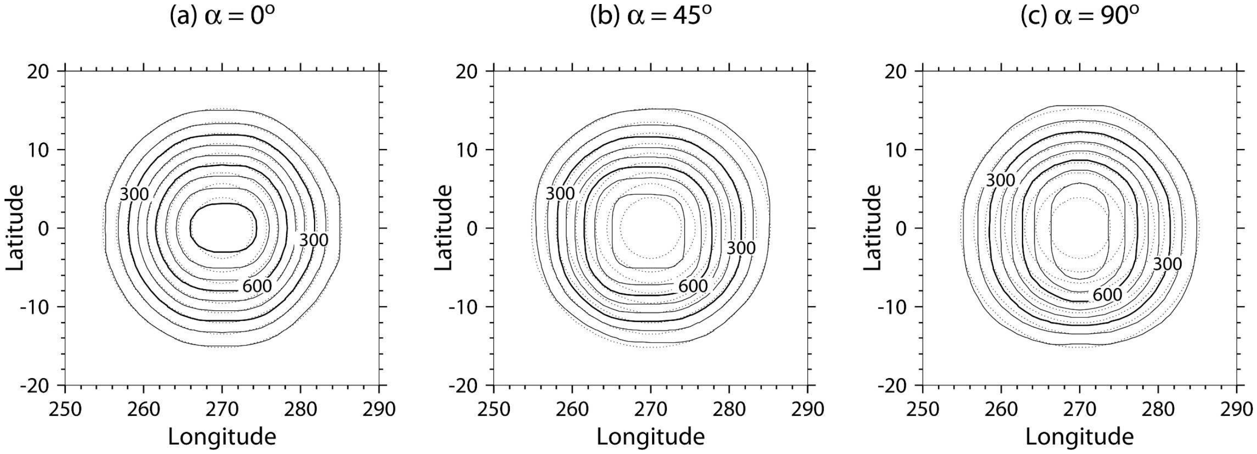


FIG. 9. Geopotential height of the cosine bell after one revolution (12 days) with one refinement level (2.5°) and different rotation angles α: (a) $\alpha = 0°$: advection along the equator, (b) $\alpha = 45°$: advection at a 45° angle, and (c) $\alpha = 90°$: advection over the poles. The analytic solution is overlaid (dotted contours). Contour intervals are 100 m, and the zero contour is omitted.

demonstrated in Fig. 10f. Figure 10f displays the difference of the cosine bell at refinement level 4 with the analytic reference state. The differences mainly develop along the edges in the 45° flow direction. In particular, the leading edge in the upper-right corner shows slightly enhanced geopotential height values, whereas the tail in the lower left corner drops below the reference state. These errors are small in comparison to the peak amplitude of the cosine bell. In general, the same type of error patterns can also be found in nonadapted model runs.

In addition, it is interesting to assess the adaptive model performance with a cross polar flow field at $\alpha = 90°$. Dynamic adaptations in polar regions are demanding since they involve many more adapted blocks close to the poles as discussed before. Nevertheless, the adapted advection scheme performs satisfactorily at all three refinement levels displayed in Fig. 11. Here, three snapshots of the cosine bell at day 2, 3, and 4 (from the bottom to the top) in a North Polar stereographic projection are shown. There are no visible distortions of the height field at any resolution as the cosine bell approaches, passes over, and leaves the North Pole. The increased resolution clearly helps preserve the shape and peak amplitude.

The overall performance of the adaptive advection tests with different rotation angles and increasing refinement levels is summarized in Table 2. The table shows not only the normalized l_1, l_2, l_∞, and $h_{\max}$ errors after one revolution but also contains information on the final peak amplitude, the minimum and maximum number of blocks during the 12-day forecast period, the number of total time steps as well as the CPU time measured on one processor of a SUN Ultra 60 workstation. In addition, the error statistics of selected uniform-grid model runs (1.25°) are listed for comparison. It can clearly be observed that the cosine bell is successfully tracked in all adapted model simulations. This is not only true for the l_1 and l_2 error norms that assess the overall shape, but also for the peak amplitudes evaluated by the l_∞ errors. The peak amplitudes improve considerably with increasing refinement levels. An approximately second order convergence rate can be found for the rotation angles $\alpha = 0°$ and $\alpha = 90°$. For $\alpha = 45°$ though, the convergence rate drops slightly below second order.

Table 2 furthermore shows that the CPU time in the adapted runs is significantly reduced in comparison to the uniform-resolution model simulations. For example, when comparing the uniform 1.25° runs with the corresponding adapted runs at refinement level 2 the speedup factors are approximately 7, 33, and 10 for the rotation angle $\alpha = 0°$, $\alpha = 45°$, and $\alpha = 90°$, respectively. The speedup is determined by two main factors. First, the number of blocks and therefore the overall workload is decreased in the adapted simulations. Second, the total number of time steps necessary for the 12-day integration is drastically reduced. The latter is due to the fact that the adaptive time step can be greatly increased if the fine grid does not cover the polar regions. In particular, this effect can be seen in the $\alpha = 45°$ case where the time step in the uniform run solely depends on the CFL restriction at the poles. This consequently leads to a rather short time step and large number of iterations. In the corresponding adapted run, the grid around the poles is mostly kept at the coarse resolution so that the time step is mainly determined by the traveling cosine bell. The latter is also true for the

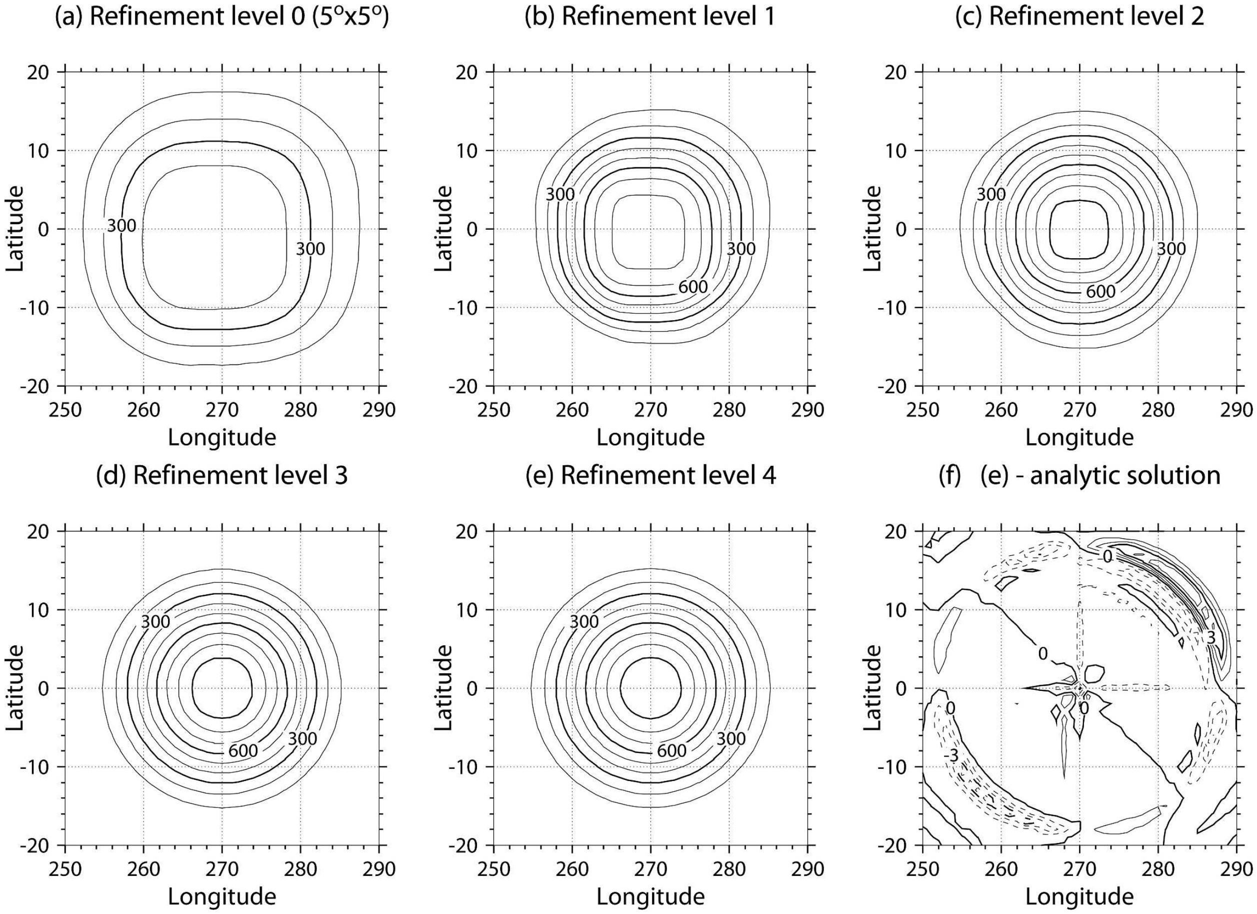
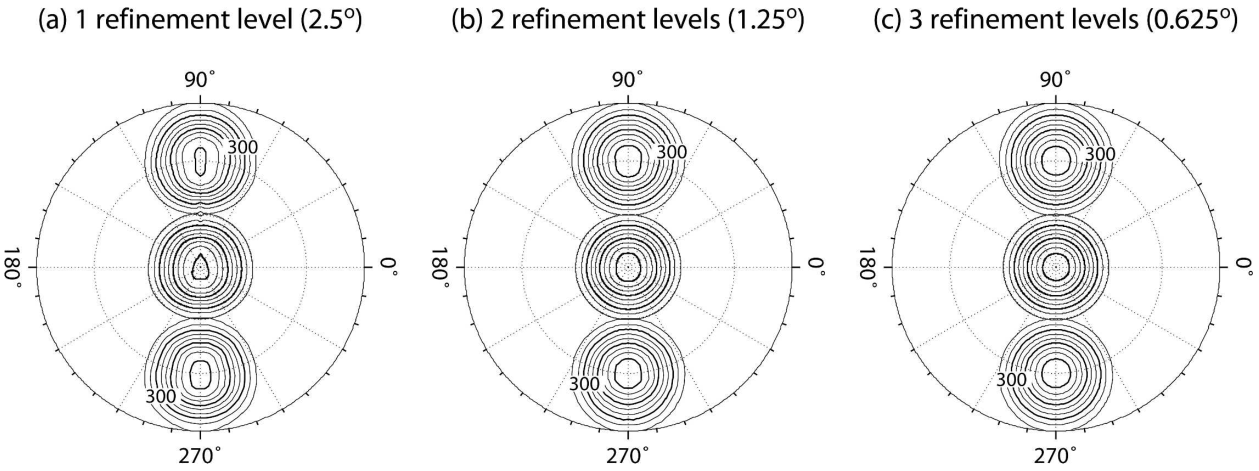


FIG. 10. (a)–(e) Geopotential height of the cosine bell and (f) height errors after one revolution (12 days) with rotation angle α = 45°. (a) No refinements (5° resolution), (b) one refinement level (2.5°), (c) two refinement levels (1.25°), (d) three refinement levels (0.625°), (e) four refinement levels (0.3125°), and (f) difference of the solution on the finest mesh [case (e)] with the analytic solution. Contour intervals are 100 m in (a)–(e) and 1 m in (f). The zero contour is omitted in (a)–(e).

$\alpha = 0°$ test case, where the time step exclusively relies on the true advective wind speeds and grid distances at the equator. Therefore, an identical number of time steps is required for both the uniform and adapted runs at the 1.25° resolution and the computational savings are exclusively due to the reduced workload.

The error norms for the adaptive simulations in Table 2 compare well to similar monotonic and conservative advection schemes presented in the literature. For example, the errors of the $\alpha = 90°$ run with one refinement level (2.5°) closely resemble the corresponding error measures for the weighted average flux (WAF) scheme in Hubbard and Nikiforakis (2003), the FFSL-3 algorithm in LR96 and the cell-integrated semi-Lagrangian method with monotonic option (CISL-M) in Nair and Machenhauer (2002) at comparable resolutions (and a slightly wider bell radius $R = a7\pi/64$). This is important to note since the latter two algorithms are semi-Lagrangian-type schemes that only require very few time steps for one revolution of the cosine bell. Both schemes finish after 256 integration steps, whereas the adaptive finite-volume model needs 4085 time steps due to the CFL restriction. Therefore, it is emphasized that the error norms match despite the large difference in the number of time steps. The sheer number of integration steps has significant implications on the model results. This has already been documented in Table 2, which shows that the error norms of the adapted runs at refinement level 2 for nonzero rotation angles are in fact slightly lower than the errors of the uniform-grid runs. The effect can be linked to the reduced number of integration steps. An even clearer example is given in Table 3, which assesses the performance of the $\alpha = 0°$ advection test with different maximum CFL numbers. As the CFL numbers decrease, the resulting number of time steps for one revolution increases, which leads to considerably less accurate results in this constant-flow test case. Here, an increased


(a) 1 refinement level (2.5°)

(b) 2 refinement levels (1.25°)

(c) 3 refinement levels (0.625°)

FIG. 11. (from the bottom to the top) Polar stereographic projections of the cosine bell transported over the North Pole (the outer circle is located at 45°N). Snapshots are taken after 2, 3, and 4 days, respectively, with refinement levels (a) 1, (b) 2, and (c) 3. Contour intervals of the geopotential height fields are 100 m, and the zero contour is omitted.

number of time steps adds numerical diffusion to the transport problem, which flattens the peak of the cosine bell. A similar time-stepping effect was also observed by Stevens and Bretherton (1996).

The time traces of the normalized l_2 and l_∞ error norms with $\alpha = 45°$ and $\alpha = 90°$ are illustrated in Fig. 12. The figure displays the evolution of the errors at three different refinement levels that show the expected decline of the solution error at finer resolutions. The norms are slightly sensitive to the rotation angle, especially in comparison to the $\alpha = 0°$ values listed in Table 2. This is in contrast to findings by Taylor et al. (1997) who utilized a high-order spectral element method on a cubed-sphere computational grid. The

TABLE 2. Error measures and statistics for the solid-body rotation of the cosine bell after one revolution (12 days) for different refinement levels and rotation angles α. The CFL number is 0.95. The number of blocks reflects the minimum and maximum during the 12-day run. The CPU time is the user time on a single processor of a SUN Ultra 60 workstation.

Base resolution	Refinement	Height error norms				Final h	No. of blocks		No. of	CPU
$\Delta\lambda$, $\Delta\varphi$	level	l_1	l_2	l_∞	$h_{\max}$	(m)	Min	Max	time steps	time (s)
$\alpha = 0°$										
1.25°	0	0.0073	0.0078	0.0107	−0.0106	984.2	768		312	230
5°	0	0.1157	0.1001	0.0949	−0.0869	838.0	48		84	4
5°	1	0.0341	0.0301	0.0317	−0.0305	949.1	54	60	156	12
5°	2	0.0097	0.0103	0.0150	−0.0106	984.2	72	84	312	32
5°	3	0.0016	0.0021	0.0044	−0.0036	995.0	174	204	612	159
5°	4	0.0003	0.0005	0.0014	−0.0012	998.4	348	384	1224	642
$\alpha = 45°$										
1.25°	0	0.0264	0.0259	0.0557	−0.0555	939.5	768		19 872	13 873
5°	0	0.5077	0.4194	0.4835	−0.4762	480.8	48		1440	64
5°	1	0.0927	0.0911	0.1525	−0.1514	830.7	54	60	1440	97
5°	2	0.0278	0.0251	0.0507	−0.0505	944.4	72	106	3333	418
5°	3	0.0088	0.0080	0.0159	−0.0159	982.8	171	210	3871	994
5°	4	0.0031	0.0030	0.0053	−0.0039	995.8	348	462	4120	2407
$\alpha = 90°$										
1.25°	0	0.0250	0.0256	0.0421	−0.0420	953.0	768		27 936	19 551
5°	0	0.4683	0.3860	0.3923	−0.3902	559.7	48		2016	97
5°	1	0.0924	0.0898	0.1253	−0.1250	856.6	60	72	4085	305
5°	2	0.0244	0.0240	0.0405	−0.0405	954.5	72	200	11 152	1885
5°	3	0.0048	0.0052	0.0125	−0.0125	986.2	192	480	33 288	15 465
5°	4	0.0010	0.0011	0.0038	−0.0038	995.9	348	1440	107 199	159 708

TABLE 3. Error measures and statistics for the solid-body rotation of the cosine bell along the equator after one revolution (12 days) with refinement level 2 (1.25°). Results are shown for different CFL numbers. The CPU time is the user time on a single processor of a SUN Ultra 60 workstation.

Base resolution	Refinement	CFL	Height error norms			Final h	Base time	No. of	CPU
$\Delta\lambda$, $\Delta\varphi$	level	number	l_1	l_2	l_∞	(m)	step (s)	time steps	time (s)
$\alpha = 0°$									
5°	2	0.95	0.0097	0.0103	0.0150	984.2	3420	312	32
5°	2	0.5	0.0170	0.0184	0.0279	967.0	1800	576	58
5°	2	0.1	0.0295	0.0275	0.0354	959.6	360	2880	275

trace of the l_∞ norm is rather noisy at low resolutions and, additionally, shows distinct spikes when the cosine bell is transported over the poles. This was also observed by Rasch (1994) and Nair and Machenhauer (2002). These spikes are not a consequence of the dynamically adaptive grid implementation, but also occur in uniform-grid model runs. As pointed out by Jakob-Chien et al. (1995) the source of the noise is the discrete sampling of the numerical and reference solution. Here the reference solution is computed analytically during the course of the integration. This leads to an occasional small increase in the peak amplitude of the reference field depending on the distance of the center to the closest grid point. This effect diminishes with increasing resolution.

2) TRANSPORT OF A SLOTTED CYLINDER

In the second example the cosine bell is replaced with a slotted cylinder. All other parameters and flow fields stay the same as discussed above. Such an advection test was originally proposed by Zalesak (1979) and has recently been applied in spherical geometry by Nair et al. (2003) and Lipscomb and Ringler (2005). The slotted cylinder exhibits very sharp nonsmooth edges in comparison to the rather smoothly varying cosine bell. Therefore, it challenges not only the numerical scheme but also the interpolation algorithm in an adaptive model simulation. The radius of the cylinder is set to $R = a \times \pi/4$ with a slot of width $a \times \pi/8$ and length $a \times 3\pi/8$. This rather wide slotted cylinder is chosen to allow grid coarsenings within the slot. As in Lipscomb and Ringler (2005), the initial height of the cylinder is set to $h = 1000$ m, whereas h is set to zero for all $r > R$ and inside the slot. The long axis of the slot is perpendicular to the equator and the cylinder is initially centered at $(\lambda_c, \varphi_c) = (3\pi/2, 0)$. The flow orientation angle $\alpha = 30°$, which avoids any grid symmetries along the trajectory path, is selected.

The model is run for 12 days, which completes a full revolution of the advected cylinder. As before, the initial conditions then serve as the true reference solution. Four refinement levels are used that start from a coarse 5° × 5° base grid. Here, a grid-scale-dependent gradient criterion is applied that flags a block for refinement if one or more grid points with grid indices (i, j) exceed the user-defined threshold

$$\xi_{i,j} = \max(|h_{i+1,j} - h_{i,j}|, |h_{i,j+1} - h_{i,j}|) \geq 10 \text{ m} \quad (32)$$

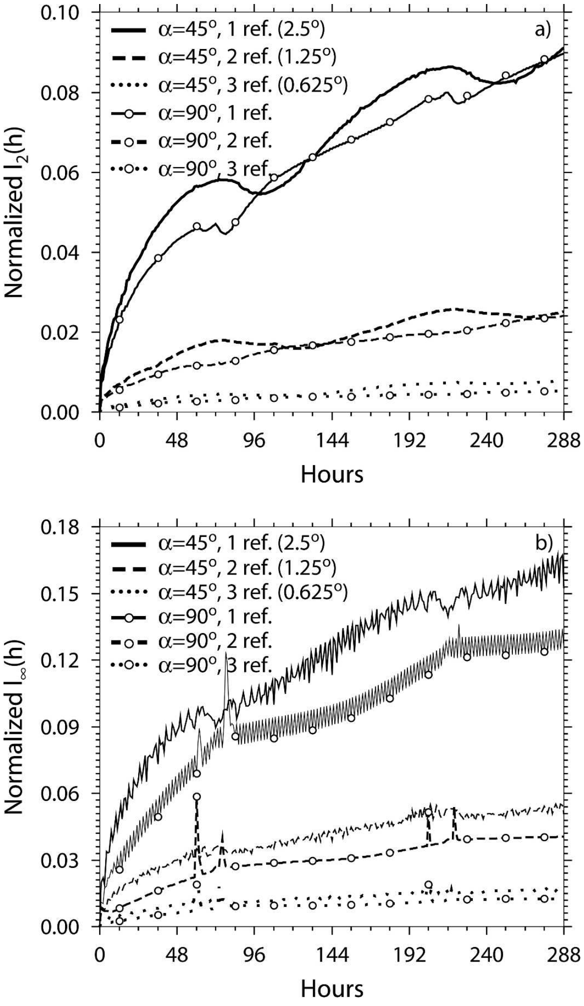


FIG. 12. Normalized (a) l_2 and (b) l_∞ height errors of the cosine bell for different refinement levels and rotation angles $\alpha = 45°$ and $\alpha = 90°$.

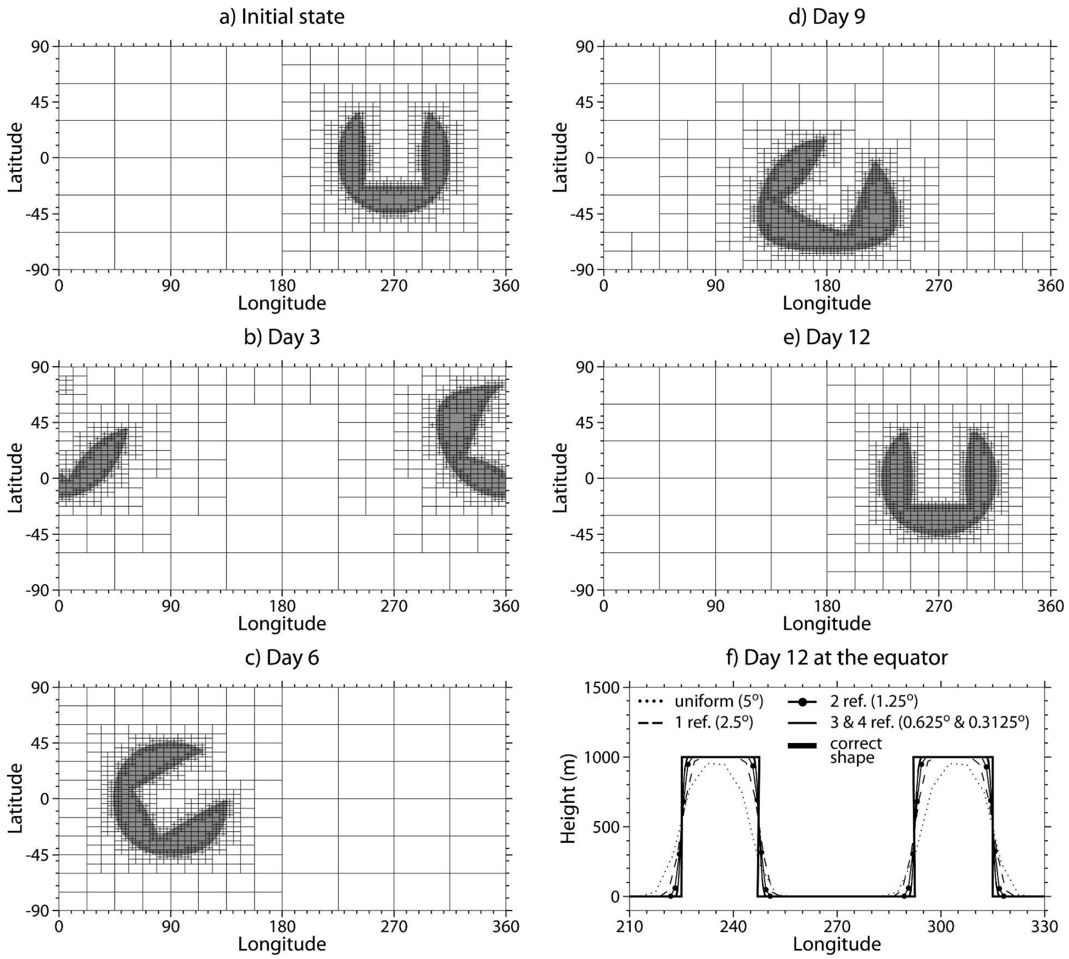


FIG. 13. Advection of a slotted cylinder with rotation angle $\alpha = 30°$ and four refinement levels (0.3125°). The positive height field is shaded in gray. (a) Initial height field ($h = 1000$ m), (b) height field at day 3, (c) day 6 (half a revolution), (d) day 9, and (e) day 12 (full revolution). (f) A cross section of h along the equator at day 12 for different refinement levels. The curves for refinement levels 3 and 4 overlay each other. The adapted blocks are guided by the resolution-dependent criterion $\xi \geq 10$ m.

for $0 \leq i \leq N_x$ and $0 \leq j \leq N_y$. This criterion evaluates the local height difference between neighboring cells, which incorporates some ghost cell information along a block boundary. A similar criterion was also formulated by Hubbard and Nikiforakis (2003) who evaluated ξ with centered differences.

Figures 13a–e show snapshots of the adaptive simulation at model day 0, 3, 6, 9, and 12. The positive height field is shaded in gray with overlaid adapted blocks. It can clearly be seen that the grid is refined along the sharp edges of the advected cylinder and kept coarse elsewhere. At this high resolution within the refined area (0.3125°) there is some minor numerical diffusion along the sharp edges. This leads to a slight broadening of the cylinder and its refined area during the course of the simulation. The broadening can also be seen in Fig. 13f, which shows a cross section of the slotted cylinder along the equator for four different refinement levels at day 12. Here, the slotted cylinder can be compared to its reference shape. The adapted grids clearly help preserve the sharpness of the edges with increased resolution. This trend seems to level off at refinement levels 3 and 4, which visually overlay each other.

The performance of the refined and corresponding uniform-resolution model runs after one revolution (day 12) is quantitatively measured in Table 4. The table lists not only the normalized l_2 and l_∞ geopotential height error norms but also gives information on the minimum and maximum number of blocks during the 12-day simulation, the total number of time steps, and the CPU time on a single AMD Opteron processor. The model runs are grouped according to their finest grid resolution. It can be seen that the l_2 errors of the adaptive and corresponding uniform-resolution runs

TABLE 4. Error measures and statistics for the solid-body rotation of the slotted cylinder after one revolution (12 days) with rotation angle $\alpha = 30°$. Model runs with different refinement levels and uniform-grid simulations are compared and grouped together according to their finest grid resolution. The CFL number is 0.95. The number of blocks reflects the minimum and maximum during the 12-day run. The CPU time is the user time on a single AMD Opteron processor.

Base resolution	Refinement	Height error norms		No. of blocks		No. of	CPU
$\Delta\lambda$, $\Delta\varphi$	level	l_2	l_∞	Min	Max	time steps	time (s)
5°	0	0.5042	0.7725	48		1152	2
2.5°	0	0.3082	0.7506	192		3744	23
5°	1	0.3076	0.7492	72	96	3061	18
1.25°	0	0.2290	0.7682	768		14 400	342
5°	2	0.2296	0.7667	174	246	6822	103
0.625°	0	0.1738	0.7532	3072		56 160	5545
5°	3	0.1770	0.7555	420	573	8667	320
0.3125°	0	0.1313	0.7899	12 288		223 488	99 924
5°	4	0.1375	0.7968	948	1299	9399	834

closely match and slowly decrease with increasing resolution. This error reduction cannot be observed for the l_∞ norms that measure the maximum deviations from the reference solution. In particular, the l_∞ norms stay almost constant. Here, the reduced convergence rate of both error norms is most likely due to the highly nonsmooth characteristics of the initial dataset. Overall, the AMR simulations show considerable speedup factors of about 17 and 120 at the two highest resolutions.

3) SMOOTH DEFORMATIONAL FLOW

An even more challenging advection test is the deformational flow (cyclogenesis) problem. This test was first suggested by Doswell (1984) and has been formulated for spherical geometries by Nair et al. (1999, 2002). Here, the smooth variant of the test that describes the simultaneous roll-up of two vortices is chosen.

The vortices are centered at the poles of a rotated spherical coordinate system (λ', φ'). This rotation is defined with respect to the regular (λ, φ) coordinates such that the North Pole of the rotated system is located at (λ_0, φ_0) in the (λ, φ) reference frame. The vortices are characterized by the normalized tangential velocity

$$V_t = \frac{3\sqrt{3}}{2}\operatorname{sech}^2(\rho')\tanh(\rho'), \tag{33}$$

with the radial distance of the vortex $\rho' = r_0 \cos\varphi'$ and the constant $r_0 = 3$. The representation of the wind speeds (u, v) in the unrotated coordinates is described in Nair et al. (1999). Note that all variables are nondimensional and that the radius of the earth a is set to 1.

The angular velocity ω' is defined by

$$\omega'(\varphi') = \begin{cases} 0 & \text{if} \quad \rho' = 0, \\ \dfrac{V_t}{\rho'} & \text{if} \quad \rho' \neq 0 \end{cases}, \tag{34}$$

which is used for the computation of the scalar field Ψ. In particular, Ψ is given by

$$\Psi(\lambda', \varphi', t) = 1 - \tanh\left[\frac{\rho'}{d}\sin(\lambda' - \omega' t)\right], \tag{35}$$

which also serves as the analytic reference solution at times $t > 0$. Here, the constant $d = 5$ determines the characteristic width of the two vortices. In addition, the parameters $(\lambda_0, \varphi_0) = (\pi/2, \pi/18)$ are chosen which point to the location (10°N, 90°E). This avoids any grid symmetry effects during the simulation.

The adaptive advection model is run for three nondimensional time units. As before, up to four refinement levels are tested that start from the coarse 5° × 5° initial grid. The chosen refinement criterion is based on the gradient of the scalar field Ψ. In particular, if the magnitude of the gradient exceeds the threshold $|\nabla\Psi| \geq 1$ at one or more grid points in a given block the block is flagged for refinement. Note that such a criterion is independent of the block resolution and therefore contrasts the resolution-sensitive nearest neighbor refinement strategy (ξ) discussed in the previous section. An example of an adapted model run with four refinement levels is displayed in Fig. 14. The figure shows the scalar field Ψ with its corresponding block distribution at times 0, 1, and $t = 3$. As it can be seen in Figs. 14a and 14d no initial adaptations are invoked. The first refinements are triggered shortly before $t = 1$ is reached. This is illustrated in Fig. 14e, which only exhibits two of the four maximum refinement levels. The missing two refinement stages are invoked imme-

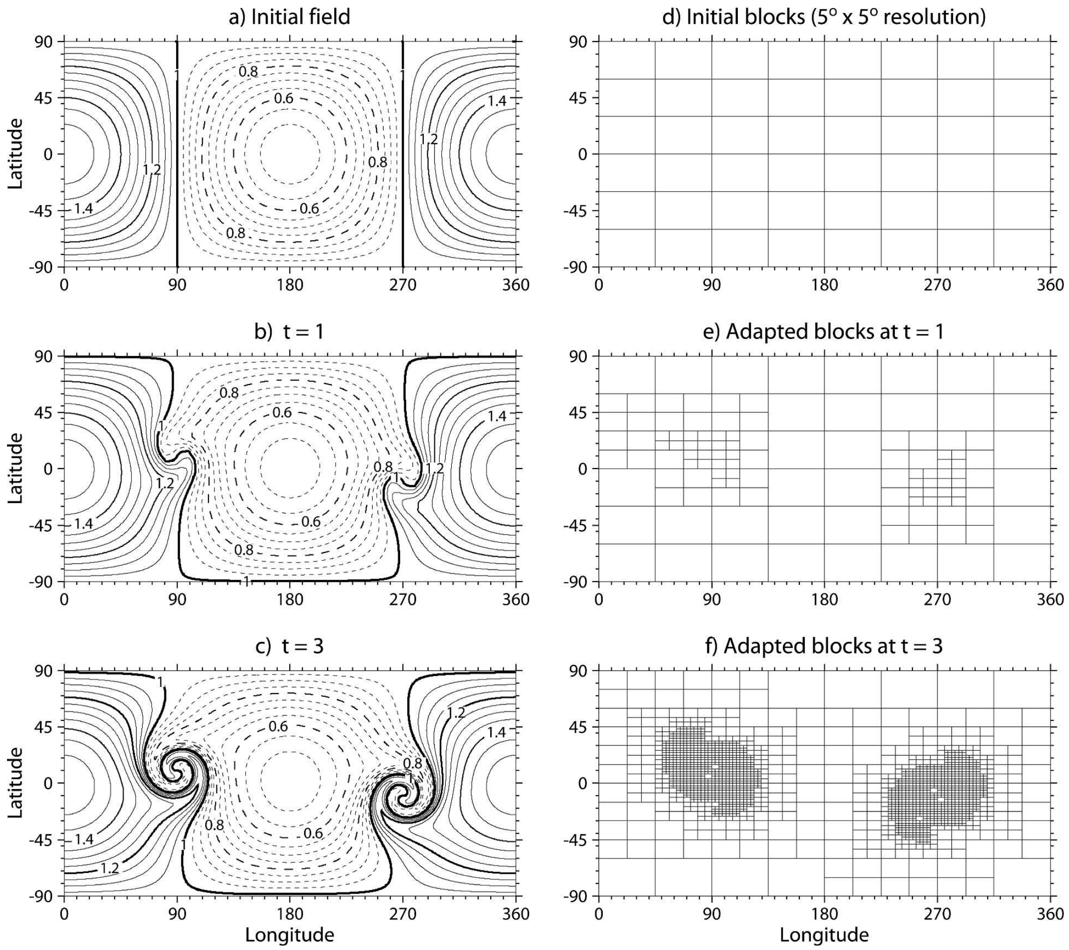


FIG. 14. Smooth deformational vortex test case with four refinement levels (0.3125°). Shown on the left are the scalar field Ψ at the nondimensional times (a) $t = 0$, (b) $t = 1$, and (c) $t = 3$. (d)–(f) The corresponding adapted blocks that are steered by the nondimensional gradient criterion $|\nabla\Psi| \geq 1$.

diately after this time stamp. At $t = 3$ the two vortices have matured and developed very sharp gradients that are now covered by an extended refined area. Overall, the evolving roll-up of the scalar is well captured by the adaptive model simulation. No noise or distortions at fine–coarse-grid boundaries are visible.

A quantitative comparison of the refined and corresponding uniform model runs is presented in Table 5. The table lists the normalized l_2 and l_∞ error norms of the scalar field Ψ, an overview of the CPU times on a single AMD Opteron processor as well as the number of blocks and time steps at $t = 3$. As before, the model runs are grouped according to their finest grid resolution. It is interesting to note that the coarse 5° base-resolution runs only show small improvements in the error statistics when refined up to a refinement level of 4. On the contrary, the errors slightly increase at refinement level 4. It suggests that the initial errors on the coarse mesh (until $t = 1$) cannot be successfully recovered by later refinement regions that are initialized via interpolations from the coarser mesh. Therefore, even an AMR run must reasonably represent the solution on the coarsest mesh if accuracy is expected to be gained in the refined areas. A good compromise between accuracy and computational costs are the 1.25° model simulations with one or two refinement levels. They show the smallest error measures even in comparison to the uniform high-resolution model runs 0.625° and 0.3125°. This is mainly attributable to the reduced number of integration steps that also lead to the considerable speedup factors of ≈8.6 and ≈108, respectively.

5. Conclusions

A spherical 2D AMR technique has been applied to a revised version of the conservative and monotonic Lin and Rood (1996) advection algorithm. The adaptive grid design is based on two building blocks: a block-structured data layout and a spherical AMR grid library

Table 5. Error measures and statistics for the smooth deformational flow test case at $t = 3$. Model runs with different refinement levels and uniform-grid simulations are compared and grouped together according to their finest grid resolution. The CFL number is 0.95. The CPU time is the user time on a single AMD Opteron processor.

Base resolution	Refinement	Error norms		No. of	No. of	CPU
$\Delta\lambda$, $\Delta\varphi$	level	$l_2(\Psi)$	$l_\infty(\Psi)$	blocks	time steps	time (s)
5°	0	0.7672×10^{-2}	0.4488×10^{-1}	48	15	0.5
2.5°	0	0.1718×10^{-2}	0.9974×10^{-2}	192	33	1.6
5°	1	0.2694×10^{-2}	0.1347×10^{-1}	84	23	0.9
1.25°	0	0.5640×10^{-3}	0.4031×10^{-2}	768	126	14
2.5°	1	0.7170×10^{-3}	0.3427×10^{-2}	294	49	3.1
5°	2	0.1917×10^{-2}	0.7226×10^{-2}	240	41	2.2
0.625°	0	0.3273×10^{-3}	0.1688×10^{-2}	3072	497	163
1.25°	1	0.3154×10^{-3}	0.1269×10^{-2}	1089	126	19
2.5°	2	0.4789×10^{-3}	0.1933×10^{-2}	699	86	8.0
5°	3	0.1891×10^{-2}	0.8465×10^{-2}	672	74	6.6
0.3125°	0	0.2904×10^{-3}	0.1021×10^{-2}	12 288	1978	4996
0.625°	1	0.2560×10^{-3}	0.9501×10^{-3}	4164	497	229
1.25°	2	0.2044×10^{-3}	0.8627×10^{-3}	2280	188	46
2.5°	3	0.4385×10^{-3}	0.1841×10^{-2}	1920	160	33
5°	4	0.1901×10^{-2}	0.9168×10^{-2}	1902	141	30

for parallel computer architectures. The library contains special provisions for ghost cell exchanges in polar regions and supports both static and dynamic grid adaptations. Furthermore, it has built-in parallel communication and load-balancing support, which reduces the development time for parallel AMR applications. The latter is especially important for more complex, nonlinear AMR models in 3D.

Three advection examples with increasing complexity have been tested. These include the advection of a cosine bell around the sphere at different rotation angles, the transport of a slotted cylinder with sharp edges, and a smooth deformational cyclogenesis test case that describes the roll-up of two vortices. Up to four refinement levels have been tested with a minimal mesh spacing of 0.3125°. The adaptations were guided by user-defined adaptation criteria. In particular, simple height- and gradient-based refinement strategies were chosen that triggered refinements whenever the problem-specific threshold values were reached. All three test examples showed that the chosen features of interest were reliably detected and tracked by the refinement regions. The additional resolution clearly helped preserve the shape and amplitude of the transported field while saving computing resources in comparison to uniform-grid model runs. Overall, the adaptive simulations demonstrate that AMR might be a viable option for atmospheric transport schemes. For practical applications, the adaptation criterion can then be tailored toward the specific advected quantities and flow conditions.

The AMR block design is readily applicable to nonlinear model configurations that can utilize a block-data structure on the sphere. This research effort is therefore a step toward a statically and dynamically adaptive GCM that can focus its resolution on user-determined regions or features of interest. The refined domains could, for example, include static adaptations in mountainous terrain or dynamic adaptations for cyclones or convective regions. Tests with an adaptive nonlinear 2D and 3D version of the finite-volume dynamical core have already been successfully performed and will be discussed in a follow-up paper.

Whether adaptive mesh approaches for climate and weather research will prevail in the future will crucially depend on two major aspects. First, it must be shown that adaptive atmospheric modeling is not just feasible, but also accurate with respect to the resulting flow patterns and furthermore, capable of detecting the features of interest reliably in 3D setups. Second, adaptive model simulations must also be computationally less expensive than comparable uniform high-resolution runs. We argue that both goals are within reach for future atmospheric model generations.

Acknowledgments. We thank S.-J. Lin (Geophysical Fluid Dynamics Laboratory, Princeton, New Jersey) and K. Yeh (NASA Goddard Space Flight Center, Greenbelt, Maryland) for their support and discussions about the Lin–Rood advection algorithm. We also wish to thank the two anonymous reviewers for their suggestions and helpful comments. This work was supported by NASA Headquarters under the Earth System Science Fellowship Grant NGT5-30359. In addition, partial funding was provided by the Department of Energy under the SciDAC Grant DE-FG02-01ER63248.

REFERENCES

Bacon, D. P., and Coauthors, 2000: A dynamically adapting weather and dispersion model: The Operational Multiscale Environment Model with Grid Adaptivity (OMEGA). *Mon. Wea. Rev.,* **128,** 2044–2076.

Behrens, J., 1996: An adaptive semi-Lagrangian advection scheme and its parallelization. *Mon. Wea. Rev.,* **124,** 2386–2395.

——, and J. Zimmermann, 2000: Parallelizing an unstructured grid generator with a space-filling curve approach. *Euro-Par 2000, Lecture Notes in Computer Science,* Vol. 1900, A. Bode, Ed., Springer-Verlag, 815–823.

——, K. Dethloff, W. Hiller, and A. Rinke, 2000: Evolution of small-scale filaments in an adaptive advection model for idealized tracer transport. *Mon. Wea. Rev.,* **128,** 2976–2982.

Berger, M., and J. Oliger, 1984: Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.,* **53,** 484–512.

——, and P. Colella, 1989: Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.,* **82,** 64–84.

Blayo, E., and L. Debreu, 1999: Adaptive mesh refinement for finite-difference ocean models: First experiments. *J. Phys. Oceanogr.,* **29,** 1239–1250.

Boybeyi, Z., N. N. Ahmad, D. P. Bacon, T. J. Dunn, M. S. Hall, P. C. S. Lee, and R. A. Sarma, 2001: Evaluation of the Operational Multiscale Environment Model with Grid Adaptivity against the European tracer experiment. *J. Appl. Meteor.,* **40,** 1541–1558.

Bryan, G. H., J. C. Wyngaard, and J. M. Fritsch, 2003: Resolution requirements for the simulation of deep moist convection. *Mon. Wea. Rev.,* **131,** 2394–2416.

Carpenter, R. L., K. K. Droegemeier, P. R. Woodward, and C. E. Hane, 1990: Application of the Piecewise Parabolic Method (PPM) to meteorological modeling. *Mon. Wea. Rev.,* **118,** 586–612.

Colella, P., and P. R. Woodward, 1984: The Piecewise Parabolic Method (PPM) for gas-dynamical simulations. *J. Comput. Phys.,* **54,** 174–201.

Dennis, J. M., 2003: Partitioning with space-filling curves on the cubed-sphere. *Proc. Int. Parallel and Distributed Processing Symposium (IPDPS),* Nice, France, IEEE/ACM, CD-ROM, 269a.

Doswell, C. A., 1984: A kinematic analysis associated with a nondivergent flow. *J. Atmos. Sci.,* **41,** 1242–1248.

Fox-Rabinovitz, M. S., G. L. Stenchikov, M. J. Suarez, and L. L. Takacs, 1997: A finite-difference GCM dynamical core with a variable-resolution stretched grid. *Mon. Wea. Rev.,* **125,** 2943–2968.

Fulton, S. R., 1997: A comparison of multilevel adaptive methods for hurricane track prediction. *Electron. Trans. Numer. Anal.,* **6,** 120–132.

——, 2001: An adaptive multigrid barotropic tropical cyclone track model. *Mon. Wea. Rev.,* **129,** 138–151.

Gombosi, T. I., and Coauthors, 2004: Solution adaptive MHD for space plasmas: Sun-to-Earth simulations. *Comput. Sci. Eng.,* **6,** 14–35.

Hubbard, M. E., and N. Nikiforakis, 2003: A three-dimensional, adaptive, Godunov-type model for global atmospheric flows. *Mon. Wea. Rev.,* **131,** 1848–1864.

Iselin, J. P., J. M. Prusa, and W. J. Gutowski, 2002: Dynamic grid adaptation using the MPDATA scheme. *Mon. Wea. Rev.,* **130,** 1026–1039.

Jablonowski, C., 2004: Adaptive grids in weather and climate modeling. Ph.D. dissertation, University of Michigan, Ann Arbor, 292 pp.

——, M. Herzog, J. E. Penner, R. C. Oehmke, Q. F. Stout, and B. van Leer, 2004: Adaptive grids for weather and climate models. *Proc. Seminar on Recent Developments in Numerical Methods for Atmosphere and Ocean Modeling,* Reading, United Kingdom, ECMWF, 233–250.

Jakob-Chien, R., J. J. Hack, and D. L. Williamson, 1995: Spectral transform solutions to the shallow water test set. *J. Comput. Phys.,* **119,** 164–187.

Kessler, M., 1999: Development and analysis of an adaptive transport scheme. *Atmos. Environ.,* **33,** 2347–2360.

LeVeque, R. J., 2002: *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press, 558 pp.

Lin, S.-J., and R. B. Rood, 1996: Multidimensional flux-form semi-Lagrangian transport scheme. *Mon. Wea. Rev.,* **124,** 2046–2070.

——, and ——, 1997: An explicit flux-form semi-Lagrangian shallow water model on the sphere. *Quart. J. Roy. Meteor. Soc.,* **123,** 2477–2498.

Lipscomb, W. H., and T. D. Ringler, 2005: An incremental remapping transport scheme on a spherical geodesic grid. *Mon. Wea. Rev.,* **133,** 2335–2350.

MacNeice, P., K. M. Olson, C. Mobarry, R. deFainchtein, and C. Packer, 2000: PARAMESH: A parallel adaptive mesh refinement community toolkit. *Comput. Phys. Comm.,* **126,** 330–354.

Nair, R., and B. Machenhauer, 2002: The mass-conservative cell-integrated semi-Lagrangian advection scheme on the sphere. *Mon. Wea. Rev.,* **130,** 649–667.

——, J. Côté, and A. Stanisforth, 1999: Cascade interpolation for semi-Lagrangian advection over the sphere. *Quart. J. Roy. Meteor. Soc.,* **125,** 1445–1468.

——, J. S. Scroggs, and F. H. M. Semazzi, 2002: Efficient conservative global transport schemes for climate and atmospheric chemistry models. *Mon. Wea. Rev.,* **130,** 2059–2073.

——, ——, and ——, 2003: A forward-trajectory global semi-Lagrangian transport scheme. *J. Comput. Phys.,* **190,** 275–294.

Odman, M. T., R. Mathur, K. Alapaty, R. K. Srivastava, D. S. McRae, and R. J. Yamartino, 1997: Nested and adaptive grids for multiscale air quality modeling. *Next Generation Environmental Models and Computational Methods,* G. Delic and M. F. Wheeler, Eds., Society for Industrial and Applied Mathematics, 59–68.

Oehmke, R. C., 2004: High performance dynamic array structures. Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 93 pp.

——, and Q. F. Stout, 2001: Parallel adaptive blocks on a sphere. *Proc. 10th Conf. on Parallel Processing for Scientific Computing,* Portsmouth, VA, SIAM, CD-ROM.

Prusa, J. M., and P. K. Smolarkiewicz, 2003: An all-scale anelastic model for geophysical flows: Dynamic grid deformation. *J. Comput. Phys.,* **190,** 601–622.

Rančić, M., 1992: Semi-Lagrangian piecewise biparabolic scheme for two-dimensional horizontal advection of a passive scalar. *Mon. Wea. Rev.,* **120,** 1394–1406.

Rasch, P., 1994: Conservative shape-preserving two-dimensional transport on a spherical reduced grid. *Mon. Wea. Rev.,* **122,** 1337–1350.

Sarma, A., N. Ahmad, D. P. Bacon, Z. Boybeyi, T. J. Dunn, M. S. Hall, and P. C. S. Lee, 1999: Application of adaptive grid refinement to plume modeling. *Air Pollution VII,* C. A. Brebbia, M. Jacobson, and H. Powell, Eds., Computational Mechanics Publications, 59–68. [Also available from WIT Press as *Advances in Air Pollution,* Vol. 7.]

Skamarock, W. C., 1989: Truncation error estimates for refinement criteria in nested and adaptive models. *Mon. Wea. Rev.,* **117,** 872–886.

——, and J. B. Klemp, 1993: Adaptive grid refinements for two-dimensional and three-dimensional nonhydrostatic atmospheric flow. *Mon. Wea. Rev.,* **121,** 788–804.

——, J. Oliger, and R. L. Street, 1989: Adaptive grid refinements for numerical weather prediction. *J. Comput. Phys.,* **80,** 27–60.

Srivastava, R. K., D. S. McRae, and M. T. Odman, 2000: An adaptive grid algorithm for air-quality modeling. *J. Comput. Phys.,* **165,** 437–472.

Stevens, D. E., and S. Bretherton, 1996: A forward-in-time advection scheme and adaptive multilevel flow solver for nearly incompressible atmospheric flows. *J. Comput. Phys.,* **129,** 284–295.

Stout, Q. F., D. L. DeZeeuw, T. I. Gombosi, C. P. T. Groth, H. G. Marshall, and K. G. Powell, 1997: Adaptive blocks: A high-performance data structure. *Proc. SC1997,* San Jose, CA, ACM/IEEE, CD-ROM.

Taylor, M., J. Tribbia, and M. Iskandarani, 1997: The spectral element method for the shallow water equations on the sphere. *J. Comput. Phys.,* **130,** 92–108.

Temperton, C., 2004: Horizontal representation by double Fourier series on the sphere. *Proc. Seminar on Recent Development in Numerical Methods for Atmosphere and Ocean Modeling,* Reading, United Kingdom, ECMWF, 135–137.

Tomlin, A., M. Berzins, J. Ware, J. Smith, and M. J. Pilling, 1997: On the use of adaptive gridding methods for modelling chemical transport from multi-scale sources. *Atmos. Environ.,* **31,** 2945–2959.

van Leer, B., 1974: Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *J. Comput. Phys.,* **14,** 361–370.

——, 1977: Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *J. Comput. Phys.,* **23,** 276–299.

——, 1985: Upwind-difference methods for aerodynamic problems governed by the Euler equations. *Lectures Appl. Math.,* **22,** 327–336.

Veldman, A. E. P., and R. W. C. P. Verstappen, 1998: Symmetry-conserving discretization with application to the simulation of turbulent flow. *Numerical Methods for Fluid Dynamics VI,* M. J. Baines, Ed., Will Print, 539–545.

Williamson, D. L., J. B. Drake, J. J. Hack, R. Jakob, and P. N. Swarztrauber, 1992: A standard test set for numerical approximations to the shallow water equations in spherical geometry. *J. Comput. Phys.,* **102,** 211–224.

Zalesak, S. T., 1979: Fully multidimensional flux-corrected transport algorithms for fluids. *J. Comput. Phys.,* **31,** 335–362.