Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 190 (2003) 275–294
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A forward-trajectory global semi-Lagrangian transport scheme

Ramachandran D. Nair [1], Jeffrey S. Scroggs *, Frederick H.M. Semazzi

Department of Marine, Earth and Atmospheric Sciences, North Carolina State University, Raleigh, NC 27695-8208, USA
Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8208, USA

Received 1 April 2002; received in revised form 12 May 2003; accepted 13 May 2003

Abstract

A forward-trajectory semi-Lagrangian scheme for advection on the surface of the sphere is proposed. The advection scheme utilizes the forward (downstream) trajectory originating at Eulerian grid points and cascade interpolation, a sequence of 1D interpolations, to transfer data from the downstream Lagrangian points to the Eulerian points. A new and more accurate algorithm determines pole values. The resulting forward-trajectory semi-Lagrangian scheme can easily incorporate high-order trajectory integration methods. This avoids the standard iterative process in a typical backward-trajectory scheme. Two third-order accurate schemes and a second-order accurate scheme are presented. A mass-conservative version of the forward-trajectory semi-Lagrangian scheme is also derived within the cascade interpolation framework. Mass from a Lagrangian cell is transferred to the corresponding Eulerian cell with two 1D remappings through an intermediate cell system. Mass in the polar region is redistributed by way of an efficient local approximation. The resulting scheme is globally conservative, but restricted to meridional Courant number, $C_\theta \leqslant 1$.

Keywords: Semi-Lagrangian advection; Spherical geometry; Forward trajectory; Cascade interpolation; Mass conservation; Runge–Kutta method

1. Introduction

Semi-Lagrangian (SL) advection combined with semi-implicit time-stepping was introduced in meteorology by Robert [1,2] nearly two decades ago. The method has been applied at scales of motion ranging from convection to the atmospheric general circulation [3,4]. Many current operational numerical weather prediction models [3,5] implement SL advection, due to the inherent accuracy and less restrictive time step. Recently, conservative SL transport schemes have been considered in climate modeling [6], where long time integration demands computationally efficient transport algorithms.

* Corresponding author.
E-mail addresses: rnair@ucar.edu (R.D. Nair), scroggs@unity.ncsu.edu (J.S. Scroggs), fred_semazzi@ncsu.edu (F.H.M. Semazzi).
[1] Present address: Scientific Computing Division, NCAR, Boulder, CO 80305, USA.

0021-9991/$ - see front matter
doi:10.1016/S0021-9991(03)00274-2

Semi-Lagrangian methods trace fluid motion by following particle paths or trajectories [7,37]. The equation governing trajectories is a nonlinear ordinary differential equation (ODE) and can be solved using a variety of numerical methods. Our method solves the equation forward in time, with the departure point location as the initial condition, and finds the downstream arrival point. Note that upstream and downstream refer to integrating a particle trajectory backward or forward in time, respectively. Traditional SL methods solve the trajectory equation backward in time, from the arrival point, by using a second-order accurate implicit midpoint method requiring several iterations [1,7,16]. This iterative procedure is computationally expensive [9,19]. Moreover, backward-trajectory schemes do not easily lend themselves to the application of high-order methods [10]. The iterative process can be avoided by explicit methods. Both upstream and downstream treatments require interpolation between regularly spaced Eulerian points and (possibly distorted) Lagrangian points. Thus, accuracy is dictated by the accuracy of both trajectory and interpolation calculations. In this paper, an explicit Taylor series method is compared with a Runge–Kutta scheme for accuracy and computational efficiency.

The choice of interpolation order in SL advection schemes strongly impacts computational efficiency. Cubic interpolation is commonly used [7], yielding fourth-order spatial accuracy. Two difficulties with multi-dimensional polynomial interpolation are the number of operations required and the lack of formal conservation. The cascade interpolation scheme introduced by Purser and Leslie [8] reduces the computational cost without degrading accuracy. Cascade schemes employ a sequence of one-dimensional (1D) interpolations, thus reducing computational overhead. For example, a typical tensor product interpolation in 3D requires $\mathcal{O}(N^3)$ operations per grid point per field, where N is the order of accuracy; whereas a cascade scheme requires only $\mathcal{O}(N)$ operations [8]. Cascade interpolation can be used in either forward- or backward-trajectory SL advection, as shown in [17]. Recently, Nair et al. [17,18] reduced the computational overhead in intermediate grid generation and extended the cascade procedure to spherical geometry.

Another important feature of cascade interpolation is the use of forward (downstream) trajectories originating on the Eulerian grid and terminating on the flow-distorted Lagrangian grid [8,17]. Forward-trajectory SL (hereafter referred to as FSL) schemes have the same computational complexity as traditional SL schemes. Purser and Leslie [9–11] further demonstrated that a high-order, SL scheme can be applied in the context of a forward-trajectory approach in Cartesian geometry.

The cascade interpolation algorithm developed by Nair et al. [18], computes the intersection of Lagrangian longitudes and the Eulerian latitudes as intermediate grid points. In [18], intermediate grid lines in the θ-direction are formed by "wrapping a (Lagrangian) longitude all around the globe," and grid lines in the λ-direction are Eulerian latitudes. To form such a grid system, the number of longitudes m must be even. A unique pole value is determined by an averaging procedure. This approach may not be suitable for a more general grid system. However, the global FSL scheme considered here does not have such limitations. Moreover, pole values are determined by a more accurate procedure.

Mass-conservation is an important issue for climate and atmospheric chemistry models. Traditional SL advection schemes are not inherently conservative. This is a major disadvantage for constituent transport in climate models. However, SL schemes can be made mass-conservative with a finite-volume approach (see [10,14,27,28], especially [30] where the incompressible Navier–Stokes equations are discussed). Our conservative version of the FSL scheme adopts the 1D piecewise parabolic method (PPM) of [35] as a basic component in the cascade interpolation framework. Spherical geometry requires special treatment of the poles and several mass-conservative SL methods have been developed for spherical geometry [15,34,26,29,25], suitable for global climate modeling. Here, we present a globally conservative FSL scheme, following some of the ideas used in the recent work of Nair et al. [26].

In this paper, we focus on three major aspects of the global FSL scheme. First, we consider two third-order numerical methods to solve the trajectory equations. Following that, we present the development of a general FSL scheme on the sphere based on cascade interpolation. Finally, the development of a mass-conservative version of the global FSL scheme (hereafter referred to as FSLc) is presented in the cascade framework.

2. Forward-trajectory SL advection

In FSL schemes, trajectories originate from regularly spaced Eulerian grid points at time t, and termini (arrival points) are located at the flow-distorted Lagrangian mesh at time $t + \Delta t$. Fig. 1 depicts two-dimensional (2D) forward-trajectory advection of a grid cell, where arrival and departure points are marked by open and closed circles, respectively. The shaded region shows a departure cell (rectangular) and the corresponding arrival cell (polygon) formed by the arrival and departure points, respectively.

The advection equation for a scalar field $F(\mathbf{x}, t)$ is written as

$$\frac{\mathrm{d}F}{\mathrm{d}t} \equiv \frac{\partial F}{\partial t} + (\mathbf{u} \cdot \nabla) F = 0, \tag{1}$$

where $F(\mathbf{x}, t) = F(\mathbf{x}(t_0), t_0)$ is the initial condition. The position vector $\mathbf{u}$ is determined from the velocity vector $\mathbf{u}(\mathbf{x}, t)$ by integrating the trajectory equation,

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}(\mathbf{x}, t). \tag{2}$$

Eq. (1) implies that the value of F is constant along trajectories described by (2).

Here, a two-time-level forward SL method integrates the material (substantial) derivative along the forward trajectory originating at the grid point $(\mathbf{x}_0, t)$ and terminating at the Lagrangian point $(\mathbf{x}^*, t + \Delta t)$. The location of Lagrangian points can be estimated by approximating the solution of the ordinary differential equation (2) using numerical methods, such as those discussed in [24]. Integrating in time along the trajectory governed by Eq. (2) with initial condition $\mathbf{x}(t_0) = \mathbf{x}_0$ results in the semi-discrete form

$$\frac{F(\mathbf{x}^*, t + \Delta t) - F(\mathbf{x}_0, t)}{\Delta t} = 0. \tag{3}$$

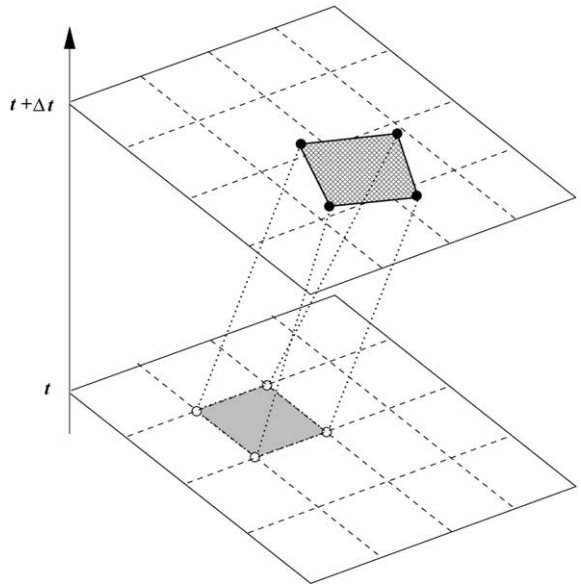

Fig. 1. A schematic of forward-trajectory semi-Lagrangian advection. Trajectories are dotted lines originating at Eulerian grid points and terminating at Lagrangian points at time-level t and $t + \Delta t$, respectively. Eulerian grids at both time-levels are depicted as dashed lines. Departure (Eulerian) points are marked by open circles and arrival (Lagrangian) points are marked by filled circles.

Generally, the Lagrangian grid is curvilinear and the arrival points (Lagrangian points) do not fall on the regular Eulerian grid intersections.

In order to formulate a consistent forward time-integration scheme, the value of the advected field $F(\mathbf{x}^*, t+\Delta t)$ at arrival points is interpolated onto the Eulerian grid. This process is repeated at every time step, distinguishing semi-Lagrangian methods from Lagrangian methods. Accurate transfer of data from the flow-distorted Lagrangian mesh to the fixed Eulerian mesh is accomplished by using cascade interpolation.

2.1. Computation of the forward trajectories

The choice of trajectory integration scheme has fundamental implications for the accuracy and efficiency of SL methods. The traditional SL scheme employs a backward trajectory method using an implicit $\mathcal{O}(\Delta t^2)$ iterative algorithm [7,16,21]. Three to five iterations are typically required to converge [7,19]. McGregor [19] introduced a high-order accurate and efficient explicit integration method based on a Taylor series expansion. This method was also shown to be accurate when applied to unstructured spherical geodesic grids [23]. We will compare McGregor's method (hereafter referred to as Ty3) with Runge–Kutta methods in the FSL scheme context.

2.1.1. McGregor's method

McGregor's method [19] uses a Taylor series to approximate Eq. (2) and is described here for completeness. The Taylor series expansion of the arrival point $\mathbf{x}(t+\Delta t)$ about the departure point $\mathbf{x}(t)$ along the trajectory is

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \sum_{v=1}^{N} \frac{(\Delta t)^v}{v!} \frac{\mathrm{d}^v \mathbf{x}(t)}{\mathrm{d}t^v} + \mathcal{O}(\Delta t^{N+1}). \tag{4}$$

The time derivative is approximated as

$$\frac{\mathrm{d}}{\mathrm{d}t} \approx \hat{\mathbf{u}} \cdot \nabla,$$

where

$$\hat{\mathbf{u}} = \mathbf{u}\left(\mathbf{x}(t), t + \frac{\Delta t}{2}\right)$$

is the time-centered wind field.

Computation of trajectories on the surface of the sphere can be performed by using 3D Cartesian (x, y, z) coordinates. For a sphere of radius $a = 1$,

$$\left.\begin{aligned} x &= \cos\theta \cos\lambda, \\ y &= \cos\theta \sin\lambda, \\ z &= \sin\theta. \end{aligned}\right\} \tag{5}$$

The corresponding inverse equations are,

$$\left.\begin{aligned} \lambda &= \tan^{-1}(y/x), \\ \theta &= \sin^{-1}(z). \end{aligned}\right\} \tag{6}$$

The Cartesian (x, y, z) coordinate values vary smoothly across the poles, unlike longitude-latitude (λ, θ) coordinates.

Departure points are located at the grid points $\mathbf{x} = (x, y, z)$ for the FSL scheme, so the velocity components determined from (5), are

$$\mathbf{u}(\mathbf{x}, t) = \frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \left(\frac{\mathrm{d}x}{\mathrm{d}t}, \frac{\mathrm{d}y}{\mathrm{d}t}, \frac{\mathrm{d}z}{\mathrm{d}t}\right) = (-u\sin\lambda - v\sin\theta\cos\lambda, u\cos\lambda - v\sin\theta\sin\lambda, v\cos\theta), \tag{7}$$

where $u = \cos\theta\,\frac{\mathrm{d}\lambda}{\mathrm{d}t}$ and $v = \frac{\mathrm{d}\theta}{\mathrm{d}t}$ are components of velocity vector along the λ- and θ-directions, respectively.

Second and higher-order time derivatives required in (4) are computed by repeatedly applying a finite-difference approximation of the operator

$$\hat{\mathbf{u}} \cdot \nabla = \frac{\hat{u}}{a\cos\theta}\frac{\partial}{\partial\lambda} + \frac{\hat{v}}{a}\frac{\partial}{\partial\theta} \tag{8}$$

for each component of (x, y, z), except at the poles where the operator is singular [19]. Time-centered wind fields $\hat{\mathbf{u}} = (\hat{u}, \hat{v})$ are extrapolated from velocities at previous time steps [7,19].

2.1.2. Runge–Kutta methods

Numerical solutions of Eq. (2) may not be sufficiently accurate near the poles unless a special method is used [38], because of the factor $\cos\theta$ appearing in $u = \cos\theta\,\frac{\mathrm{d}\lambda}{\mathrm{d}t}$. Initial wind data is converted from (λ, θ) coordinates to corresponding Cartesian components using (7), and the trajectory equations are solved in Cartesian (x, y, z)-space. Arrival points are converted back to (λ, θ) coordinates using (6).

Our third-order accurate Runge–Kutta scheme (RK3) for Eq. (2) is [20]

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \frac{1}{4}\mathbf{k}_1 + \frac{3}{4}\mathbf{k}_3, \tag{9}$$

where

$$\begin{aligned}
\mathbf{k}_1 &= \Delta t\,\mathbf{u}(\mathbf{x}_n, t_n),\\
\mathbf{k}_2 &= \Delta t\,\mathbf{u}\left(\mathbf{x}_n + \frac{1}{3}\mathbf{k}_1, t_n + \frac{1}{3}\Delta t\right),\\
\mathbf{k}_3 &= \Delta t\,\mathbf{u}\left(\mathbf{x}_n + \frac{2}{3}\mathbf{k}_2, t_n + \frac{2}{3}\Delta t\right).
\end{aligned}$$

Interpolation of Cartesian velocity components in 3D, required in Eq. (9), is expensive. However, this is achieved by interpolating the Cartesian velocity components in (λ, θ)-space, utilizing Eqs. (6).

For comparison, a second-order accurate Runge–Kutta (RK2) scheme was tested,

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{k}_2, \tag{10}$$

where

$$\begin{aligned}
\mathbf{k}_2 &= \Delta t\,\mathbf{u}\left(\mathbf{x}_n + \frac{1}{2}\mathbf{k}_1, t_n + \frac{1}{2}\Delta t\right),\\
\mathbf{k}_1 &= \Delta t\,\mathbf{u}(\mathbf{x}_n, t_n).
\end{aligned}$$

The interpolation procedures are analogous to those described for (9).

Note that the vector sum in (9) and (10) represents a point on the unit sphere, and is subject to the condition $\|\mathbf{x}_{n+1}\|_2 = 1$. This constraint is enforced [22,23] before transforming the Cartesian coordinate values back onto the (λ, θ) grid using (6).

3. Cascade interpolation

In the FSL scheme, cascade interpolation transfers values of the field from Lagrangian points to Eulerian points, by means of 1D operations. The grid-to-grid transfer of data in the cascade cycle is achieved via an intermediate grid system, corresponding to the intersection of Lagrangian and Eulerian grid lines. For the present study, the intermediate grid is defined by the intersection of Lagrangian longitudes (Θ_i-curves) and Eulerian latitudes. Fig. 2 displays schematically the position of the Eulerian points (open squares) and the Lagrangian points (filled circles) over a region away from poles. The Eulerian grid lines are marked by dashed lines and the corresponding Lagrangian longitudes are denoted as curves. Intermediate grid points are determined by the intersections of Θ_i-curves with the latitudes $\theta = \theta_j$, denoted by slashes in Fig. 2. Details of an efficient implementation of the FSL scheme in 2D Cartesian geometry are given in [17]. In this paper, we focus on the formulation of the FSL scheme in spherical geometry.

To summarize our numerical method, each Lagrangian longitude, Θ_i is approximated by piecewise great-circle segments [18]. This is analogous to using piecewise straight line segments in the case of a 2D plane [17]. In addition, the Θ_i-curves are extended beyond the poles for interpolation in the θ-direction. The procedure for finding intermediate grid points is the same as in [18].

The cascade algorithm consists of two 1D interpolation phases, first along Θ_i-curves (vertical direction in Fig. 2), followed by 1D interpolation along the latitude circles (horizontal direction). In the first phase, the Lagrangian points are taken as nodes (see Fig. 2) and the intermediate points are target points. For the second phase, intermediate points are the nodes and Eulerian grid points are the target points. Interpolation along the Θ_i-curve is performed assuming the great-circle distance (arc length) along the curve is the independent variable [18].

3.1. Cascade interpolation over the polar zones

Here, we introduce an efficient and accurate algorithm to compute the pole values. When the Eulerian grid is uniformly spaced in the λ-direction, with an even number of points, Θ_i-curves are extended by adding a few segments of the piecewise great circle along the Lagrangian longitude, corresponding to the Eulerian longitude $\lambda = \lambda_i \pm \pi$. If this is not the case, then the Θ_i-curve may be extended by adding segments

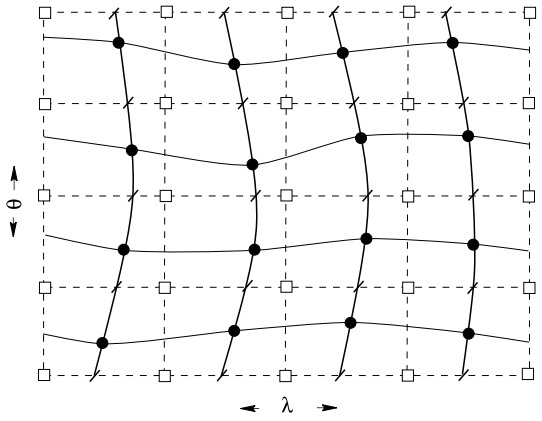

Fig. 2. Schematic diagram illustrating Eulerian (open squares) and Lagrangian (solid circles) points over the equatorial region of a sphere. Eulerian grid lines are denoted by dashed lines and Lagrangian longitudes (Θ_i-curves) and latitudes are respectively denoted by vertical and horizontal solid curves. Intermediate grid intersections of the Lagrangian longitudes and the Eulerian latitudes are marked by the slashes.

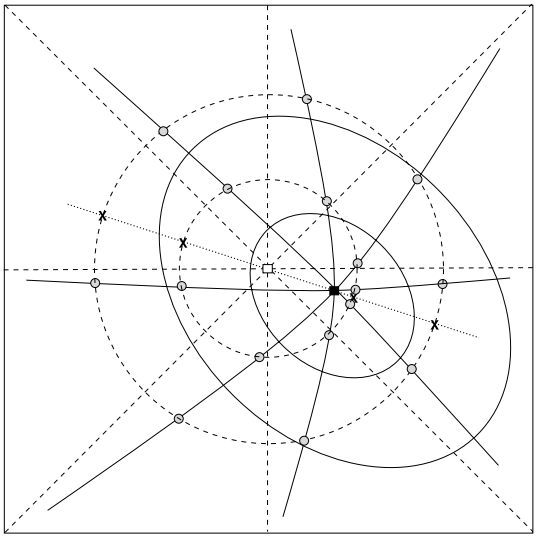

Fig. 3. Schematic diagram illustrating cascade interpolation over the polar zones. Dashed lines depict Eulerian latitudes and longitudes, respectively. The corresponding Lagrangian latitudes and longitudes (Θ_i-curves) are represented by solid curves. Eulerian and Lagrangian poles are shown as an open square and a filled square, respectively. Intermediate grid intersections are marked by filled circles. Dotted line indicates a great-circle trajectory connecting the Eulerian and Lagrangian poles.

to the Lagrangian longitude nearest to the Θ_i-curve. The number of segments to be added depends on the location of the Lagrangian pole with respect to the Eulerian pole. Fig. 3 illustrates the polar cascade stencil as projected onto a polar stereographic plane. The regular Eulerian grid consists of dashed lines, while the corresponding Lagrangian grid lines are solid curves. The Eulerian and Lagrangian poles are marked by open and filled squares, respectively.

Cascade interpolation is modified for the polar regions. Values of the advected field located at Lagrangian grid points are transferred to intermediate locations (marked as filled circles in Fig. 3) by 1D cubic (or high-order) interpolations. These are followed by 1D interpolations along latitude circles to transfer data from intermediate locations to Eulerian grid points. However, values at the Eulerian pole points are still undetermined. There are several ways to compute these values [18].

In this paper, the pole values are determined by interpolating along a great-circle trajectory joining the Eulerian and Lagrangian pole points. This trajectory is schematically shown in Fig. 3 as a dotted line, and cross marks ($\times$) denote the intersection with regular Eulerian latitudes. Values at cross marks are obtained by 1D interpolations performed along the latitude circles, during the second phase of a cascade cycle. One-dimensional interpolations along the trajectories, determine the values at Eulerian pole points, with the cross marks ($\times$) as nodes.

4. Conservative FSL schemes

We now proceed to construct a conservative algorithm based on the cascade procedure described above. The "advective" form of the transport equation (1) will not, in general, guarantee mass-conservation. Here, we consider the integral form of the continuity equation,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{u} = 0, \tag{11}$$

as given in [12],

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V(t)} \rho \,\mathrm{d}V = 0, \tag{12}$$

where $\mathbf{u}$ is the velocity vector, $\mathrm{d}/\mathrm{d}t$ is the total derivative, ρ is the density of the fluid, $\mathrm{d}V$ is the element of volume integral, and the integration is over the time-dependent Lagrangian volume $V(t)$. The value of $\int_{V(t)} \rho \,\mathrm{d}V$ is the mass of the fluid enclosed in the volume $V(t)$, and Eq. (12) states that the value is constant as $V(t)$ moves with the local fluid velocity.

Integrating (12) from t to $t + \Delta t$ results in the semi-discretized form

$$\int_{V(t+\Delta t)} \rho \,\mathrm{d}V = \int_{V(t)} \rho \,\mathrm{d}V. \tag{13}$$

In our conservative FSL scheme (FSLc), $V(t)$ is the Eulerian cell and $V(t + \Delta t)$ is the Lagrangian cell, as shown by the shaded regions in Fig. 1. In order to formulate a forward time marching scheme, the mass in Lagrangian cells at time $t + \Delta t$ must be transferred to regular Eulerian cells at the same time-level. This process is often referred to as remapping [13,25–27].

4.1. Intermediate cells for the FSLc

Conservative cascade remapping is performed through an intermediate cell system (compared to the intermediate grid system of the previous section). Recently, Nair et al. [26] developed a conservative cascade scheme (hereafter referred to as CCS) on the sphere for backward-trajectory SL advection. Their scheme is computationally efficient and applicable for polar meridional Courant number $C_\theta \leqslant 1$. Some of the ideas introduced in [26] are adopted here; however, the intermediate cell generation and special treatment for the polar zones are modified in the context of forward trajectories.

The spherical (λ, θ) domain is transformed to a Cartesian (λ, μ)-plane, where $\mu = \sin\theta$, with the area-preserving transformation described in [15,25,26], introducing variable resolution along the μ-direction. The CCS is implemented in the (λ, μ)-system, including special treatment for polar zones, by restricting the polar meridional Courant number, $C_\theta \leqslant 1$. Thus, the Lagrangian pole is located within the first array of Eulerian cells near the pole line $(\mu = \pm 1)$.

To begin, we describe the cascade scheme away from the pole. The computational procedure is analogous to the FSL described in the previous section. Lagrangian cells are approximated as polygons with sides parallel to either the λ or the μ-axis [15,25]. In Fig. 4, Eulerian cells are shown as rectangles with corner points depicted as open squares, the dotted lines indicate walls of an approximated Lagrangian cell, hereafter referred to as a *computational cell*.

Initially, we construct the intermediate cells required in the first phase of the cascade interpolation. Intermediate points are depicted as slashes in Fig. 4, and are computed as described previously for the FSL scheme. Intermediate grid lines along the Θ_i-curves are approximated by piecewise straight line segments through the midpoints of two adjacent intermediate points. These are shown as vertical piecewise dotted lines in Fig. 4, will be the east or west walls of intermediate cells. The north and south walls of intermediate cells are line segments of the corresponding horizontal Eulerian grid lines.

Computational cells are formed using intermediate cells. North and south walls are the horizontal lines through midpoints between Lagrangian corner points. These walls are depicted as horizontal dotted lines in Fig. 4. The east and west walls are constructed from intermediate cells as shown in Fig. 4. The shaded region in Fig. 4 shows a typical computational cell, and such cells span the entire domain without overlapping or introducing disjoint regions.

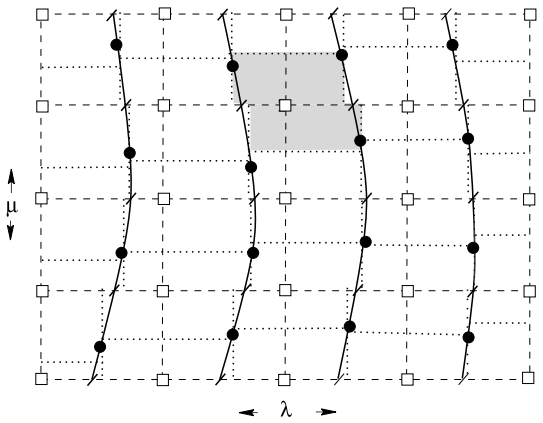

Fig. 4. Schematic diagram illustrating the conservative cascade scheme (CCS) for FSL. Eulerian and Lagrangian points are shown as open square and filled circles, respectively. Intermediate grid intersections are marked by slashes. Lagrangian cells are approximated as polygons with sides (dotted lines) parallel to coordinate axes. The shaded region shows a single approximate Lagrangian cell (computational cell).

4.2. Conservative cascade remapping

Consider a "column" of contiguous computational cells bounded by two Θ_i-curves as depicted in Fig. 4. First, mass is transferred to intermediate cells by a 1D remapping. This is followed by 1D conservative remapping where the mass in intermediate cells is transferred to Eulerian cells. Thus, in the first phase of the CCS algorithm, the Lagrangian grid is the source grid and the intermediate grid is the target grid. In the second phase, the intermediate grid is the source grid and the Eulerian grid is the target grid.

The conservative remapping can be described in general terms using ξ as the independent variable. In practice, ξ could be either λ or μ. In 1D, cell k is the region $\xi_{k-1/2} \leqslant \xi \leqslant \xi_{k+1/2}$ and has cell width $\Delta\xi_k = \xi_{k+1/2} - \xi_{k-1/2}$. The density distribution function for cell k is denoted by $h_k(\xi)$. The corresponding target cell is identified by ξ'_k with width $\Delta\xi'_k = \xi'_{k+1/2} - \xi'_{k-1/2}$ such that $\sum_k \Delta\xi_k = \sum_k \Delta\xi'_k$. Let $\overline{h}_k$ be the average density

$$\overline{h}_k = \frac{1}{\Delta\xi_k} \int_{\xi_{k-1/2}}^{\xi_{k+1/2}} h_k(\xi)\,\mathrm{d}\xi. \tag{14}$$

The cell-averaged density in the target cell ξ'_k is

$$\overline{h}'_k = \frac{1}{\Delta\xi'_k} \int_{\xi'_{k-1/2}}^{\xi'_{k+1/2}} h(\xi)\,\mathrm{d}\xi, \tag{15}$$

where $h(\xi)$ is defined piecewise in the interval $[\xi'_{i-1/2}, \xi'_{i+1/2}]$. Mass in the target cell can be computed based on a 1D conservative remapping scheme [13,25,27]. The computation of $\overline{h}_k$ at a new time level using (13) is discussed in [25,26], and will not be presented here.

The density distribution ($h_k(\xi)$) in the source cell is typically represented by a polynomial. However, we adopt the PPM developed by Colella and Woodward [35], which uses a second-degree polynomial. This method has proven to be cost-effective for meteorological modeling [15,25,26,29]. Note that, the coefficients of the parabola $h_k(\xi)$ may be further modified to preserve monotonicity of the flow field [25,26,29].

4.3. FSLc scheme over the polar regions

Application of the conservative cascade scheme requires special treatment near the poles. This is mainly due to the polar singularities associated with the latitude–longitude grid system. Cell-based conservative SL schemes require that the grid cells are well-defined (i.e., cell walls do not cross and the cells do not collapse in a single time step, see [25,28,29]). Even with smaller time steps, this condition is likely to be violated for the Lagrangian cell that includes the Eulerian pole because of the grid singularity [25,26].

Lagrangian cell walls are curved and poorly approximated as straight line segments in polar zones of the (λ, μ) system. In [25,26], additional points are introduced in the μ-direction to circumvent this problem. For the FSLc scheme considered here, the initial mass distribution is approximated without using any additional grid points. However, this may lead to a crude approximation of the mass for cells in the polar zones, and may adversely affect local mass conservation, even though this approach is globally conservative. In order to improve local mass conservation for these cells, we apply a modified version of the redistribution technique described in [25,26].

To illustrate the cascade procedure near the polar zone, consider the north polar zone as shown in Fig. 5. Typically, Lagrangian cells located in the first row away from the (Eulerian) pole line are the most deformed. (Note that, the pole depicted as an open square in Fig. 3, has been mapped to the top boundary line segment in Fig. 5). The west and east walls are constructed with vertical line segments through the intermediate grid points (shown as cross marks) of the south walls. The north walls of these cells coincide with the polar line segment $\mu = 1$. First, the total mass enclosed in a region formed by an array of computational cells along the latitude is computed. Then, the total mass is redistributed to the target cells using the "weights" derived from the FSL method. The FSL scheme described in the previous section is not only efficient but is also accurate for cross-polar advection. These properties of the FSL scheme can be exploited for polar mass redistribution.

To calculate the weights, let d_i be the density in cell i, obtained by downstream cascade interpolation. The approximate mass obtained by the FSL scheme for cell 'i' is $M_i = |d_i|A_i$, where A_i is the area of the target cell. Consider a horizontal belt of computational cells in the polar zone with total mass M. This total mass is initially computed by applying the cascade method in the constituent cells, then taking the sum. Using the weight

$$w_i = \frac{M_i}{\sum_l M_l}, \tag{16}$$

the new mass in target cell 'i' is $C_i = Mw_i$. Note that $\sum_i C_i = M$, so that the scheme is conservative.

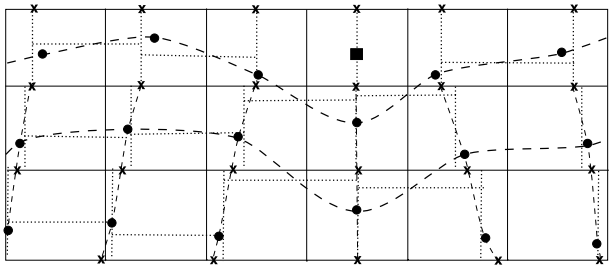

Fig. 5. Schematic illustration of the computational cells used near the polar zones of (λ, μ)-plane. The Lagrangian latitudes are shown as dashed horizontal curves with filled circles denoting Lagrangian points, and cross marks denote intermediate grid points. The Lagrangian pole is shown as a filled square and located in an Eulerian cell with north and south walls bounded by the pole line and first latitude from the pole, respectively ($C_0 \leqslant 1$). The dotted straight line segments represent the walls of the computational cells.

Once the redistribution in horizontal cells (λ-direction) is complete, the same approach can be applied in the vertical direction (μ-direction), but only near the poles. The modified mass along the horizontal belts is used for redistribution in the vertical direction, where cells from the belt on either side of the pole point (when viewed from above the pole) are symmetrically arranged in a column with the pole point in the middle. In the present context, a column of six cells, three each from either side of the pole, is used. The weights for vertical redistribution are determined from the FSL method using Eq. (16), however, the summation for six vertical cells and corresponding values of M_i are used. This represents an improvement of the method originally introduced in [25].

When the polar meridional Courant number, $C_\theta > 1$, the conservative option of the cascade scheme fails [26]. Therefore, the present form of FSLc is not applicable. A possible remedy is to combine the FSLc scheme with a 2D remapping algorithm, such as one presented in [25], over the polar regions. However, this is beyond the scope of present study.

5. Numerical experiments

Cross-polar advection is considered to be the most challenging aspect of global transport, especially when attempting to maintain accuracy. Here, we consider solid-body rotation and deformational flow tests, over the poles, to evaluate the performance of both the FSL and FSLc schemes. If $F = F(\lambda, \theta, t)$ is the advected field (density), then the normalized errors are defined by [39],

$$
\begin{aligned}
l_1 &= \frac{I(|F - F_{\mathrm{T}}|)}{I(|F_{\mathrm{T}}|)}, \\
l_2 &= \left[\frac{I[(F - F_{\mathrm{T}})^2]}{I[(F_{\mathrm{T}})^2]} \right]^{1/2}, \\
l_\infty &= \frac{\max_{\forall \lambda,\theta} |F - F_{\mathrm{T}}|}{\max_{\forall \lambda,\theta} |F_{\mathrm{T}}|}, \\
M &= \frac{I(F) - I(F_{\mathrm{T}})}{I(F_0)}, \\
F_{\max} &= \frac{\max_{\forall \lambda,\theta}(F) - \max_{\forall \lambda,\theta}(F_{\mathrm{T}})}{\Delta F_0} \\
F_{\min} &= \frac{\min_{\forall \lambda,\theta}(F) - \min_{\forall \lambda,\theta}(F_{\mathrm{T}})}{\Delta F_0},
\end{aligned}
$$

where F_{T} and F_0 are, respectively, the true solution and its initial value, ΔF_0 is the difference between the maximum and minimum solution at initial time, and the global integral I is as defined follows,

$$
I(F) = \frac{1}{4\pi} \int_0^{2\pi} \int_{-\pi/2}^{\pi/2} F(\lambda, \theta, t) \cos\theta \, \mathrm{d}\lambda \, \mathrm{d}\theta. \tag{17}
$$

The integral $I(F)$ is computed as the sum over grid point values for discrete truncation.

5.1. Solid-body rotation tests

First, we consider the "cosine bell" test problem proposed by Williamson et al. [39]. This problem is widely used to evaluate global advection schemes (see [18,25,29,38]). Because the exact trajectory of the

advecting field and exact solution for the problem are known, this test is very useful for checking the accuracy of the computed trajectory and the numerical solutions.

The initial scalar (density) distribution is defined as follows,

$$F(\lambda,\theta)=\begin{cases}\frac{1}{2}[1+\cos(\pi r/R)] & \text{if } r<R,\\ 0 & \text{if } r\geqslant R,\end{cases} \tag{18}$$

where r is the great-circle distance between (λ,θ) and the bell center. The bell radius R is $7\pi/64$, is initially centered at $(3\pi/2,0)$. The computational domain consists of 128×65 grid points, including the poles [25,26]. For the FSLc scheme, cell-centered values (128×64) are used. Velocity components of the advecting wind field are

$$u=u_0(\cos\alpha\cos\theta+\sin\alpha\cos\lambda\sin\theta),$$
$$v=-u_0\sin\alpha\sin\lambda,$$

where α is the angle between the axis of solid-body rotation and the polar axis of the spherical coordinate system [39]. When α is zero or $\pi/2$, flow is along the equator (zonal) or along the pole-to-pole (meridional) direction, respectively. Here, we focus on cross-polar advection where the flow is meridional, $\alpha=\pi/2$ or $\alpha=\pi/2-0.05$. The latter corresponds to an offset polar flow that avoids symmetry with respect to the spherical grid system.

We also consider the "slotted cylinder" problem [41], where the initial data is not smooth, unlike the previous case. The cylinder has radius $10\pi/64$ units initially located on the equator at $(3\pi/2,0)$. A constant value (0.05) is added to the initial field so that the maximum height of the cylinder is 1.05 units.

5.1.1. Trajectory error

Forward trajectories are computed using Ty3, RK3 and RK2, as described in Section 2. The accuracy of the trajectories is measured using the L_2 norm as follows,

$$L_2=\left(I[(\mathbf{x}_a-\mathbf{x}_a^*)^2]\right)^{1/2}, \tag{19}$$

where $\mathbf{x}_a^*$ and $\mathbf{x}_a$ are, respectively, the exact and computed arrival points, and I is the global integral defined by (17).

The Runge–Kutta schemes use either cubic or linear interpolation to estimate the wind fields at intermediate time levels. Table 1 shows the absolute L_2 error for these schemes.

Third-order schemes give better results, as expected. However, RK3 with cubic interpolation is computationally more expensive than Ty3, where the wind fields are interpolated twice per time step. RK2 with cubic interpolation is slightly less accurate compared to RK3, but requires only one interpolation per time step. The third-order RK3 scheme with linear interpolation does not show an improvement in accuracy over the less expensive RK2. One possible reason for this could be the dominance of spatial error in RK3 [31]. The error analysis study of Falcone and Ferretti [32], for backward-trajectory SL schemes indicates that the overall error is of the form $\mathcal{O}(\Delta t^k+\Delta x^{p+1}/\Delta t)$, where k refers to the time integration scheme order and p to the interpolation order. A similar analysis would be required [16,33], in the forward trajectory context, to establish the role of spatial and temporal errors.

Table 1
The absolute L_2 errors for the three different trajectory integration schemes

Scheme	Ty3	RK3 (cubic)	RK2 (cubic)	RK3 (linear)	RK2 (linear)
L_2 error	7.827×10^{-7}	8.038×10^{-7}	4.139×10^{-6}	1.423×10^{-5}	1.285×10^{-5}

5.1.2. Solid-body rotation test results

Fig. 6(a) shows the numerical solution with the FSL scheme combined with cubic Lagrange interpolation. Here, the computed trajectories (Ty3) are used. (The results with exact trajectories and those computed with RK3 are found to be visually indistinguishable.) In Fig. 6(a), the top left and right panels show the bell approaching and passing over the north pole. The bottom left panel shows the bell leaving the north pole after 72 time steps while the bottom right panel shows the bell after one complete revolution (256

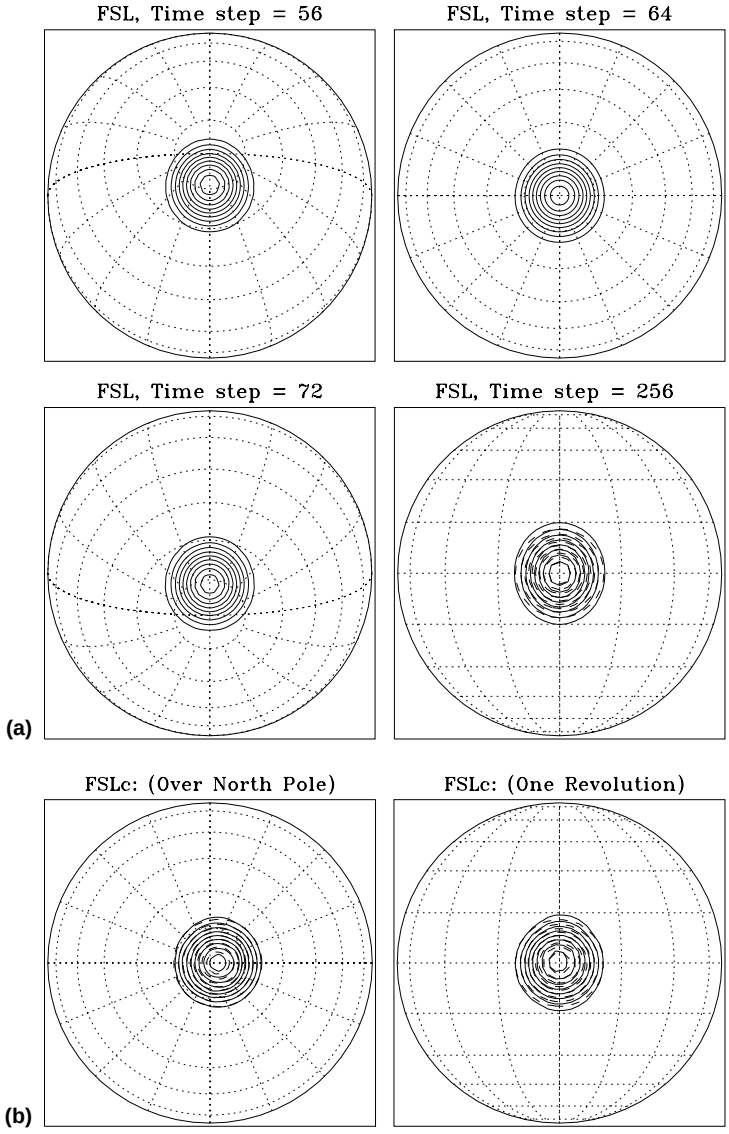

Fig. 6. (a) Orthographic projection for solid-body rotation of a cosine-bell approaching, passing over and leaving the north polar region of the sphere at time-steps 56, 64 and 72, respectively. The bottom right panel shows the cosine-bell after one revolution (256 time steps) located at the initial position, the exact solution is shown as dashed contours. (b) Orthographic projection for the offset ($\alpha = \pi/2 - 0.05$) solid-body rotation of a cosine-bell over the north pole (left panel) and on the equator (right panel) after one complete revolution. Conservative version of FSL scheme is used for the numerical integration, the exact solution is shown as dashed contours.

time steps). The numerical solution does not appear to exhibit any noise while crossing over the pole. Note that, in Fig. 6(a) the cosine-bell moves from the bottom to the top for each panel and contour values vary from 0.1 to 0.9 from the outermost to the innermost, respectively. When compared to the exact solution (dashed contour), the pattern seems to have slightly stretched along the flow direction, which is typical for cubic Lagrange interpolation (see similar results in [18]). The quality of the numerical solution may be further improved using a 1D version of cubic splines or Hermite polynomials for interpolation [18,38].

For the FSLc scheme, these experiments were repeated. Here, the initial field (density) is treated as cell-centered averages. There are 128×64 cells spanning the surface of the sphere. Fig. 6(b) shows the numerical solution with the FSLc scheme, where the wind flow is off-centered ($\alpha = \pi/2 - 0.05$). The left panel in Fig. 6(b) shows the numerical solution over the pole after 64 time steps, where the dashed contours are the exact solution. The right panel in Fig. 6(b) shows the numerical solution after one revolution (256 time steps) centered on the equator (initial position). The exact solution is marked by dashed contours, the numerical solution deforms slightly from the exact solution as the cosine-bell moves around the sphere.

Numerical results with solid-body rotation of the cosine bell are presented in Table 2. Exact trajectories are used for this set of experiments, so the interpolation scheme is the only source of error. Results from the backward-trajectory SL scheme with cascade cubic interpolation (BSL) [18] are included in Table 2 for comparison. The normalized errors $F_{\max}$ and $F_{\min}$ show that FSL scheme performs slightly better than the BSL. The l_∞ error is much better for FSL scheme as compared to BSL. The polar cascade procedure used for the FSL scheme leads to a significant improvement in the error, particularly the l_∞ error.

Fig. 7 shows a cross-sectional view of the scalar fields after one complete revolution, located on the equator. The top panels in Fig. 7 show the cosine bell for the FSL (left panel) and FSLc (right panel) schemes. The height of the bell has reduced for the numerical solution (dashed-dotted line) as compared to the exact solution (solid thin line). The lower panels in Fig. 7 show the numerical solution for the slotted cylinder, for FSL (left panel) and the FSLc scheme. The numerical solution has spurious oscillations near sharp edges, however, the monotonic version of the FSLc scheme completely removes the noise (solid thick line in the bottom right panel).

The FSLc results are listed in the last two rows of Table 2. The PPM-based conservative scheme is more accurate in l_1, l_2 and $F_{\min}$ as compared to FSL and BSL. For the CCS in [26], the corresponding l_1, l_2 and l_∞ errors for solid-body rotation case ($\alpha = \pi/2$) are 0.053, 0.047 and 0.049, respectively. These are more accurate than the results obtained with the present version of FSLc scheme. The FSLc does not use a filter to smooth out noise while crossing over the pole. Accuracy of the FSLc scheme might be further enhanced at the cost of additional computation, by adding grid points along the vertical walls of the Lagrangian cell near the polar zones, as suggested in [25,26]. Nevertheless, we do not use this option to improve the accuracy for FSLc, because our aim is to introduce a simple and efficient polar mass redistribution method for the conservative cascade scheme.

Computational efficiency of the FSLc scheme was measured with respect to the CCS scheme of [26], without the code optimization. For the solid-body rotation test with exact trajectories, the FSLc was found

Table 2
The normalized standard global errors for the FSL schemes, the conservative version of FSL scheme (FSLc) is also shown

Scheme	l_1	l_2	l_∞	M	$F_{\max}$	$F_{\min}$
FSL ($\alpha = \pi/2$)	0.231	0.141	0.113	0.0063	−0.113	−0.031
FSL ($\alpha = \pi/2 - 0.05$)	0.231	0.141	0.117	0.0037	−0.117	−0.029
BSL ($\alpha = \pi/2$)	0.235	0.144	0.121	0.0067	−0.121	−0.034
BSL ($\alpha = \pi/2 - 0.05$)	0.234	0.144	0.120	0.0039	−0.120	−0.030
FSLc ($\alpha = \pi/2$)	0.132	0.115	0.146	0.0[a]	−0.136	−0.007
FSLc ($\alpha = \pi/2 - 0.05$)	0.133	0.115	0.151	0.0[a]	−0.134	−0.006

[a] Note that the mass errors M for the FSLc scheme are of the order of machine precision, and therefore these error values are set to 0.0.

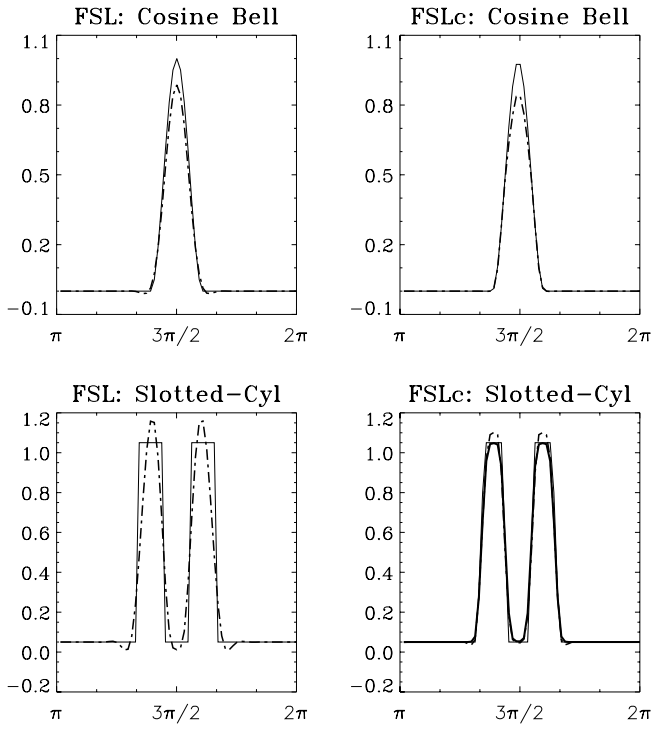

Fig. 7. Cross-sectional plots (along the equator) for the cosine-bell (top two panels) and the slotted cylinder (bottom two panels), after one revolution along the pole-to-pole direction. The initial solutions are shown as thin lines and the dashed-dotted lines show the numerical solutions. The monotonic solution for the FSLc is shown as thick line, in the bottom right panel.

to be approximately 1.1 times faster than the CCS (without filters) in [26]. This also indicates that the FSLc is more than twice as efficient as conventional BSL scheme (see [26]).

5.1.3. Order of accuracy of the FSLc scheme

In this section, we provide computational evidence that the method is at least second-order accurate. First, the time step is refined, demonstrating that the error is dominated by spatial terms (see Table 3). Here, we compute the absolute errors ℓ_1, ℓ_2 and ℓ_∞ corresponding to the non-normalized l_1, l_2 and l_∞ errors, respectively. Even though the ℓ_1 error decays slightly, the ℓ_2 and ℓ_∞ errors are nearly identical, indicating the spatial dominance. The second experiment examines the spatial discretization errors and the third experiment involves refinement in both space and time.

Represent the error as a truncation of a discretization parameter δ as $E(\delta) = \|U_{\text{computed}} - U_{\text{exact}}\|$, where $\Delta t = \mathcal{O}(\delta)$ and the spacing between grid points has the same magnitude as δ. Based on the assumption that $E(\delta) = \mathcal{O}(\delta^p)$ for some p, we make the ansatz $E(\delta) = K\delta^p$, where K is constant across all experiments. Of

Table 3
Absolute errors for the FSLc with different time steps

Grid	Time step	ℓ_1	ℓ_2	ℓ_∞
64×33	512	0.1174	0.1247	0.4519
64×33	256	0.0796	0.1181	0.4450
64×33	128	0.0659	0.1074	0.4101

course, K will have some variation. However, this ansatz gives an estimate of the order. Thus, we estimate $p \approx p_{\text{est}}$ with the formula

$$p_{\text{est}} = \log_2\left(\frac{E(\delta)}{E(\delta/2)}\right).$$

Results from refining the spatial grid (keeping the temporal spacing constant) are presented in Table 4. Most truncation error analyses utilize the ℓ_2-norm, and the ℓ_2-norm results indicate second-order accuracy ($p_{\text{est}} \approx 2$). Refining space and time together is studied in Table 5, where we see additional evidence of second-order accuracy. These tests do not prove the order of accuracy; rather, we seek to simply provide an indication that the methods are second-order accurate. To conclude the method is formally second-order, a truncation error analysis is required.

5.2. Deformational flow test

For the deformational flow test, we use the idealized cyclogenesis problem [36] in spherical geometry [18]. The smooth deformational flow test on the surface of the sphere is described in [25,26], and is considered here for FSLc scheme. Two steady vortices are simulated such that each vortex center is located near the north and south poles, respectively. The normalized tangential velocity of the vortex field is given by

$$V_{\text{t}} = \frac{3\sqrt{2}}{2}\operatorname{sech}^2(\rho')\tanh(\rho'),$$

where ρ' is "radius" of the vortex. The exact solution at time t is

$$F(\lambda', \theta', t) = 1 - \tanh\left[\frac{\rho'}{\gamma}\sin(\lambda' - \omega' t)\right],$$

where (λ', θ') are the rotated spherical coordinates with poles at the vortex centers, γ is the parameter that defines characteristic width [18], $\omega' = \omega'(\theta')$ is the angular velocity and $\rho' = \rho'(\theta')$. For the definition of these and other parameters defining the vortex, see [25,26].

Table 4
Absolute errors for the three different grid resolution for FSLc with same number of time steps

Grid	Time step	ℓ_1		ℓ_2		ℓ_∞	
		$E(\delta)$	p_{est}	$E(\delta)$	p_{est}	$E(\delta)$	p_{est}
64×33	512	0.1174	(...)	0.1247	(...)	0.4520	(...)
128×65	512	0.0176	**(2.7)**	0.0345	**(1.8)**	0.1684	**(1.4)**
256×129	512	0.0021	**(3.0)**	0.0045	**(2.9)**	0.0313	**(2.4)**

Table 5
Absolute errors for the three different grid resolution for the FSLc with variable time steps

Grid	Time step	ℓ_1		ℓ_2		ℓ_∞	
		$E(\delta)$	p_{est}	$E(\delta)$	p_{est}	$E(\delta)$	p_{est}
64×33	128	0.0659	(...)	0.1074	(...)	0.4101	(...)
128×65	256	0.0144	**(2.2)**	0.0291	**(1.9)**	0.1425	**(1.6)**
256×129	512	0.0021	**(2.8)**	0.0045	**(2.7)**	0.0316	**(2.2)**

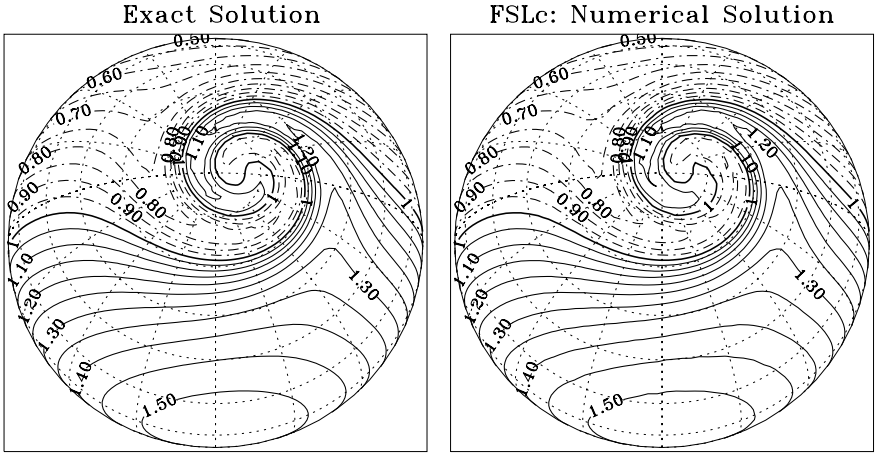

Fig. 8. Orthographic projection for the polar vortex with center located near the north pole. The exact and numerical solution with FSLc are shown respectively, in the left and the right panels, at $t = 3$, and that corresponds to 64 time steps for the FSLc scheme.

The vortex centers are located at approximately 81° north and south latitudes, respectively. The initial values $F(\lambda', \theta', t = 0)$ are generated at the cell centers on a 128×64 grid and integrated for 3 time units with 64 time steps, corresponding to directional maximum Courant numbers $C_\lambda = 19.1$ and $C_\theta = 0.82$, respectively. Fig. 8 shows exact and numerical solutions with FSLc after $t = 3$ time units. The relative errors l_1, l_2 and l_∞ for the FSLc scheme are 0.0015, 0.0043 and 0.0334, respectively. These results are slightly less accurate than that of the CCS scheme in [26], but better than a traditional SL scheme with bicubic Lagrange interpolation (see, Table 2 of [26]). Exact trajectories are used for numerical integration. The numerical solution is similar to the exact solution with the overall structure of the vortex well captured by FSLc. However, a slight degradation near the center of the vortex is visible for the numerical solution, comparable to the results given in [25,26].

6. Summary and conclusions

A computationally efficient forward-trajectory semi-Lagrangian scheme (FSL) for advection on the surface of a sphere was developed. The scheme uses forward trajectories, originating on a regular grid (Eulerian points) and terminating at the downstream (Lagrangian) points. The accuracy of the FSL scheme was evaluated using two test problems, solid-body rotation and deformational flow near the poles.

At every time step, the advecting field is transferred from the irregular Lagrangian grid to the regular Eulerian grid by means of cascade interpolation through an intermediate grid system. The cascade method presented here can be used for a SL scheme (with forward or backward trajectories) on a latitude–longitude spherical grid. The global cascade scheme developed by Nair et al. [18], employed an alternative intermediate grid system where the number of longitudes is assumed to be even. In general, this approach cannot be used for a variable-resolution global grid, and the pole values are estimated in an arbitrary manner. The cascade procedure proposed here is free from these limitations. Moreover, the FSL scheme can handle the pole problem efficiently, because the pole values are estimated accurately by interpolating along the polar trajectory connecting Lagrangian and Eulerian poles.

Most SL schemes employ $\mathcal{O}(\Delta t^2)$ backward-trajectory iterative schemes for estimating upstream positions. This is a computationally expensive component of the SL advection process. Backward trajectories imply simple upstream interpolation, where the nodes are regularly spaced grid points. A forward-trajectory SL scheme necessitates downstream interpolation, where data are transferred from an irregular to

regular grid. On the sphere, standard multi-dimensional interpolation is computationally prohibitive. However, with the global cascade approach presented here, calculations are more efficient. The FSL scheme avoids the iterative search process and high-order trajectory integration (ODE) schemes can be easily incorporated. Numerical experiments with the FSL scheme show that its accuracy is comparable to the traditional SL scheme.

Two $\mathcal{O}(\Delta t^3)$ accurate trajectory algorithms based on the Taylor series expansion [19] and Runge–Kutta (RK3) scheme were evaluated. The ODEs are solved in Cartesian (x, y, z) coordinates, but function values are estimated in (λ, θ) coordinates and transformed back to the corresponding Cartesian coordinates. This approach is particularly useful for solving the trajectory equations in the polar zones. A second-order accurate Runge–Kutta scheme was tested for comparison. To maintain high-order accuracy, the Runge–Kutta schemes optionally implements cubic interpolation for estimating function values at intermediate time-levels. However, this may be expensive, particularly for third or higher-order Runge–Kutta schemes. For practical applications, a low-storage Runge–Kutta scheme [40] may be desirable. McGregor's scheme, based on the Taylor method (Ty3) was found to be much less expensive compared to RK3 (with cubic interpolation), and had comparable accuracy. For practical application on the sphere, Ty3 is an appropriate choice, because it is easy to implement and the order of accuracy can be increased in a less expensive way. High-order Adam–Bashforth schemes may be an interesting alternative for forward-trajectory integration [8,11].

A conservative version of the FSL algorithm, FSLc was developed and tested using solid-body rotation and deformational flow. The scheme employs 1D piecewise parabolic remapping, and was developed in the cascade interpolation framework. The FSLc scheme is approximately 10% more efficient than the backward-trajectory conservative cascade scheme. The accuracy of FSLc is comparable to or better than that of the standard SL scheme employing cubic Lagrange interpolation. A simple polar mass redistribution method works well for maintaining accuracy, and the FSLc does not use any polar filter. The accuracy of the FSLc can be further improved by introducing additional cell arrays around the pole as described in [25,26]. At present, the FSLc scheme is restricted to meridional polar Courant number $C_\theta \leqslant 1$ and is free from zonal Courant number C_λ restriction. The major difficulty in using finite-volume based methods for the conventional (λ, θ) grid system (or the equivalent (λ, μ) grid), is related to the remapping of cells which deform. Mass redistribution methods such as the one used here, may be an easy and cost-effective way to avoid this problem. However, they do not guarantee local mass conservation for the polar regions. A comprehensive polar remapping strategy (free from C_θ, C_λ restriction) is under development.

Acknowledgements

This project was partially supported by the NSF Grant ATM-95-25286. The authors would like to thank Dr. Steve Thomas (NCAR) for his helpful suggestions. Support of the Department of Mathematics at North Carolina State University and the computer time provided by North Carolina Supercomputing Center (NCSC) are appreciated. The comments of the anonymous reviewers are also appreciated.

References

[1] A. Robert, A stable numerical integration scheme for the primitive meteorological equations, Atmos. Ocean. 19 (1981) 35.
[2] A. Robert, A semi-Lagrangian semi-implicit numerical integration scheme for the primitive meteorological equations, J. Meteorolog. Soc. Japan 60 (1982) 319.
[3] J. Côté, S. Gravel, A. Methot, A. Patoine, M. Roch, A. Staniforth, The operational CMC/MRB Global Environmental Multiscale (GEM) model, Mon. Weather Rev. 126 (1998) 1373.
[4] J.-H. Qian, F.H.M. Semazzi, J.S. Scroggs, A global nonhydrostatic semi-Lagrangian model with orography, Mon. Weather Rev. 126 (1998) 747.

[5] C. Temperton, M. Hortal, A. Simmons, A two-time-level semi-Lagrangian global spectral model, Quart. J. Roy. Meteorolog. Soc. 127 (2001) 111.
[6] D.L. Williamson, J.G. Olson, A comparison of semi-Lagrangian and Eulerian polar climate simulations, Mon. Weather Rev. 126 (1998) 991.
[7] A. Staniforth, J. Côté, 1991: Semi-Lagrangian integration schemes for the atmospheric models – a review, Mon. Weather Rev. 119 (1991) 2206.
[8] R.J. Purser, L.M. Leslie, An efficient interpolation procedure for high-order three-dimensional semi-Lagrangian models, Mon. Weather Rev. 119 (1991) 2492.
[9] R.J. Purser, L.M. Leslie, An efficient semi-Lagrangian scheme using third-order semi-implicit time integration and forward trajectories, Mon. Weather Rev. 122 (1994) 745.
[10] L.M. Leslie, R.J. Purser, Three-dimensional mass-conserving semi-Lagrangian schemes employing forward trajectories, Mon. Weather Rev. 123 (1995) 2551.
[11] R.J. Purser, L.M. Leslie, Generalized Adams–Bashforth time integration schemes for a semi-Lagrangian model employing the second-derivative from of the horizontal momentum equations, Quart. J. Roy. Meteorolog. Soc. 122 (1996) 737.
[12] J.K. Dukowicz, J.R. Baumgardner, Incremental remapping as a transport/advection algorithm, J. Comput. Phys. 160 (1999) 318.
[13] J.K. Dukowicz, J.W. Kodis, Accurate conservative remapping (rezoning) for arbitrary Lagrangian Eulerian computations, SIAM J. Statist. Comput. 8 (1987) 305.
[14] J.P.R. Laprise, A. Plante, A class of semi-Lagrangian integrated mass (SLIM) numerical transport scheme, Mon. Weather Rev. 123 (1995) 553.
[15] B. Machenhauer, M. Olk, On the development of a cell-integrated semi-Lagrangian shallow water model on the sphere, in: Proceedings of the ECMWF Workshop on semi-Lagrangian methods, November 6–8, 1995, 1996, p. 213.
[16] A.V. Malevsky, S.J. Thomas, Parallel algorithms for semi-Lagrangian advection, Int. J. Numer. Meth. Fluids 25 (1997) 455.
[17] R.D. Nair, J. Côté, A. Staniforth, Monotonic cascade interpolation for semi-Lagrangian advection, Quart. J. Roy. Meteorolog. Soc. 125 (1999) 197.
[18] R.D. Nair, J. Côté, A. Staniforth, Cascade interpolation for semi-Lagrangian advection over the sphere, Quart. J. Roy. Meteorolog. Soc. 125 (1999) 1445.
[19] J.L. McGregor, Economical determination of the departure points for the semi-Lagrangian models, Mon. Weather Rev. 121 (1993) 221.
[20] M. Abramowitz, I.A. Stegun, Handbook of mathematical functions, National Bureau of Standards Applied Mathematics Series .55, US Department of Commerce, 1964.
[21] A. McDonald, An examination of alternative extrapolation to find the departure point position in a two-time-level semi-Lagrangian integration, Mon. Weather Rev. 127 (1999) 1985.
[22] H. Ritchie, Semi-Lagrangian advection on a Gaussian grid, Mon. Weather Rev. 115 (1987) 608.
[23] F.X. Giraldo, Trajectory calculations for spherical geodesic grids in Cartesian space, Mon. Weather Rev. 127 (1999) 1653.
[24] C.W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, NJ, 1971, 253 pp.
[25] R.D. Nair, B. Machenhauer, The mass-conservative cell integrated semi-Lagrangian advection scheme over the the sphere, Mon. Weather Rev. 130 (2002) 649.
[26] R.D. Nair, J.S. Scroggs, F.H.M. Semazzi, Efficient conservative global transport schemes for climate and atmospheric chemistry models, Mon. Weather Rev. 130 (2002) 2059.
[27] M. Rančić, An efficient, conservative, monotone remapping for semi-Lagrangian transport algorithm, Mon. Weather Rev. 123 (1995) 1213.
[28] J.S. Scroggs, F.H.M. Semazzi, A conservative semi-Lagrangian method for multidimensional applications, Num. Meth. PDE 11 (1995) 445.
[29] S.-J. Lin, R.B. Rood, Multi-dimensional flux-form semi-Lagrangian transport schemes, Mon. Weather Rev. 124 (1996) 1213.
[30] T.N. Phillips, A.J. Williams, A semi-Lagrangian finite volume method for Newtonian contraction flows, SIAM J. Sci. Comput. 22 (2001) 2152.
[31] P. Bartello, S.J. Thomas, The cost-effectiveness of semi-Lagrangian advection, Mon. Weather Rev. 124 (1996) 2883.
[32] M. Falcone, R. Ferretti, Convergence analysis for a class of high-order semi-Lagrangian advection schemes, SIAM J. Numer. Anal. 35 (1998) 909.
[33] D. Xiu, G.E. Karniadakis, A semi-Lagrangian high-order method for Navier–Stokes equations, J. Comput. Phys. 172 (2001) 658.
[34] P.J. Rasch, Recent development in transport schemes at NCAR, Technical Report No. 265, Max Planck Institute for Meteorology, Hamburg, Germany, 65, 1998.
[35] P. Colella, P.R. Woodward, Piecewise parabolic method for gas-dynamical simulations, J. Comput. Phys. 54 (1984) 174.
[36] C.A. Doswell, A kinematic analysis of frontogenesis associated with a nondivergent vortex, J. Atmos. Sci. 41 (1984) 1242.
[37] P.K. Smolarkiewicz, Pudykiewicz, A class of semi-Lagrangian approximation for fluids, J. Atmos. Sci. 49 (1992) 2082.

[38] D.L. Williamson, P. Rasch, Two-dimensional semi-Lagrangian transport with shape preserving interpolation, Mon. Weather Rev. 117 (1989) 102.
[39] D.L. Williamson, J.B. Drake, J. Hack, R. Jacob, P.N. Swartztrauber, A standard test for numerical approximation to the shallow water equations in spherical geometry, J. Comput. Phys. 102 (1992) 211.
[40] J. Williamson, Low-storage Runge–Kutta schemes, J. Comput. Phys. 35 (1980) 48.
[41] S.T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys. 31 (1979) 335.