Journal of Computational Physics 230 (2011) 1567–1584

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A low communication and large time step explicit finite-volume solver for non-hydrostatic atmospheric dynamics

Matthew R. Norman [a,*], Ramachandran D. Nair [b], Fredrick H.M. Semazzi [a]

[a] *Department of Marine, Earth, and Atmospheric Science, North Carolina State University, Raleigh, NC, USA*
[b] *Institute for Mathematics Applied to Geosciences, National Center for Atmospheric Research, Boulder, CO, USA*

ARTICLE INFO

Article history:
Received 24 April 2010
Received in revised form 28 October 2010
Accepted 12 November 2010
Available online 21 November 2010

Keywords:
Finite volume
Atmospheric dynamics
Non-hydrostatic
Riemann solver
Fully discrete
Flux vector splitting

ABSTRACT

An explicit finite-volume solver is proposed for numerical simulation of non-hydrostatic atmospheric dynamics with promise for efficiency on massively parallel machines via low communication needs and large time steps. Solving the governing equations with a single stage lowers communication, and using the method of characteristics to follow information as it propagates enables large time steps. Using a non-oscillatory interpolant, the method is stable without post-hoc filtering. Characteristic variables (built from interface flux vectors) are integrated upstream from interfaces along their trajectories to compute time-averaged fluxes over a time step. Thus we call this method a Flux-Based Characteristic Semi-Lagrangian (FBCSL) method. Multidimensionality is achieved via a second-order accurate Strang operator splitting. Spatial accuracy is achieved via the third- to fifth-order accurate Weighted Essentially Non-Oscillatory (WENO) interpolant.

We implement the theory to form a 2-D non-hydrostatic compressible (Euler system) atmospheric model in which standard test cases confirm accuracy and stability. We maintain stability with time steps larger than CFL = 1 (CFL number determined by the acoustic wave speed, not advection) but note that accuracy degrades unacceptably for most cases with CFL > 2. For the smoothest test case, we ran out to CFL = 7 to investigate the error associated with simulation at large CFL number time steps. Analysis suggests improvement of trajectory computations will improve error for large CFL numbers.

1. Introduction

Inclusion of non-hydrostatic dynamics in atmospheric models has already become the norm for mesoscale and synoptic scale models [1,2] because of the resolved atmospheric features on non-hydrostatic scales. It is also quickly becoming a desirable feature of even global climate models to accommodate future multi-scale techniques such as adaptive grid refinement. Even quasi-uniform resolution global models are approaching grid spacings in the tens of kilometers [3] which means non-hydrostatic dynamics are an application of importance.

Efficient numerical integration of non-hydrostatic atmospheric dynamical equations is complex with large distributed memory parallel machines in the picture. In the reign of Moore’s law (exponentially increasing single CPU compute power), Semi-Implicit Semi-Lagrangian (SISL) methods provided excellent efficiency [4–6]. In the current era of Massively Parallel Machines (MPMs) which distribute system memory among 10,000s of nodes, communicating data between nodes is costly. High communication requirements of SISL methods force diminishing returns before reaching the capacity of current

[*] Corresponding author. Tel.: +1 9198009774.
E-mail addresses: mrnorman@ncsu.edu (M.R. Norman), rnair@ucar.edu (R.D. Nair), semazzi@ncsu.edu (F.H.M. Semazzi).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.11.022

machines. Therefore, these methods along with fully implicit methods [7,8] have difficulty scaling current machines in Amdahl's sense [9] with a reasonable problem size and throughput.

We take ''scalable'' to mean throughput scaling, closely related to Gustafson's notion of weak scaling [10]. Suppose a method can simulate on a given grid at the rate of five Simulated Years Per Day (SYPD), a number generally accepted in the climate modeling community for future projections. If we refine the grid such that the overall problem size increases by some factor *f*, that method scales perfectly if it retains five SYPD by spreading the larger problem across *f* times more processors.

Explicit methods cannot scale in this manner due to the Courant–Friedrichs–Lewy (CFL) condition which requires time step reduction with grid refinement. Please note that our references to the CFL constraint are always relative to the acoustic wave speed and not the advection, and it has a precise meaning: $\mathrm{CFL} = c_{max}\Delta t/\Delta x$ where c_{max} is the maximum wave speed (acoustic), Δt is the time step, and Δx is the grid spacing. However, because explicit methods have low communication needs, they scale by Amdahl's definition of taking a fixed problem size and spreading it among as many processors as possible. Low-communication explicit methods have shown good utilization of MPMs and throughput at high resolutions for atmospheric flow [3]. Therefore, we explore here a new explicit method with a balance of time step and communication that we believe forms a competitive middle ground in terms of efficiency among existing methods. First, we review previous explicit methods used for atmospheric dynamics.

A class of popular temporal discretization methods for the atmosphere are central semi-discrete solvers with the split-explicit sub-cycling (SESC) treatment applied [11–15]. "Central" means there is no upwinding for the full dynamical equation set. The CFL number of these methods is limited generally to one (though moderately higher for some of the semi-discrete solvers). In particular, multi-stage solvers require communication between each stage which can increase communication burden on MPMs. SESC linearizes off the fast waves (sound waves/external gravity waves) and solves them on a smaller time step with cheap small-stencil methods, solving the slow (transport) waves on a larger time step. It is typical in implementation for the fast waves to be sub-cycled within each stage of a multi-stage solver. This can be a burden on communication constraints, but sub-cycling enables significantly larger time steps with little additional computation which increases efficiency.

Upwind Godunov-type methods are another FV class [16–18]. For the atmosphere, they are an emerging application even though one was applied as far back as two decades ago [16]. Of particular interest to the present paper are methods of this class which implement fully discrete time solvers (meaning one step, one stage) because they exhibit low communication requirements in parallel. To a large extent, the schemes developed herein were motivated by the theory and attributes of these types of methods. These all have CFL number limitations of one, and high-order accuracy is obtained via Taylor series expansions limited with van Leer type limiters.

Galerkin methods [19–24] belong to the Finite Element class. They generally come in two flavors: Discontinuous Galerkin (DG) and Continuous Galerkin/Spectral Element (SE). They have differing CFL number limitations for stability, but both time steps decrease as the order of accuracy increases as well as when grid spacing decreases. Because Galerkin methods use multiple degrees of freedom (either nodal or modal), they can perform high-order accurate reconstruction without communication leading to a very low communication burden. Non-oscillatory limiting of the spatial approximations within elements is an active area of research [25] which tends to add to the communication burden. Even with time step limitations, because of the low communication requirements the spectral element method has already been shown to perform very well for atmospheric flows [26,3].

Constrained interpolation profile (CIP) methods [27–31] are FV methods that also evolve point values (and derivatives) at cell boundaries to make reconstruction more local. CIP methods using characteristics [28,31] have much in common with the method being presented here. Both compute interface fluxes by tracing characteristic variables out from the interface using characteristic trajectories. Thus, they can simulate at large CFL numbers. Here, we use flux-based characteristic variables for easing the maintenance of hydrostatic balance, and we do not make the approximation that the averaged flux vector is equal to the flux computed using averaged state variables (see Section 2.4.1). Additionally, we do not make use of cell interface point values, but we evolve only the cell means.

Here, we propose a new solver for atmospheric dynamics with the potential for competitive efficiency through low communication requirements (enabling better scaling on large parallel machines) and large time steps (improving overall efficiency). It is a high-order accurate, upwind, fully discrete, and non-oscillatory solver based on a flux vector splitting analog of *f*-waves [32,33]. The flux vector splitting is what enables large time steps, and we extend our method to large CFL numbers in this paper to determine and discuss the errors and complications involved with explicit large CFL number simulation. We call this method a Flux-Based Characteristic Semi-Lagrangian (FBCSL) method.

The closest comparison of the FBCSL method as far as algorithmic similarity to an operational method is the class of central semi-discrete methods usually coupled with the SESC treatment. Questions naturally arise regarding the comparative communication costs and overall efficiency with regard to SESC methods. We remark that the SESC treatment is algorithmically distinct from the central, semi-discrete solvers they typically accompany. It can be adapted to other methods, and in fact, our characteristics-based method has a natural extension since the fast waves are separated from slow waves in the characteristic decomposition. Therefore, the more critical comparison regarding communication burden (leaving SESC extensions to future research) is between our single stage FBCSL method and multi-stage solvers.

Also, we would like the reader to note that this paper is a proof of concept regarding the FBCSL method, giving rise to at least two implications. First, this method is not intended to be presented as operational. Future adaptations and modifications (e.g. the SESC treatment perhaps) are necessary for adequate efficiency, and significantly more testing is required of any proposed integration method in more realistic contexts. Second, this would be most effective as a horizontal scheme in typical synoptic and global scale models. Because of the high aspect ratios between horizontal and vertical grid spacing, vertical fast dynamics are either handled hydrostatically (removing vertical sound waves) or implicitly in the non-hydrostatic case.

Given the scheme's flexibility to accommodate any spatial interpolant, we use the third- to fifth-order accurate Weighted Essentially Non-Oscillatory (WENO) method as implemented in Capdeville [34]. The WENO philosophy of Shu and Osher [35] and Shu [36] involves a weighted sum of polynomials where the least oscillatory polynomials are weighted the highest. One could also use, for instance, non-polynomial interpolants as well [37,38]. There tends to be concern with the expense of WENO interpolation. Since our scheme can handle any interpolant, this isnot of consequence to the FBCSL framework. However, in large parallel simulation, higher computation to communication ratios resulting from the added expense actually hide some of this expense due to increased parallel efficiency. But overall, it is a design choice available to the user. For positive definite simulation, a monotonic interpolation (e.g. the Piecewise Parabolic Method) is an obvious choice.

Three standard 2-D non-hydrostatic test cases will be performed to validate the proposed method. The method is described in Section 2, validation through numerical simulation of non-hydrostatic test cases is given in Section 3, and concluding remarks and future work are given in Section 4.

2. Numerical method

2.1. 2-D compressible non-hydrostatic equation set

In this study, we use a 2-D, compressible, non-hydrostatic model (essentially the Euler system of equations) which explicitly conserves mass, momentum, and potential temperature. A Cartesian rectangular grid is used for spatial discretization. The equation set is as follows:

$$\frac{\partial \mathbf{U}}{\partial t}+\frac{\partial \mathbf{F}(\mathbf{U})}{\partial x}+\frac{\partial \mathbf{H}(\mathbf{U})}{\partial z}=\mathbf{S} \tag{1}$$

$$\mathbf{U}=\begin{bmatrix}\rho\\ \rho u\\ \rho w\\ \rho\theta\end{bmatrix},\quad \mathbf{F}(\mathbf{U})=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uw\\ \rho u\theta\end{bmatrix},\quad \mathbf{H}(\mathbf{U})=\begin{bmatrix}\rho w\\ \rho wu\\ \rho w^2+p\\ \rho w\theta\end{bmatrix},\quad \mathbf{S}(\mathbf{U})=\begin{bmatrix}0\\ 0\\ -\rho g\\ 0\end{bmatrix} \tag{2}$$

where ρ is the density, u is the horizontal wind, w is the vertical wind, p is the pressure, and θ is the potential temperature which is related to the actual temperature, T, by $\theta = T(p_0/p)^{R_d/c_p}$. The equation set is closed by the equation of state: $p = C_0(\rho\theta)^\gamma$ where the constant C_0 is defined by: $C_0 = R_d^\gamma p_0^{-R_d/c_p}$. The constants are $\gamma = c_p/c_v \approx 1.4$, $R_d = 287\ \mathrm{J\ kg^{-1}\ K^{-1}}$, $c_p = 1004\ \mathrm{J\ kg^{-1}\ K^{-1}}$, $c_v = 717\ \mathrm{J\ kg^{-1}\ K^{-1}}$, and $p_0 = 10^5$ Pa.

2.2. Fully discrete FV framework

In FV models, the spatial domain is spanned by cells, and the cell-averaged state variables are evolved between them by fluxes through cell interfaces. To approximate the equations, the entire equation set is integrated over one of these computational cells with a domain of $\Omega_{i,j} \in [x_{i-1/2,j}, x_{i+1/2,j}] \times [z_{i,j-1/2}, z_{i,j+1/2}]$ where $x_{i\pm1/2,j} = x_{i,j} \pm \Delta x/2$ and $z_{i,j\pm1/2} = z_{i,j} \pm \Delta z/2$ refer to cell interface locations and Δx and Δz are the horizontal and vertical grid spacing, respectively. Next, the Gauss divergence theorem is applied to the flux divergence integrals, transforming them into line integrals of the normal flux over the cell boundaries. On a rectangular, Cartesian grid, this gives:

$$\frac{\partial \overline{\mathbf{U}}_{i,j}}{\partial t}+\frac{1}{\Delta x}[\mathbf{F}_{i+1/2,j}(\mathbf{U})-\mathbf{F}_{i-1/2,j}(\mathbf{U})]+\frac{1}{\Delta z}[\mathbf{H}_{i,j+1/2}(\mathbf{U})-\mathbf{H}_{i,j-1/2}(\mathbf{U})]=\overline{\mathbf{S}}_{i,j} \tag{3}$$

The flux evaluations in time will be fully discrete, meaning the equations are integrated in time directly. Since an integral is the product of the average and the interval of integration, this can be rewritten as:

$$\overline{\mathbf{U}}_{i,j}^{n+1}=\overline{\mathbf{U}}_{i,j}^{n}-\frac{\Delta t}{\Delta x}\left[\widehat{\mathbf{F}}_{i+1/2,j}(\mathbf{U})-\widehat{\mathbf{F}}_{i-1/2,j}(\mathbf{U})\right]-\frac{\Delta t}{\Delta z}\left[\widehat{\mathbf{H}}_{i,j+1/2}(\mathbf{U})-\widehat{\mathbf{H}}_{i,j-1/2}(\mathbf{U})\right]+\Delta t\widehat{\overline{\mathbf{S}}}_{i,j} \tag{4}$$

where the hat above a variable denotes an average over the time step, a superscript n denotes the variable valid at time $n\Delta t$, and Δt is the time step.

2.3. Strang splitting: multidimensionality and source term

Finally, a second-order accurate Strang splitting is applied to integrate the fluxes in a sequence of 1-D sweeps. Consider the following update operators on any given cell:

$$\mathsf{F}_x(\overline{\mathbf{U}}^n) = \overline{\mathbf{U}}^n_{i,j} - \frac{\Delta t}{\Delta x}\left[\widehat{\mathbf{F}}_{i+1/2,j}\left(\overline{\mathbf{U}}^n_{i-\frac{s-1}{2},j},\ldots,\overline{\mathbf{U}}^n_{i+\frac{s+1}{2},j}\right) - \widehat{\mathbf{F}}_{i-1/2,j}\left(\overline{\mathbf{U}}^n_{i-\frac{s+1}{2},j},\ldots,\overline{\mathbf{U}}^n_{i+\frac{s-1}{2},j}\right)\right] \tag{5}$$

$$\mathsf{F}_z(\overline{\mathbf{U}}^n) = \overline{\mathbf{U}}^n_{i,j} - \frac{\Delta t}{\Delta z}\left[\left(\widehat{\mathbf{H}}_{i,j+1/2}\left(\overline{\mathbf{U}}^n_{i,j-\frac{s-1}{2}},\ldots,\overline{\mathbf{U}}^n_{i,j+\frac{s+1}{2}}\right) - \widehat{\mathbf{H}}_{i,j-1/2}\left(\overline{\mathbf{U}}^n_{i,j-\frac{s+1}{2}},\ldots,\overline{\mathbf{U}}^n_{i,j+\frac{s-1}{2}}\right)\right)\right] \tag{6}$$

$$\mathsf{F}_S(\overline{\mathbf{U}}^n) = \overline{\mathbf{U}}^n_{i,j} + \Delta t\widehat{\mathbf{S}}_{i,j}\left(\overline{\mathbf{U}}^n_{i,j}\right) \tag{7}$$

where s is the size of the stencil used for spatial reconstruction (see Section 2.4.1). Our WENO approximation has a value $s = 5$. We introduce the stencil size in here to show that the flux computations only depend on a local set of cells. The operator splitting procedure (splitting dimensions and the gravity source term) is implemented as follows for all cells i, j:

$$\begin{aligned}
\overline{\mathbf{U}}^*_{i,j} &= \mathsf{F}_x\left(\overline{\mathbf{U}}^n\right) \\
\overline{\mathbf{U}}^{**}_{i,j} &= \mathsf{F}_z\left(\overline{\mathbf{U}}^*\right) \\
\overline{\mathbf{U}}^{n+1}_{i,j} &= \mathsf{F}_S\left(\overline{\mathbf{U}}^{**}\right) \\
\overline{\mathbf{U}}^*_{i,j} &= \mathsf{F}_S\left(\overline{\mathbf{U}}^{n+1}\right) \\
\overline{\mathbf{U}}^{**}_{i,j} &= \mathsf{F}_z\left(\overline{\mathbf{U}}^*\right) \\
\overline{\mathbf{U}}^{n+2}_{i,j} &= \mathsf{F}_x\left(\overline{\mathbf{U}}^{**}\right)
\end{aligned} \tag{8}$$

With the exception of the dimensional splitting and the source term, the accuracy depends entirely on the approximation of the time-averaged interface fluxes.

2.4. Flux evaluations

2.4.1. Flux-Based Characteristic Variables (CVs)

The equation set given in 1 is classified as hyperbolic because after applying the chain rule to the fluxes, the resulting matrix (see below), called the flux Jacobian, can be decomposed into eigenvalues and eigenvectors that are guaranteed to have real (non-imaginary) values. Put in characteristic form, the homogeneous equation set (considering only the x-direction for clarity) becomes:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial x} = 0 \tag{9}$$

where $\partial\mathbf{F}/\partial\mathbf{U} = A$ is the flux Jacobian matrix. From here, we must operate under the assumption of a "locally frozen" Jacobian to use linear characteristic theory. At each interface, the flux Jacobian is locally held constant in time and uniform in space during a time step, computed by state variables which are representative of the local spatiotemporal fluid environment. Once locally froze, the flux Jacobian can be diagonalized into eigenvectors and eigenvalues: $A = R\Lambda L$ where R is a matrix whose columns are right eigenvectors, L is a matrix whose rows are left eigenvectors, Λ is a diagonal matrix whose diagonal components are eigenvalues, and $L = R^{-1}$. For the potential temperature-based Euler equation set (1), the eigenvectors are (in the x- and z-directions):

$$R_x = \begin{bmatrix} 1 & 0 & 1 & 1 \\ u & 0 & u - c_s & u + c_s \\ 0 & 1 & w & w \\ 0 & 0 & \theta & \theta \end{bmatrix}, \quad R_z = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & u & u \\ 0 & w & w - c_s & w + c_s \\ 0 & 0 & \theta & \theta \end{bmatrix} \tag{10}$$

$$L_x = \begin{bmatrix} 1 & 0 & 0 & -\frac{1}{\theta} \\ 0 & 0 & 1 & -\frac{w}{\theta} \\ \frac{u}{2c_s} & -\frac{1}{2c_s} & 0 & \frac{1}{2\theta} \\ -\frac{u}{2c_s} & \frac{1}{2c_s} & 0 & \frac{1}{2\theta} \end{bmatrix}, \quad L_z = \begin{bmatrix} 0 & 1 & 0 & -\frac{u}{\theta} \\ 1 & 0 & 0 & -\frac{1}{\theta} \\ \frac{w}{2c_s} & 0 & -\frac{1}{2c_s} & \frac{1}{2\theta} \\ -\frac{w}{2c_s} & 0 & \frac{1}{2c_s} & \frac{1}{2\theta} \end{bmatrix} \tag{11}$$

where $c_s = \sqrt{\gamma p/\rho}$ is the speed of sound. Also, the corresponding eigenvalues are: $\Lambda_x = \text{diag}(u, u, u - c_s, u + c_s)$ and $\Lambda_z = \text{diag}(w, w, w - c_s, w + c_s)$.

Leveque [33] shows that, for any hyperbolic equation set, the difference in the flux across an interface can be described as a weighted sum of the right eigenvectors:

$$\mathbf{F}(\overline{\mathbf{U}}_i) - \mathbf{F}(\overline{\mathbf{U}}_{i-1}) \equiv \Delta\mathbf{F}_{i-1/2} = \sum_p \beta^p_{i-1/2}\mathbf{r}^p_{i-1/2} \tag{12}$$

Throughout, a superscript p is not an exponent but refers to one of the four characteristic waves admitted by this equation set. A characteristic wave is defined by a right eigenvector $\mathbf{r}^p_{i-1/2}$ (the pth column of R computed at an interface), a left eigenvector $\mathbf{l}^p_{i-1/2}$ (the pth row of L computed at the interface), and an eigenvalue λ^p (the pth diagonal element of Λ). The value, $\beta^p_{i-1/2}$, is a flux difference based CV calculated by $\beta^p_{i-1/2} = \mathbf{l}^p_{i-1/2} \cdot \Delta\mathbf{F}_{i-1/2}$. The eigenvalues, λ^p, define the velocities of the trajectories along which CVs are materially conserved. To compute the interface eigenvectors, L and R, one can use any state variables that are representative of the local surroundings because the CVs are based on the flux vector and not state variables (see [18,32]. Also, Roe-averaging [39] has not yet been derived for the equation set we are using. Therefore, we take a simple average of the surrounding state variables at the interface to construct L and R.

Alternatively, the flux vector itself at an interface (rather than the difference across an interface) can be described as a weighted sum of the right eigenvectors, and this is the approach taken here. To obtain the flux through an interface at a given time, we need the flux vector based CVs arriving at the interface at that time: $w^p_{i-1/2}(t) = \mathbf{l}^p_{i-1/2} \cdot \mathbf{F}(\mathbf{U}, t)$. Then, the flux at a given time is:

$$\mathbf{F}_{i-1/2}(t) = \sum_p w^p_{i-1/2}(t)\mathbf{r}^p_{i-1/2} \tag{13}$$

Now, to obtain time-averaged fluxes, we need to integrate the CVs arriving at the interface over a time step, and this will be discussed in Section 2.4.2.

Advantages of flux-based CVs: There are two main advantages to using flux-based CVs rather than the traditional type built on state variables in this context. First, hydrostatic balance is more easily treated, and we specifically mean the pressure term in the vertical momentum equation. Though we cannot use the highly convenient method of Ahmad and Lindeman [18] because our method is a flux vector splitting, we simply subtract the basic state pressure from the true pressure along the upwind trajectory to achieve a good balance using separate reconstructions. Please note that though we have referenced Ahmad and Lindeman [18] several times regarding the use of flux-based CVs, aside from this similarity the two schemes are dissimilar in the algorithmic implementation and mathematical formulation of flux computation. They differ most fundamentally in that our scheme uses flux vector-based CVs rather than flux difference-based CVs.

If the state variables were cast into characteristics, we could no longer do this. Second, if we computed the time-averaged state variable passing through the interface and computed the flux from them, the flux would no longer be formally high-order accurate. This is because the flux is a non-linear function of state variables so that the true time-averaged flux is not the same as the flux built on time-averaged state variables: $\widehat{\mathbf{F}}(\mathbf{U}) \neq \mathbf{F}\left(\widehat{\mathbf{U}}\right)$. In fact, to equate those two is formally only first-order accurate. Since we compute the flux directly along the upstream trajectory, this assumption is not necessary.

Notes on conservation: In traditional flux-difference splitting schemes, using flux-based CVs enables conservation without having to define a Roe-averaging of the left and right eigenvectors. In this case, however, because of the manner in which this flux vector splitting is performed, conservation is always guaranteed. This is because all a flux form FV method needs to ensure conservation is a single-valued interface flux for each interface.

2.4.2. Temporally averaged fluxes

Time-averaged fluxes may be obtained using the time-averaged CVs along their upstream characteristic trajectories, a result of simply integrating (13) with respect to time: $\widehat{\mathbf{F}}_{i-1/2} = \sum_p \widehat{w}^p_{i-1/2}\mathbf{r}^p_{i-1/2}$. Therefore, the crux of this computation is the time integral of the CVs:

$$\hat{w}^p_{i-1/2} = \frac{1}{\Delta t}\int_0^{\Delta t} w^p_{i-1/2}(t)dt \tag{14}$$

Because the CVs are conserved along characteristic trajectories (again, a consequence of linear characteristics arising from a locally frozen Jacobian) whose velocities are given by the eigenvalues, λ^p, we can trace the CVs *upstream* from the interface using the negated velocity (eigenvalues) and the amount of time they have traveled. A CV arriving a cell interface has the upstream (backwards) trajectory: $x(t) = x_{i-1/2} - \lambda^p t$ which locates the departure points of CVs arriving at the interface at an arbitrary time. Again, because CVs are conserved along their trajectories, the CV value at its departure location is the same as its value at the arrival location (the interface). Assuming a high-order reconstruction of state variables within each cell, $\widetilde{\mathbf{U}}$, the CV value at its departure point is:

$$w^p_{i-1/2}(t) = \mathbf{l}^p_{i-1/2} \cdot \mathbf{F}\left(\widetilde{\mathbf{U}}(x_{i-1/2} - \lambda^p t)\right) \tag{15}$$

2.4.3. Integration procedure

To find the CV values at departure locations, we first reconstruct the state variables, $\overline{\mathbf{U}}_i$, themselves over a stencil to provide a functional approximation, $\widetilde{\mathbf{U}}_i(x)$, inside each cell on the domain. Next, we integrate $w^p = \mathbf{l}^p \cdot \mathbf{F}(\widetilde{\mathbf{U}})$ in time along the upwind trajectory via Gauss–Legendre (GL) quadrature up to a desired accuracy. Assuming GL weights, ω_m, corresponding to GL point locations, $x_{m,p} = x_{i-1/2} - \lambda^p \Delta t \kappa_m$, the pth time-averaged CV passing through interface $x_{i-1/2}$ is:

$$\hat{w}^p_{i-1/2} = \mathbf{l}^p_{i-1/2} \cdot \left[\sum_{m=1}^{n_G} \omega_m \mathbf{F}\left(\widetilde{\mathbf{U}}_{i+\alpha^p}(x_{m,p})\right)\right]$$

where $\alpha^p \in \{-1, 0\}$ is used to locate the upwind cell, n_G is the number of GL points, and κ_m are the standard GL weights transformed to the domain $[-1, 1] \to [0, \Delta t]$. After computing the time-averaged CVs, we multiply them by the interface right eigenvectors and sum to recover the time-averaged interface flux to complete the flux computation. This process has been described in the *x*-direction neglecting the *j* subscript for simplicity of notation. The process is the same in the *z* direction.

In this study, we use a fifth-order accurate WENO reconstruction [35,34,36]. The WENO philosophy involves computing polynomials over multiple stencils and weighting the least oscillatory ones the most. This weighted sum produces a smooth and non-oscillatory interpolant. Because of up to fifth-order accuracy of the spatial interpolant, we use a 3-point (sixth-order accurate) GL quadrature rule for present simulations. This method can use any single-moment spatial interpolant and works on any hyperbolic equation set. Even if the interpolation is non-conservative, the simulation will still locally and globally conserve state variables. This is because a single-valued interface flux is sufficient for conservation in a flux vector based FV method, and we obtain this regardless of the interpolant.

For cases in which there is a hyperbolic and non-hyperbolic portion to the governing equation set, a typical splitting technique can be employed as it is here. Particularly, in convectively dominated flows, viscous fluxes can usually be split off even in a first-order accurate manner. In fact, in the Straka density current test cases employed here, we use this simple approach for the viscous updates to avoid any additional communication in the model as a whole and still obtain the expected solutions. However, splitting due to multidimensional simulation and due certain important source terms (like gravity treated here or Coriolis effect in global models) needs more careful consideration. As mentioned, here we use a second-order accurate alternating Strang splitting for more tightly coupled components.

2.4.4. Extension to large CFL numbers

In order to extend the method to a larger CFL number (here we go up to 7 for the smoothest test case), we simply perform the integration procedure from Section 2.4.3 over more than one cell along the upstream trajectory. To respect the discontinuities across cell boundaries due to the WENO reconstruction, we perform a separate quadrature within each cell. A path-length-weighted sum of the individual cell averages along the upwind trajectory renders the averaged characteristic variable which can then be cast into flux components and summed to obtain the time-averaged flux over more than one cell. Note that we are assuming a constant wind speed from the locally frozen Jacobian for the present, and therefore we will experience some accuracy degradation. We leave it to future research to improve the trajectories and characteristics computations. A schematic of the process for computing time-averaged characteristic variables is given in Fig. 1.

2.5. Hydrostatic balance and material boundaries

For density stratified fluids, hydrostatic balance (defined by $\partial p/\partial z \approx -\rho g$, a balance between the vertical pressure gradient and gravity source term) dominates the vertical momentum equation. Solid wall (material surface) boundary conditions in the vertical direction can be difficult to maintain in the presence of hydrostatic balance. A traditional (and the simplest) technique for implementing solid wall boundaries in the vertical direction while maintaining high-order accuracy down to the surface is to enter "ghost cell" values past the boundary which mirror the interior dynamics. This enforces a gradient of zero for density, transverse momentum, potential temperature, and pressure, and it enforces zero normal velocity. In the case of hydrostatic balance, the pressure gradient is clearly non-zero at the material surfaces, making this technique invalid.

Typically for non-hydrostatic models, hydrostatic balance is subtracted from the vertical momentum equation (see e.g. [40], leaving only perturbations from hydrostatic balance for the vertical pressure gradient and for density in the gravity source term. This is done for many reasons, the dominant reason being that in the presence of terrain, large cancellation errors can occur otherwise which would lead to poor representation of the pressure gradient forcing (because the vertical gradient dominates). When the dominant balance is removed, we again have similar gradient magnitudes in the horizontal and vertical direction. Also, with perturbations from hydrostatic state, we can then assume that the vertical gradient of the pressure perturbation is zero (at least for this flat geometry, not sloping terrain necessarily), enabling a simplistic ghost cell treatment of the boundaries. We adopt this technique, giving the following equations:

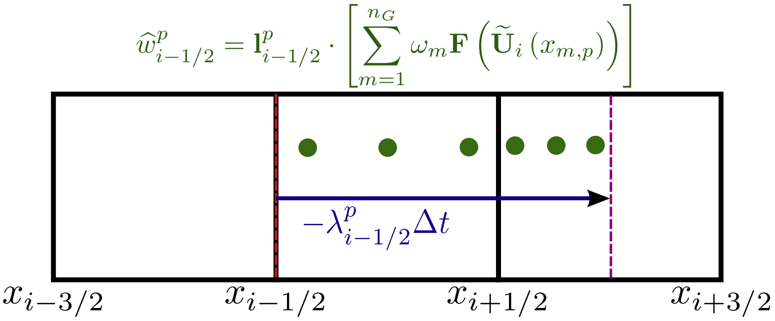

$\widehat{w}^p_{i-1/2} = \mathbf{l}^p_{i-1/2} \cdot \left[\sum_{m=1}^{n_G} \omega_m \mathbf{F}\left(\widetilde{\mathbf{U}}_i\left(x_{m,p}\right)\right) \right]$
$-\lambda^p_{i-1/2}\Delta t$
$x_{i-3/2}$
$x_{i-1/2}$
$x_{i+1/2}$
$x_{i+3/2}$

Fig. 1. Schematic of the process for computing time-averaged characteristic variables with CFL = 1.5 for the interface with a red dashed line. The blue arrow is the upwind trajectory, and the violet dashed line is the departure location. Dark green circles denote quadrature points at which the flux is calculated from reconstructions to compute characteristic variables at locations $x_{m,p}$. Note separate quadrature within each cell. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho w\\ \rho\theta\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uw\\ \rho u\theta\end{bmatrix}+\frac{\partial}{\partial z}\begin{bmatrix}\rho w\\ \rho wu\\ \rho w^2+(p-p_H)\\ \rho w\theta\end{bmatrix}=\begin{bmatrix}0\\ 0\\ -(\rho-\rho_H)g\\ 0\end{bmatrix} \tag{16}$$

Handling the source term is trivial because we only need the cell mean perturbation by subtracting off the cell-mean hydrostatic state. However, handling the pressure term in the vertical flux vector is not as simple. We need a high-order accurate approximation to the perturbation along arbitrary characteristic trajectories. We found it sufficient to perform an initial reconstruction of the hydrostatic basic state of potential temperature $\widetilde{(\rho\theta)}_H(z)$ (where an H subscript is a hydrostatic basic state and a $\sim$ superscript is a reconstruction) and compute a hydrostatic pressure from this $p_H(z) = p(\widetilde{(\rho\theta)}_H(z))$. Then, the difference can be computed along the trajectory at quadrature points as $p - p_H = p(\widetilde{\rho\theta}(z)) - p(\widetilde{\rho\theta_H}(z))$ at each quadrature point along the upstream trajectory. The perturbation flux is then used to compute CVs. If the present method is insufficient for curvilinear geometry or other factors in another application, then the perturbations will need to be computed cell-wise and then reconstructed (rather than separately reconstructed and sampled), increasing computational requirements.

As for the characteristics and their interaction with material boundaries, the normal slow waves should never reach the material boundary. This is because the normal velocity is identically zero at the boundary, meaning the wave asymptotically may not reach it. There are ways of handling this, standard in the semi-Lagrangian model literature (see, e.g. [41]. We do not currently need to employ this because our Mach numbers are sufficiently small that for the CFL numbers we are using, the slow waves will not reach the material boundaries anyway. As for the acoustic waves, algorithmically we reflect them back into the physical domain, (to emulate mirrored ghost cell reconstructions), and we negate the normal velocity to ensure it is zero at the boundary.

2.6. Temporal accuracy

In this fully discrete method, the temporal integral is computed directly (essentially forward Euler in nature). We cast the time integral over one time step into a spatial integral over the upwind trajectory path for each CV. If the wave speed remains constant over the time step, the two integrals are identical. If the spatial reconstruction is accurate to $O(\Delta x^n)$, then the accuracy in time is also $O(\Delta x^n)$ if wind is constant. Also, because of the CFL restriction, $\Delta t \propto \Delta x$ which means that the temporal accuracy is also $O(\Delta t^n)$. However, the wave speeds are not in general constant in time, so this assumption obviously incurs error temporally, and we will address relaxing this assumption in future research. Temporal accuracy is still formally restricted to second-order regardless of the 1-D truncation error in individual sweeps because of the dimensional splitting we use. Still, for CFL restricted problems (i.e. explicit time integration), spatial error dominates the total truncation error. This argument is similar to the many Lagrangian single-step, single-stage transport methods used in atmospheric models.

2.7. Flux computation summary

Here, we summarize the algorithm for computing an interface flux. In the vertical direction, assume a reconstruction of a hydrostatic basic state potential temperature, $\widetilde{\rho\theta_H}$, is subtracted from the cell mean potential temperature $\widetilde{\rho\theta}$ in step 2. → (a) → ii.

1. Form left and right eigenvectors and eigenvalues (wave speeds) by averaging left and right limits from $\widetilde{\mathbf{U}}$ at the interface
2. For each of the four waves
   - (a) For each quadrature point in time
     - i. Trace quadrature point upstream in time using eigenvalues
     - ii. Compute the flux vector at this location from $\widetilde{\mathbf{U}}$
     - iii. Compute the CV for this wave via a dot product of the left eigenvector and the flux vector
   - (b) Compute the time-averaged CV from quadrature points
   - (c) Compute the flux update: a product of the right eigenvector and the time-averaged CV
3. Sum the four flux updates to compute the time-averaged interface flux

2.8. Comparison of communication burden

An exhaustive comparison of overall performance on large parallel platforms is certainly beyond the scope of this paper. However, we do wish to provide a quantitative discussion of the communication needs of our method compared to a multi-stage method. There are design constraints when considering a parallel algorithm via domain decomposition such as how many communications are performed and how much duplication of work there is. Regarding the former, we will consider the communication burden as proportional to the total amount of communication required per amount of time in simulation.

For the latter, we will consider multiple scenarios for each algorithm. For the FBCSL algorithm, we take two scenarios: (1) no duplicated work and (2) duplicated reconstructions. We choose only duplicated reconstructions for scenario 2 because the flux stage is much more expensive than the reconstruction stage. For the multi-stage algorithm, we will take again two scenarios: (1) no duplicated work and (2) duplicated reconstruction and fluxes. This is not overly optimistic for the multi-stage communication patterns because the reconstructions and fluxes are both relatively cheap in a communication dominated context. Quantitative details of the scenarios are given in the following itemization.

Consider n to be the number of variables per cell, c to be the CFL number, and s to be the number of stages for the multi-stage solvers. The variable c arises to account for the fact that the FBCSL methods can perform at large CFL numbers and thus need to communicate more than just one cell of information. For instance, suppose our reconstruction requires five cells in total: the cell in question plus two on either side. The halo region would be two cells. A method limited to CFL < 1 will only need to reconstruct the cell adjacent to a domain decomposition boundary (DDB). Therefore, in all, three cells are communicated to reconstruct that adjacent cell. If the CFL is raised to CFL < 3, now we need to reconstruct three cells past the DDB. Thus, in all, five cells are communicated for the purpose of reconstruction. So generalizing this, $c + 2$ (where 2 is the halo region) cells must be communicated for reconstruction. Here, I will use the term "float" to refer to a 4-byte floating point variable and "double" to refer to an 8-byte floating point variable.

- *FBCSL Scenario* 1 We need to communicate $2n$ floats (two cells of state vectors) across a domain decomposition boundary to form the reconstructions, and then $5nc$ floats to communicate c reconstructions (5 coefficients per state variable per cell) to form fluxes at large CFL numbers, and then n floats (one flux vector) to communicate interface fluxes for updating. Doing this in two directions gives a total of $2nc + 3)$ floats of communication.
- *FBCSL Scenario* 2 We need to communicate $(2 + c)n$ floats (enough state vectors to reconstruct c cells) for reconstructions, and n floats (one flux vector) to communicate interface fluxes. Doing this in two directions gives a total of $2n(c + 3)$ floats of communication.
- *Multi-stage Scenario* 1 Each stage will need to communicate $2n$ floats (two cells of state vectors) for reconstruction, n floats for the interpolated interface state vector, and n floats for the flux vector. With s stages and two directions, this would be: $2c(4ns)$.
- *Multi-stage Scenario* 2 Each stage will need to communicate $2n$ floats for reconstruction, and the interpolation and flux computation will be duplicated. In s stages, this is $2ncs$ communications. In two directions, this is: $4ncs$.

We will assume a 3-stage method (typically 3rd-order accurate), and for fair comparison, we will compare like scenarios. Computing the ratio of communications for Scenario 1, we get a ratio (FBCSL/multi-stage) of about $0.417 + 0.25/c$. The ratio for scenario 2 is $0.167 + 0.5/c$. One can see that simulating at higher CFL number (larger c) is relatively more effective for scenario 2 than it is for scenario 1. Regardless, however, the FBCSL method requires at most 2/3 the amount of communication required by the 3-stage solver. For CFL number values of 2–3, the FBCSL methods requires only 1/3 to 1/2 as much communication. The question of overall efficiency regarding the tradeoffs of duplication and communication and the overall cost of the two algorithms will be highly machine and implementation dependent and is, again, beyond the scope of this paper. We are simply demonstrating that the FBCSL method can operate with less overall communication requirements.

3. Numerical results

Some standard benchmark test cases are performed to evaluate the ability of the proposed solver to effectively simulate non-hydrostatic atmospheric dynamics: a rising convective thermal, a density current, and internal gravity waves in the non-hydrostatic regime. These are the same test cases as in Ahmad and Lindeman [18], Straka et al. [42] and references therein. The convective bubble test case embodies a phenomenon of great interest to mesoscale type flows. The Straka density current mimics cold outflow from a convective system and tests a methods ability to control oscillations when run without numerical viscosity. Finally, the internal gravity waves test case tests a model's resolution of a smooth phenomenon on non-hydrostatic scales which transfers sizable amounts of energy on both mesoscales and global scales.

Because these test cases have no analytical solution, they must be evaluated qualitatively. Full conservation was achieved in all state variables in each test case except the density current because the diffusion prescribed for the test case was not in conservation form.

Hydrostatic initialization.

Constant potential temperature. To initialize hydrostatic balance, it is easiest to obtain a vertical profile for Exner pressure, π, rather than pressure directly. Exner pressure is a function of pressure only, given by:

$$\pi = \left(\frac{p}{p_0}\right)^{R_d/c_p} \tag{17}$$

And its hydrostatic balance equation is given in terms of only potential temperature:

$$\frac{d\pi}{dz} = -\frac{g}{c_p\theta} \tag{18}$$

The first two test cases assume a constant potential temperature basic state. Therefore, the hydrostatically balanced Exner pressure profile is trivial:

$$\bar{\pi}(z) = \bar{\pi}_{sfc} - \frac{gz}{c_p\theta_0} \tag{19}$$

where we assume $\bar{\pi}_{sfc} = 1$ meaning $p = p_0$ at the surface.

Constant Brunt–Vaisala frequency. In the gravity wave test case, a constant Brunt–Vaisala frequency, N_0, is specified. The Brunt–Vaisala frequency is given in terms of fractional vertical gradient of potential temperature:

$$\sqrt{\frac{g}{\theta}\frac{d\theta}{dz}} = N_0$$

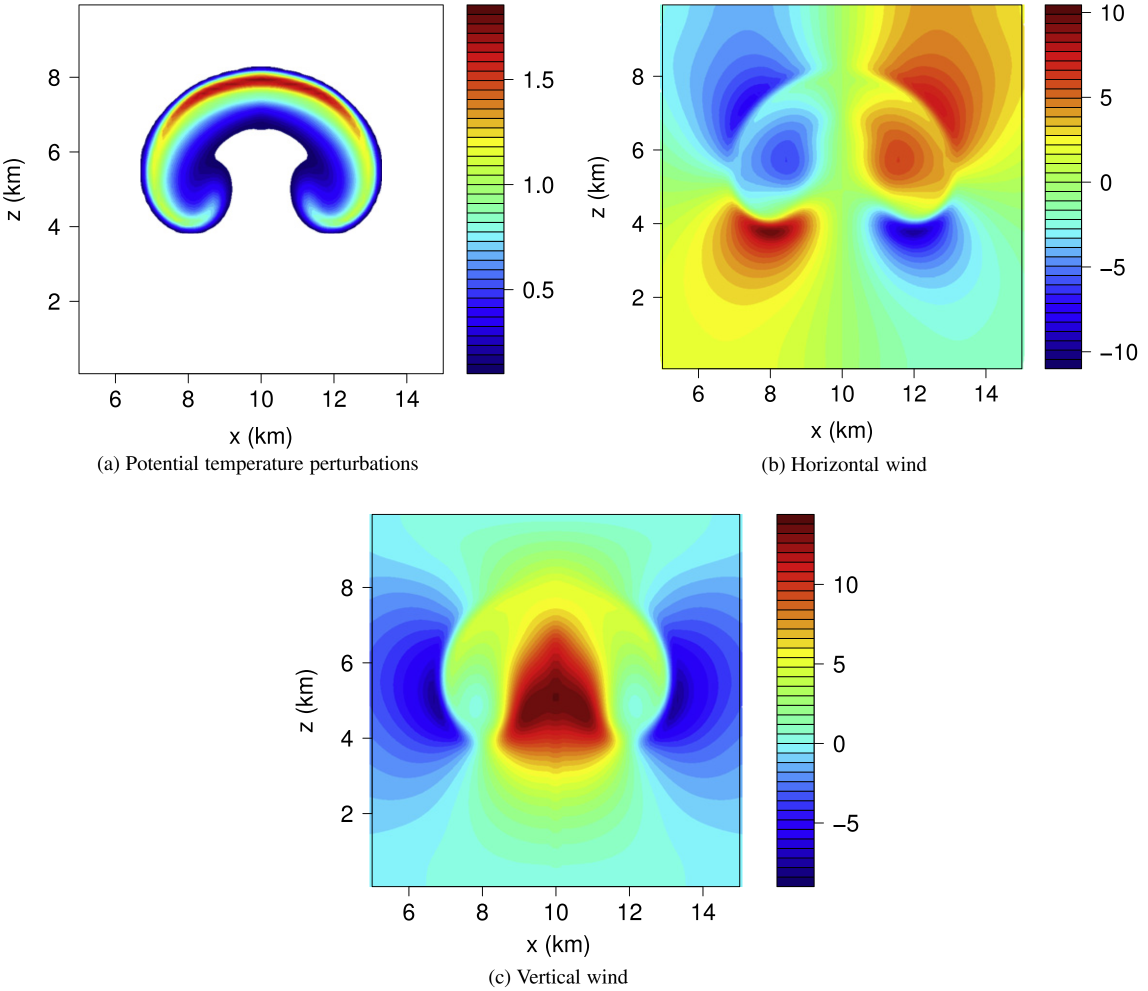
Therefore,

$$\theta(z) = \theta_{sfc}e^{\frac{N_0^2}{g}z}$$

Plugging this into (18), we eventually obtain the following vertical profile:

$$\bar{\pi}(z) = \bar{\pi}_{sfc} - \frac{g^2}{c_pN_0^2}\left(\frac{\theta(z) - \theta_{sfc}}{\theta(z)\theta_{sfc}}\right) \tag{20}$$

We set the constants as follows: $\bar{\pi}_{sfc} = 1,\ \theta_{sfc} = 300$ K, and $N_0 = 10^{-2}\ \text{s}^{-1}$.



(a) Potential temperature perturbations

(b) Horizontal wind

(c) Vertical wind

Fig. 2. Plots for the convective thermal test case with a 125 m grid spacing after 1,000 s of simulation. *x*- and *y*-axes are in km, potential temperature perturbations are in K, and winds are in ms^{-1}.

Pressure can be obtained from (19) or (20) by the Exner pressure equation, (17). Then, density may be extracted from the pressure by the equation of state. Many studies simply use the cell mid-point value for initialization (at most second-order accurate) in a FV context. In this study, however, a five-point (ninth-order accurate) Gauss–Legendre quadrature is used to initialize the cell means. Perturbations for the various test cases are included in the quadrature for a high-order accurate initialization. Also, hydrostatic balance is not restored after addition of the perturbations in this case, so the initial thermal should be interpreted as an instantaneous shock to the hydrostatically balanced system.

3.1. Convective thermal

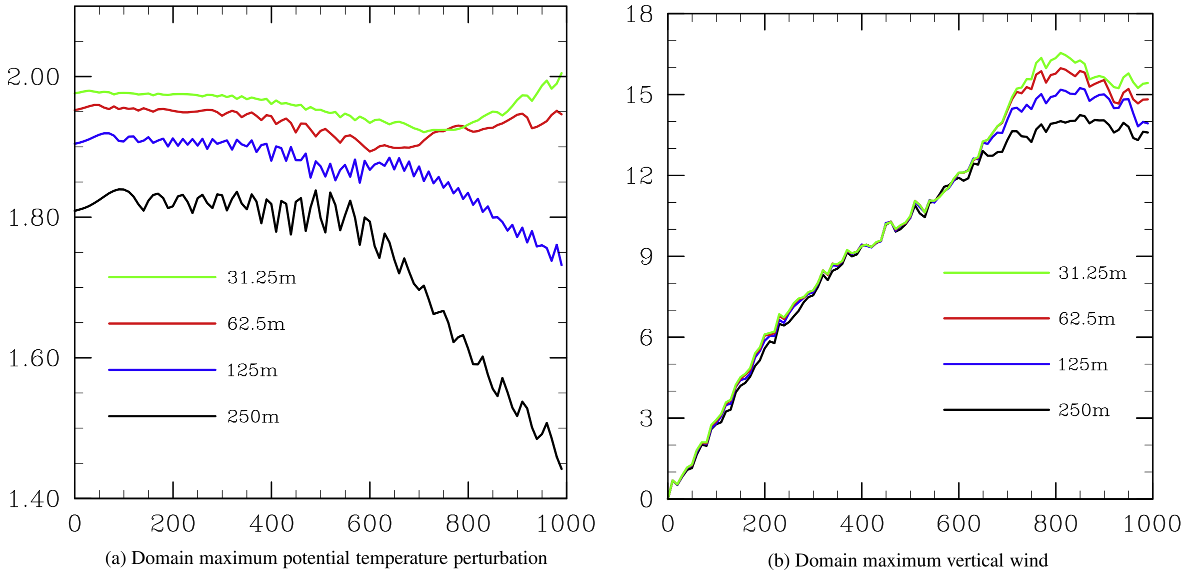
The convective thermal uses a hydrostatic balance based on a uniform potential temperature, $\theta_0 = 300$,K, and then adds the following perturbation in potential temperature: $\theta = \theta_0 + \Delta\theta\max(0, 1 - D/R)$ where R is the radius of the bubble, and D is the distance from the center of the bubble given by: $D = \sqrt{(x - x_0)^2 + (z - z_0)^2}$. For this test case, we define $\Delta\theta = 2$ K, $R = 2$ km, $x_0 = 10$ km, and $z_0 = 2$ km. The model domain is $[0,20] \times [0,10]$ km. The horizontal and vertical wind are both initialized to zero, and the thermal is simulated at a maximum Courant number of 0.98.

The potential temperature perturbations, horizontal wind, and vertical wind with a grid spacing of 125 m are contoured in Fig. 2. Time traces (sampled every 10 s) of the domain maximum potential temperature perturbation and vertical wind are also given in Fig. 3. The maximum wave speed at all four resolutions was $|\vec{V}| + c_s \approx 349$ ms^{-1}.

Qualitatively, the simulation matches well with other studies, and because we initialize cell averages and not cell mid-points, the results will not be exactly alike. The flow at this time and on these scales exhibits no large-scale turbulence, leading to a sense of convergence as the grid is refined. We run at multiple resolutions to show how the standard resolution (125 m) for this test case compares to a higher resolution solution (31.25 m). We start with a time step of 0.35 s at $\Delta x = 125$ m and decrease it linearly with the grid spacing from there.

Time traces in Fig. 3 seem to show oscillations, but they occur because the maximum in a variable may be split between two cells at one time and in a single cell at another time. This is further supported by the observation that they decrease substantially as the grid spacing decreases. With 31.25 m grid spacing, the domain maximum potential temperature exceeds the initial value towards the end of the simulation, potentially a numerical overshoot. There is considerable agreement at all resolutions for the domain maximum vertical wind until about 650 s. At this point, the higher resolutions are resolving smaller-scale flows which concentrate the potential temperature near the top of the "mushroom".

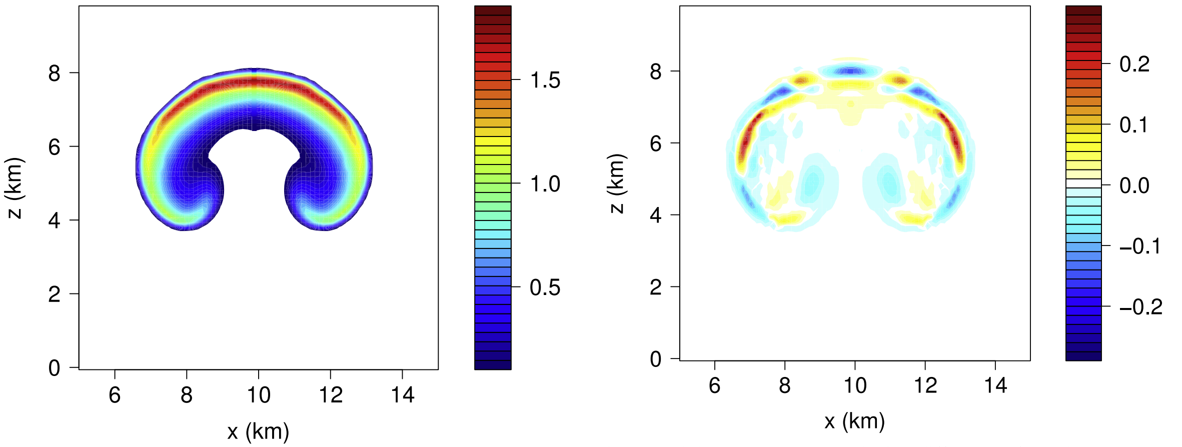
Fig. 4 gives a plot of the potential temperature perturbations for a convective thermal test case run with a maximum CFL number of 1.96 with 125 m grid spacing and differences from the previous run. The largest magnitude difference from the previous run is 0.27 which represents about a 13% departure at that location. Given the likelihood that the 0.98 CFL number run is more accurate, we can take from this that there probably needs to be some adjustments to the trajectory accuracy in



(a) Domain maximum potential temperature perturbation

(b) Domain maximum vertical wind

Fig. 3. Domain maximum potential temperature and vertical wind traces for the convective thermal test case over a range of grid spacings. x-axis is time in seconds and y-axis is K for potential temperature trace and ms^{-1} for vertical wind trace.



(a) Potential temperature perturbations

(b) Potential temperature perturbation differences from the CFL=0.98 run.

Fig. 4. Plots for the convective thermal test case with a 125 m grid spacing after 1,000 s of simulation with a CFL number of 1.96. *x*- and *y*-axes are in km, potential temperature perturbations are in K.

order to increase the accuracy of the overall simulation. Stability is achieved with the larger CFL number despite any accuracy concerns.

3.2. Straka density current

This test case uses a hydrostatic balance, again based on a uniform potential temperature, $\theta_0 = 300$ K, and then adds the following perturbation in potential temperature:

$$\theta = \begin{cases} \theta_0 & \textit{if} \quad L > 1 \\ & \textit{OR} \\ \theta_0 + \Delta\theta[\cos(\pi L) + 1]/2 & \textit{otherwise} \quad L \leqslant 1 \end{cases}$$

$$L = \sqrt{\left(\frac{x - x_0}{x_R}\right)^2 + \left(\frac{z - z_0}{z_R}\right)^2}$$

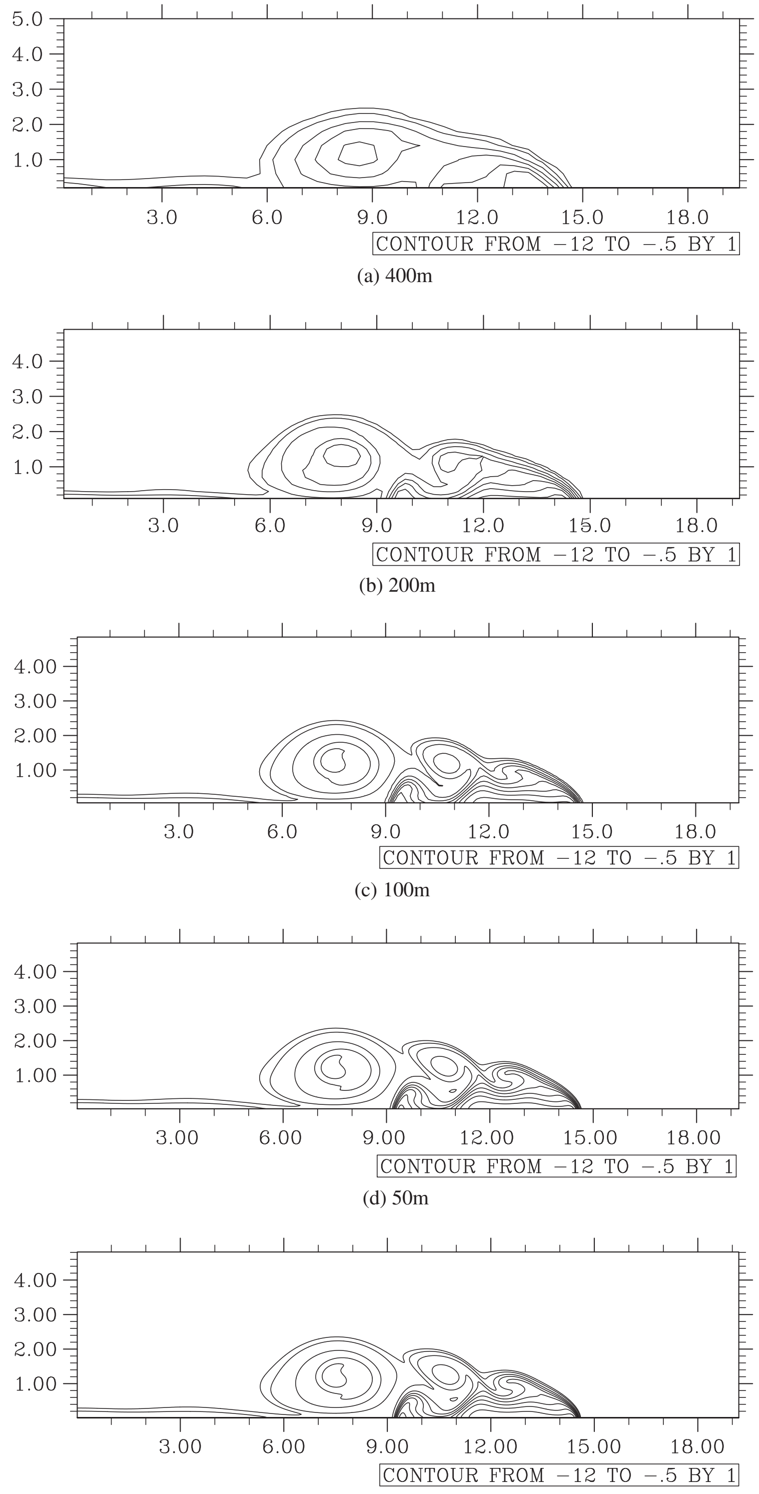
We define $\Delta\theta = -15$ K, $x_R = 4$ km, $z_R = 2$ km, $x_c = 0$ km, and $z_c = 3$ km. The model domain is $[-26.5, 26.5] \times [0, 6.4]$ km with a grid spacing varying from 25 m to 400 m. The simulation is run for 900 s with a maximum Courant number of 0.97 and a maximum wave speed of $|\vec{V}| + c_s \approx 385$ ms^{-1}. This is a good test case for examining the oscillatory properties of a scheme because of the many strong gradients in both the wind and potential temperature.

For proper grid convergence, the test case prescribes a dynamic viscosity. To accommodate this, we include simple centered second-order accurate finite difference approximations to diffusion terms for the momentum equations and potential temperature equation. The horizontal momentum, vertical momentum, and potential temperature diffusion terms are given, respectively, by $\rho K(u_{xx} + u_{zz})$, $\rho K(w_{xx} + w_{zz})$, and $\rho K(\theta_{xx} + \theta_{zz})$ where $K = 75$ m^2 s^{-1} is the coefficient of diffusion. Even though viscosity is present, we use free slip boundary conditions at the domain bottom (meaning the wind is not forced to zero there).

Fig. 5 shows potential temperature perturbation contours for resolutions at 400 m, 200 m, 100 m, 50 m, and 25 m. Table 1 gives the maximum and minimum potential temperature perturbation at 900 s of simulation. The lack of a 0 K perturbation contour is solely for the sake of plot clarity. We start with a time step of 1 s at $\Delta x = 400$ m and decrease it linearly with the grid spacing from there.

The plots are in line with other studies. One thing to keep in mind which makes this simulation slightly different than others is that we initialize the model with cell averages computed via quadrature and not cell midpoint values. Therefore, our minimum potential temperature perturbation will be slightly smaller in magnitude than other studies.

We also ran this test without explicit viscosity to demonstrate the stabilizing properties of the WENO interpolant at Courant numbers near 1 and near 2. Fig. 6 shows a plot of the density current test case with only implicit numerical viscosity run at a maximum CFL number of 0.98 and 1.95. We ran the near-unity CFL number test case with a time step of 0.126 s, and both runs had a grid spacing of 50 m. The strong gradients create difficulty for a numerical method to remain stable, but the WENO interpolants provide sufficient implicit diffusion for stability. For the simulation without explicit diffusion at CFL$_{\max} = 0.98$, the minimum θ' was -12.1 K and the maximum θ' was 0.383 K. There are noticeable differences between



(a) 400m

(b) 200m

(c) 100m

(d) 50m

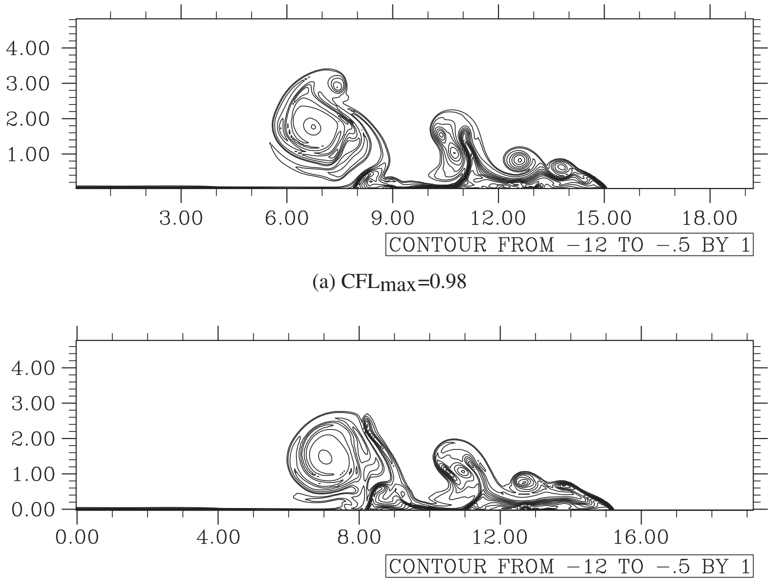
(e) 25m

Fig. 5. Plots for the Straka density current test case after 900 s with diffusion for grid spacings ranging from 25 m to 400 m. *x*- and *y*-axes are in km and potential temperature perturbations are in K.

the two runs at near unity CFL number and larger CFL number which demonstrate the potential advantages of improving trajectory calculations. Still, for a turbulent flow they are similar, and stability was maintained at the higher CFL number without explicit diffusion.

Table 1
Maximum and minimum potential temperature perturbation at 900 s for the Straka density test case with diffusion.

	Grid spacing				
	400 m	200 m	100 m	50 m	25 m
Time step (s)	1	0.5	0.25	0.125	0.0625
Max θ' (K)	1.52e−1	5.20e−2	9.59e−3	6.52e−3	1.22e-07
Min θ' (K)	−6.09	−7.45	−8.48	−8.70	−8.74



(a) $\mathrm{CFL}_{\max}$=0.98

(b) $\mathrm{CFL}_{\max}$=1.95

Fig. 6. Contours of potential temperature perturbation for the Straka density current test case without explicit numerical viscosity after 900 s with a grid spacing of 50 m. *x*- and *y*-axes are in km and potential temperature perturbations are in K. For visual clarity, they are plotted with half the number of contours as Fig. 5.

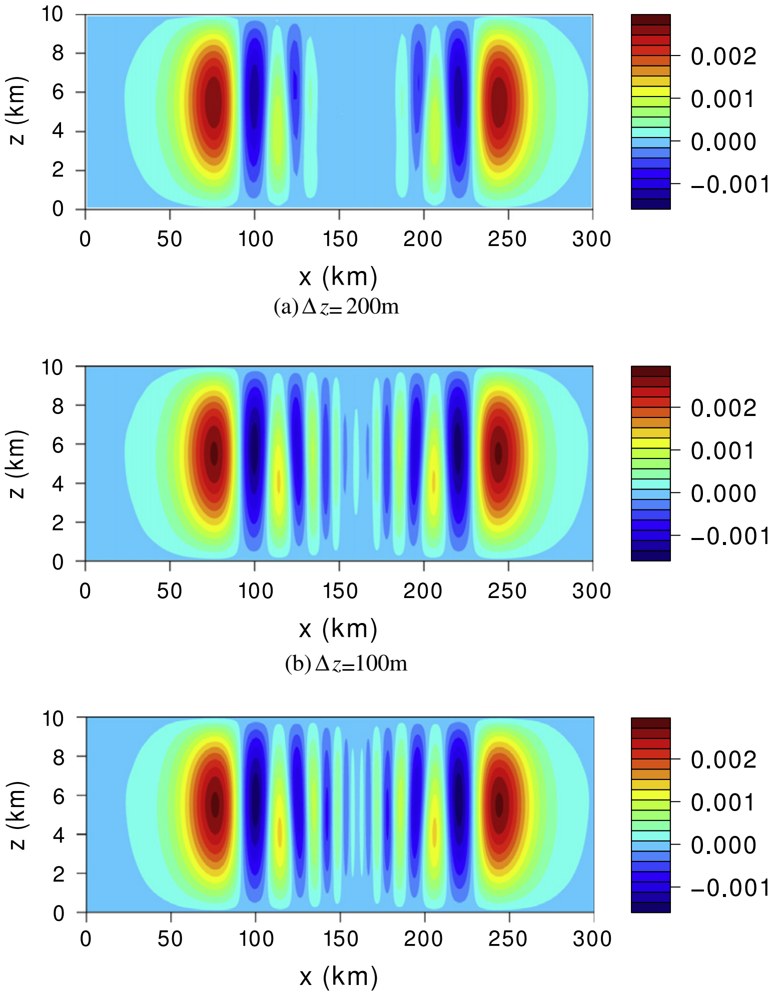
3.3. Non-hydrostatic internal gravity waves

Here, we test the proposed scheme in its handling of Internal Gravity Waves (IGWs) on a non-hydrostatic scale. The domain is initialized hydrostatically with a constant Brunt–Vaisala frequency of $N = 10^{-2}\ \mathrm{s}^{-1}$ to admit IGWs. A potential temperature perturbation is added to the potential temperature field as follows:

$$\theta = \theta_0(z) + \Delta\theta \frac{\sin(\pi z/H)}{1 + (x - x_0)^2/a^2}$$

where $H = 10$ km, $\Delta\theta = 10^{-2}$ K, $a = 5$ km, and $x_0 = 100$ km. The simulation is run for 3,000 s on a domain of $[0,300] \times [0,10]$ km with a maximum Courant number of about 0.99. The initial vertical wind is set to 0 ms^{-1}, and the initial horizontal wind is set to 20 ms^{-1} to advect the entire IGW train in the positive *x*-direction. We ran this simulation at three vertical grid spacings ranging from 50 m to 200 m. The horizontal grid spacing is always ten times greater than the vertical in this test case. We ran with a time step of 0.54 s for $\Delta z = 200$ m and decreased it linearly with grid spacing from there.

Fig. 7 gives contour plots of the potential temperature perturbation, and Fig. 8 gives the potential temperature perturbations along the line $z = 5$ km after 3,000 s. These results agree well with previous studies, and it is the only test case which really converges to a solution as the grid is refined without requiring diffusion. Therefore, we use this test case to get some notion of the numerical convergence of the scheme as both grid and time step are refined using the 25 m grid spacing as the "exact" solution. If we consider the error as a function of grid spacing, $E(\Delta x) = C(\Delta x^n)$ with C being a constant with respect to Δx, then the order of convergence is given by $n = \ln(E(\Delta x)/E(\Delta x/f))/\ln f$. We estimate error norms by regridding the 50 m potential temperature perturbations to the coarser grid. Table 2 summarizes the error norms, and Fig. 9 shows a log–log plot with the slope showing the convergence. The L_1 error norms seem to be asymptoting to near fourth-order convergence, and



(a) $\Delta z = 200$m

(b) $\Delta z = 100$m

(c) $\Delta z = 50$m

Fig. 7. Plots for the internal gravity waves test case after 3,000 s with a range of grid spacings with $\text{CFL}_{\text{max}} = 0.99$. $\Delta x = 10\Delta z$ for all simulations. The *x*- and *y*-axes are in km and potential temperature perturbations are in K.

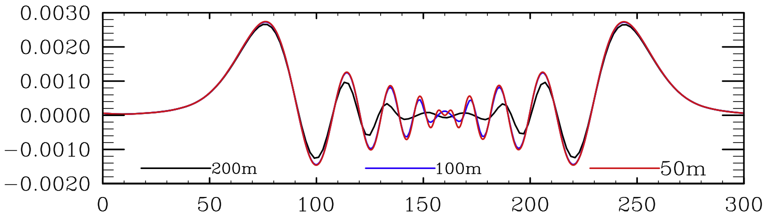


Fig. 8. Plot of potential temperature perturbations along the line $z = 5$ km for the internal gravity waves test case after 3,000 s with a range of grid spacings. $\Delta x = 10\Delta z$ for all simulations. The *x*-axis is in km and the *y*-axis is in K.

Table 2
Error norms in the potential temperature field compared to 25 m results.

Δz	400 m	200 m	100 m	50 m
L_1	0.422339	0.156702	0.0236569	0.00172828
L_2	0.449548	0.187334	0.0356143	0.00300876
L_∞	0.480062	0.234141	0.0703918	0.00899815

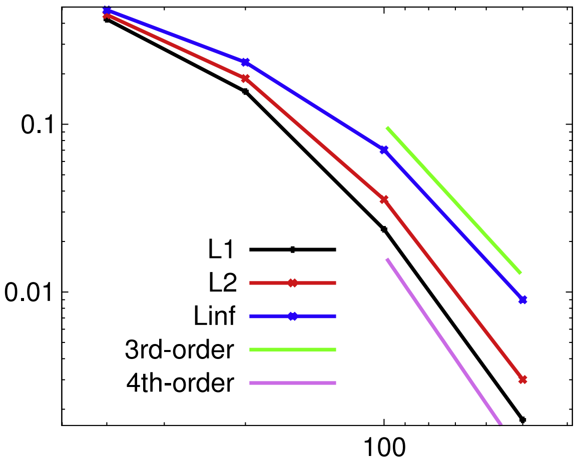
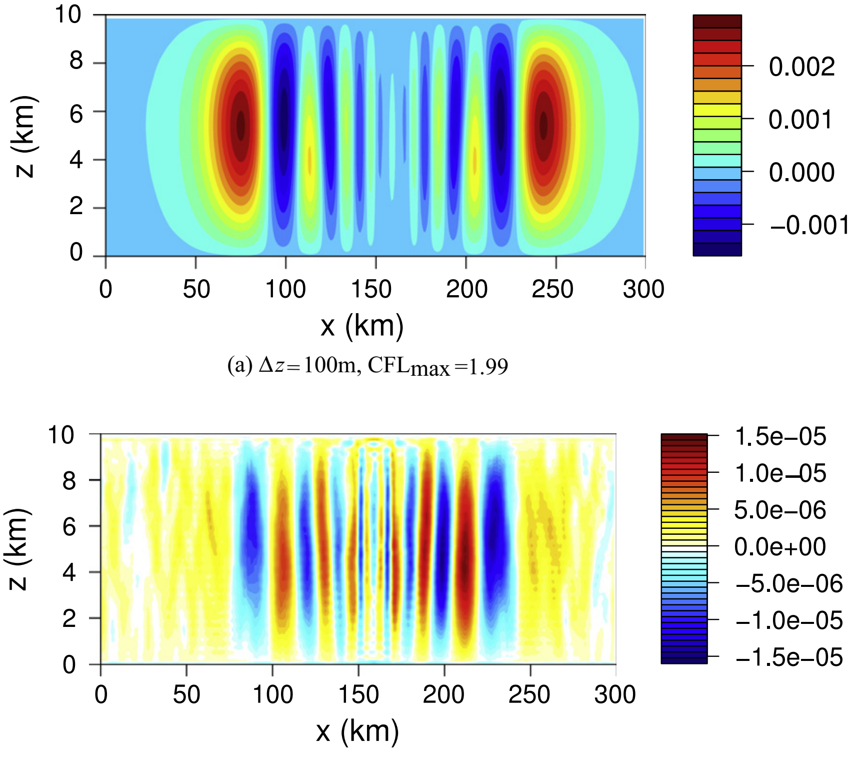


Fig. 9. Log–log plot of potential temperature errors (increasing upward) as a function of grid spacing (decreasing to the right). We used the 25 m run to represent the exact answer and add lines whose slopes show visually the order of convergence.

the L_∞ norms converge more slowly which is typical. We note that this test case is very smooth, and this sort of convergence is very unlikely in realistic flows.

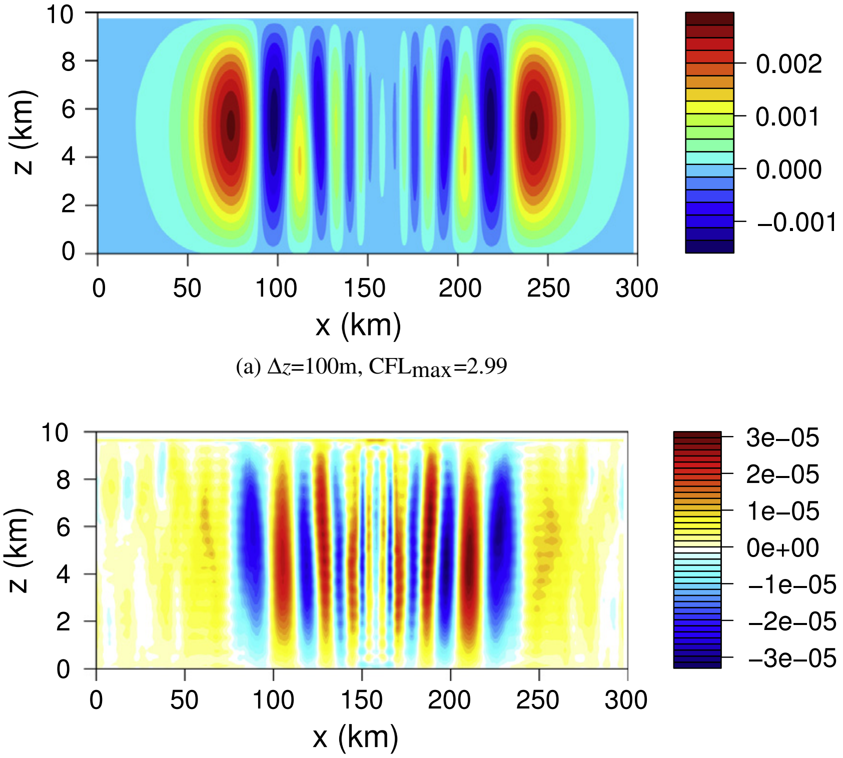
We also ran the 100 m (vertical) grid spacing run with CFL_{max}=1.99, and the potential temperature shaded contours are given in Fig. 10 along with the differences from the previous 100 m run. The maximum deviation from the previous run was only 1.5×10^{-5} K which represents 0.5%. Because this is such a smooth flow with such small changes in characteristic velocities, we also ran the test case at a CFL number of 2.99 and 3.99 with plots shown in Figs. 11 and 12. For CFL_{max} = 2.99 and CFL_{max} = 3.99, the maximum deviations from the standard run were 1.1% and 2.0%, respectively. The maximum difference in potential temperature from CFL = 0.99 as CFL number increases (running with CFL numbers up to 7) fits a quadratic function



(a) Δz=100m, CFL_{max}=1.99

(b) Differences between CFL=1.99 and CFL=0.99

Fig. 10. Plots for the internal gravity waves test case after 3,000 s with a maximum CFL number of 1.99. $\Delta x = 10\Delta z$ = 1,000 m. The x- and y-axes are in km and potential temperature perturbations are in K.



(a) Δz=100m, CFL$_{\text{max}}$=2.99

(b) Differences between CFL=2.99 and CFL=0.99

Fig. 11. Plots for the internal gravity waves test case after 3,000 s with a maximum CFL number of 2.99. $\Delta x = 10\Delta\ z = 1{,}000$ m. The *x*- and *y*-axes are in km and potential temperature perturbations are in K.

Table 3
Maximum absolute difference of potential temperature from the CFL = 0.99 run with increasing CFL number.

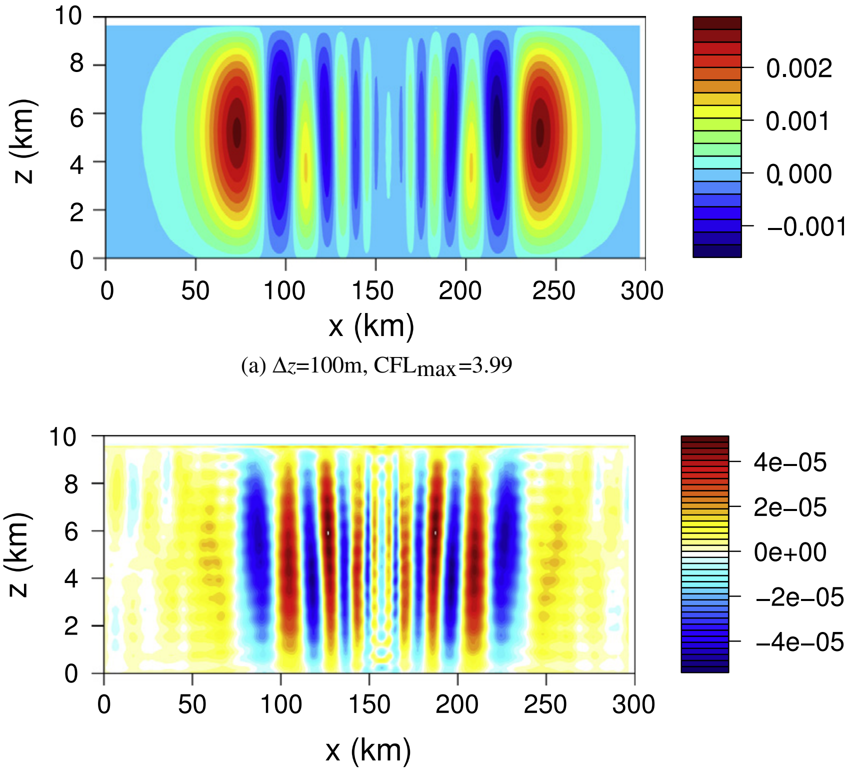
CFL number	1.99	2.99	3.99	4.99	5.99	6.99
Max. abs. diff.	1.505e−5	3.259e−5	5.295e−5	8.370e−5	1.383e−4	1.709e−4

with a squared residual of 0.997 (see Table 3). Future work is necessary to find what leads to this quadratic relationship. The fact that we obtained better results for smoother flows in which characteristic trajectory gradients are small coincides with the findings in Toda et al. [28], Ii and Xiao [31] that large CFL characteristic schemes of this type can perform stably and accurately provided the characteristic speeds do not change much in a given time step.

With the runs of CFL > 1, there are vertically-oriented oscillations of wavelength 2CFL$_{\text{max}}\Delta z$ in the difference plots. We hypothesize by the wavelength scale and vertical orientation that the errors must be related to something unique about the vertical direction. We see several things unique about the vertical direction: (1) there is a gradient in the acoustic wave speed, and (2) the gravity source term applies only in the vertical, and (3) potential temperature has a vertical gradient due to the stable stratification. The error from the first situation could arise from assumed constant trajectories (from freezing the Jacobian). As the CFL increases, this effect will also increase linearly. The second situation with the gravity source term could mean the error is arising from time step to time step imbalances as the source term and pressure forcing are split further apart with large CFL. We note also that dimensional splitting is the other major assumption in this discretization and likely plays a role in determining this error as well, or at least amplifies the error arising from the aforementioned issues.

4. Conclusions and future work

We have presented a new FV solver for numerical simulation of atmospheric dynamics offering competitive efficiency in terms of low communication requirements and large time steps. It is fully discrete (one-step, one-stage), upwind, spatially local, stable at large CFL numbers, and it accommodates any single-moment spatial interpolant. We have described the theory and implementation of the method and performed standard non-hydrostatic atmospheric test cases for validation. The method performed accurately and stably in each of the test cases without the need for post-hoc diffusion to stabilize even at



(a) Δz=100m, CFL_{max}=3.99

(b) Differences between CFL=3.99 and CFL=0.99

Fig. 12. Plots for the internal gravity waves test case after 3,000 s with a maximum CFL number of 3.99. $\Delta x = 10\Delta\, z = 1{,}000$ m. The *x*- and *y*-axes are in km and potential temperature perturbations are in K.

large Courant numbers. In the non-hydrostatic internal gravity waves test case, we estimated a numerical convergence of roughly third- to fourth-order (depending on the error norm) for that test case.

We would like to reemphasize that on modern computing architectures (which are distributed memory machines that are getting larger by the year), communication is quickly becoming the dominant limitation for many schemes. There is a great need to investigate new methods which may offer promise in this regard. In comparison to finite element type methods such as spectral element and discontinuous Galerkin, the present method allows larger time steps while keeping lower communication requirements than traditional FV methods.

Results at large CFL numbers showed degraded accuracy compared to simulation at CFL numbers near unity. Though we still had stability at CFL numbers greater than two, the accuracy degradation beyond CFL = 2 was too large for two of the test cases. Running the smoothest test case up to CFL = 7 revealed some characteristics of the errors associated with large CFL number simulation. It suggests that improving trajectories may alleviate much of the error and that properly resolving large gradients in characteristic speeds will require the most attention.

One limitation to this method in the current theory and implementation is that it is fixed to a dimensionally split framework. We will be investigating the potential for a genuinely multi-dimensional extension for use in curvilinear geometries that do not perform as well when dimensionally split.

All simulations in this study were performed on Graphics Processing Units (GPUs). Code run on an Nvidia GTX 280 GPU in double precision with relatively simple memory optimizations (making use of fast local shared memory) performed roughly 9–10× faster than an Intel Core2 Duo T7500 CPU. The CPU code being compared against was optimized with OpenMP (with linear speed-up) and SSE, and all data transfer times on the GPU are included as is file I/O with end-to-end wall timers on each for a fair comparison. In single precision, the GPU code ran 16–17× faster than the CPU code.

Acknowledgements

The first author gratefully acknowledges funding by the Department of Energy Computational Science Graduate Fellowship and also partial support by the National Center for Atmospheric Research Advanced Study Program. We are also grateful to the reviewers for their insightful comments that enhanced the manuscript's quality.

References

[1] W.C. Skamarock, J.B. Klemp, A time-split nonhydrostatic atmospheric model for weather research and forecasting applications, Journal of Computational Physics 227 (2008) 3465–3485.
[2] P.K. Smolarkiewicz, L.G. Margolin, A.A. Wyszogrodzki, A class of nonhydrostatic global models, Journal of the Atmospheric Sciences 58 (2001) 349–364.
[3] J.M. Dennis, A. Fournier, W.F. Spotz, A. St-Cyr, M.A. Taylor, S.J. Thomas, H.M. Tufo, High-resolution mesh convergence properties and parallel efficiency of a spectral element atmospheric dynamical core, International Journal of High Performance Computing Applications 19 (2005) 225–235.
[4] A. Staniforth, J. Cote, Semi-Lagrangian integration schemes for atmospheric models: a review, Monthly Weather Review 119 (9) (1991) 2206–2223.
[5] F.H.M. Semazzi, J.H. Quian, J.S. Scroggs, A global nonhydrostatic semi-Lagrangian atmospheric model without orography, Monthly Weather Review 123 (1995) 2534.
[6] L. Bonaventura, A semi-implicit semi-Lagrangian scheme using the height coordinate for a nonhydrostatic and fully elastic model of atmospheric flows, Journal of Computational Physics 227 (2000) 3849–3877.
[7] D.A. Knoll, D.E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, Journal of Computational Physics 193 (2004) 357–397.
[8] K.J. Evans, W.I. Rouson, A.G. Salinger, M.A. Taylor, W. Weijer, J.B. White, A scalable and adaptable solution framework within components of the community climate system model, in: Computational Science – ICCS 2009, Lecture Notes in Computer Science, vol. 5545, Springer, 2009, pp. 332–341, doi:10.1007/978-3-642-01973-9. ISBN:0302-9743.
[9] G.M. Amdahl, Validity of the single processor approach to achieving large scale computing capabilities, in: Proceedings of the April 18-20, 1967, spring joint computer conference, AFIPS Joint Computer Conferences, 1967, pp. 483–485.
[10] J.L. Gustafson, Reevaluating Amdahl's law, Communications of the ACM 31 (1988) 532–533.
[11] J.B. Klemp, W.C. Skamarock, J. Dudhia, Conservative split-explicit time integration methods for the compressible nonhydrostatic equations, Monthly Weather Review 135 (2007) 2897–2913.
[12] M.J.P. Cullen, T. Davies, A conservative split-explicit integration scheme with fourth-order horizontal advection, Quarterly Journal of the Royal Meteorological Society 117 (1991) 993–1002.
[13] D.R. Durran, The third-order Adams–Bashforth method: an attractive alternative to leapfrog time differencing, Monthly Weather Review 119 (1991) 702–720.
[14] A.J. Gadd, A split explicit integration scheme for numerical weather prediction, Quarterly Journal of the Royal Meteorological Society 104 (1978) 569–582.
[15] D. Majewski, D. Liermann, P. Prohl, B. Ritter, M. Buchhold, T. Hanisch, G. Paul, W. Wergen, J. Baumgardner, The operational global icosahedral–hexagonal gridpoint model GME: description and high-resolution tests, Monthly Weather Review 130 (2002) 319–338.
[16] R.L. Carpenter, K.K. Droegemeier, P.R. Woodward, C.E. Hane, Application of the piecewise parabolic method (PPM) to meteorological modeling, Monthly Weather Review 118 (1990) 586–612.
[17] N. Ahmad, The f-wave Riemann solver for Meso- and Micro-scale flows, AIAA Paper 2008-465.
[18] N. Ahmad, J. Lindeman, Euler solutions using flux-based wave decomposition, International Journal for Numerical Methods in Fluids 54 (2007) 47–72.
[19] R.D. Nair, S.J. Thomas, R.D. Loft, A discontinuous Galerkin transport scheme on the cubed sphere, Monthly Weather Review 133 (4) (2005) 814–828.
[20] J.M. Dennis, M. Levy, R.D. Nair, H.M. Tufo, T. Voran, Towards an efficient and scalable discontinuous Galerkin atmospheric model, in: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS05), Workshop 13, vol.14, 2005, 8 p.
[21] F.X. Giraldo, M. Restelli, A study of spectral element and discontinuous Galerkin methods for the Navier–Stokes equations in nonhydrostatic mesoscale atmospheric modeling: equation sets and test cases, Journal of Computational Physics 227 (2008) 3849–3877.
[22] M.N. Levy, R.D. Nair, H.M. Tufo, High-order Galerkin methods for scalable global atmospheric models, Computers and Geosciences 33 (2007) 1022–1035.
[23] R.D. Nair, H.-W. Choi, H.M. Tufo, Computational aspects of a scalable high-order discontinuous Galerkin atmospheric dynamical core, Computers and Fluids 38 (2009) 309–319.
[24] A. St-Cyr, D. Neckels, A fully implicit Jacobian-Free high-order discontinuous Galerkin mesoscale flow solver, in: Lecture Notes in Computer Sciences, vol. LNCS 5545, Springer, 2009, pp. 243–252.
[25] D.S. Balsara, C. Altmann, C.-D. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG+HWENO schemes, Journal of Computational Physics 226 (2007) 586–620.
[26] H. Wang, J.J. Tribbia, F. Baer, A. Fournier, M.A. Taylor, A spectral element version of CAM2, Monthly Weather Review 135 (2007) 3825–3840.
[27] S. Ii, F. Xiao, High order multi-moment constrained finite volume method. Part I: Basic formulation, Journal of Computational Physics 228 (2009) 3669–3707.
[28] K. Toda, Y. Ogata, T. Yabe, Multi-dimensional conservative semi-Lagrangian method of characteristics CIP for the shallow water equations, Journal of Computational Physics 228 (2009) 4917–4944.
[29] T. Yabe, Y. Ogata, Conservative semi-Lagrangian CIP technique for the shallow water equations, Computational Mechanics, doi:10.1007/s00466-009-0438-8.
[30] Y. Matsumoto, K. Seki, Implementation of the CIP algorithm to magnetohydrodynamic simulations, Computer Physics Communications 179 (2008) 289–296.
[31] S. Ii, F. Xiao, CIP/multi-moment finite volume method for Euler equations: a semi-Lagrangian characteristic formulation, Journal of Computational Physics 222 (2007) 849–871.
[32] D. Bale, R.J. Leveque, S. Mitran, J.A. Rossmanith, A wave propagation method for conservation laws and balance laws with spatially varying flux functions, SIAM Journal on Scientific Computing 24 (3) (2002) 955–978.
[33] R.J. Leveque, Finite Volume Methods for Hyperbolic Problems, Cambridge University Press, Cambridge, MA, 2002.
[34] G. Capdeville, A central WENO scheme for solving hyperbolic conservation laws on non-uniform meshes, Journal of Computational Physics 227 (2008) 2977–3014.
[35] C.W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, Journal of Computational Physics 77 (1988) 439–471.
[36] C.W. Shu, High order ENO and WENO schemes for computational fluid dynamics, in: T.J. Barth, H. Deconinck (Eds.), High-Order Methods for Computational Physics, Lecture Notes in Computational Science and Engineering, vol. 9, Springer, 1999, pp. 439–582.
[37] M.R. Norman, R.D. Nair, Inherently conservative non-polynomial based remapping schemes: application to semi-Lagrangian transport, Monthly Weather Review 136 (2008) 5044–5061.
[38] M.R. Norman, F.H.M. Semazzi, R.D. Nair, Conservative cascade interpolation on the sphere: an intercomparison of various non-oscillatory reconstructions, Quarterly Journal of the Royal Meteorological Society 135 (2009) 795–805.
[39] P.L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, Journal of Computational Physics 43 (1981) 357–372.
[40] W.C. Skamarock, J.B. Klemp, J. Dudhia, D.O. Gill, D.M. Barker, W. Wang, J.G. Powers, A Description of the Advanced Research WRF Version 2, Tech. Rep. NCAR/TN-468+STR, National Center for Atmospheric Research, Boulder, CO, USA, 2005.
[41] N. Wood, A. Staniforth, A. White, Determining near-boundary departure points in semi-Lagrangian models, Quarterly Journal of the Royal Meteorological Society 135 (2009) 1890–1896.
[42] J.M. Straka, R.B. Wilhelmson, L.J. Wicker, J.R. Anderson, K.K. Droegemeier, Numerical solutions of a non-linear density current: a benchmark solution and comparisons, International Journal for Numerical Methods in Fluids 17 (1993) 1–22.