A SCALABLE FULLY IMPLICIT COMPRESSIBLE EULER SOLVER FOR MESOSCALE NONHYDROSTATIC SIMULATION OF ATMOSPHERIC FLOWS*

CHAO YANG† AND XIAO-CHUAN CAI‡

Abstract. A fully implicit solver is developed for the mesoscale nonhydrostatic simulation of atmospheric flows governed by the compressible Euler equations. To spatially discretize the Euler equations on a height-based terrain-following mesh, we apply a cell-centered finite volume scheme, in which an AUSM$^+$-up method with a piecewise linear reconstruction is employed to achieve second-order accuracy for the low-Mach flow. A second-order ESDIRK method with adaptive time stepping is applied to stabilize physically insignificant fast waves and accurately integrate the Euler equations in time. The nonlinear system arising at each time step is solved by using a Jacobian-free Newton-Krylov-Schwarz algorithm. To accelerate the convergence and improve the robustness, we employ a class of additive Schwarz preconditioners in which the subdomain Jacobian matrix is constructed using a first-order spatial discretization. Several test cases are used to validate the correctness of the scheme and examine the performance of the solver. Large-scale results on a supercomputer with up to 18,432 processor cores are provided to show the parallel performance of the proposed method.

Key words. fully implicit method, Newton-Krylov-Schwarz, nonhydrostatic model, compressible Euler equations, parallel scalability

AMS subject classifications. 65Y05, 65M55, 65F08, 86A10, 35L65

1. Introduction.

The atmosphere contains multiscale dynamics that support a variety of wave motions. Fast waves, such as the acoustic wave and the inertial-gravity wave, often impose restrictive time step constraints for explicit schemes. To deal with the rapidly traveling, physically insignificant fast waves, one can either (i) simplify the governing equations based on, e.g., a hydrostatic, an incompressible or an anelastic assumption; or (ii) employ a more advanced time integration scheme with a weaker stability requirement. In the first approach, the compressible Euler equations are replaced with simplified ones that are often easier to solve in an explicit manner because certain fast waves are filtered out. For example, when the hydrostatic primitive equations are employed, the atmosphere is assumed to be in vertical balance and, as a result, is free of the internal acoustic mode. However, the hydrostatic assumption becomes invalid when the horizontal scale is smaller than about 10 km. Even for other simplified equations that might be more accurate than the hydrostatic primitive equations, it is still not clear if they are valid for all scales [20, 43]. Therefore, when high resolution is of interest as in mesoscale and cloud-resolving atmospheric simulations, fast and efficient solution of the fully compressible Euler equations becomes desirable.

The second approach to stabilize fast waves is to make use of a more advanced time integration scheme, which is usually based on either (i) modifying a fully explicit scheme to increase the maximum allowable time step size; or (ii) reducing the cost of a fully implicit scheme. In this study, we focus on the latter method and only briefly mention some examples of the former one, such as the split-explicit method [13, 21],

* This work was supported in part by NSF grants DMS-0913089 and CCF-1216314. The first author was also supported in part by NSFC grants 61170075, 91130023 and 61120106005, and by 973 Program of China 2011CB309701.

† Institute of Software, Chinese Academy of Sciences, Beijing 100190, China and State Key Laboratory of High Performance Computing, Changsha 410073, China (yangchao@iscas.ac.cn).

‡ Department of Computer Science, University of Colorado Boulder, Boulder, CO 80309, USA (cai@cs.colorado.edu).

the fractional-step method [46], the semi-implicit method [25], the semi-implicit semi-Lagrangian method [2, 42] and the horizontally-explicit vertically-implicit method [36, 43]. The basic idea behind these methods is operator splitting. Although some of them allow substantially larger time steps than a fully explicit scheme, the dependency between the time step length and the horizontal mesh resolution still persists. In addition, due to the inconsistent time integration of different terms in the governing equations, the solution obtained in an operator splitting method may violate the nonlinear consistency which in turn leads to large splitting errors and accuracy degradation [22, 29].

Compared to operator splitting, fully implicit methods enjoy two major advantages: (i) the time step size depends only on the accuracy requirement; and (ii) the discretized equations are nonlinearly consistent. However, at each time step, due to the need of solving a nonlinear system, a fully implicit method might be much more expensive than an explicit or an operator splitting method. Despite of some successes on computing fully implicit solutions of the global shallow water equations [10, 11, 48, 49], it is rare to see the application of fully implicit methods in non-hydrostatic atmospheric simulations. One of the contributions is [40], in which a discontinuous Galerkin solver for mesoscale flows using a fully implicit Rosenbrock scheme is proposed and shown to be superior to an explicit Runge-Kutta method. However, in their Jacobian-free Newton-Krylov framework for solving the nonlinear system, the preconditioner, which is critical for the performance of the solver, is not studied.

On the other hand, we have made some efforts on employing domain decomposition based algorithms in the fully implicit solution of the global shallow water equations [48]. The solver has been shown in several numerical experiments to be highly scalable to tens of thousands of processors in terms of both strong and weak scalabilities. The purpose of this study is to extend the Newton-Krylov algorithm, especially the additive Schwarz preconditioner, for the fully implicit solution of the compressible Euler equations. An alternative approach for preconditioning fast (or stiff) wave problems is the physics-based preconditioner [24], which has also been applied for low speed compressible flows [32, 34]. In a physics-based method, the preconditioning operator has certain advantages, such as being diagonally dominant. But a comprehensive comparison between the additive Schwarz preconditioner and the physics-based preconditioner is beyond the scope of this study. We point out that the basic idea of the additive Schwarz preconditioner studied in this paper is still applicable for inverting the physics-based preconditioning operator.

The paper is organized as follows. In Section 2, the compressible Euler equations in a nondimensionalized form are presented. We then provide in Section 3 some details of a cell-centered finite volume scheme for the spatial discretization, including an AUSM$^+$-up Riemann solver, a second-order piecewise linear reconstruction and the numerical approximation of the boundary conditions. In Section 4, we present an L-stable second-order ESDIRK method with an adaptive time step control strategy for the temporal integration. Details of a Jacobian-free Newton-Krylov solver preconditioned by an additive Schwarz method are then introduced in Section 5. We validate the discretization scheme and study the parallel performance of the fully implicit solver by presenting numerical results on several test cases in Section 6. The paper is concluded in Section 7.

2. The compressible Euler equations. The compressible Euler equations for the atmosphere in the $x-z$ plane are written as a system of conservation laws [15]:

$$\begin{cases} \dfrac{\partial}{\partial t}\rho + \dfrac{\partial}{\partial x}(\rho u) + \dfrac{\partial}{\partial z}(\rho w) = 0, \\ \dfrac{\partial}{\partial t}(\rho u) + \dfrac{\partial}{\partial x}(\rho u u + p) + \dfrac{\partial}{\partial z}(\rho u w) = 0, \\ \dfrac{\partial}{\partial t}(\rho w) + \dfrac{\partial}{\partial x}(\rho u w) + \dfrac{\partial}{\partial z}(\rho w w + p) + \rho g = 0, \\ \dfrac{\partial}{\partial t}(\rho\theta) + \dfrac{\partial}{\partial x}(\rho u \theta) + \dfrac{\partial}{\partial z}(\rho w \theta) = 0, \end{cases} \tag{2.1}$$

where the primitive variables are the density ρ, the horizontal velocity u, the vertical velocity w and the potential temperature θ. The gravity force is represented in the equations as a source term ρg, where $g = 9.80665\,\mathrm{m/s^2}$ is the effective gravity constant. The system is closed with the equation of state

$$p = p_{00}\left(\frac{\rho R\theta}{p_{00}}\right)^{\gamma}, \tag{2.2}$$

where $p_{00} = 1013.25\,\mathrm{hPa}$ is the ground level pressure, $R = 287.04\,\mathrm{J/(kg\cdot K)}$ is the gas constant for dry air and $\gamma = 1.4$.

To reduce the roundoff error, we apply a nondimensionalization to (2.1). Given a reference potential temperature θ_c, we select the reference pressure $p_c = 1013.25\,\mathrm{hPa}$, scale the gas constant by $R_c = 287.04\,\mathrm{J/(kg\cdot K)}$ and the gravity constant by $g_c = 9.80665\,\mathrm{m/s^2}$. Then the nondimensionalization can be done by introducing the reference values of the density, the velocity, the length and the time respectively as

$$\rho_c = \frac{p_c}{R_c\theta_c}, \quad u_c = \sqrt{\frac{p_c}{\rho_c}}, \quad \ell_c = \frac{u_c^2}{g_c}, \quad t_c = \frac{\ell_c}{u_c}.$$

It is easy to verify that after the nondimensionalization, the Euler equations (2.1) and the equation of state (2.2) both remain the same, but the physical constants are normalized to be

$$g = 1, \quad p_{00} = 1, \quad R = 1. \tag{2.3}$$

The dynamics of the atmosphere, in most situations, are relatively small perturbations of the hydrostatic equilibrium, which assumes the gravity is balanced by the pressure gradient force, i.e.,

$$\frac{\partial \overline{p}}{\partial z} = -\overline{\rho} g, \tag{2.4}$$

where $\overline{p}$, $\overline{\rho}$ are usually independent of time but may depend on both x and z. Therefore, in order to reduce the approximation error of the hydrostatic state, it is preferable [15, 40] to introduce in the equations the following perturbed values

$$\rho' = \rho - \overline{\rho}, \quad p' = p - \overline{p}, \quad (\rho\theta)' = \rho\theta - \overline{\rho\theta},$$

in which $\overline{\rho}$, $\overline{p}$, $\overline{\rho\theta}$ satisfy the hydrostatic condition (2.4) and the equation of state (2.2). After splitting out the hydrostatic background state, we rewrite the Euler equations (2.1) in a compact form:

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial z} + S = 0, \tag{2.5}$$

where

$$Q = \begin{pmatrix} \rho' \\ \rho u \\ \rho w \\ (\rho\theta)' \end{pmatrix},\ F = \begin{pmatrix} \rho u \\ \rho uu + p' \\ \rho uw \\ \rho u\theta \end{pmatrix},\ G = \begin{pmatrix} \rho w \\ \rho wu \\ \rho ww + p' \\ \rho w\theta \end{pmatrix},\ S = \begin{pmatrix} 0 \\ 0 \\ \rho' g \\ 0 \end{pmatrix}.$$

When necessary, a physical dissipation term

$$S_\nu = -(0, \nabla\cdot(\nu\rho\nabla u), \nabla\cdot(\nu\rho\nabla w), \nabla\cdot(\nu\rho\nabla\theta'))^T \tag{2.6}$$

is added to the left-hand side of (2.5), where ν is a constant to control the viscosity.

3. Spatial discretization. We solve the Euler equations (2.5) on the physical domain

$$\Omega = \{(x,z)^T \big| x_{min} < x < x_{max}, z_b(x) < z < z_{max}\},$$

with the bottom boundary $z_b(x)$ describing the topography of the region. The bottom topography may have a significant impact on the dynamics of atmosphere, especially at high resolutions. In order to accurately approximate the topography, we employ the height-based terrain-following coordinates [14] given by

$$\begin{aligned} \hat{x} &= x - x_{min}, \\ \hat{z} &= \frac{z_{max} - z_{min}}{z_{max} - z_b}(z - z_b), \end{aligned} \tag{3.1}$$

where $(\hat{x},\hat{z})^T$ are the transformed coordinates and $z_{min} = \inf(z_b)$. The coordinates transform (3.1) maps the physical domain Ω to a regular computational domain

$$\hat{\Omega} = \{(\hat{x},\hat{z})^T \big| 0 \le \hat{x} \le \ell_1, 0 \le \hat{z} \le \ell_2\},$$

where $\ell_1 = x_{max} - x_{min}$, $\ell_2 = z_{max} - z_{min}$. In $\hat{\Omega}$ we introduce a uniform mesh consisting of points $\hat{\mathbf{x}}_{ij} = (\hat{x}_i, \hat{z}_j)^T = (i\ell_1/n_1, j\ell_2/n_2)^T$, $i = 0,1,\cdots,n_1$, $j = 0,1,\cdots,n_2$. Using the coordinates transform (3.1), we obtain a structured mesh on the physical domain Ω, with mesh points $\mathbf{x}_{ij} = (x_{ij}, z_{ij})^T$ given by

$$\begin{aligned} x_{ij} &= \hat{x}_i + x_{min}, \\ z_{ij} &= \frac{z_{max} - z_b(x_{ij})}{z_{max} - z_{min}}\hat{z}_j + z_b(x_{ij}). \end{aligned} \tag{3.2}$$

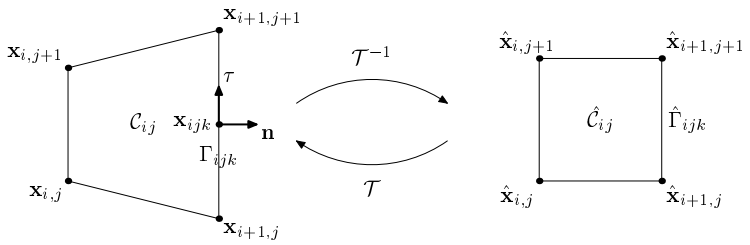

FIG. 3.1. *A coordinate transform between the physical mesh and the computational mesh.*

3.1. Cell-centered finite volume scheme. A cell-centered finite volume scheme is applied to discretize the Euler equation (2.5). Based on (3.2), we define a nonsingular mapping $\mathcal{T} : \hat{\mathbf{x}}_{ij} \to \mathbf{x}_{ij}$, for $i = 0, 1, \cdots, n_1$, $j = 0, 1, \cdots, n_2$, as shown in Fig. 3.1. Denote $\mathcal{C}_{ij}$ as a mesh cell formed by mesh points $\mathbf{x}_{ij}$, $\mathbf{x}_{i+1,j}$, $\mathbf{x}_{i+1,j+1}$, $\mathbf{x}_{i,j+1}$, and $\hat{\mathcal{C}}_{ij}$ as a mesh cell formed by mesh points $\hat{\mathbf{x}}_{ij}$, $\hat{\mathbf{x}}_{i+1,j}$, $\hat{\mathbf{x}}_{i+1,j+1}$, $\hat{\mathbf{x}}_{i,j+1}$. It is easy to verify that the mapping $\mathcal{T}$ is bilinear on each mesh cell $\hat{\mathcal{C}}_{ij}$. We define the approximate solution at time t as

$$Q_{ij}(t) = \frac{1}{|\mathcal{C}_{ij}|} \int_{\mathcal{C}_{ij}} Q(\mathbf{x}, t) d\mathbf{x}, \quad i = 0, 1, \cdots, n_1 - 1, \ j = 0, 1, \cdots, n_2 - 1. \tag{3.3}$$

Integrating (2.5) over $\mathcal{C}_{ij}$ and applying the Gauss divergence theorem, we have

$$\frac{\partial Q_{ij}(t)}{\partial t} + \frac{1}{|\mathcal{C}_{ij}|} \int_{\partial \mathcal{C}_{ij}} (F(Q(\mathbf{x}, t))n_x + G(Q(\mathbf{x}, t))n_z)\, ds + S_{ij}(t) = 0, \tag{3.4}$$

where $(n_x, n_z)^T$ is the unit outward normal of $\partial \mathcal{C}_{ij}$.

In (3.4), the source term is approximated as

$$S_{ij}(t) = \frac{1}{|\mathcal{C}_{ij}|} \int_{\mathcal{C}_{ij}} S(Q(\mathbf{x}, t)) d\mathbf{x} \approx S(Q_{ij}(t)) \tag{3.5}$$

with second-order accuracy. In order to evaluate the numerical fluxes of F and G in (3.4), we decompose the boundary of $\mathcal{C}_{ij}$ into four segments, i.e., $\partial \mathcal{C}_{ij} = \cup_{k=1}^{4} \Gamma_{ijk}$, as shown in Fig. 3.1. On Γ_{ijk}, we denote the unit outward normal as $n = (n_x, n_z)^T$ and correspondingly the unit tangent vector as $\tau = (-n_z, n_x)^T$. Then n and τ form a new Cartesian coordinates. Given a state variable $Q = (\rho', \rho u, \rho w, (\rho\theta)')^T$, we introduce a new state variable $q = (\rho', \rho u_n, \rho u_\tau, (\rho\theta)')^T$, where u_n, u_τ are respectively the normal and tangent components of $\mathbf{v} = (u, w)^T$. It is easy to see that $q = T_{ijk} Q$, where $T_{ijk} = \mathrm{diag}\{1, L_{ijk}, 1\}$ and $L_{ijk}^T = (n, \tau)$. By using T_{ijk}, we have

$$\begin{aligned} \int_{\Gamma_{ijk}} (F(Q(\mathbf{x}, t))n_x + G(Q(\mathbf{x}, t))n_z)\, ds &= T_{ijk}^{-1} \int_{\Gamma_{ijk}} F(T_{ijk} Q(\mathbf{x}, t)) ds \\ &\approx T_{ijk}^{-1} |\Gamma_{ijk}| F(q(\mathbf{x}_{ijk}, t)), \end{aligned} \tag{3.6}$$

which is a second-order approximation, provided that $\mathbf{x}_{ijk}$ is the mid-center of edge Γ_{ijk}. The numerical flux $F(q(\mathbf{x}_{ijk}, t))$ is then estimated using a Riemann solver $\mathcal{F}(q^-(\mathbf{x}_{ijk}, t), q^+(\mathbf{x}_{ijk}, t))$, based on the inward and outward reconstructed values $q^-(\mathbf{x}_{ijk}, t)$ and $q^+(\mathbf{x}_{ijk}, t)$. Different choices of the Riemann solver and different ways to perform the reconstruction lead to different spatial discretizations.

3.2. AUSM$^+$-up method. Since the atmosphere flow is nearly incompressible, the corresponding Mach number is usually small. Classical Riemann solvers frequently found in aerodynamics often result in excessive numerical dissipation errors for low-Mach flows. Therefore, we employ a modified Advection Upstream Splitting Method (AUSM$^+$-up) [27] which is accurate for all Mach numbers. This method has been studied in a global shallow water model [44] and recently in a compressible Euler model [43]. The basic idea of the AUSM$^+$-up method is to split the numerical flux into a convective component and a pressure component, i.e.,

$$\mathcal{F}_{\text{AUSM}^+\text{-up}}(q^-, q^+) = \mathcal{F}^{(c)}(q^-, q^+) + \mathcal{F}^{(p)}(q^-, q^+). \tag{3.7}$$

The convective component in (3.7) is calculated by

$$\mathcal{F}^{(c)}(q^-, q^+) = \begin{cases} \dot{c}\,\dot{m}\,(1, u_n^-, u_\tau^-, \theta^-)^T, & \text{if } \dot{m} > 0, \\ \dot{c}\,\dot{m}\,(1, u_n^+, u_\tau^+, \theta^+)^T, & \text{otherwise,} \end{cases} \tag{3.8}$$

where $\dot{c} = (\sqrt{\gamma p^-/\rho^-} + \sqrt{\gamma p^+/\rho^+})/2$ is the interface sound speed and $\dot{m}$ is the interface Mach number given by

$$\dot{m} = \mathcal{M}_4^+(m^-) + \mathcal{M}_4^-(m^+) - \frac{K_p}{f_a}\max(1 - \sigma\bar{M}^2, 0)\frac{(p^-)' - (p^+)'}{\dot{\rho}\,\dot{c}^2}. \tag{3.9}$$

In (3.9), the interface density $\dot{\rho}$ is obtained from $\dot{\rho} = (\rho^- + \rho^+)/2$, the average Mach number $\dot{m}$ satisfies $\dot{m}^2 = ((m^-)^2 + (m^+)^2)/2$ where $m^\pm = u_n^\pm/\dot{c}$, and the split Mach numbers $\mathcal{M}_2^\pm$, $\mathcal{M}_4^\pm$ are polynomials of degree 2:

$$\mathcal{M}_2^\pm(m) = \pm(m \pm 1)^2/4, \tag{3.10}$$

and degree 4:

$$\mathcal{M}_4^\pm(m) = \begin{cases} (m \pm |m|)/2, & \text{if } |m| \geq 1, \\ \mathcal{M}_2^\pm(m)\left[1 \mp 16\beta\mathcal{M}_2^\mp(m)\right], & \text{otherwise,} \end{cases} \tag{3.11}$$

respectively.

The pressure component in (3.7) is calculated by

$$\mathcal{F}^{(p)}(q^-, q^+) = (0, \dot{p}', 0, 0)^T, \tag{3.12}$$

in which $\dot{p}'$ is the perturbation of the interface pressure given by

$$\begin{aligned} \dot{p}' &= \mathcal{P}_5^+(m^-)(p^-)' + \mathcal{P}_5^-(m^+)(p^+)' \\ &\quad - K_u f_a \mathcal{P}_5^+(m^-)\mathcal{P}_5^-(m^+)(\rho^- + \rho^+)\dot{c}(u_n^+ - u_n^-), \end{aligned} \tag{3.13}$$

where

$$\mathcal{P}_5^\pm(m) = \begin{cases} (1 \pm \text{sign}(m))/2, & \text{if } |m| \geq 1, \\ \mathcal{M}_2^\pm(m)\left[(\pm 2 - m) \mp 16\alpha m\,\mathcal{M}_2^\mp(m)\right], & \text{otherwise,} \end{cases} \tag{3.14}$$

As suggested in [43], the parameters used in (3.9)-(3.14) are set to be $K_p = 1/4$, $K_u = 3/4$, $f_a = 1$, $\sigma = 1$, $\alpha = 3/16$ and $\beta = 1/8$. We remark here that because of the shifting of the Euler equations according to the hydrostatic state, the formulation of the AUSM$^+$-up scheme is slightly different from its original form [27] in the way that the pressure perturbation instead of the pressure is used in (3.9), (3.12) and (3.13).

3.3. State reconstruction. In the Riemann solver $\mathcal{F}(q^-(\mathbf{x}_{ijk}, t), q^+(\mathbf{x}_{ijk}, t))$, the reconstructed values $q^\pm(\mathbf{x}_{ijk}, t)$ are needed. In order to obtain $q^\pm(\mathbf{x}_{ijk}, t)$, we first reconstruct $\hat{Q}^\pm(\hat{\mathbf{x}}_{ijk}, t) = Q^\pm(\mathbf{x}_{ijk}, t)$ on the computational mesh (due to the mesh uniformity) and then calculate $q^\pm(\mathbf{x}_{ijk}, t) = T_{ijk}Q^\pm(\mathbf{x}_{ijk}, t) = T_{ijk}\hat{Q}^\pm(\hat{\mathbf{x}}_{ijk}, t)$. Without loss of generality, suppose $\hat{\Gamma}_{ijk} =$ is the eastward edge of $\hat{\mathcal{C}}_{ij}$, as seen in Fig. 3.1. We employ the following piecewise linear reconstruction:

$$\begin{aligned} \hat{Q}^-(\hat{\mathbf{x}}_{ijk}, t) &= \frac{2-\kappa}{2}Q_{ij}(t) - \frac{1-\kappa}{4}Q_{i-1,j}(t) + \frac{1+\kappa}{4}Q_{i+1,j}(t), \\ \hat{Q}^+(\hat{\mathbf{x}}_{ijk}, t) &= \frac{2-\kappa}{2}Q_{i+1,j}(t) + \frac{1+\kappa}{4}Q_{ij}(t) - \frac{1-\kappa}{4}Q_{i+2,j}(t), \end{aligned} \tag{3.15}$$

where $\kappa \in [0, 1)$. The above reconstruction results in the κ-scheme for linear problems; in particular, $\kappa = 0$, $1/2$ and $1/3$ lead to the Fromm scheme [12], the QUICK scheme [26] and the QUICKEST scheme [26]. Although several studies have been carried out for these schemes for some linear problems, it is not clear which one is optimal for the Euler equations. In [48], $\kappa = 0$ is used for the global shallow water equations, while in [43], $\kappa = 1/3$ is used for the vertical discretization of the compressible Euler equations. In our finite volume scheme, we choose $\kappa = 1/2$ because of its low numerical dissipation as observed in our numerical experiments.

The piecewise linear reconstruction (3.15), together with the AUSM$^+$-up Riemann solver, leads to a formally second-order accurate scheme for the Euler equations (2.5), provided that the solution is sufficiently smooth. Otherwise, spurious oscillations may occur when the solution contains strong shocks or discontinuities. Slope limiters may need to be added in (3.15) to reduce the spurious oscillations and improve the stability of the scheme when necessary. We remark that when $\kappa = 1/3$ the local truncation error of the reconstruction is formally third-order; but it needs to be incorporated with a third-order numerical flux to become a third-order spatial discretization.

3.4. Boundary conditions. The bottom boundary is the only physical boundary of the domain Ω. We therefore specify a no-flux (rigid wall) boundary condition along it. In the calculation of the numerical flux (3.6), the no-flux boundary condition requires that the normal velocity vanishes along the boundary, i.e., $u_n = 0$, which can be easily incorporated. For the physical dissipation term (2.6), the no-flux boundary condition requires that $n \cdot (\nu\rho\nabla u) = n \cdot (\nu\rho\nabla w) = n \cdot (\nu\rho\nabla\theta') = 0$, so that there is no loss of mass on the boundary.

For problems that involve mountain waves, it is desirable that all energy transport is properly radiated out of the top and lateral boundaries. To this end, we apply absorbing sponge layers [8] in which a Rayleigh damping is applied to the Euler equations (2.5) in the form of

$$\frac{\partial Q}{\partial t} + (1-\varphi)\left(\frac{\partial F}{\partial x} + \frac{\partial G}{\partial z} + S\right) + \varphi(Q - \tilde{Q}) = 0, \tag{3.16}$$

where $\tilde{Q}$ is a predetermined reference solution and φ is the damping factor. Given a physical domain Ω, we first choose a domain of interest $\Omega^* = \{(x,z)\big| x^*_{min} < x < x^*_{max}, h(x) < z < z^*_{max}\}$ and then apply the absorbing layers in $\Omega\backslash\Omega^*$ by setting

$$\varphi = 1 - (1-\varphi_x)(1-\varphi_z) = \varphi_x + \varphi_z - \varphi_x\varphi_z.$$

Here φ_x and φ_z are nonzero only in the lateral and top absorbing layers respectively, and they are calculated in an analogous manner [15], e.g.,

$$\varphi_z = \left(\frac{z - z^*_{max}}{z_{max} - z^*_{max}}\right)^4.$$

Note that in Ω^* we set $\varphi = 0$ to turn off the absorbing layer.

4. Fully implicit adaptive time stepping. After spatially discretizing the Euler equations (2.5) with the cell-centered finite volume scheme, we obtain a semi-discrete system

$$\frac{\partial}{\partial t}X(t) + \mathcal{L}(X(t)) = 0. \tag{4.1}$$

Here we organize the solution vector in a point-block natural order

$$X(t) = (Q_{0,0}(t), Q_{1,0}(t), Q_{2,0}(t), \cdots, Q_{0,1}(t), Q_{1,1}(t), Q_{2,1}(t), \cdots)^T, \tag{4.2}$$

and the discrete operator $\mathcal{L}(X(t))$ is organized in the same point-block order with each component given by

$$L_{ij}(t) = \frac{1}{|\mathcal{C}_{ij}|} \sum_{\cup \Gamma_{ijk} = \partial \mathcal{C}_{ij}} T_{ijk}^{-1} |\Gamma_{ijk}| \mathcal{F}(T_{ijk} Q^-(\mathbf{x}_{ijk}, t), T_{ijk} Q^+(\mathbf{x}_{ijk}, t)) + S(Q_{ij}(t)).$$

Many temporal discretization schemes can be used to integrate the semi-discrete system (4.1). Due to the existence of fast waves in the compressible Euler equations, explicit methods suffer from stability restrictions on the time step size from fast waves or stiff waves, although the advective time scale is often of interest. To quantitatively analyze the property of different temporal discretization schemes, we define the respective Courant-Friedrichs-Lewy (CFL) number for both the fast acoustic wave and the advection as:

$$CFL_f = \Delta t / \Delta t_f, \quad CFL_a = \Delta t / \Delta t_a, \tag{4.3}$$

where Δt is the employed time step size and

$$\Delta t_f = \min\{\Delta x, \Delta z\}/u_f, \quad \Delta t_a = \min\{\Delta x, \Delta z\}/u_a, \tag{4.4}$$

are the respective time scales. In (4.4), $u_f = \sqrt{\gamma p/\rho}$ is the speed of the fast acoustic wave and u_a is the maximum advection speed. For comparison purpose, we implement an explicit second-order Strong Stability Preserving Runge-Kutta (SSP RK-2) method

$$\begin{aligned} X^{(1)} &= X(t_m) - \Delta t \mathcal{L}\left(X(t_m)\right), \\ X(t_{m+1}) &= \frac{1}{2}(X(t_m) + X^{(1)}) - \frac{\Delta t}{2}\mathcal{L}(X^{(1)}), \end{aligned} \tag{4.5}$$

in which we use a fixed time step size Δt determined from $CFL_f \approx 0.5$.

In order to relax the time step limit and control the time step size according to the accuracy, we employ a family of Explicit-first-step, Single-diagonal-coefficient, Diagonally Implicit Runge-Kutta (ESDIRK) methods:

$$\begin{aligned} &X^{(0)} = X(t_m), \\ &\frac{1}{\Delta t_m}\left(X^{(p)} - X^{(0)}\right) + \sum_{q=0}^{p} a_{pq}\mathcal{L}(X^{(q)}) = 0, \quad p = 1, .., s, \\ &X(t_{m+1}) = X^{(s)}, \end{aligned} \tag{4.6}$$

where $s \geq 1$ is the number of implicit stages, $X(t_m)$ is the solution vector and $\Delta t_m = t_{m+1} - t_m$ is the corresponding time step size at the m^{th} time step. We denote an ESDIRK method with s implicit stages as ESDIRK(s). The method has a single diagonal coefficient because we always set $a_{pp} \equiv c$ for all $p = 1, \cdots, s$. In particular, for ESDIRK(1), there are two special cases, namely the backward Euler method with $a_{10} = 0$, $c = 1$, and the Crank-Nicolson method with $a_{10} = c = 1/2$. The backward Euler method is L-stable but is only first-order accurate and the Crank-Nicolson method is second-order accurate but not L-stable. We find that neither of them is

efficient and accurate enough to be incorporated with an adaptive time stepping. Therefore, we focus on an ESDIRK(2) method with coefficients

$$a_{10} = c = 1 - \sqrt{2}/2, \quad a_{20} = a_{21} = \sqrt{2}/4,$$

which is both L-stable and second-order accurate [45].

We adaptively control the time step size Δt_m by using a strategy that is analogous to the switched evolution/relaxation approach [17, 30]. More specifically, we start with a relatively small time step size Δt_0 and adjust its value according to

$$\Delta t_{m+1} = \min\left(\Delta t_{max}, \max\left(\frac{1}{r}, \min\left(r, \left(\frac{\|\mathcal{L}(X_m)\|_2}{\|\mathcal{L}(X_{m+1})\|_2}\right)^{\eta}\right)\right)\Delta t_m\right), \tag{4.7}$$

for $m = 0, 1, 2, \cdots$. Here Δt_{max} is the maximum allowable time step size in the simulation, $r \in (0, +\infty)$ is a safeguard to avoid excessive change of the time step size between any two immediate time steps, and $\eta \in (0, 1)$ is used the control the adjustment of the time step size. In practice, we set the adaptivity parameters to be $r = 1.5$ and $\eta = 0.75$. We remark that because the ESDIRK(2) method is embedded, one may also control the time step size by calculating the difference of two possible solutions of different orders at each time step, as done in, e.g., [18]. We find that the performance and efficiency of this embedded adaptation strategy is similar to that of (4.7).

It is worth mentioning that, although other fully implicit schemes may have certain advantages, we choose the ESDIRK(2) method to demonstrate the efficiency of the preconditioning techniques to be studied in this paper. The resulting nonlinear/linear systems arising in different fully implicit schemes often have similar structures; for example, there is only a slight modification on the diagonal part of the Jacobian matrix if the ESDIRK(2) method is replaced by another ESDIRK method or a Rosenbrock method. Both ESDIRK and Rosenbrock methods, when incorporated with suitable adaptive time stepping, are widely studied in computational fluid dynamics, especially in solving stiff problems that admit a variety of time scales; see, e.g. [19, 28] for further references.

5. Newton-Krylov-Schwarz solver. In the ESDIRK(2) method, there are two implicit stages that lead to two nonlinear systems at each time step. In order to solve the nonlinear systems efficiently, we employ a Newton-Krylov-Schwarz (NKS) algorithm described as follows.

5.1. Newton-Krylov iteration. Given a nonlinear system $N(X) = 0$, the Newton's iteration is to update the current approximate solution X_n to obtain a new approximation X_{n+1} through

$$X_{n+1} = X_n + \lambda_n \delta X_n, \quad n = 0, 1, \cdots. \tag{5.1}$$

Here X_0 is chosen as the solution at the previous time step, λ_n is the steplength determined by a linesearch procedure (see, e.g., [6, Sec. 6.3]) and δX_n is the Newton correction obtained by solving the Jacoian system as discussed later. To achieve a more uniform distribution of residual errors of all time steps [48], the stopping condition for the Newton iteration (5.1) is adaptively determined by

$$\|N(X_{n+1})\|_2 \leq \min\left\{\hat{\varepsilon}_a, \max\left\{\check{\varepsilon}_{a,n}, \varepsilon_r \|N(X_0)\|_2\right\}\right\}. \tag{5.2}$$

Here the relative tolerance ε_r and the safeguard $\hat{\varepsilon}_a$ are both fixed for all time steps, and the absolute tolerance $\check{\varepsilon}_{a,n}$ is chosen as $\check{\varepsilon}_{a,0} \in [0, \hat{\varepsilon}_a)$ at the first time step and then adaptively determined by

$$\check{\varepsilon}_{a,n} = \max\left\{\check{\varepsilon}_{a,n-1}, \|N(X_{n-1})\|_2\right\}.$$

In (5.1), the Newton correction vector δX_n is calculated by approximately solving the Jacoian system

$$J_n \delta X_n = -N(X_n), \tag{5.3}$$

where $J_n = N'(X_n)$ is the Jacobian matrix. To improve the convergence of the Jacobian solve, instead of (5.3), we solve the right-preconditioned system

$$J_n M_n^{-1}(M_n \delta X_n) = -N(X_n) \tag{5.4}$$

by using a Krylov subspace method such as GMRES. Here M_n^{-1} is a preconditioner based on the domain decomposition method. The GMRES iteration stops when the linear residual $r_n = J_n \delta X_n + N(X_n)$ satisfies

$$\|r_n\|_2 \leq \max\left\{\zeta_r \|N(X_n)\|_2, \zeta_a\right\},$$

where $\zeta_r, \zeta_a > 0$ are the relative and absolute tolerances respectively. Here ζ_r, also called the nonlinear forcing term, is fixed in our study. Some more advanced techniques for setting ζ_r, e.g., [9], may further improve the performance of the solver.

At each Newton step, when solving the Jacobian system by GMRES, the Jacobian matrix itself is not explicitly needed; instead, only the matrix-vector multiplication is required. Therefore we use a Jacobian-free method [23] in which the matrix-vector multiplication of J_n and Y is approximated by

$$J_n Y \approx \frac{N(X_n + \epsilon Y) - N(X_n)}{\epsilon}, \tag{5.5}$$

where $\epsilon > 0$ is a small number calculated by using a technique suggested in [33]. We remark here that the Jacobian matrix can also be generated analytically, or by using some other methods such as the multi-coloring finite difference method [5] or the automatic differentiation method [16]. An advantage of generating the Jacobian matrix explicitly is that the sparse matrix-vector multiplication can be carried out with fewer floating-point operations than using (5.5). However, some extra local memory is needed to store the matrix and the efficiency of the sparse matrix-vector multiplication might be lower than that of (5.5) on modern heterogeneous supercomputers, on which memory bandwidth is often a bottleneck. For the above reason, in this study we choose to use the Jacobian-free method instead of generating the Jacobian matrix explicitly. It is worth mentioning that, for the preconditioning operation, we do need to build an approximate Jacobian matrix explicitly as discussed in the next subsection.

5.2. Restricted additive Schwarz preconditioner. To define the preconditioner M^{-1} in (5.4), we first decompose the computational domain Ω into np nonoverlapping subdomains Ω_p $(p = 1, \cdots, np)$, such that $\overline{\Omega} = \cup_{p=1}^{np} \overline{\Omega}_p$ and $\Omega_p \cap \Omega_{p'} = \emptyset, \forall p \neq p'$. Here np is the number of subdomains and also the number of processor cores. Then within Ω each subdomain Ω_p is extended by δ layers of mesh cells to

Ω_p^δ, resulting in an overlapping decomposition $\Omega = \cup_{p=1}^{np}\Omega_p^\delta$. The classical additive Schwarz (AS(δ) or simply AS, [7]) preconditioner is defined as

$$M_{AS(\delta)}^{-1} = \sum_{p=1}^{np}(R_p^\delta)^T B_p R_p^\delta. \tag{5.6}$$

Here B_p represents a certain subdomain solver, R_p^δ and $(R_p^\delta)^T$ are the restriction and prolongation operators respectively. Given a solution vector (4.2) defined on all mesh cells in Ω, R_p^δ restricts the vector to a vector that is defined only on the mesh cells within the overlapping subdomain Ω_p^δ; while $(R_p^\delta)^T$ prolongates the restricted vector back to a vector defined on all mesh cells in the whole domain Ω with zeros filled to the components corresponding to mesh cells outside Ω_p^δ. In particular when the overlap is zero, the AS preconditioner becomes the block Jacobi preconditioner.

There are several modifications of the AS preconditioner that may have some potential advantages. Among them a popular one is the left restricted additive Schwarz (RAS(δ) or simply RAS, [3]) preconditioner that reads

$$M_{RAS(\delta)}^{-1} = \sum_{p=1}^{np}(R_p^0)^T B_p R_p^\delta. \tag{5.7}$$

The only difference between the RAS preconditioner and the AS preconditioner is the extension operator. Instead of $(R_p^\delta)^T$, the RAS preconditioner uses $(R_p^0)^T$ which puts zeros at components corresponding to mesh cells outside the non-overlapping subdomain Ω_p.

In an additive Schwarz preconditioner, it is a common practice to set the subdomain solver to be $B_p = \text{inv}(J_{n,p})$, which is either a direct or an approximate inverse of the subdomain Jacobian matrix

$$J_{n,p} = R_p^\delta J_n (R_p^\delta)^T.$$

However, using the above formula to calculate B_p could be expensive due to the high bandwidth and the large number of non-zeros in the sparse matrix $J_{n,p}$. Therefore, instead of using the original second-order spatial discretization, we generate the subdomain Jacobian matrix associated with a first-order spatial discretization, and calculate B_p based on it, i.e.,

$$B_p = \text{inv}(J_{n,p}^{1st}), \quad J_{n,p}^{1st} = R_p^\delta J_n^{1st} (R_p^\delta)^T. \tag{5.8}$$

Here the first-order spatial discretization is obtained by simply replacing the piecewise linear reconstruction (3.15) with a piecewise constant reconstruction

$$\hat{Q}^-(\hat{\mathbf{x}}_{ijk}, t) = Q_{ij}(t), \quad \hat{Q}^+(\hat{\mathbf{x}}_{ijk}, t) = Q_{i+1,j}(t), \tag{5.9}$$

in the cell-centered finite volume scheme. By using the first-order scheme, the number of non-zeros in the Jacobian matrix is reduced nearly by half. It has been reported that, in addition to reducing the cost of subdomain solves, using a first-order scheme to generate the subdomain Jacobian matrix may further improve the convergence of the linear solver in some cases studied in [47, 48].

We remark here that neither direct nor approximate inverse of $J_{n,k}^{1st}$ is explicitly calculated in solving subdomain problems. Instead, only the matrix-vector multiplications of $\text{inv}(J_{n,k}^{1st})$ are required. Since the point-block ordering is used for both

the unknowns and the discretized equations, each entry of the Jacobian matrix is a 4×4-block. We then use the point-block version of either the sparse LU or the sparse incomplete LU (ILU) factorization for subdomain solves. The fill-in level ℓ for the ILU method is adjusted to achieve good performance in terms of the total compute time. It is also worth mentioning that since the Jacobian matrices of different Newton iterations have similar structures, it helps save some compute time by performing the LU factorization only once and reusing the factorized matrix within the same nonlinear solve, as done in, e.g., [47].

6. Numerical tests. We carry out numerical experiments on a newly announced supercomputer, Tianhe-2, which tops the Top-500 list as of June, 2013. The computing nodes of Tianhe-2 are interconnected via a proprietary high performance network, with two 12-core Intel Ivy Bridge Xeon CPUs and 24GB local memory in each node. We implement algorithms proposed in this paper on top of the Portable, Extensible Toolkits for Scientific computations (PETSc) library [1]. In the numerical experiments we use all 24 CPU cores in each node and assign one subdomain to each processor core. The Xeon Phi MIC processors equipped on Tianhe-2 are not utilized in the simulation. No physical dissipations are employed unless explicitly mentioned.

In the fully implicit solver, we set the tolerances as follows. For the Newton iteration, a relative tolerance of $\varepsilon_r = 10^{-6}$ is utilized, the absolute tolerance is initially set to $\check{\varepsilon}_{a,0} = 10^{-9}$ and then adaptively controlled in (5.2) with safeguard $\hat{\varepsilon}_a = 10^{-5}$; and for the GMRES iteration, the relative and absolute tolerances are respectively $\zeta_r = 10^{-3}$ and $\zeta_a = 10^{-11}$. The restarting parameter in GMRES is set to be 30.

6.1. Validation by several test cases. In this subsection, we validate the discretization scheme and the fully implicit solver by running several previously published test cases[1].

6.1.1. Test case 1: Density current. The first test case, proposed by Straka [41], describes the dynamics of a density current (cold air bubble), including the shape-shearing of the bubble as it travels downwardly and the development of Kelvin-Helmholtz instabilities after it hits the ground. The simulation is done on a rectangular domain $(-25.6\,\text{km}, 25.6\,\text{km}) \times (0, 6.4\,\text{km})$ with rigid-walls on the boundaries. The flow is initially homogeneous in the horizontal and at the hydrostatic balance with $\overline{\mathbf{v}} = 0$ and $\overline{\theta} = 300\,\text{K}$. A density current is introduced in Ω by perturbing the temperature field $\overline{T} = \overline{p}/(\overline{\rho}R)$ with

$$T' = \begin{cases} T_m \cos^2(\pi L/2), & \text{if } L \le 1.0, \\ 0\,\text{K}, & \text{otherwise}, \end{cases} \qquad L = \sqrt{\frac{(x-x_c)^2}{x_r^2} + \frac{(z-z_c)^2}{z_r^2}}, \tag{6.1}$$

where $T_m = -15\,\text{K}$, $(x_c, z_c) = (0\,\text{km}, 3\,\text{km})$ and $x_r = 4\,\text{km}$, $z_r = 2\,\text{km}$. To obtain mesh-converged solutions in the test, a physical dissipation with $\nu = 75.0\,\text{m}^2/\text{s}$ is utilized in the calculation.

Due to the symmetry of the problem, we conduct the simulation only on the right half of the domain. We show in Figure 6.1 the computed results obtained on a 1024×256 mesh by using the fully implicit ESDIRK(2) method. Starting with $\Delta t_0 = 2.5$ seconds and adjusting the time step size by (4.7), we finish the simulation at $t = 900\,\text{s}$ after 213 time steps, leading to an average time step size $\Delta t_{avg} = 4.2$

[1]Geometric and physical parameters in this section are given without nondimensionalization although they are nondimensionalized in the implementation of the algorithms.

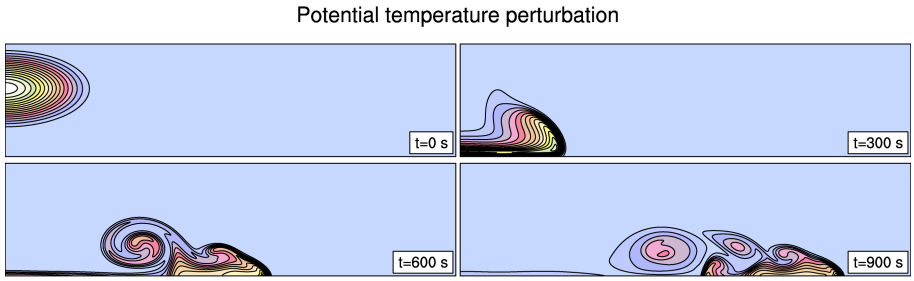

FIG. 6.1. *Test case 1: Density current. Mesh resolution:* $\Delta x = \Delta z = 50\,\mathrm{m}$*, time step size:* $\Delta t_0 = 2.5\,\mathrm{s}$*,* $\Delta t_{avg} = 4.2\,\mathrm{s}$*. The four panels show contour plots of the potential temperature perturbation at* $t = 0\,\mathrm{s}$*,* $300\,\mathrm{s}$*,* $600\,\mathrm{s}$ *and* $900\,\mathrm{s}$*, respectively. The contour range is between* $-14.5\,\mathrm{K}$ *and* $-0.5\,\mathrm{K}$ *with a contour interval of* $1\,\mathrm{K}$*.*

seconds. Contour plots of the potential temperature perturbation at $t = 0$, 300, 600, and 900 seconds are given in the figure. Only the portion of $[0, 18\,\mathrm{km}] \times [0, 5\,\mathrm{km}]$ is shown in the plots. We find that the results are consistent to published solutions in, e.g., [31, 43]. In particular, the general pattern of the three well-developed Kelvin-Helmholtz rotors found at $t = 900\,\mathrm{s}$ are similar to reference ones.

Denote the total discrete summation of a physical variable ϕ as

$$\sigma(\phi) = \frac{1}{n_1 n_2} \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} \phi_{ij},$$

where n_1, n_2 are the numbers of mesh cells in the x and z directions respectively. Then the numerical conservation error of the total mass at $t = t_m$ is defined as

$$\frac{\sigma(\rho(t_m) - \rho(t_0))}{\sigma(\rho(t_0))}. \tag{6.2}$$

Due to the consistent calculation of numerical fluxes on the edges of each mass cell, the spatial discretization is exactly conservative in terms of the total mass. To examine the mass conservation (6.2), we run the test case again on a 1024×256 mesh by using both the fully implicit ESDIRK(2) and the explicit SSP RK-2 methods. In the fully implicit run, we use the same time step size as in the previous run; while in the explicit run, the time step size is fixed to be $\Delta t = 0.03\,\mathrm{s}$. Figure 6.2 shows the results on the mass conservation (6.2). Note that although the same spatial discretization is used in both methods, the behavior of mass conservation is different. For the explicit method, the mass conservation is exact to the machine precision, as shown in the left panel of the figure. However, for the fully implicit method, due to the inexact solution of the nonlinear system at each time step, the conservation error is beyond the machine precision, as seen in the right panel of the figure. Overall, this error is relatively small compared to the nonlinear residual and more importantly, it does not accumulate over time, which is essential in atmospheric modeling.

This test case is also used to examine the accuracy of the fully implicit scheme. To exclude the influence from the error introduced by the NKS solver, we use a more rigid stopping condition for the Newton iteration: $\varepsilon_r = 10^{-8}$ and $\check{\varepsilon}_{a,0} = \hat{\varepsilon}_a = 10^{-11}$, although we find later that the default stopping condition is sufficiently accurate.

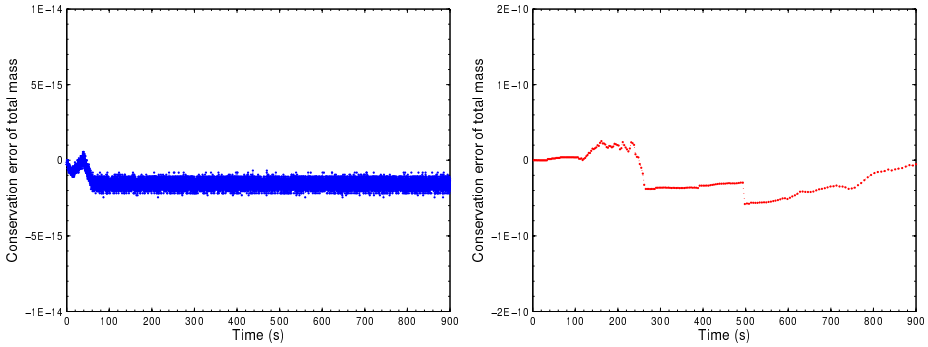

FIG. 6.2. *Mass conservation results for the density current problem on a* 1024×256 *mesh. Left panel: the mass conservation error of the explicit SSP RK-2 method with* $\Delta t = 0.03\,\mathrm{s}$. *Right panel: the mass conservation error of the fully implicit ESDIRK(2) method using the adaptive time stepping started with* $\Delta t_0 = 2.5\,\mathrm{s}$.

Following Straka [41], the L^2-error of θ' at $t = 900\,\mathrm{s}$ is defined as

$$L^2(\theta') = \sqrt{\sigma\left(\left[\theta'_{ij} - (\theta')^{(ref)}_{ij}\right]^2\right)}, \tag{6.3}$$

where $(\theta')^{(ref)}_{ij}$ is obtained from a reference solution. In order to quantify the error of the spatial discretization and cancel out the temporal error, we fix the time step size to $\Delta t = 1\,\mathrm{s}$ and use the solution of 6.25 m resolution (corresponding to a 4096×1024 mesh) as the reference solution. The L^2-errors with respect to different mesh resolutions are shown in the left panel of Figure 6.3, in which we also draw the ideal second-order convergence line. It is clear to see that the spatial discretization is second-order accurate. We then fix the mesh resolution to be 25 m and use the

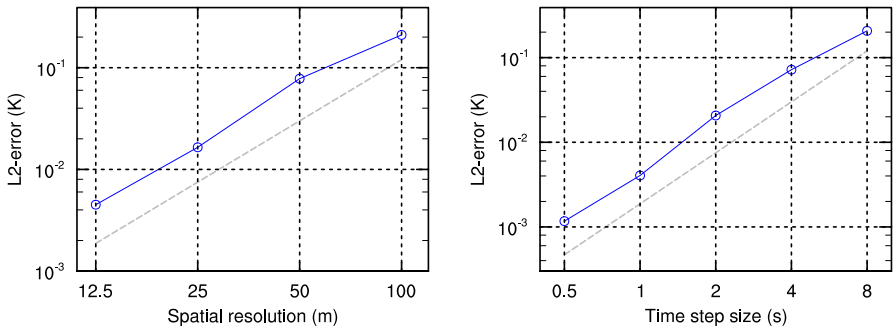

FIG. 6.3. *Convergence analysis of the density current problem. Left panel: the* L^2*-error of* θ' *with respect to different mesh resolutions (fixed time step size* $\Delta t = 1\,\mathrm{s}$*); the reference solution is obtained with* 6.25 m *resolution. Right panel: the* L^2*-error of* θ' *with respect to different time step sizes (fixed mesh resolution* 25 m*); the reference solution is obtained with* $\Delta t = 0.1\,\mathrm{s}$. *In both panels, the gray lines indicate the ideal second-order convergence.*

solution of $\Delta t = 0.1\,\mathrm{s}$ as the reference solution in (6.3). The L^2-errors with respect to different time step size are provided in the right panel of Figure 6.3, where the ideal second-order convergence line is also shown. Again, it is evident that second-order accuracy is achieved with the ESDIRK(2) temporal integration scheme.

6.1.2. Test case 2: Interacting bubbles. The second test case, proposed by Robert [35], studies the interaction of a rising warm bubble and a descending cold bubble inside a square domain $\Omega = (0, 1\,\mathrm{km})^2$. Analogous to the first test case, the flow is initially at the hydrostatic rest with $\overline{\mathbf{v}} = 0$ and $\overline{\theta} = 303.15\,\mathrm{K}$. The two bubbles added to the domain both have a Gaussian profile:

$$\theta' = \begin{cases} \theta_m, & \text{if } L \le s, \\ \theta_m e^{-\frac{(L-s)^2}{r^2}}, & \text{otherwise,} \end{cases} \qquad L = \sqrt{(x - x_c)^2 + (z - z_c)^2}, \tag{6.4}$$

which describes a bubble, centered at (x_c, z_c), that has an amplitude of θ_m inside a flat area of radius s and decays with a sharpness parameter r outside the area. In the test, the warm bubble has an amplitude of 0.5 K and is placed at (500 m, 300 m) with a flat radius of 150 m. The cold bubble is on top of the warm one at (560 m, 640 m) with an amplitude of $-0.15\,\mathrm{K}$ but without a flat core. Both bubbles have the same sharpness parameter of 50 m. No-flux boundary conditions are applied along the boundaries.

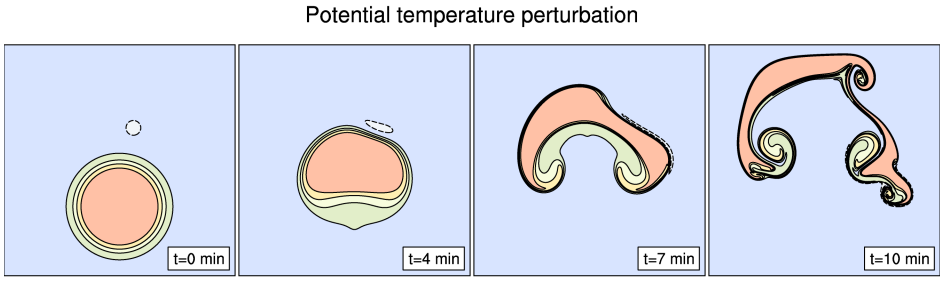

FIG. 6.4. *Test case 2: Interacting bubbles. Mesh resolution:* $\Delta x = \Delta z = 1.25\,\mathrm{m}$*, time step size:* $\Delta t_0 = 0.5\,\mathrm{s}$*,* $\Delta t_{avg} = 2.2\,\mathrm{s}$*. The four panels show contour plots of the potential temperature perturbation at* $t = 0\,\mathrm{min}$*,* $4\,\mathrm{min}$*,* $7\,\mathrm{min}$ *and* $10\,\mathrm{min}$*, respectively. The contour range is between* $-0.1\,\mathrm{K}$ *and* $0.45\,\mathrm{K}$ *with a contour interval of* $0.1375\,\mathrm{K}$*. Dashed contour lines are used to emphasize the* $-0.1\,\mathrm{K}$ *contour level.*

We perform the simulation by using the fully implicit ESDIRK(2) method on a 800×800 mesh. The time step size is initially set to be $\Delta t_0 = 0.5\,\mathrm{s}$ and then adaptively adjusted according to (4.7). The simulation is finished at $t = 10\,\mathrm{min}$ after 270 time steps, resulting in an average time step size $\Delta t_{avg} = 2.2\,\mathrm{s}$. Contour plots of the potential temperature perturbation at $t = 0$, 4, 7 and 10 minutes are shown in Figure 6.4. We find that the resemblance between the simulated results and the published results in [31, 35] is remarkable. The fully implicit method successfully resolves both the large and small scales introduced by the two bubbles in the test.

6.1.3. Test case 3: Inertia-gravity wave. The third test case, introduced by Skamarock & Klemp [38], describes an inertia-gravity wave propagating in a horizontally periodic channel $\Omega = (0, 300\,\mathrm{km}) \times (0, 10\,\mathrm{km})$ with rigid walls along the bottom and top boundaries. The initial condition is set based on a constant horizontal flow of $\overline{u} = 20\,\mathrm{m/s}$ in a uniformly stratified atmosphere with a Brunt-Väisälá frequency of $\mathcal{N} = 0.01\,/\mathrm{s}$. Here the Brunt-Väisälá frequency is defined as $\mathcal{N}^2 = g\frac{d\ln\overline{\theta}}{dz}$. It immediately follows that $\overline{\theta} = \overline{\theta}_0 \exp(\mathcal{N}^2 z/g)$, where the ground temperature is set to be $\overline{\theta}_0 = 300\,\mathrm{K}$. The wave is excited by adding the following perturbation to the potential

temperature field:

$$\theta' = \frac{\theta_m \sin(\frac{\pi z}{z_m})}{1 + \left(\frac{x - x_c}{x_r}\right)^2},$$

where $\theta_m = 0.01\,\mathrm{K}$, $z_m = 10.0\,\mathrm{km}$, $x_c = 100.0\,\mathrm{km}$ and $x_r = 5.0\,\mathrm{km}$.

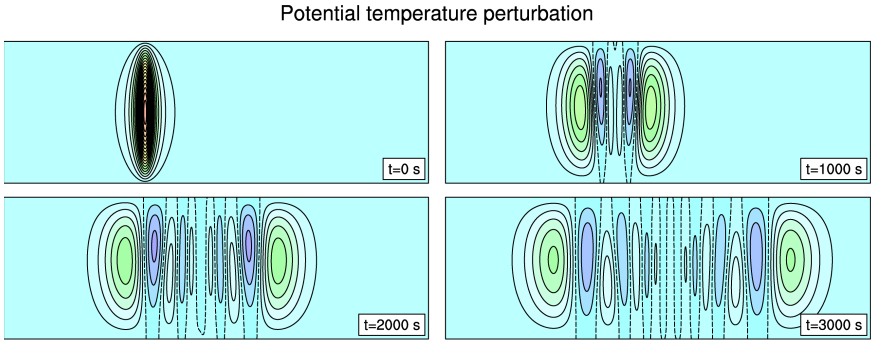

FIG. 6.5. *Test case 3: Inertia-gravity wave. Mesh resolution:* $\Delta x = \Delta z = 100\,\mathrm{m}$*, time step size:* $\Delta t_0 = 10.0\,\mathrm{s}$*,* $\Delta t_{avg} = 30.9\,\mathrm{s}$*. The four panels show contour plots of the potential temperature perturbation at* $t = 0\,\mathrm{s}$*,* $1000\,\mathrm{s}$*,* $2000\,\mathrm{s}$ *and* $3000\,\mathrm{s}$*, respectively. The contour range is between* $-0.0015\,\mathrm{K}$ *and* $0.01\,\mathrm{K}$ *with a contour interval of* $0.0005\,\mathrm{K}$*. Dashed contour lines are used to emphasize the zero contour level.*

Figure 6.5 shows the computed results obtained on a 3000×100 mesh by using the fully implicit ESDIRK(2) method. The contour plots of the potential temperature perturbation at $t = 0$, 1000, 2000, and 3000 seconds are provided in the figure. In the simulation, we set the initial time step size to be $\Delta t_0 = 10.0\,\mathrm{s}$ and then adaptively vary it according to (4.7). The simulation is stopped at $3000\,\mathrm{s}$ after 97 time steps, resulting in an average time step size $\Delta t_{avg} = 30.9\,\mathrm{s}$. We find that the computed results are comparable to those obtained by using a fixed time step of $\Delta t = 5\,\mathrm{s}$; but both differ from the original reference results provided by Skamarock & Klemp in [38], where the solutions are visually symmetric in the vertical direction. We remark that, in contrast to [38] that uses the Boussinesq approximation, our simulations are based on the unapproximated, fully compressible Euler equations, and are consistent with [4, 15], where similar vertical asymmetry was observed.

6.1.4. Test case 4: Linear hydrostatic mountain. The fourth test case, studied in [8, 39], intends to test the ability of a model to accurately capture vertically propagating linear hydrostatic mountain waves. The initial condition of the atmosphere is assumed to be a constant horizontal flow of $\overline{u} = 20\,\mathrm{m/s}$ in an isothermal state with a constant temperature of $\overline{\theta} = 250\,\mathrm{K}$. It can be verified that the Brunt-Väisälá frequency satisfies $\mathcal{N} = g/\sqrt{c_p \theta}$ due to the isothermal assumption, where $c_p = R\gamma/(\gamma - 1)$. The simulation is done on $\Omega = (0, 240\,\mathrm{km}) \times (h, 30\,\mathrm{km})$, where the mountain profile h is described by a witch of Agnesi curve

$$h(x) = \frac{h_m}{1 + \left(\frac{x - x_c}{x_r}\right)^2}. \tag{6.5}$$

The mountain is centered at $x_c = 120\,\mathrm{km}$ with half-width $x_r = 10\,\mathrm{km}$ and height $h_m = 1\,\mathrm{m}$. These conditions ensure that the problem is close to linear and the

flow is within the hydrostatic range because: (i) the meteorological Froude number $F_r = \overline{u}/(\mathcal{N}h_m)$, which measures the linearity, is about 1000; and (ii) the long wave parameter $\mu = \overline{u}/(\mathcal{N}x_r)$, which controls the dispersive effects deviating from the hydrostatic balance, is around 0.1. No-flux boundary conditions are specified along the bottom boundary and sponge layers are utilized outside the domain of interest $\Omega^* = (80\,\text{km}, 160\,\text{km}) \times (h, 12\,\text{km})$.

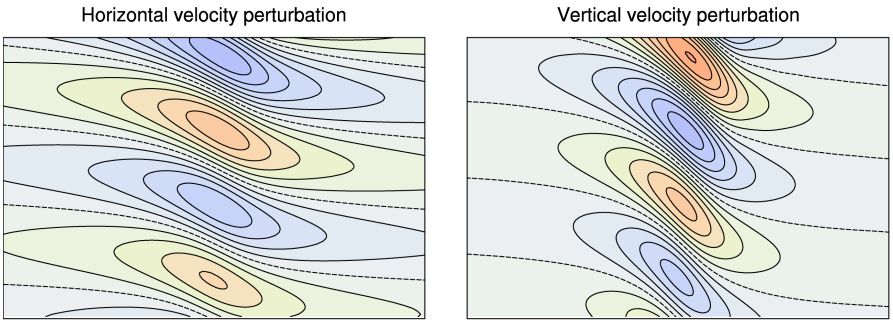

FIG. 6.6. *Test case 4: Linear hydrostatic mountain. Mesh resolution:* $\Delta x = 600\,\text{m}$, $\Delta z = 125\,\text{m}$, *time step size:* $\Delta t_0 = 10.0\,\text{s}$, $\Delta t_{avg} = 666.7\,\text{s}$. *Left panel: the contour plot of the horizontal velocity perturbation at* $t = 10\,\text{h}$, *where the contour range is between* $-0.05\,\text{m/s}$ *and* $0.05\,\text{m/s}$ *with a contour interval of* $0.005\,\text{m/s}$. *Right panel: the contour plot of the vertical velocity perturbation at* $t = 10\,\text{h}$, *where the contour range is between* $-0.005\,\text{m/s}$ *and* $0.005\,\text{m/s}$ *with a contour interval of* $0.0005\,\text{m/s}$. *Dashed contour lines are used to emphasize the zero contour level.*

We carry out the simulation by using the fully implicit ESDIRK(2) method on a 400×240 mesh. Starting with $\Delta t_0 = 10.0\,\text{s}$ and following the adaptation in (4.7), the simulation is finished at $t = 10$ hours after 54 time steps, leading to an average time step size $\Delta t_{avg} = 666.7\,\text{s}$. For this test, the adaptive time stepping method is able to adjust the time step size by over three orders of magnitude, and the simulated results, as shown in Figure 6.6, agree well with published results in, e.g., [15].

6.1.5. Test case 5: Linear nonhydrostatic mountain. The fifth test case differs from the fourth one by having both hydrostatic and nonhydrostatic wave components. It is used to test the capability of a model to simulate nonhydrostatic topographic flows that have small amplitudes. The simulation is carried out on a domain $(0, 144\,\text{km}) \times (h, 30\,\text{km})$ which is horizontally smaller than that in test case 4. Analogous to test case 4, the mountain profile h is given in (6.5) with height $h_m = 1\,\text{m}$, but with a different center position $x_c = 72\,\text{km}$ and with a different half-width $x_r = 1\,\text{km}$. The initial condition of the atmosphere is taken to be a constant horizontal flow of $\overline{u} = 10\,\text{m/s}$ in a uniformly stratified atmosphere with a Brunt-Väisälá frequency of $\mathcal{N} = 0.01\,/\text{s}$ and a ground temperature of $\overline{\theta}_0 = 280\,\text{K}$. The problem is also close to linear because $F_r = \overline{u}/(\mathcal{N}h_m) = 1000$; but is beyond the hydrostatic balance since $\mu = \overline{u}/(\mathcal{N}x_r) = 1$. No-flux boundary conditions are specified along the bottom topography and non-reflecting boundary conditions are enforced along the lateral and top boundaries by adding sponge layers outside the domain of interest $\Omega^* = (60\,\text{km}, 115\,\text{km}) \times (h, 12\,\text{km})$.

We run the test on a 800×200 mesh by using the fully implicit ESDIRK(2) method. In the simulation, we again use an initially small time step size $\Delta t_0 = 10.0\,\text{s}$ and adaptively change it according to (4.7). We find that the efficiency of the adaptive time stepping method for this test case is comparable to the previous one. The simulation ends at $t = 5$ hours after 33 time steps, with an average time step size

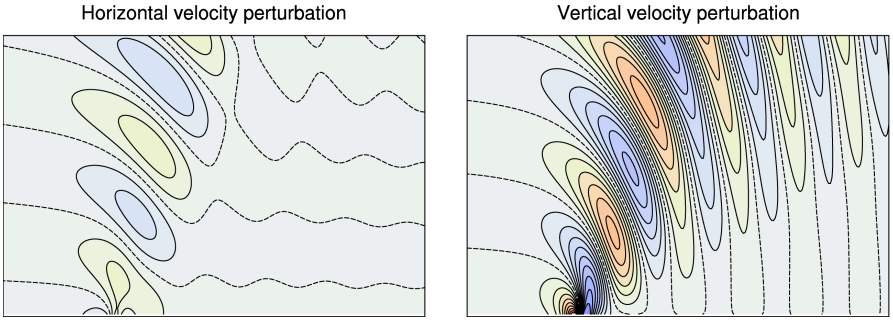

FIG. 6.7. *Test case 5: Linear nonhydrostatic mountain. Mesh resolution:* $\Delta x = 180\,\mathrm{m}$, $\Delta z = 150\,\mathrm{m}$, *time step size:* $\Delta t_0 = 10.0\,\mathrm{s}$, $\Delta t_{avg} = 545.5\,\mathrm{s}$. *Left panel: the contour plot of the horizontal velocity perturbation at* $t = 5\,\mathrm{h}$, *where the contour range is between* $-0.01\,\mathrm{m/s}$ *and* $0.01\,\mathrm{m/s}$ *with a contour interval of* $0.0025\,\mathrm{m/s}$. *Right panel: the contour plot of the vertical velocity perturbation at* $t = 5\,\mathrm{h}$, *where the contour range is between* $-0.01\,\mathrm{m/s}$ *and* $0.01\,\mathrm{m/s}$ *with a contour interval of* $0.0005\,\mathrm{m/s}$. *Dashed contour lines are used to emphasize the zero contour level.*

$\Delta t_{avg} = 545.5\,\mathrm{s}$. We show in Figure 6.7 the contour plots of the horizontal and the vertical velocity perturbations at $t = 5\,\mathrm{hr}$. The adaptive control of time step size is successful and the simulated results agree well with reference solutions in, e.g., [15].

6.1.6. Test case 6: Schär mountain. This test case was introduced by Schär et al. in [37]. The initial state of the flow has a constant horizontal velocity $\overline{u} = 10\,\mathrm{m/s}$ and is in a uniformly stratified atmosphere with a Brunt-Väisälá frequency of $\mathcal{N} = 0.01\,/\mathrm{s}$ and a ground temperature of $\overline{\theta}_0 = 280\,\mathrm{K}$. We run the test on the domain $\Omega = (-25\,\mathrm{km}, 25\,\mathrm{km}) \times (h, 21\,\mathrm{km})$ whose bottom boundary h is given by

$$h(x) = h_m e^{-(\frac{x}{r})^2} \cos^2\left(\frac{\pi x}{\lambda_c}\right),$$

where $h_m = 250\,\mathrm{m}$, $r = 5\,\mathrm{km}$ and $\lambda_c = 4\,\mathrm{km}$. Compared to the previous two test cases with single-peak mountains, the mountain profile in this test case contains rapidly decaying peaks located at $x = (4n)$ km, for $n = 0, \pm 1, \pm 2, \ldots$ We specify no-flux boundary condition along the bottom boundary and apply non-reflecting boundary conditions by adding sponge layers outside the domain of interest $\Omega^* = (-10\,\mathrm{km}, 10\,\mathrm{km}) \times (h, 10\,\mathrm{km})$.

Figure 6.8 shows the computed results obtained on a 500×200 mesh by using the fully implicit ESDIRK(2) method. The contour plots of both horizontal and vertical velocity perturbations at $t = 10$ hours are drawn. In the simulation, similar to the previous two test cases with topography, we set the initial time step size to be $\Delta t_0 = 10.0\,\mathrm{s}$ and then adaptively control it by using (4.7). Once again, the adaptive time stepping strategy works well in the test, with an average time step size $\Delta t_{avg} = 947.4\,\mathrm{s}$. In addition, the large time step size doesn't degrade the accuracy, the computed results agree well with published results in, e.g., [15, 43], although much smaller time steps are used in their works.

6.1.7. Summary of CFL numbers. We summarize in Table 6.1 the respective CFL numbers due to both the fast acoustic wave and the advection defined in (4.3) for the six test cases. It is obvious that for different test cases, the adaptive time stepping method works very differently. For the three test cases that involve steady-state simulation of mountain waves, the fully implicit time step is adjusted to be

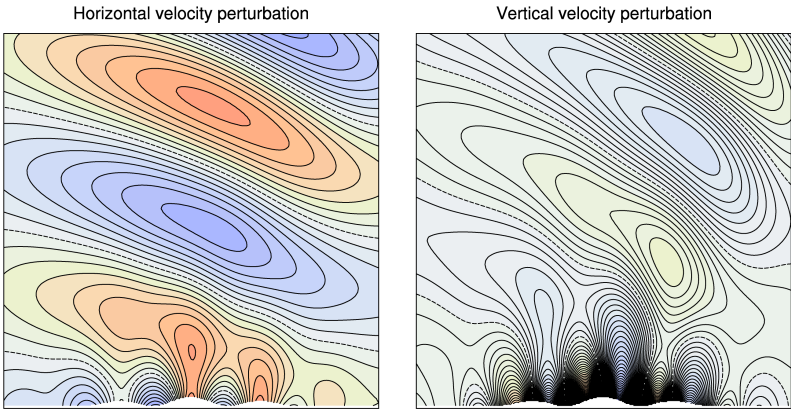

FIG. 6.8. *Test case 6: Schär mountain. Mesh resolution:* $\Delta x = 100\,\mathrm{m}$, $\Delta z = 105\,\mathrm{m}$, *time step size:* $\Delta t_0 = 10.0\,\mathrm{s}$, $\Delta t_{avg} = 947.4\,\mathrm{s}$. *Left panel: the contour plot of the horizontal velocity perturbation at* $t = 10\,\mathrm{h}$, *where the contour range is between* $-2\,\mathrm{m/s}$ *and* $2\,\mathrm{m/s}$ *with a contour interval of* $0.2\,\mathrm{m/s}$. *Right panel: the contour plot of the vertical velocity perturbation at* $t = 10\,\mathrm{h}$, *where the contour range is between* $-2\,\mathrm{m/s}$ *and* $2\,\mathrm{m/s}$ *with a contour interval of* $0.05\,\mathrm{m/s}$. *Dashed contour lines are used to emphasize the zero contour level.*

TABLE 6.1
Summary of the acoustic and the advective CFL numbers for test case 1 to 6.

Test case	1	2	3	4	5	6
Averaged CFL_f	58.3	617.2	106.6	1418.5	1220.3	3158.1
Averaged CFL_a	8.5	7.4	6.3	106.7	36.4	112.0

thousands of times larger as compared to the explicit method. For the other three test cases, the time step size is moderate, stepping over acoustic time scale by at least 50 times, while keeping within the range of one magnitude larger than the advective time scale. We remark that by increasing the spatial resolution, the advantage of the fully implicit method with adaptive time stepping is more evident, as seen in the weak-scaling tests to be discussed in the next subsection.

6.2. Performance of the fully implicit algorithm. In this subsection, we focus on the parallel performance of the proposed fully implicit solver.

6.2.1. Influence of different overlaps and subdomain solvers. There are several performance-related parameters in the NKS algorithm. To investigate the influence of these parameters, we first focus on the Schär mountain problem by runing the tests with 288 processor cores. The experiments are carried out on a 3072×1536 mesh with a fixed time step size $\Delta t = 10\,\mathrm{s}$ for the first ten time steps. We adjust the overlap from 1 to 5 and try different subdomain solvers including the sparse LU factorization and the sparse ILU factorization with different levels of fill-in. It is observed that the total number of Newton iterations is insensitive to the parameters and is always 2 for each nonlinear solve. The number of linear iterations and the compute time, however, both vary with the parameters, as summarized in Table 6.2. We remark that when the overlap is zero, the RAS preconditioner degenerates to the block-Jacobi preconditioner and the convergence of the solver is quite slow. Therefore we do not include the results of the non-overlapping preconditioner in the table.

From Table 6.2, we first note that by increasing the overlap or by increasing the

TABLE 6.2
Performance of NKS when using different overlaps and subdomain solvers. Tests are done for solving the Schär mountain problem on a 3072×1536 *mesh with* 288 *processor cores. The time step size is fixed to* $\Delta t = 10\,\mathrm{s}$ *and the results are averaged over the first ten time steps.*

	GMRES/Newton					Compute time (s)/Newton				
δ	1	2	3	4	5	1	2	3	4	5
ILU(0)	48.1	41.7	39.8	39.6	40.0	1.69	1.47	1.47	1.48	1.49
ILU(1)	44.8	39.2	38.6	38.1	37.9	1.66	1.45	1.46	1.46	1.47
ILU(2)	44.4	38.9	38.2	37.9	36.2	1.69	1.46	1.48	1.49	1.50
ILU(3)	43.2	38.2	37.4	37.1	36.9	1.78	1.58	1.59	1.59	1.60
ILU(4)	41.9	36.9	35.9	35.5	35.3	1.87	1.65	1.66	1.66	1.68
ILU(5)	40.0	35.4	34.1	33.6	33.4	1.95	1.70	1.71	1.71	1.73
LU	25.6	22.3	21.4	19.9	20.7	2.52	2.42	2.44	2.53	2.59

fill-in level, the number of GMRES iterations becomes smaller, but the compute time doesn't necessarily reduce because of the increased cost of subdomain solves. In particular, when the sparse LU factorization instead of ILU is used as the subdomain solver, GMRES converges with fewer iterations but the compute time is larger because LU is more expensive than ILU. Overall, we find that the optimal choice in terms of compute time is RAS(2) with ILU(1) subdomain solver. Note that the performance is quite close for the combination of RAS(2), RAS(3) or RAS(4) with ILU(0), ILU(1), ILU(2) or ILU(3). For comparison, we also try to replace the RAS(2) preconditioner with the AS(2) preconditioner in the same test. And we find that both the number of GMRES iterations and the compute time increase when AS(2) is used. This observation is in consistency with other reports [3].

6.2.2. Strong scaling results. We perform the strong scaling test using the same test case on a fixed 6144×3072 mesh and different number of processor cores. The test is run with a fixed time step $\Delta t = 10\,\mathrm{s}$ to $t = 100\,\mathrm{s}$. In the test, we use RAS(2) with both sparse ILU(1) and LU as the subdomain solver. We show in Table 6.3 the average numbers of Newton and GMRES iterations as well as the total compute time. From the table we observe that, when the number of processor cores (np in the

TABLE 6.3
Strong scaling results for solving the Schär mountain problem on a 6144×3072 *mesh to* $t = 100\,\mathrm{s}$. *For the explicit SSP RK-2 method, the time step size is* $\Delta t = 0.01\,\mathrm{s}$*; for the fully implicit method, the time step size is* $\Delta t = 10\,\mathrm{s}$ *and the overlap is* $\delta = 2$.

	Newton/Step		GMRES/Newton		Compute time (s)		
np	ILU(1)	LU	ILU(1)	LU	ILU(1)	LU	Exp.
576	2.1	2.1	69.6	34.6	222.6	299.8	748.7
1152	2.1	2.1	70.9	37.3	108.3	140.2	368.4
2304	2.1	2.1	74.5	41.9	57.9	72.3	182.1
4608	2.1	2.1	75.7	44.8	29.1	37.8	91.4
9216	2.1	2.1	82.1	52.0	16.1	20.2	46.9
18432	2.1	2.1	84.5	55.5	8.3	10.4	23.7

table) is doubled, the number of Newton iterations does not change and the number of GMRES iterations increases mildly in the NKS solver. In terms of the total compute time, both methods scale well with up to $18,432$ processor cores. For comparison purpose, we also include the performance results of the explicit SSP RK-2 method,

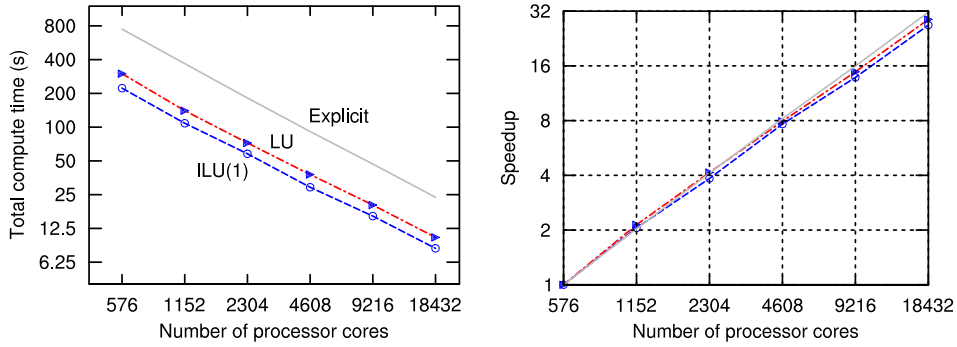

FIG. 6.9. *Strong scaling results for solving the Schär mountain problem as in Table 6.3. The dashed lines with circle markers represent for the fully implicit method preconditionered by RAS(2) with ILU(1) subdomain solver. The dot-dashed lines with triangle markers represent for the fully implicit method preconditionered by RAS(2) with LU subdomain solver. The gray lines indicate the results of the explicit method.*

which requires a lot more time steps due to the CFL restriction. Figure 6.9 displays the total compute time and the speedup with respect to the number of processor cores. We observe from the figure that both the explicit and the implicit methods have nearly linear speedup, and the implicit methods outperform the explicit method by several times in terms of the total compute time.

6.2.3. Weak scaling results. A weak scaling test is carried out to further examine the performance of the solver when the problem size is increased in proportion to the number of processor cores. In this test, we are particularly interested in stopping the calculation at a target simulation time. Table 6.4 shows the performance results in the weak scaling test, in which both the inertial-gravity wave and the Schär mountain problems are studied. For the inertial-gravity wave case, we start with 96 processor cores and a 3072×96 mesh; and for the Schär mountain, we start with 72 processor cores and a 576×288 mesh. The number of processor cores is increased as the mesh is refined accordingly. We use an initial time step size of $\Delta t_0 = 10\,\mathrm{s}$ in (4.7) for the inertial-gravity wave problem and finish the simulation at $t = 3000\,\mathrm{s}$; while for the Schär mountain problem, the initial time step size is $\Delta t_0 = 10\,\mathrm{s}$ and the simulation is terminated at $t = 10\,\mathrm{hr}$. We show in the table the results obtained with the fully implicit ESDIRK(2) method and the explicit SSP RK-2 method. The preconditioner for the fully implicit solver is RAS(2) with LU as the subdomain solver. We comment that ILU may take shorter computing time but LU is more stable especially when the time step size is large.

Below we list several observations made from Table 6.4.

- For the explicit method, the total number of time steps as well as the total compute time doubles as the mesh is refined for both test cases; this is due to the stability restriction on the time step size.
- For the inertial-gravity wave case, both the total number of implicit time steps and the total number of Newton iterations increase very slowly as more processor cores are used; but the total compute time increases more rapidly due to the increased number of GMRES iterations. Overall, the implicit method is always a few times faster than the explicit method.
- For the Schär mountain case, the total number of implicit time steps is more sensitive to the mesh resolution than the case of inertial-gravity wave. And

TABLE 6.4
Weak scaling results of the fully implicit ESDIRK(2) method and the explicit SSP RK-2 method. Both the inertial-gravity wave problem and the Schär mountain problem are tested.

	Inertia-gravity wave			Schär mountain		
np	96	384	1536	72	288	1152
Mesh size (in x)	3072	6144	12288	576	1152	2304
Mesh size (in z)	96	192	384	288	576	1152
Implicit time steps	96	96	97	38	66	89
Total Newton	235	239	244	135	187	241
Total GMRES	2971	4075	5850	14035	28332	54537
Total compute time (s)	58.3	84.2	122.3	115.7	254.7	502.9
Explicit time steps	20000	40000	80000	360000	720000	1440000
Total compute time (s)	122.5	245.1	491.4	1863.7	3716.6	7456.3

as a result, the total numbers of Newton and GMRES iterations, as well as the total compute time, increase rapidly as the mesh is refined. Although not ideally scaling, the implicit method still outperforms the the explicit method by over 13 times.

It is worth mentioning that for the weak scaling tests, the theoretically ideal situation is that the total compute time remains as a constant as the number of processor cores is increased. As shown by the experiments, neither the explicit method nor the implicit method reaches the ideal performance. In order to improve the weak scaling performance of the fully implicit solver, we believe coarse level corrections are required in the additive Schwarz preconditioner and plan to look into this issue in the future.

6.2.4. Performance on using different time steps. To analyze the behavior of the implicit solver as the time step size is changed, we run again the Schär mountain test on a fixed 1152×576 mesh using 288 processor cores. In the test, the adaptive time stepping is turned off and different time step sizes are tried. The simulation is stopped at $t = 600$ seconds. The results on the average numbers of Newton and GMRES iterations as well as the total compute time are summarized in Table 6.5. It is

TABLE 6.5
Performance results on using different time step sizes. The Schär mountain problem is solved on 1152×576 *mesh using* 288 *processor cores. The simulation is stopped at* $t = 600$ *seconds.*

Δt	2	5	10	20	50
Implicit time steps	300	120	60	30	12
Newton/Step	4.0	4.0	4.0	4.1	4.2
GMRES/Newton	7.7	10.5	13.3	18.3	30.2
Total compute time (s)	192.7	69.6	43.8	19.6	12.8

observed from the results that, the average number of Newton iterations for each time step is nearly independent of the time step size; but the average number of GRMRES iterations increases as the time step size is increased. Overall, fewer time steps are required when we use larger time steps, leading to the reduce of the total compute time. Since increasing the time step size may cause larger discretization error, there is a trade-off between the accuracy and the time step size, which is the reason for using the adaptive time stepping.

7. Concluding remarks. In this paper, we developed a fully implicit method for solving the compressible Euler equations in mesoscale atmospheric simulations. A cell-centered finite volume scheme based on an AUSM$^+$-up method and a second-order accurate piecewise linear reconstruction is applied to spatially discretize the Euler equations. In order to stabilize physically insignificant fast waves and accurately integrate the Euler equations in time, we employ a second-order ESDIRK method together with an adaptive time-stepping strategy. The nonlinear system arising at each implicit time step is solved by using a Jacobian-free Newton-Krylov-Schwarz algorithm. To reduce the computation and communication cost of the preconditioning operator, the additive Schwarz preconditioner is built based on a first-order spatial discretization. Numerical results on several test cases are provided to validate the accuracy and examine the parallel performance of the proposed fully implicit method.

Acknowledgments. The authors would like to express their appreciations to the anonymous reviewers for the invaluable comments that have greatly improved the quality of the manuscript.

REFERENCES

[1] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc users manual*, Tech. Report ANL-95/11 - Revision 3.4, Argonne National Laboratory, July 2013.

[2] L. Bonaventura, *A semi-implicit semi-Lagrangian scheme using the height coordinate for a nonhydrostatic and fully elastic model of atmospheric flows*, J. Comput. Phys., 158 (2000), pp. 186–213.

[3] X.-C. Cai and M. Sarkis, *A restricted additive Schwarz preconditioner for general sparse linear systems*, SIAM J. Sci. Comput., 21 (1999), pp. 792–797.

[4] X. Chen, N. Andronova, B. Van Leer, J. E. Penner, J. P. Boyd, C. Jablonowski, and S.-J. Lin, *A control-volume model of the compressible Euler equations with a vertical Lagrangian coordinate*, Mon. Wea. Rev., 141 (2013), pp. 2526–2544.

[5] T. F. Coleman and J. J. Moré, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

[6] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, Philadelphia, 1996.

[7] M. Dryja and O. Widlund, *Domain decomposition algorithms with small overlap*, SIAM J. Sci. Comput., 15 (1994), pp. 604–620.

[8] D. R. Durran and J. B. Klemp, *A compressible model for the simulation of moist mountain waves*, Mon. Wea. Rev., 111 (1983), pp. 2341–2361.

[9] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 1064–8275.

[10] K. J. Evans, D. W. Rouson, A. G. Salinger, M. A. Taylor, W. Weijer, and J. B. White, III, *A scalable and adaptable solution framework within components of the community climate system model*, in Proceedings of the 9th Intl. Conf. on Computational Science (ICCS 2009), part II, Lecture Notes in Computer Science, vol. 5545, Berlin, Heidelberg, 2009, Springer-Verlag, pp. 332–341.

[11] K. J. Evans, M. A. Taylor, and J. B. Drake, *Accuracy analysis of a spectral element atmospheric model using a fully implicit solution framework*, Mon. Wea. Rev., 138 (2010), pp. 3333–3341.

[12] J. E. Fromm, *A method for reducing dispersion in convective difference schemes*, J. Comput. Phys., 3 (1968), pp. 176–189.

[13] A. J. Gadd, *A split explicit integration scheme for numerical weather prediction*, Q. J. R. Meteorol. Soc., 104 (1978), pp. 569–582.

[14] T. Gal-Chen and R. C. J. Somerville, *On the use of a coordinate transformation for the solution of the Navier-Stokes equations*, J. Comput. Phys., 17 (1975), pp. 209–228.

[15] F. X. Giraldo and 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*, J. Comput. Phys., 227 (2008), pp. 3849–3877.

[16] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, SIAM, Philadelphia, 2000.
[17] W. D. Gropp, D. K. Kaushik, D. E. Keyes, and B. Smith, *Performance modeling and tuning of an unstructured mesh CFD application*, in Proceedings of the 2000 ACM/IEEE conference on Supercomputing (SC '00), IEEE Computer Society, 2000.
[18] K. Gustafsson and G. Söderlind, *Control strategies for the iterative solution of nonlinear equations in ODE solvers*, SIAM J. Sci. Comput., 18 (1997), pp. 23–40.
[19] V. John and J. Rang, *Adaptive time step control for the incompressible Navier-Stokes equations*, Comput. Meth. Appl. Mech. Eng., 199 (2010), pp. 514–524.
[20] R. Klein, U. Achatz, D. Bresch, O.M. Knio, and P. Smolarkiewicz, *Regime of validity of soundproof atmospheric flow models*, J. Atmos. Sci., 67 (2010), pp. 3226–3237.
[21] J. Klemp and R. B. Wilhelmson, *The simulation of three dimensional convective storm dynamics*, J. Atmos. Sci., 35 (1978), pp. 1070–1096.
[22] D. A. Knoll, L. Chacón, L. G. Margolin, and V. A. Mousseau, *On balanced approximations for time integration of multiple time scale systems*, J. Comput. Phys., 185 (2003), pp. 583–611.
[23] D. A. Knoll and D. E. Keyes, *Jacobian-free Newton-Krylov methods: A survey of approaches and applications*, J. Comput. Phys., 193 (2004), pp. 357–397.
[24] D. A. Knoll, V. A. Mousseau, L. Chacón, and J. Reisner, *Jacobian-free Newton-Krylov methods for the accurate time integration of stiff wave systems*, 25 (2005), pp. 213–230.
[25] M. Kwizak and A. J. Robert, *A semi-implicit scheme for grid point atmospheric models of the primitive equations*, Mon. Wea. Rev., 99 (1971), pp. 32–36.
[26] B. P. Leonard, *A stable and accurate convective modelling procedure based on quadratic upstream interpolation*, Comput. Meth. Appl. Mech. Eng., 19 (1979), pp. 59–98.
[27] M.-S. Liou, *A sequel to AUSM, part II: AUSM$^+$-up for all speeds*, J. Comput. Phys., 214 (2006), pp. 137–170.
[28] P. Lucas, H. Bijl, and A. H. van Zuijlen, *Efficient unsteady high Reynolds number flow computations on unstructured grids*, Comput. Fluids, 39 (2010), pp. 271–282.
[29] V. A. Mousseau, D. A. Knoll, and J. M. Reisner, *An implicit nonlinearly consistent method for the two-dimensional shallow-water equations with Coriolis force*, Mon. Wea. Rev., 130 (2002), pp. 2611–2625.
[30] W. A. Mulder and B. Van Leer, *Experiments with implicit upwind methods for the Euler equations*, J. Comput. Phys., 59 (1985), pp. 232–246.
[31] A. Müller, J. Behrens, F. X. Giraldo, and V. Wirth, *Comparison between adaptive and uniform discontinuous Galerkin simulations in dry 2D bubble experiments*, J. Comput. Phys., 235 (2013), pp. 371–393.
[32] H. Park, R. R. Nourgaliev, R. C. Martineau, and D. A. Knoll, *On physics-based preconditioning of the Navier-Stokes equations*, J. Comput. Phys., 228 (2009), pp. 9131–9146.
[33] M. Pernice and H. F. Walker, *NITSOL: A Newton iterative solver for nonlinear systems*, SIAM J. Sci. Comput., 19 (1998), pp. 302–318.
[34] J. Reisner, A. Wyszogrodzki, V. A. Mousseau, and D. A. Knoll, *An efficient physics-based preconditioner for the fully implicit solution of small-scale thermally driven atmospheric flows*, J. Comput. Phys., 189 (2003), pp. 30–44.
[35] A. Robert, *Bubble convection experiments with a semi-implicit formulation of the Euler equations*, J. Atmos. Sci., 50 (1993), pp. 1865–1873.
[36] M. Satoh, *Conservative scheme for the compressible nonhydrostatic models with the horizontally explicit and vertically implicit time integration scheme*, Mon. Wea. Rev., 130 (2002), pp. 1227–1245.
[37] C. Schär, D. Leuenberger, O. Fuhrer, D. Lüthic, and C. Girard, *A new terrain-following vertical coordinate formulation for atmospheric prediction models*, Mon. Wea. Rev., 130 (2002), pp. 2459–2480.
[38] W. C. Skamarock and J. B. Klemp, *Efficiency and accuracy of the Klemp-Wilhelmson time-splitting technique*, Mon. Wea. Rev., 122 (1994), pp. 2623–2630.
[39] R. B. Smith, *The influence of mountains on the atmosphere*, Adv. Geophys., 21 (1979), pp. 87–230.
[40] A. St-Cyr and D. Neckels, *A fully implicit Jacobian-free high-order discontinuous Galerkin mesoscale flow solver*, in Proceedings of the 9th Intl. Conf. on Computational Science (ICCS 2009), part II, Lecture Notes in Computer Science, vol. 5545, Berlin, Heidelberg, 2009, Springer-Verlag, pp. 243–252.
[41] J. M. Straka, R. B. Wilhelmson, L. J. Wicker, J. R. Anderson, and K. K. Droegemeier, *Numerical solutions of a non-linear density current: a benchmark solution and comparisons*, Int. J. Numer. Methods Fluids, 17 (1993), pp. 1–22.

[42] C. Temperton and A. Staniforth, *An efficient two-time-level semi-Lagrangian semi-implicit integration scheme*, Q. J. R. Meteorol. Soc., 113 (1987), pp. 1025–1039.

[43] P. A. Ullrich and C. Jablonowski, *Operator-split Runge-Kutta-Rosenbrock methods for nonhydrostatic atmospheric models*, Mon. Wea. Rev., 140 (2012), pp. 1257–1284.

[44] P. A. Ullrich, C. Jablonowski, and B. van Leer, *High-order finite-volume methods for the shallow-water equations on the sphere*, J. Comput. Phys., 229 (2010), pp. 6104–6134.

[45] C. Völcker, J. B. Jørgensen, P. G. Thomsen, and E. H. Stenby, *Adaptive stepsize control in implicit Runge-Kutta methods for reservoir simulation*, in Proceedings of the 9th Intl. Symp. on Dynamics and Control of Process Systems (DYCOPS 2010), 2010, pp. 509–514.

[46] N. N. Yanenko, *The Method of Fractional Steps: The Solution of Problems of Mathematical Physics in Several Variables*, Springer Verleg, 1971.

[47] C. Yang and X.-C. Cai, *Newton-Krylov-Schwarz method for a spherical shallow water model*, in Proceedings of the 19th Intl. Conf. on Domain Decomposition Methods, Lecture Notes in Computational Science and Engineering, Y. Huang, R. Kornhuber, O. Widlund, and J. Xu, eds., vol. 78, Berlin, Heidelberg, 2011, Springer-Verlag, pp. 149–155.

[48] C. Yang and X.-C. Cai, *Parallel multilevel methods for implicit solution of shallow water equations with nonsmooth topography on the cubed-sphere*, J. Comput. Phys., 230 (2011), pp. 2523–2539.

[49] C. Yang, J. Cao, and X.-C. Cai, *A fully implicit domain decomposition algorithm for shallow water equations on the cubed-sphere*, SIAM J. Sci. Comput., 32 (2010), pp. 418–438.