Journal of Computational Physics 520 (2025) 113511

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Local time-stepping for the shallow water equations using CFL optimized forward-backward Runge-Kutta schemes

Jeremy R. Lilly [a,b,*], Giacomo Capodaglio [b], Darren Engwirda [c], Robert L. Higdon [a], Mark R. Petersen [b]

[a] *Department of Mathematics, Oregon State University, Corvallis, OR, 97331, USA*
[b] *Computational Physics and Methods Group, Los Alamos National Laboratory, Los Alamos, NM, 87545, USA*
[c] *Fluid Dynamics and Solid Mechanics Group, Los Alamos National Laboratory, Los Alamos, NM, 87545, USA*

ARTICLE INFO

Keywords:
Shallow water
Time-stepping
CFL condition
Storm surge
MPAS-Ocean
TRiSK

ABSTRACT

The Courant–Friedrichs–Lewy (CFL) condition is a well known, necessary condition for the stability of explicit time-stepping schemes that effectively places a limit on the size of the largest admittable time-step for a given problem. We formulate and present a new local time-stepping (LTS) scheme optimized, in the CFL sense, for the shallow water equations (SWEs). This new scheme, called FB-LTS, is based on the CFL optimized forward-backward Runge-Kutta schemes from Lilly et al. [16]. We show that FB-LTS maintains exact conservation of mass and absolute vorticity when applied to the TRiSK spatial discretization [21], and provide numerical experiments showing that it retains the temporal order of the scheme on which it is based (second order). We implement FB-LTS, along with a certain operator splitting, in MPAS-Ocean to test computational performance. This scheme, SplitFB-LTS, is up to 10 times faster than the classical four-stage, fourth-order Runge-Kutta method (RK4), and 2.3 times faster than an existing strong stability preserving Runge-Kutta based LTS scheme with the same operator splitting (SplitLTS3). Despite this significant increase in efficiency, the solutions produced by SplitFB-LTS are qualitatively equivalent to those produced by both RK4 and SplitLTS3.

1. Introduction

The computational performance of explicit time-stepping schemes is often limited by the so-called Courant–Friedrichs–Lewy (CFL) condition. Given a system of partial differential equations (PDEs) with a finite speed of propagation and a spatial discretization, the CFL condition bounds the time-step above in terms of the size of the spatial cells and a quantity related to the speed of the dynamics of the problem. Formally, the CFL condition states that it is necessary for stability that

$$\nu = c\frac{\Delta t}{\Delta x} \leq \nu^{\max}, \tag{1}$$

where c is a characteristic speed, such as that of a gravity wave, ν is referred to as the Courant number, and $\nu^{\max}$ is the maximum admittable Courant number, which depends on the model problem itself and the chosen time and space discretizations. For many applications, particularly those that require high performance computing (HPC) resources, the size of the desired spatial discretization

* Corresponding author.
E-mail address: jlilly@lanl.gov (J.R. Lilly).

https://doi.org/10.1016/j.jcp.2024.113511
Received 16 May 2024; Received in revised form 14 October 2024; Accepted 15 October 2024

Available online 18 October 2024

Δx and the speed c can vary wildly across the computational domain (e.g. Francois et al. [6], Hoffman et al. [11], Larour et al. [14], Marsh et al. [17], Xu and Di Vittorio [23], Zhou et al. [24]). In cases like these, traditional explicit global time-stepping schemes are forced to satisfy a single, global CFL condition, perhaps enforced by only a small portion of the domain. Local time-stepping (LTS) methods provide an answer to this problem by allowing a scheme to take time-steps depending on local values of c and Δx, satisfying a local CFL condition rather than a global one. In practice, this means that a domain can be divided into *coarse* regions where large time-steps are used, and *fine* regions, where smaller time-steps are used. In contrast, a global scheme requires the small time-step native to a fine region to be used everywhere, resulting in the need for more evaluations of right-hand-side terms to advance forward in time, increasing the overall computational burden.

In this work, we introduce a new LTS scheme for the shallow water equations (SWEs) that has been optimized in the CFL sense. This scheme, called FB-LTS, is based on the three-stage, second-order, forward-backward Runge-Kutta scheme FB-RK(3,2) developed by Lilly et al. [16]. FB-RK(3,2) is an explicit time-stepping scheme designed for coupled systems of PDEs. In the context of the SWEs, the scheme uses a forward-backward (FB) average of available thickness data to advance the momentum equation at each Runge-Kutta stage. The weights of these FB averages have been optimized to produce a scheme that has maximal $\nu^{\max}$ when applied to the SWEs. It was shown in Lilly et al. [16] that FB-RK(3,2) outperforms a popular three-stage, third-order strong stability preserving Runge-Kutta scheme (SSPRK3) in admittable time step by factors roughly between 1.6 and 2.2, making the scheme approximately twice as computationally efficient with little to no effect on solution quality. The new FB-LTS scheme introduced here takes advantage of the CFL performance of FB-RK(3,2) and combines it with the benefits of a local time-stepping scheme.

The particular algorithm by which regions of the domain using different time steps communicate, which we refer to here as the LTS framework, was originally developed by Hoang et al. [10] for use with a TRiSK spatial discretization [21]. TRiSK is a finite volume-type spatial discretization made for unstructured, variable-resolution polygonal grids, and is the discretization used in the Model for Prediction Across Scales-Ocean (MPAS-O) [19–21]. The application of this LTS framework in the context of a TRiSK spatial discretization is a major topic of this work. The scheme presented in Hoang et al. [10] is based on SSPRK3, and is referred to here as LTS3. Using FB-RK(3,2) as opposed to SSPRK3, our FB-LTS scheme is able to outperform LTS3 in terms of the size of the admittable time-step by factors up to 2.3.

The long-term goal of FB-LTS is to increase the computational efficiency of climate-scale models of the ocean and atmosphere running on highly variable resolution meshes, with a particular focus on the Energy Exascale Earth System Model (E3SM) being developed by the U.S. Department of Energy [8]. In this work, we implement FB-LTS, along with a certain operator splitting, for single-layer configurations in MPAS-Ocean, the ocean component of E3SM. A single-layer ocean is modeled by the SWEs, and serves as the starting point for our eventual goal to use FB-LTS in multi-layer, climate-scale models.

This paper is structured as follows. We recall the formulation of FB-RK(3,2) presented in Lilly et al. [16] for completeness, then present the FB-LTS scheme. Next, we show that the scheme exactly conserves mass and absolute vorticity in the context of a TRiSK spatial discretization. Then, we discuss the details of the implementation of the scheme in MPAS-Ocean, including a discussion of an operator splitting approach that we have adopted within the SWEs. Finally, we perform a number of numerical experiments that demonstrate the computational efficiency of FB-LTS as compared to LTS3 in MPAS-Ocean. These experiments model the storm surge in Delaware Bay caused by hurricane Sandy in 2012, and are an evolution of the simulations explored in [15].

2. Local time-stepping schemes with FB-RK(3,2)

We begin by recalling the FB-RK(3,2) scheme introduced by Lilly et al. [16] for completeness, then introduce a new LTS scheme based on this global scheme, called FB-LTS. The primary goal of FB-LTS is to solve the shallow water equations (SWEs) efficiently in the CFL sense, i.e. taking time-steps as large as possible. To facilitate discussion of our methods and the SWEs, we introduce the nonlinear SWEs on a rotating sphere, given by

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \left(\nabla \times \mathbf{u} + f\mathbf{k}\right) \times \mathbf{u} &= -\nabla \frac{|\mathbf{u}|^2}{2} - g\nabla(h + z_b) \\ \frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{u}) &= 0\,, \end{aligned} \tag{2}$$

where $\mathbf{u}(x, y, t) = \big(u(x, y, t),\, v(x, y, t)\big)$ is the horizontal fluid velocity, x and y are the spatial coordinates, t is the time coordinate, f is the Coriolis parameter, $\mathbf{k}$ is the local vertical unit vector, g is the gravitational constant, h is the fluid thickness, and z_b is the height of the bottom topography. In Section 4.1, we will introduce a similar shallow water model of particular interest that will showcase the performance of FB-LTS in a real-world test case. Throughout this work, we often refer to a given equation for the evolution of $\mathbf{u}$ as the momentum equation, and a given equation for the evolution of h as the thickness or mass equation.

2.1. FB-RK(3,2)

The time-stepping scheme presented here is an extension of the three-stage, second-order Runge-Kutta time-stepping scheme RK(3,2) from Wicker and Skamarock [22] which is used to solve a SWE-like system in MPAS-Atmosphere. This extension of RK(3,2) allows the use of the most recently obtained data for the layer thickness to update the momentum data within each Runge-Kutta stage. This is done by taking a weighted average of layer thickness data at the old time level t^n and the most recent RK stage, then applying this to the momentum equation.

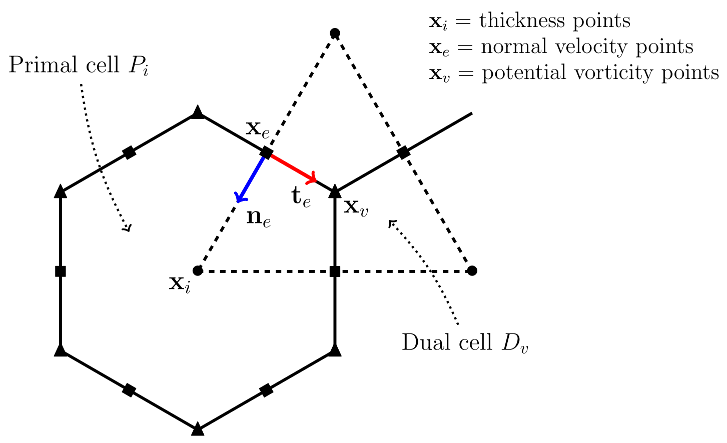

$\mathbf{x}_i$ = thickness points
$\mathbf{x}_e$ = normal velocity points
$\mathbf{x}_v$ = potential vorticity points
Primal cell P_i
$\mathbf{x}_e$
$\mathbf{t}_e$
$\mathbf{n}_e$
$\mathbf{x}_v$
$\mathbf{x}_i$
Dual cell D_v

Fig. 1. Example TRiSK grid from a Voronoi tessellation, where the primal cells are hexagons and the dual cells are triangles centered at primal cell vertices. This is the type of spatial discretzation used by MPAS-Ocean. The vector $\mathbf{n}_e$ is normal to cell edge e in a fixed, arbitrary direction. Later, in Section 2.4, we define a quantity $n_{e,i}$ as either 1 or -1 so that $n_{e,i}\mathbf{n}_e$ is the outward unit normal vector to cell i at edge e. Then, $\mathbf{t}_e = \mathbf{k} \times \mathbf{n}_e$.

Consider a general system of ODEs in independent variables $u = u(t)$ and $h = h(t)$ of the form

$$\begin{aligned} \frac{\mathrm{d}u}{\mathrm{d}t} &= \Phi(u,h) \\ \frac{\mathrm{d}h}{\mathrm{d}t} &= \Psi(u,h)\ , \end{aligned} \tag{3}$$

where t is the time coordinate. As discussed above, in the context of the SWEs u is the fluid velocity and h is the ocean layer thickness. The right-hand-side operators Φ and Ψ are referred to as the momentum and thickness (or mass) tendencies respectively. Let $u^n \approx u(t^n)$ and $h^n \approx h(t^n)$ be the numerical approximations to u and h at time $t = t^n$. Let Δt be a time-step such that $t^{n+1} = t^n + \Delta t$. Also, let $t^{n+1/m} = t^n + \frac{\Delta t}{m}$ for any positive integer m. Then, FB-RK(3,2) is given by

$$\begin{aligned} \bar{h}^{n+1/3} &= h^n + \frac{\Delta t}{3}\Psi\left(u^n, h^n\right) \\ \bar{u}^{n+1/3} &= u^n + \frac{\Delta t}{3}\Phi\left(u^n, h^*\right) \\ h^* &= \beta_1 \bar{h}^{n+1/3} + (1-\beta_1)h^n \end{aligned} \tag{4a}$$

$$\begin{aligned} \bar{h}^{n+1/2} &= h^n + \frac{\Delta t}{2}\Psi\left(\bar{u}^{n+1/3}, \bar{h}^{n+1/3}\right) \\ \bar{u}^{n+1/2} &= u^n + \frac{\Delta t}{2}\Phi\left(\bar{u}^{n+1/3}, h^{**}\right) \\ h^{**} &= \beta_2 \bar{h}^{n+1/2} + (1-\beta_2)h^n \end{aligned} \tag{4b}$$

$$\begin{aligned} h^{n+1} &= h^n + \Delta t\Psi\left(\bar{u}^{n+1/2}, \bar{h}^{n+1/2}\right) \\ u^{n+1} &= u^n + \Delta t\Phi\left(\bar{u}^{n+1/2}, h^{***}\right) \\ h^{***} &= \beta_3 h^{n+1} + (1-2\beta_3)\bar{h}^{n+1/2} + \beta_3 h^n\ . \end{aligned} \tag{4c}$$

The weights β_1, β_2, and β_3 are called the forward-backward (FB) weights. These FB-weights can be chosen so as to optimize the allowable time-step in the SWEs; it was shown in [16] that taking $(\beta_1, \beta_2, \beta_3) = (0.531, 0.531, 0.313)$ increases the admittable time-step versus RK(3,2) between factors of 1.6 and 2.2 in a number of nonlinear test cases while maintaining second-order accuracy.

2.2. FB-LTS

Here, we introduce FB-LTS in the context of a TRiSK spatial discretization [21], which is a finite volume-type spatial discretization made for unstructured, variable-resolution polygonal grids, and is the discretization used in MPAS-Ocean. TRiSK employs C-grid-type discretization [2] wherein the mass variable is computed on cell centers and the normal component of velocity is computed on cell edges. In MPAS-Ocean, these are Voronoi grids [13,18] consisting primarily of hexagons as the primal mesh, with a dual mesh consisting of triangles (Fig. 1).

Given some computational domain, let Ω_P be the set of indices for primal cells (hereafter referred to as just cells, dual cells will be referred to specifically as dual cells) and Ω_E be the set of indices for cell edges. We decompose the computational domain into two regions, a coarse region, which will be advanced with the coarse time-step Δt, and a fine region, which will be advanced with the fine time-step $\frac{\Delta t}{M}$ for some positive integer M. Note that the label of fine or coarse does not necessarily reference the size of the

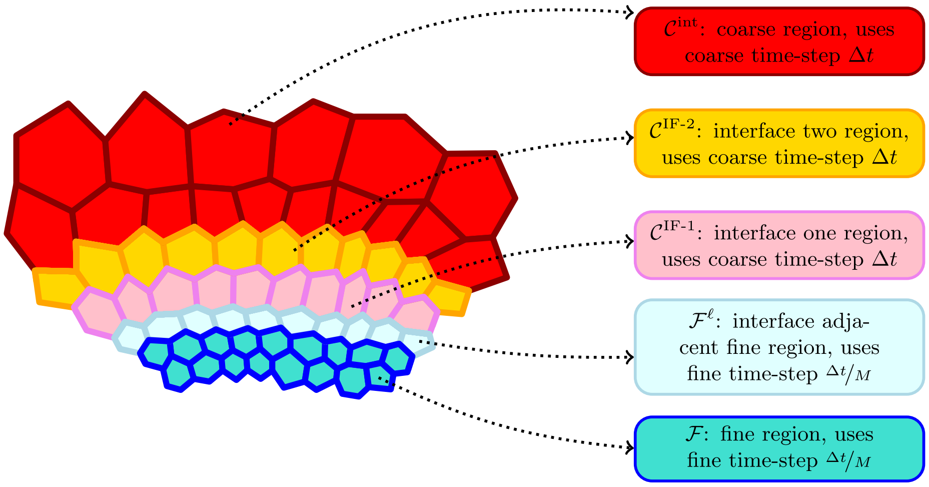

$\mathcal{C}^{\text{int}}$: coarse region, uses coarse time-step Δt
$\mathcal{C}^{\text{IF-2}}$: interface two region, uses coarse time-step Δt
$\mathcal{C}^{\text{IF-1}}$: interface one region, uses coarse time-step Δt
$\mathcal{F}^{\ell}$: interface adjacent fine region, uses fine time-step $\Delta t/M$
$\mathcal{F}$: fine region, uses fine time-step $\Delta t/M$

Fig. 2. An example mesh with cells and edges labeled for LTS. Blue cells and edges belong to $\mathcal{F}$, light blue cells and edges belong to $\mathcal{F}^{\ell} \subseteq \mathcal{F}$, pink cells and edges belong to $\mathcal{C}^{\text{IF-1}}$, yellow cells and edges belong to $\mathcal{C}^{\text{IF-2}}$, and red cells and edges belong to $\mathcal{C}^{\text{int}}$. Note that in practice, one often needs more layers of light blue, pink, and yellow cells; see Fig. 8 for a practical example. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

spatial discretization, but rather which time-step is used to advance it. The region made up of fine cells is called the fine region while the rest of the mesh is called the coarse region. While the sets Ω_P and Ω_E are formally sets of indices, throughout the text we often refer to them directly as sets of cells and edges respectively for readability.

Following the notation of Hoang et al. [10], let $\mathcal{F}_P$ be the set of cells in the fine region, and $\mathcal{C}_P$ be the set of cells in the coarse region such that $\mathcal{F}_P \cup \mathcal{C}_P = \Omega_P$. In the fine region, we define subsets $\mathcal{F}_P^{\ell} \subseteq \mathcal{F}_P$ to be the so-called, interface adjacent fine cells. Let r be the radius of the discrete tendency operators for $\mathbf{u}$ and $\mathbf{h}$, and define $\mathcal{F}_P^{\ell}$ for $\ell = 1, \cdots, 5$ so that $\mathcal{F}_P^{\ell}$ contains ℓr layers of cells in $\mathcal{F}_P$ neighboring $\mathcal{C}_P$. For example, in the implementation of FB-LTS described in Section 3 we have $r = 2$, so $\mathcal{F}_P^5$ contains 10 layers of cells bordering the interface one region (Fig. 8a). These $\mathcal{F}_P^{\ell}$ subsets are not disjoint with one another, rather $\mathcal{F}_P^1 \subseteq \cdots \subseteq \mathcal{F}_P^5$.

In the coarse region, define disjoint subsets $\mathcal{C}_P^{\text{IF-1}} \subseteq \mathcal{C}_P$, $\mathcal{C}_P^{\text{IF-2}} \subseteq \mathcal{C}_P$, and $\mathcal{C}_P^{\text{int}} \subseteq \mathcal{C}_P$ such that $\mathcal{C}_P^{\text{IF-1}} \cup \mathcal{C}_P^{\text{IF-2}} \cup \mathcal{C}_P^{\text{int}} = \mathcal{C}_P$. Call $\mathcal{C}_P^{\text{IF-1}}$ the set of interface one cells, $\mathcal{C}_P^{\text{IF-2}}$ the set of interface two cells, and $\mathcal{C}_P^{\text{int}}$ the set of interior coarse cells; all these cells advance with the coarse time-step. These collections of cells are distributed in the computational domain such that only $\mathcal{C}_P^{\text{IF-1}}$ cells border $\mathcal{F}_P$ cells, only $\mathcal{C}_P^{\text{IF-2}}$ cells border $\mathcal{C}_P^{\text{IF-1}}$ cells, and only $\mathcal{C}_P^{\text{int}}$ cells border $\mathcal{C}_P^{\text{IF-2}}$ cells.

The sets $\mathcal{F}_E$, $\mathcal{F}_E^{\ell}$, $\mathcal{C}_E^{\text{IF-1}}$, $\mathcal{C}_E^{\text{IF-2}}$, $\mathcal{C}_E^{\text{int}}$, and $\mathcal{C}_E$ give the corresponding sets of cell edges for all the sets of cells described above. An edge shared by a cell from $\mathcal{F}_P$ and a cell from $\mathcal{C}_P^{\text{IF-1}}$ belongs to $\mathcal{F}_E$, an edge shared by a cell from $\mathcal{C}_P^{\text{IF-1}}$ and a cell from $\mathcal{C}_P^{\text{IF-2}}$ belongs to $\mathcal{C}_E^{\text{IF-1}}$, and an edge shared by a cell from $\mathcal{C}_P^{\text{IF-2}}$ and a cell from $\mathcal{C}_P^{\text{int}}$ belongs to $\mathcal{C}_E^{\text{IF-2}}$. In plain language, an edge in dispute between two cells of different regions belongs to the region closest to the fine region. This domain decomposition is visualized in Fig. 2; we refer to these regions collectively as the LTS regions. Often, we use the notation $\mathcal{C}$ or $\mathcal{F}$ without a subscript P or E to refer to the whole of the corresponding region, including both cells and edges.

Finally, we assume that the LTS regions are configured in such a way that there are enough layers of $\mathcal{C}_P^{\text{IF-1}}$ and $\mathcal{C}_P^{\text{IF-2}}$ cells so that the operator stencils of the tendencies do not contain cells more than one region away. For example, the operator stencil on fine cells can only contain fine cells and interface one cells, and an operator stencil on interface one cells can only contains fine, interface one, and interface two cells, Fig. 8 shows a practical example of this, where the radius of the operator stencil is $r = 2$.

Now, consider the following system of PDE that has been discretized in space

$$\begin{aligned} \frac{\partial u_e}{\partial t} &= \Phi_e\left(\mathbf{u}, \mathbf{h}\right) \\ \frac{\partial h_i}{\partial t} &= \Psi_i\left(\mathbf{u}, \mathbf{h}\right), \end{aligned} \tag{5}$$

where $\mathbf{u} = (u_e)_{e \in \Omega_E}$ and $\mathbf{h} = (h_i)_{i \in \Omega_P}$. Under a TRiSK spatial discretization, h_i is computed at primal cell centers, and u_e is the normal component of velocity computed at primal cell edges (Fig. 1).

Set a time-step Δt, and let M be some positive integer. Let $t^{n+1/m} = t^n + \frac{\Delta t}{m}$ for any positive integer m, and $t^{n,k} = t^n + k\frac{\Delta t}{M}$, and $t^{n,k+1/m} = t^n + \left(k + 1/m\right)\frac{\Delta t}{M}$ (Fig. 3). The FB-LTS scheme proceeds as follows.

1. **Coarse Advancement:** Compute all three stages of FB-RK(3,2) on cells and edges from $\mathcal{C}_P$ and $\mathcal{C}_E$ to advance to time t^{n+1}. Note that during this step, we perform calculations on some of the interface adjacent fine cells; recall that if r is the radius of the discrete tendency operators for $\mathbf{u}$ and $\mathbf{h}$, we defined $\mathcal{F}_P^{\ell}$ for $\ell = 1, \cdots, 5$ so that $\mathcal{F}_P^{\ell}$ contains ℓr layers of cells in $\mathcal{F}_P$ neighboring $\mathcal{C}_P^{\text{IF-1}}$, and that $\mathcal{F}_E^{\ell}$ was the corresponding sets of edges for $\ell = 1, \cdots, 5$. The data computed in these sets is not used to advance the fine region, but is needed because these cells and edges are in the domain of dependence for the interface regions.

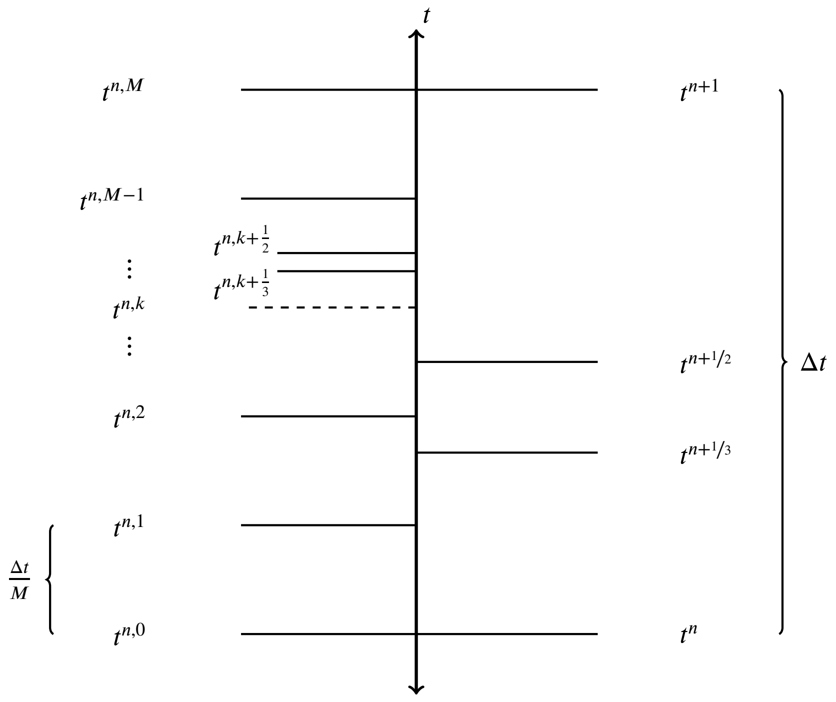


Fig. 3. Visualization and notation for the time-levels used by FB-LTS. In the coarse region (illustrated in the right half of the diagram), $t^{n+1/m} = t^n + \frac{\Delta t}{m}$ for any positive integer m; first stage data is calculated at time $t^{n+1/3}$ and second stage data is calculated at time $t^{n+1/2}$. In the fine region (illustrated in the left half of the diagram), $t^{n,k} = t^n + k\frac{\Delta t}{M}$ and $t^{n,k+1/m} = t^n + \left(k + 1/m\right)\frac{\Delta t}{M}$; first stage data is calculated at times $t^{n,k+1/3}$ and second stage data is calculated at times $t^{n,k+1/2}$ for $k = 0, \cdots, M-1$.

(a) **Thickness Stage 1:** For $i \in \mathcal{F}_P^5 \cup \mathcal{C}_P^{\text{IF-1}} \cup \mathcal{C}_P^{\text{IF-2}}$,

$$\begin{aligned} \tilde{h}_i^{n+1/3} &= h_i^n + \frac{\Delta t}{3}\Psi_i\left(\mathbf{u}^n, \mathbf{h}^n\right) \\ h_i^* &= \beta_1 \tilde{h}_i^{n+1/3} + (1-\beta_1) h_i^n \,. \end{aligned} \tag{6}$$

Note that $\tilde{h}_i$ denotes what we refer to as *uncorrected* thickness data. At the end of the scheme we will recalculate the necessary uncorrected data on the interface regions using information from the fine region advancement to obtain *corrected* data.
For $i \in \mathcal{C}_P^{\text{int}}$,

$$\begin{aligned} \bar{h}_i^{n+1/3} &= h_i^n + \frac{\Delta t}{3}\Psi_i\left(\mathbf{u}^n, \mathbf{h}^n\right) \\ h_i^* &= \beta_1 \bar{h}_i^{n+1/3} + (1-\beta_1) h_i^n \,. \end{aligned} \tag{7}$$

(b) **Velocity Stage 1:** For $e \in \mathcal{F}_E^4 \cup \mathcal{C}_E^{\text{IF-1}} \cup \mathcal{C}_E^{\text{IF-2}}$,

$$\tilde{u}_e^{n+1/3} = u_e^n + \frac{\Delta t}{3}\Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right) \,. \tag{8}$$

Note that $\tilde{u}_e$ denotes what we refer to as *uncorrected* velocity data. At the end of the scheme we will recalculate the necessary uncorrected data on the interface regions using information from the fine region advancement to obtain *corrected* data.
For $e \in \mathcal{C}_E^{\text{int}}$,

$$\bar{u}_e^{n+1/3} = u_e^n + \frac{\Delta t}{3}\Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right) \,. \tag{9}$$

(c) **Thickness Stage 2:** For $i \in \mathcal{F}_P^3 \cup \mathcal{C}_P^{\text{IF-1}} \cup \mathcal{C}_P^{\text{IF-2}}$,

$$\begin{aligned} \tilde{h}_i^{n+1/2} &= h_i^n + \frac{\Delta t}{2}\Psi_i\left(\tilde{\mathbf{u}}^{n+1/3}, \tilde{\mathbf{h}}^{n+1/3}\right) \\ h_i^{**} &= \beta_2 \tilde{h}_i^{n+1/2} + (1-\beta_2) h_i^n \,, \end{aligned} \tag{10}$$

where $\tilde{h}_i^{n+1/3} := \bar{h}_i^{n+1/3}$ and $\tilde{u}_e^{n+1/3} := \bar{u}_e^{n+1/3}$ for $i \in \mathcal{C}_P^{\text{int}}$ and $e \in \mathcal{C}_E^{\text{int}}$.

For $i \in \mathcal{C}_P^{\text{int}}$,

$$\begin{aligned} \bar{h}_i^{n+1/2} &= h_i^n + \frac{\Delta t}{2}\Psi_i\left(\bar{\mathbf{u}}^{n+1/3}, \bar{\mathbf{h}}^{n+1/3}\right) \\ h_i^{**} &= \beta_2 \bar{h}_i^{n+1/2} + (1-\beta_2)h_i^n\,, \end{aligned} \tag{11}$$

where $\bar{h}_i^{n+1/3} := \tilde{h}_i^{n+1/3}$ and $\bar{u}_e^{n+1/3} := \tilde{u}_e^{n+1/3}$ for $i \in \mathcal{C}_P^{\text{IF-2}}$ and $e \in \mathcal{C}_E^{\text{IF-2}}$.

(d) **Velocity Stage 2:** For $e \in \mathcal{F}_E^2 \cup \mathcal{C}_E^{\text{IF-1}} \cup \mathcal{C}_E^{\text{IF-2}}$,

$$\tilde{u}_e^{n+1/2} = u_e^n + \frac{\Delta t}{2}\Phi_e\left(\tilde{\mathbf{u}}^{n+1/3}, \mathbf{h}^{**}\right)\,, \tag{12}$$

where $\tilde{u}_e^{n+1/3} := \bar{u}_e^{n+1/3}$ for $e \in \mathcal{C}_E^{\text{int}}$.
For $e \in \mathcal{C}_E^{\text{int}}$,

$$\bar{u}_e^{n+1/2} = u_e^n + \frac{\Delta t}{2}\Phi_e\left(\bar{\mathbf{u}}^{n+1/3}, \mathbf{h}^{**}\right)\,, \tag{13}$$

where $\bar{u}_e^{n+1/3} := \tilde{u}_e^{n+1/3}$ for $e \in \mathcal{C}_E^{\text{IF-2}}$.

(e) **Thickness Stage 3:** For $i \in \mathcal{F}_P^1 \cup \mathcal{C}_P^{\text{IF-1}} \cup \mathcal{C}_P^{\text{IF-2}}$,

$$\begin{aligned} \tilde{h}_i^{n+1} &= h_i^n + \Delta t \Psi_i\left(\tilde{\mathbf{u}}^{n+1/2}, \tilde{\mathbf{h}}^{n+1/2}\right) \\ h_i^{***} &= \beta_3 \tilde{h}_i^{n+1} + (1-2\beta_3)\tilde{h}_i^{n+1/2} + \beta_3 h_i^n\,, \end{aligned} \tag{14}$$

where $\tilde{h}_i^{n+1/2} := \bar{h}_i^{n+1/2}$ and $\tilde{u}_e^{n+1/2} := \bar{u}_e^{n+1/2}$ for $i \in \mathcal{C}_P^{\text{int}}$ and $e \in \mathcal{C}_E^{\text{int}}$.
For $i \in \mathcal{C}_P^{\text{int}}$,

$$\begin{aligned} h_i^{n+1} &= h_i^n + \Delta t \Psi_i\left(\bar{\mathbf{u}}^{n+1/2}, \bar{\mathbf{h}}^{n+1/2}\right) \\ h_i^{***} &= \beta_3 h_i^{n+1} + (1-2\beta_3)\bar{h}_i^{n+1/2} + \beta_3 h_i^n\,, \end{aligned} \tag{15}$$

where $\bar{h}_i^{n+1/2} := \tilde{h}_i^{n+1/2}$ and $\bar{u}_e^{n+1/2} := \tilde{u}_e^{n+1/2}$ for $i \in \mathcal{C}_P^{\text{IF-2}}$ and $e \in \mathcal{C}_E^{\text{IF-2}}$.

(f) **Velocity Stage 3:** For $e \in \mathcal{C}_E^{\text{IF-1}}$,

$$\tilde{u}_e^{n+1} = u_e^n + \Delta t \Phi_e\left(\tilde{\mathbf{u}}^{n+1/2}, \mathbf{h}^{***}\right)\,, \tag{16}$$

where $\tilde{u}_e^{n+1/2} := \bar{u}_e^{n+1/2}$ for $e \in \mathcal{C}_E^{\text{int}}$.
For $e \in \mathcal{C}_E^{\text{int}}$,

$$u_e^{n+1} = u_e^n + \Delta t \Phi_e\left(\bar{\mathbf{u}}^{n+1/2}, \mathbf{h}^{***}\right)\,, \tag{17}$$

where $\bar{u}_e^{n+1/2} := \tilde{u}_e^{n+1/2}$ for $e \in \mathcal{C}_E^{\text{IF-2}}$.

2. **Interface Prediction:** Use the uncorrected data on interface one to obtain predicted values for FB-RK(3,2) data on $\mathcal{C}_P^{\text{IF-1}}$ and $\mathcal{C}_E^{\text{IF-1}}$ at times $t^{n,k}$, $t^{n,k+1/3}$, and $t^{n,k+1/2}$, for $k = 0, \cdots, M-1$.

$$\begin{bmatrix} \mathbf{h}^{n,k} \\ \mathbf{u}^{n,k} \end{bmatrix} = \frac{k}{M}\begin{bmatrix} \tilde{\mathbf{h}}^{n+1} \\ \tilde{\mathbf{u}}^{n+1} \end{bmatrix} + \left(1 - \frac{k}{M}\right)\begin{bmatrix} \mathbf{h}^{n} \\ \mathbf{u}^{n} \end{bmatrix} \tag{18a}$$

$$\begin{bmatrix} \bar{\mathbf{h}}^{n,k+1/3} \\ \bar{\mathbf{u}}^{n,k+1/3} \end{bmatrix} = \frac{k}{M}\begin{bmatrix} \tilde{\mathbf{h}}^{n+1} \\ \tilde{\mathbf{u}}^{n+1} \end{bmatrix} + \frac{1}{M}\begin{bmatrix} \tilde{\mathbf{h}}^{n+1/3} \\ \tilde{\mathbf{u}}^{n+1/3} \end{bmatrix} + \left(1 - \frac{k+1}{M}\right)\begin{bmatrix} \mathbf{h}^{n} \\ \mathbf{u}^{n} \end{bmatrix} \tag{18b}$$

$$\begin{bmatrix} \bar{\mathbf{h}}^{n,k+1/2} \\ \bar{\mathbf{u}}^{n,k+1/2} \end{bmatrix} = \frac{k}{M}\begin{bmatrix} \tilde{\mathbf{h}}^{n+1} \\ \tilde{\mathbf{u}}^{n+1} \end{bmatrix} + \frac{1}{M}\begin{bmatrix} \tilde{\mathbf{h}}^{n+1/2} \\ \tilde{\mathbf{u}}^{n+1/2} \end{bmatrix} + \left(1 - \frac{k+1}{M}\right)\begin{bmatrix} \mathbf{h}^{n} \\ \mathbf{u}^{n} \end{bmatrix}\,. \tag{18c}$$

Additionally, compute the prediction for $\mathbf{h}^{n,k}$ (18a) one additional time for $k = M$; this data will be needed to calculate the third stage of FB-RK(3,2) in the fine region with $k = M-1$. The coefficients for interpolating the uncorrected data on interface one are called the prediction coefficients, and the chosen prediction coefficients given above result in a second order approximation to $\mathbf{u}$ and $\mathbf{h}$ data at the corresponding times. These coefficients are derived in Appendix A. Additionally, one can observe that when $M = 1$, (18) reduces in such a way that the predictions are exactly the corresponding data already computed. This means that in the case where $M = 1$, FB-LTS is mathematically equivalent to FB-RK(3,2).

Note that we can also use this data to compute values for $\mathbf{h}^{*,k}$, $\mathbf{h}^{**,k}$, and $\mathbf{h}^{***,k}$ on interface one cells. For $i \in \mathcal{C}_P^{\text{IF-1}}$,

$$\begin{aligned} h_i^{*,k} &= \beta_1 \bar{h}_i^{n,k+1/3} + (1-\beta_1) h_i^{n,k} \\ h_i^{**,k} &= \beta_2 \bar{h}_i^{n,k+1/2} + (1-\beta_2) h_i^{n,k} \\ h_i^{***,k} &= \beta_3 h_i^{n,k+1} + (1-2\beta_3)\bar{h}_i^{n,k+1/2} + \beta_3 h_i^{n,k} . \end{aligned} \tag{19}$$

3. **Fine Advancement:** For $i \in \mathcal{F}_P$ and $e \in \mathcal{F}_E$, advance with FB-RK(3,2) with the fine time-step M times. For $k = 0, \cdots, M-1$,

$$\begin{aligned} \bar{h}_i^{n,k+1/3} &= h_i^{n,k} + \frac{\Delta t}{3M}\Psi_i\left(\mathbf{u}^{n,k}, \mathbf{h}^{n,k}\right) \\ \bar{u}_e^{n,k+1/3} &= u_e^{n,k} + \frac{\Delta t}{3M}\Phi_e\left(\mathbf{u}^{n,k}, \mathbf{h}^{*,k}\right) \\ h_i^{*,k} &= \beta_1 \bar{h}_i^{n,k+1/3} + (1-\beta_1) h_i^{n,k} \end{aligned} \tag{20a}$$

$$\begin{aligned} \bar{h}_i^{n,k+1/2} &= h_i^{n,k} + \frac{\Delta t}{2M}\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/3}, \bar{\mathbf{h}}^{n,k+1/3}\right) \\ \bar{u}_e^{n,k+1/2} &= u_e^{n,k} + \frac{\Delta t}{2M}\Phi_e\left(\bar{\mathbf{u}}^{n,k+1/3}, \mathbf{h}^{**,k}\right) \\ h_i^{**,k} &= \beta_2 \bar{h}_i^{n,k+1/2} + (1-\beta_2) h_i^{n,k} \end{aligned} \tag{20b}$$

$$\begin{aligned} h_i^{n,k+1} &= h_i^{n,k} + \frac{\Delta t}{M}\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/2}, \bar{\mathbf{h}}^{n,k+1/2}\right) \\ u_e^{n,k+1} &= u_e^{n,k} + \frac{\Delta t}{M}\Phi_e\left(\bar{\mathbf{u}}^{n,k+1/2}, \mathbf{h}^{***,k}\right) \\ h_i^{***,k} &= \beta_3 h_i^{n,k+1} + (1-2\beta_3)\bar{h}_i^{n,k+1/2} + \beta_3 h_i^{n,k} . \end{aligned} \tag{20c}$$

Note that in the FB average in (20c), we already have the data for $h_i^{n,k+1}$ on $\mathcal{C}^{\text{IF-1}}$ cells for $k = M-1$ because of the extra computation of (18a) for $k = M$.
After looping over k, set $h_i^{n+1} := h_i^{n,M}$ and $u_e^{n+1} := u_e^{n,M}$ for $i \in \mathcal{F}_P$ and $e \in \mathcal{F}_E$.

4. **Interface Correction:** Finally, compute the corrected data at time t^{n+1} on interface one and two. For $i \in \mathcal{C}_P^{\text{IF-1}} \cup \mathcal{C}_P^{\text{IF-2}}$ and $e \in \mathcal{C}_E^{\text{IF-1}} \cup \mathcal{C}_E^{\text{IF-2}}$,

$$\begin{aligned} h_i^{n+1} &= h_i^n + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/2}, \bar{\mathbf{h}}^{n,k+1/2}\right) \\ u_e^{n+1} &= u_e^n + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\Phi_e\left(\bar{\mathbf{u}}^{n,k+1/2}, \mathbf{h}^{***,k}\right) , \end{aligned} \tag{21}$$

where $u_e^{n,k} := u_e^n$, $h_i^{n,k} := h_i^n$, $h_i^{*,k} := h_i^*$, $\bar{u}_e^{n,k+1/3} := \tilde{u}_e^{n+1/3}$, $\bar{h}_i^{n,k+1/3} := \tilde{h}_i^{n+1/3}$, $h_i^{**,k} := h_i^{**}$, $\bar{u}_e^{n,k+1/2} := \tilde{u}_e^{n+1/2}$, $\bar{h}_i^{n,k+1/2} := \tilde{h}_i^{n+1/2}$, and $h_i^{***,k} := h_i^{***}$ for $i \in \mathcal{C}_P^{\text{IF-2}}$ and $e \in \mathcal{C}_E^{\text{IF-2}}$. Similarly, $u_e^{n,k} := u_e^n$, $h_i^{n,k} := h_i^n$, $h_i^{*,k} := h_i^*$, $\bar{u}_e^{n,k+1/3} := \bar{u}_e^{n+1/3}$, $\bar{h}_i^{n,k+1/3} := \bar{h}_i^{n+1/3}$, $h_i^{**,k} := h_i^{**}$, $\bar{u}_e^{n,k+1/2} := \bar{u}_e^{n+1/2}$, $\bar{h}_i^{n,k+1/2} := \bar{h}_i^{n+1/2}$, and $h_i^{***,k} := h_i^{***}$ for $i \in \mathcal{C}_P^{\text{int}}$ and $e \in \mathcal{C}_E^{\text{int}}$.
Note that the individual terms in the sums in (21) can be calculated and accumulated during the fine advancement step.

This ends the description of the FB-LTS scheme.

2.3. Temporal convergence

Here we describe the results of a numerical experiment in which FB-LTS is $\mathcal{O}\left((\Delta t)^2\right)$ everywhere, including on interface cells and edges (Fig. 4). We calculate the error of a FB-LTS solution against that of the classical, four-stage, fourth order Runge-Kutta method (RK4) using a small time-step on a simple model problem.

Consider a non-rotating aquaplanet (i.e. a spherical mesh with no land cells) with constant resting thickness, where the fluid velocity is initialized to zero, and the layer thickness is initialized to a Gaussian bump. This produces a simple external gravity wave, which is modeled by the system

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} = -g\nabla h \\ \frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{u}) = 0 . \end{cases} \tag{22}$$

In these equations $\mathbf{u}$ is the fluid velocity, t is the time coordinate, g is the gravitational constant, and h is the ocean thickness.

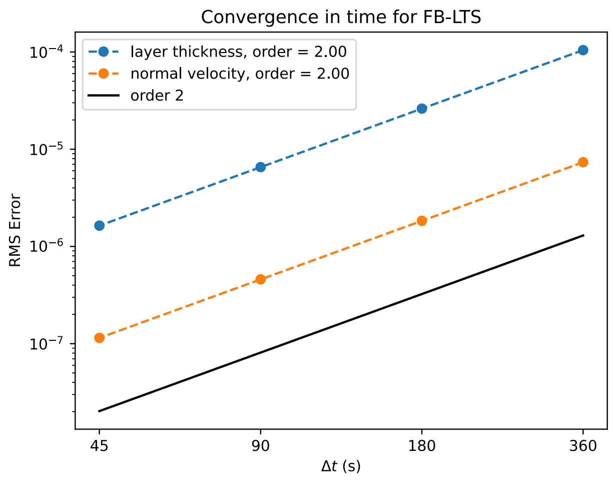


Fig. 4. Temporal convergence for FB-LTS on a test case consisting of an external gravity wave on a non-rotating aquaplanet. The time-steps on the horizontal axis are the time-steps used in the coarse region, in each case we have $M = 4$. Errors are computed against a reference solution generated by RK4 using a time-step of 10 s.

The root-mean-square (RMS) error is defined as

$$E_{\text{RMS}} = \sqrt{\frac{\sum_{i=1}^{N}(s_i - m_i)^2}{N}}\,, \tag{23}$$

where $\{s_i\}_{i=1}^{N}$ is the discrete reference solution, $\{m_i\}_{i=1}^{N}$ is the discrete model solution, and N is the number of discretization points.

We run this test case with FB-LTS using successively decreasing time-steps. By calculating errors against a reference solution generated by RK4 using a time-step of 10 s, we obtain the expected $\mathcal{O}\left((\Delta t)^2\right)$ temporal convergence rate for both layer thickness and normal velocity (Fig. 4).

2.4. Conservation of mass and absolute vorticity

An important property of FB-LTS is that it provides exact conservation of the spatially discrete representation of mass (Theorem 1) and the spatially discrete representation of absolute vorticity (Theorem 2) in the cases where there are no boundary conditions (e.g. the case of an aquaplanet), or no-flow boundary conditions (i.e. the normal velocity is zero at the boundary). In the context of the shallow water equations, the mass variable is the ocean layer thickness h, and absolute vorticity is given by $\eta = \mathbf{k}\cdot\nabla\times\mathbf{u} + f$, depending on the fluid velocity $\mathbf{u}$ and the Coriolis parameter f. To clarify what is meant by conservation here, we mean that the values of globally integrated thickness and globally integrated absolute vorticity are constant in time. We are concerned with the discrete counterparts to the conservation equations for these quantities in the continuous case. The mass equation is given by

$$\frac{\partial h}{\partial t} + \nabla\cdot(h\mathbf{u}) = 0\,, \tag{24}$$

and the absolute vorticity equation (obtained by taking the curl of the momentum equation) is given by

$$\frac{\partial \eta}{\partial t} + \nabla\cdot(\eta\mathbf{u}) = 0\,. \tag{25}$$

Often, (25) is written in terms of potential vorticity (PV), where PV is given by $q = \frac{\eta}{h}$. The absolute vorticity equation can also be written as a thickness-weighted potential vorticity equation,

$$\frac{\partial (qh)}{\partial t} + \nabla\cdot\left(q(h\mathbf{u})\right) = 0\,. \tag{26}$$

In particular, Ringler et al. [21] formulates the discrete counterpart to (25) in terms of a spatially discrete representation of PV.

Theorem 1. *FB-LTS exactly preserves the discrete representation of mass assuming either no boundary conditions or no-flow boundary conditions.*

Proof. The continuous equation giving conservation of mass for the SWEs is (24), which states that the evolution of the thickness h depends only on the divergence of the thickness flux. Within a TRiSK framework, this equation is spatially discretized as

$$\begin{aligned}\frac{\partial h_i}{\partial t} &= \Psi_i\left(\mathbf{u},\mathbf{h}\right)\\ &= \frac{-1}{A_i}\sum_{e\in\mathcal{E}_i} n_{e,i}\ell_e F_e\left(u_e,\mathbf{h}\right)\,,\end{aligned} \tag{27}$$

where $\mathbf{u} = (u_e)_{e\in\Omega_E}$ and $\mathbf{h} = (h_i)_{i\in\Omega_P}$, A_i is the area of cell i, $\mathcal{E}_i$ is the set of edges of cell i, ℓ_e is the length of edge e, $n_{e,i}$ is either 1 or -1, chosen so that $n_{e,i}\mathbf{n}_e$ (see Fig. 1) is the unit outward normal vector to cell i at edge e, and $F_e = F_e\left(u_e, \mathbf{h}\right)$ is the signed value of the thickness flux at edge e in the direction of the unit vector $\mathbf{n}_e$. Note that because of the choice of $n_{e,i}$ such that $n_{e,i}\mathbf{n}_e$ is the outward normal to cell i at edge e, the quantity $n_{e,i}F_e$ is the thickness flux leaving cell i. To show that the total mass in the system at time t^n is equal to the total mass in the system at time t^{n+1}, we will show that any given edge is neither a source nor a sink for mass as it is transported across cells. Specifically, we will show this for an edge that is shared by a fine cell and an interface one cell as this is the most delicate case; the cases of other types of edges can be shown similarly, though more simply, so we omit them for brevity.

Let $i_1 \in \mathcal{F}_P$ and $i_2 \in \mathcal{C}_P^{\text{IF-1}}$ be indices for cells P_{i_1} and P_{i_2} that have a shared edge $e_0 \in \mathcal{F}_E$. Assume that the thickness-fluxes at all other edges of P_{i_1} and P_{i_2} are zero. We do this with the goal of showing that the mass flux as computed on one side of the edge e_0 is the same as computed on the other side of the edge e_0. From this, we can conclude that the edge in question neither adds or subtracts mass from the system, giving a local conservation of mass, and from there, global conservation follows. Since we are assuming that $F_e\left(u_e, \mathbf{h}\right) = 0$ for all $e \neq e_0$, (27) simplifies to

$$\frac{\partial h_i}{\partial t} = \frac{-1}{A_i} n_{e_0,i}\ell_{e_0}F_{e_0}\left(u_{e_0}, \left(h_{i_1}, h_{i_2}\right)\right), \tag{28}$$

for $i = i_1, i_2$. The goal is to show that

$$\int\limits_{P_{i_1}\cup P_{i_2}} h_{i_*}^{n+1}\,\mathrm{d}A = \int\limits_{P_{i_1}\cup P_{i_2}} h_{i_*}^{n}\,\mathrm{d}A, \tag{29}$$

where h_{i_*} refers to the thicknesses h_{i_1} and h_{i_2} in the respective cells. From here, the central idea is to look at the discretization of the thickness on cell i_1 and i_2 at time t^{n+1}. For cell i_1, which is in the fine region, the thickness at time t^{n+1} is given by (20c) with $k = M-1$ in the *Fine Advancement* step of FB-LTS. For cell i_2, which is in the interface one region, the thickness at time t^{n+1} is given by (21) in the *Interface Correction* step. After writing out both $h_{i_1}^{n+1}$ and $h_{i_2}^{n+1}$, we will note that the total flux over their shared edge is equal, as computed from both cells. Then, we can integrate both thicknesses over their respective cells to show that the total mass in both cells at time t^{n+1} is the same as the total mass at time t^n. On P_{i_1}, we have that

$$\begin{aligned}
h_{i_1}^{n+1} &= h_{i_1}^{n,M}\\
&= h_{i_1}^{n,M-1} + \frac{\Delta t}{M}\Psi_{i_1}\left(\bar{\mathbf{u}}^{n,(M-1)+1/2}, \bar{\mathbf{h}}^{n,(M-1)+1/2}\right)\\
&= h_{i_1}^{n,M-2} + \frac{\Delta t}{M}\left[\Psi_{i_1}\left(\bar{\mathbf{u}}^{n,(M-2)+1/2}, \bar{\mathbf{h}}^{n,(M-2)+1/2}\right) + \Psi_{i_1}\left(\bar{\mathbf{u}}^{n,(M-1)+1/2}, \bar{\mathbf{h}}^{n,(M-1)+1/2}\right)\right]\\
&\vdots\\
&= h_{i_1}^{n,0} + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\Psi_{i_1}\left(\bar{\mathbf{u}}^{n,k+1/2}, \bar{\mathbf{h}}^{n,k+1/2}\right)\\
&= h_{i_1}^{n} + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\left[\frac{-1}{A_{i_1}} n_{e_0,i_1}\ell_{e_0}F_{e_0}\left(\bar{u}_{e_0}^{n,k+1/2}, \left(\bar{h}_{i_1}^{n,k+1/2}, \bar{h}_{i_2}^{n,k+1/2}\right)\right)\right].
\end{aligned} \tag{30}$$

Note that the values of the thickness fluxes $F_{e_0}\left(\bar{u}_{e_0}^{n,k+1/2}, \left(\bar{h}_{i_1}^{n,k+1/2}, \bar{h}_{i_2}^{n,k+1/2}\right)\right)$ depend only on the thickness on cell i_1 in the fine region, the thickness on cell i_2 in the interface one region, and the normal velocity at edge e_0, all computed as second stage data times $t^{n,k+1/2}$ for $k = 0, \cdots, M-1$. To simplify notation, from here on we write $F_{e_0}^{n,k+1/2} := F_{e_0}\left(\bar{u}_{e_0}^{n,k+1/2}, \left(\bar{h}_{i_1}^{n,k+1/2}, \bar{h}_{i_2}^{n,k+1/2}\right)\right)$. The central idea for the proof is that both the fine cell and the interface one cell are using the same flux for each $k = 0, \cdots, M-1$. Now, looking at P_{i_2}, after completing the correction step (21), we have

$$\begin{aligned}
h_{i_2}^{n+1} &= h_{i_2}^{n} + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\Psi_{i_2}\left(\bar{\mathbf{u}}^{n,k+1/2}, \bar{\mathbf{h}}^{n,k+1/2}\right)\\
&= h_{i_2}^{n} + \frac{\Delta t}{M}\sum_{k=0}^{M-1}\left[\frac{-1}{A_{i_2}} n_{e_0,i_2}\ell_{e_0}F_{e_0}^{n,k+1/2}\right].
\end{aligned} \tag{31}$$

Next, we integrate (30) over the cell P_{i_1} to get the total mass in the cell at time t^{n+1},

$$\int\limits_{P_{i_1}} h_{i_1}^{n+1}\,\mathrm{d}A = A_{i_1}h_{i_1}^{n} + A_{i_1}\left(\frac{\Delta t}{M}\sum_{k=0}^{M-1}\left[\frac{-1}{A_{i_1}} n_{e_0,i_1}\ell_{e_0}F_{e_0}^{n,k+1/2}\right]\right)$$

$$= A_{i_1} h_{i_1}^n - \frac{\Delta t}{M} n_{e_0,i_1} \ell_{e_0} \sum_{k=0}^{M-1} \left[F_{e_0}^{n,k+1/2} \right]$$

$$= \int_{P_{i_1}} h_{i_1}^n \, \mathrm{d}A - \frac{\Delta t}{M} n_{e_0,i_1} \ell_{e_0} \sum_{k=0}^{M-1} \left[F_{e_0}^{n,k+1/2} \right] . \tag{32}$$

Through a similar calculation on P_{i_2} starting with (31), we get

$$\int_{P_{i_2}} h_{i_2}^{n+1} \, \mathrm{d}A = \int_{P_{i_2}} h_{i_2}^n \, \mathrm{d}A - \frac{\Delta t}{M} n_{e_0,i_2} \ell_{e_0} \sum_{k=0}^{M-1} \left[F_{e_0}^{n,k+1/2} \right] . \tag{33}$$

Finally, add (32) and (33),

$$\int_{P_{i_1} \cup P_{i_2}} h_{i_*}^{n+1} \, \mathrm{d}A = \int_{P_{i_1} \cup P_{i_2}} h_{i_*}^n \, \mathrm{d}A - \frac{\Delta t}{M} \left(n_{e_0,i_1} + n_{e_0,i_2} \right) \ell_{e_0} \sum_{k=0}^{M-1} \left[F_{e_0}^{n,k+1/2} \right]$$

$$= \int_{P_{i_1} \cup P_{i_2}} h_{i_*}^n \, \mathrm{d}A , \tag{34}$$

with the final step being achieved using the fact that $n_{e_0,i_1} = -n_{e_0,i_2}$. This shows that FB-LTS exactly conserves mass under the TRiSK spatial discretization. □

Theorem 2. *FB-LTS exactly preserves the discrete representation of absolute vorticity assuming either no boundary conditions or no-flow boundary conditions.*

Proof. The thickness-weighted PV equation (26) shows that the evolution of the thickness-weighted PV (i.e., absolute vorticity) depends only on the divergence of the absolute vorticity flux. Under TRiSK, this is discretized in space as

$$\frac{\partial \eta_v}{\partial t} = \Theta_v \left(\mathbf{u}, \mathbf{h} \right)$$

$$= \frac{-1}{A_v} \sum_{e \in \mathcal{E}_v} -t_{e,v} d_e \hat{q}_e F_e^{\perp} , \tag{35}$$

where A_v is the area of dual cell v, $\mathcal{E}_v$ is the set of edges of dual cell v, $t_{v,i}$ is an indicator function, either -1 or 1 depending on whether the fixed unit vector $\mathbf{t}_e = \mathbf{k} \times \mathbf{n}_e$ (Fig. 1) is outward or inward to dual cell v (note that the signs are opposite relative to $n_{i,e}$ described above; this convention is used in Ringler et al. [21] and we keep it here for consistency), d_e is the length of edge e, $\hat{q}_e$ is the value of the PV interpolated to edge e, and $F_e^{\perp} = \mathcal{M}_e \left(\mathbf{F}(\mathbf{u}, \mathbf{h}) \right)$. Here, $\mathcal{M}_e$ is the flux mapping operator defined in Ringler et al. [21] that defines a mapping between the primal flux field $\mathbf{F} = \left(F_e \right)_{e \in \Omega_E}$ in the direction normal to primal cell edges and the dual flux field $\mathbf{F}^{\perp} = \left(F_e^{\perp} \right)_{e \in \Omega_E}$ in the direction tangent to primal cell edges, and therefore normal to dual cell edges. This mapping $\mathcal{M}$ from Ringler et al. [21] allows for the vorticity equation (25) to be defined on the dual mesh and be consistent with the underlying momentum and thickness equations defined on the primal cells; because of this mapping, Ringler et al. [21] asserts that a prognostically obtained (i.e., computed from equations derived from PDE) vorticity is equal to a diagnostically obtained (i.e., computed from prognostically obtained quantities) vorticity, up to machine precision. Finally, recall that F_e and $F_e^{\perp}$ are thickness fluxes of the form $h\mathbf{u}$, so a quantity of the form $qF_e = (qh)\mathbf{u} = \eta\mathbf{u}$ can be understood as a flux of the absolute vorticity.

Therefore, even though in practice we obtain values for the absolute vorticity diagnostically, never computing (35) directly, our goal is to show that when (35) is solved prognostically, absolute vorticity is exactly conserved in the sense that the total absolute vorticity at time t^n is equal to the total absolute vorticity at time t^{n+1}. This can be done very similarly to the conservation of mass argument above, except now we consider dual cells, looking at fluxes normal to dual cell edges (Fig. 5); effectively, because this is a conservation equation where the temporal evolution of η depends only on a divergence of a flux, it is equivalent to the thickness case shown above. As before, we will show this for an edge that is shared by a fine dual cell and an interface one dual cell, though calculations for other types of edges are similar.

Let $\mathcal{F}_D$ and $\mathcal{C}_D^{\text{IF-1}}$ be the sets of indices for dual cells, then take $v_1 \in \mathcal{F}_D$ and $v_2 \in \mathcal{C}_D^{\text{IF-1}}$ so that dual cells D_{v_1} and D_{v_2} share a dual cell edge that is orthogonal to a primal cells edge $e_0 \in \mathcal{C}_E^{\text{IF-1}}$. Assume that the thickness-fluxes at all other edges of D_{i_1} and D_{v_2} are zero. Since we are assuming that $F_e^{\perp} = 0$ for all $e \neq e_0$, (35) simplifies to

$$\frac{\partial \eta_v}{\partial t} = \frac{1}{A_v} t_{e_0,v} d_{e_0} \hat{q}_{e_0} F_{e_0}^{\perp} . \tag{36}$$

See Fig. 5 for an illustration of this situation and note how (36) is similar to (28). From here, the result follows from a calculation very similar to the above for the conservation of mass (we omit the details because the calculation is so similar to the above); the value of $\eta_{v_1}^{n+1}$ is given by summing the fluxes in and out of D_{v_1} over M fine time-steps, and the value of $\eta_{v_2}^{n+1}$ is given by performing

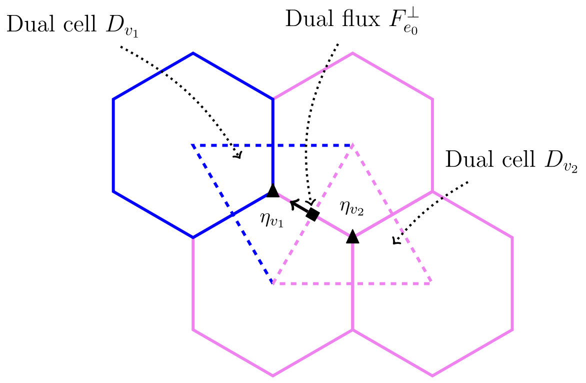


Fig. 5. TRiSK grid showing dual cells at the boundary between fine and interface one regions. Blue denotes the fine region and pink denotes the interface one region as in Fig. 2.

the interface correction step similar to (21). Then, we use the fact that $t_{e_0,v_1} = -t_{e_0,v_2}$ to conclude that the total flux entering D_{v_1} is equal to the total flux exiting D_{v_2}. This shows that a given edge is neither a source or a sink for absolute vorticity and therefore that the absolute vorticity is conserved globally. □

Remark 1. Both Ringler et al. [21] and Hoang et al. [10] state that TRiSK and LTS3, respectively, conserve globally integrated PV. This requires the clarification that in this case, conservation means that the value of the *volume* integral of PV is constant in time, as opposed to the *area* integral as in the case for conservation of thickness and absolute vorticity. FB-LTS also conserves PV in this sense; this follows from Theorem 2 and the fact that q_{v_*} is independent of the vertical position. That is,

$$\begin{aligned}\int q_{v_*}\,\mathrm{d}V &= \int h_{v_*} q_{v_*}\,\mathrm{d}A \\ &= \int \eta_{v_*}\,\mathrm{d}A\,.\end{aligned} \tag{37}$$

Note that since PV is defined on dual cells, this requires that the thickness be defined on dual cells as well. TRiSK defines an auxiliary thickness equation defined on dual cells in such a way that the dual cell thickness and the primal cell thickness are equivalent with each other in the sense that a prognostically obtained dual cell thickness is equal to the dual cell thickness obtained diagnostically by an interpolation of the primal cell thickness up to machine precision [21].

Remark 2. It is important to note that while Theorem 1 is proven in the context of a TRiSK spatial discretization, the result will hold for any conservative finite volume scheme. The result depends only on the fact that the fluxes as computed from either side of an edge shared by a fine and interface one cell are equal. In contrast, the proof of Theorem 2 does depend on the TRiSK scheme, specifically on the flux mapping operator $\mathcal{M}_e$ described in the proof above.

3. Implementation in MPAS-Ocean

Here, we describe some particulars relating to the implementation of the FB-LTS and LTS3 codes in MPAS-Ocean and compare to the now outdated implementation of LTS3 used in Lilly et al. [15] and Capodaglio and Petersen [5]. While the discussion in this section is largely in the context of MPAS-Ocean, the methods described can be applied in general. In particular, any implementation of FB-LTS would likely benefit from a load balancing procedure similar to that described at the end of Section 3.1. Similarly, the operator splitting from Section 3.2 has no intrinsic connection to MPAS-Ocean, but rather coincidentally helps to overcome a particular limitation of the MPAS-Ocean tendency routines.

3.1. MPI domain decomposition and parallelization

Balancing load and achieving efficient parallelization are non-trivial tasks when it comes to local time-stepping due to the asynchronous way in which the solution is advanced, causing an inherent load imbalance that needs to be properly addressed with ad-hoc implementation strategies. In Capodaglio and Petersen [5] and Lilly et al. [15], the authors obtained load balancing and parallel scalability by assigning to each MPI rank a well balanced number of cells from the coarse, interface, and fine LTS regions. This means that these three regions were treated as separate sets and each was partitioned across MPI ranks in a well balanced way. This process required the interface one and interface two regions to be augmented with additional cells in order for the MPI partition to provide at least 100 interface cells per rank, as a rule of thumb. Since the correction terms for the interface are computed during the sub-stepping procedure to minimize storage requirements, the proper number of additional interface cells needed to be tuned for best performance, as shown in Capodaglio and Petersen [5] and Lilly et al. [15].

The reason for treating the interface cells as a separate entity when it comes to load balancing was motivated by the need to address another crucial feature to guarantee computational performance, which is the ability to compute the right hand side terms only on the specific regions in which the solution will be advanced. A limitation of the underlying MPAS-Ocean framework is that the tendency routines only calculate tendency terms globally, with no way to only perform calculations on a given subset of cells. That is, if a given MPI rank owned both fine and coarse cells, it would not be possible to calculate tendencies only on fine cells, or only on coarse cells. In our previous works, we overcame this limitation by assigning each MPI rank multiple memory blocks, and looping over each memory block in serial. Each memory block would only contain one type of cell (fine, interface, or coarse), so when the tendency terms were called, the tendencies would only be calculated on those cells. Then, each MPI rank was assigned three memory blocks, each containing fine, interface, and coarse cells respectively. This allowed us to implement the LTS algorithms without changes to the underlying MPAS-Ocean framework.

Although convenient and effective, this procedure introduced a major issue in terms of parallel communication: Each memory block had its own halo cells on which parallel communication with other MPI ranks occurred, hence the communication cost of running with n tasks was actually that of $3n$ tasks. As outlined in Table 3 from Lilly et al. [15], the gains of using local time-stepping over a global time-stepping method were significantly reduced due to these communication issues becoming predominant when the number of cells per MPI rank became small, with 3000 cells per rank being observed as an approximate minimum for acceptable performance.

For the current work, the LTS3 and FB-LTS algorithms have been implemented in MPAS-Ocean within the main branch of the Energy Exascale Earth System Model (E3SM) [8]. This new implementation abandons the use of multiple memory blocks per MPI rank, which leads to two primary benefits. First, the amount of parallel communication does not grow by a factor of 3, hence the benefits of using local time-stepping still persist when the number of cells per MPI rank becomes small. Second, we no longer assign one block to each LTS region, hence we no longer need to tune the number of additional cells in the interface layers, and only use the minimum number of cells in the interface layers required by the numerical algorithms. Concerning the load balancing procedure, the mesh cells are divided in two sets, one containing the cells in the fine LTS region, and the other containing the coarse and interface LTS regions. Then, each MPI rank is assigned a well balanced number of cells from each set. That is, each MPI rank is has a well balanced number of both fine and coarse cells within the same memory block. The interface cells are not treated as a separate entity anymore on the load balancing procedure and are considered coarse cells, since the coarse time-step is used to advance the solution on them.

Discontinuing the use of multiple memory blocks per MPI rank required us to find a computationally efficient way to compute tendency terms only on the specific LTS region on which the solution is to be advanced. This has been achieved in part via an operator splitting approach, as described in the next section.

3.2. Operator splitting

It is well known that the ocean admits dynamics on a vast range of time-scales, with the most rapid motions being up to two orders of magnitude faster than the slowest. To account for this, climate-scale ocean models employ a barotropic/baroclinic splitting in which the fast 2D motions modeled by the vertically integrated barotropic subsystem are solved separately from the slow 3D motions in the baroclinic subsystem [9]. This fast barotropic subsystem is essentially the SWEs (2); however, even within this fast subsystem, there are still a range of time-scales at play that are often solved monolithically. The most rapid motions in a shallow water system are due to external gravity waves, which propagate at a rate of $\sqrt{gH}$, where H is the fluid resting depth. This means that, assuming an average ocean depth of 4000 m, gravity waves propagate at a rate of approximately 200 m s^{-1}. This is two orders of magnitude faster than Eulerian current velocities which are at most a few meters per second.

With the goal of exploiting this range of time-scales for computational performance, we introduce a certain operator splitting for the SWEs. We identify the external gravity-wave subsystem as *fast* and the remaining system, including the nonlinear momentum advection and forcing, as *slow*. The external gravity-wave subsystem is given by removing all forcing terms from the momentum equation except for the pressure gradient term, e.g. (22). In practice, the splitting works by evaluating the slow tendencies at time $t = t^n$ and then skipping their update during the subsequent time-stepping procedure. In particular, with a time discretization of the momentum equation in (3), the right hand side term Φ at a generic time t^*, i.e. $\Phi\left(\mathbf{u}(t^*), h(t^*)\right)$, is approximated as

$$\Phi\left(\mathbf{u}(t^*), h(t^*)\right) \approx \Phi^{\text{fast}}\left(\mathbf{u}(t^*), h(t^*)\right) + \Phi^{\text{slow}}\left(\mathbf{u}(t^n), h(t^n)\right) . \tag{38}$$

The approximation above can be seen as an additive Lie-Trotter splitting. Note that because the external gravity-wave system is treated as fast, there is no splitting within the thickness equation, only in the momentum equation. That is,

$$\begin{aligned} \Phi^{\text{fast}}(\mathbf{u}, h) &= -g\nabla(h + z_b) \\ \Psi^{\text{fast}}(\mathbf{u}, h) &= -\nabla \cdot (h\mathbf{u}) . \end{aligned} \tag{39}$$

We have made this choice of the fast terms motivated by the fact that the external gravity-waves are the fastest motions within the SWEs and impose the tightest CFL restriction. A notable exclusion from the fast terms is the Coriolis term, which is well known to cause stability issues. In particular, the Coriolis term influences the dispersion of Poincaré waves whose frequencies are close to the inertial frequency. It is possible that the omission of the Coriolis term from the fast terms could result in related instability. Whether the particular choice of the fast terms in (39) is optimal is an open question.

As a result of this splitting, the more computationally expensive slow terms are only evaluated once per coarse time-step, saving a significant number of floating point operations and MPI communication calls. This splitting does introduce a first-order error into the temporal local truncation error of our models, but we will show in Section 4.1 that the quality of the solution is not affected (Fig. 6). This is due to the fact that in complex real-world applications driven by observed data, like the one shown in Section 4.1, errors coming from the spatial discretization, model parameterization, and observed data are dominant. Further, the slow dynamics evolve on the order of hours, so taking time-steps on the order of hundreds of seconds, as in Section 4.1, does not introduce meaningful errors.

As mentioned above in Section 3.1, the splitting also helps with the practical concern of needing a way to restrict tendency calculations to certain regions of a mesh for a LTS scheme. MPAS-Ocean tendency routines are very complex and further rely on other routines that compute diagnostic quantities. In order to efficiently compute tendencies only on given LTS regions without using multiple memory blocks, one would need to reimplement the MPAS-Ocean tendency routines from scratch in a way that allows the domain of the calculation to be restricted. This would have been impractical to do for the full tendency routines, but because of the operator splitting, we only need to reimplement the parts of the routines for the fast terms. The versions of FB-LTS and LTS3 implemented in E3SM are implemented in the context of this operator splitting. In the numerical experiments in Section 4, we use the labels SplitFB-LTS and SplitLTS3 to refer to these schemes with this operator splitting. In the convergence test from Section 2.3, the same implementation of FB-LTS was used, but on a problem with all the slow terms disabled. This effectively removes the operator splitting, so FB-LTS shows the expected second-order convergence.

A final point of interest is how this splitting effects the CFL performance of the model. That is, we are interested in how this splitting effects the size of the admittable time-step in both the fine and coarse regions. In Section 4.2, we show two cases, one in which the CFL performance is not effected and one in which it is, and provide evidence that this depends on the ratio of the resolution of cells in the coarse region to the resolution of cells in the fine region.

4. Numerical experiments

Here, we present a series of numerical experiments that showcase the performance of SplitFB-LTS as compared to SplitLTS3, SplitFB-RK(3,2), and RK4. As described in Section 3.2, both LTS3 and FB-LTS are used in the context of the first-order operator splitting of the fast/slow subsystems within the SWEs. Additionally, note that the SplitFB-RK(3,2) scheme is obtained by running SplitFB-LTS with a uniform time-step everywhere on the mesh, i.e. taking $M = 1$. For these experiments, we model the storm surge caused by hurricane Sandy in and around Delaware Bay off the eastern coast of the United States, running on meshes that have been regionally refined to very high resolutions (2 km and 125 m) around Delaware Bay (Table 1 and Fig. 7).

4.1. *Hurricane model*

The momentum and thickness equations for the hurricane Sandy model are given by

$$\begin{aligned}\frac{\partial \mathbf{u}}{\partial t} + \left(\nabla\times\mathbf{u} + f\mathbf{k}\right)\times\mathbf{u} &= -\nabla K - \frac{1-\beta}{\rho_0}\nabla p^s - g\nabla\left(\xi - \xi_{EQ} - \beta\xi\right)\\ &\quad - \chi\frac{C\mathbf{u}}{H} - C_{\mathrm{D}}\frac{|\mathbf{u}|\,\mathbf{u}}{h} + C_{\mathrm{W}}\frac{|\mathbf{u}_{\mathrm{W}} - \mathbf{u}|\left(\mathbf{u}_{\mathrm{W}} - \mathbf{u}\right)}{h}\\ \frac{\partial h}{\partial t} + \nabla\cdot(h\mathbf{u}) &= 0\,,\end{aligned} \tag{40}$$

where $\mathbf{u}$ is the horizontal fluid velocity, t is the time coordinate, f is the Coriolis parameter, $\mathbf{k}$ is the local vertical unit vector, $K = \frac{|\mathbf{u}|^2}{2}$ is the kinetic energy per unit mass, β is the self-attraction and loading coefficient [1], ρ_0 is the (constant) fluid density, p^s is the surface pressure, g is the gravitational constant, ξ is the sea-surface height perturbation, ξ_{EQ} is the sea-surface height perturbation due to equilibrium tidal forcing [3], $\frac{C}{H}$ is a spatially varying internal tide dissipation coefficient [12], χ is a scalar tuning factor optimized for barotropic tides response [4]. H is the resting depth of the ocean, h is the total ocean thickness such that $h = H + \xi$, C_{D} is the bottom drag coefficient, C_{W} is the wind stress coefficient [7], and $\mathbf{u}_{\mathrm{W}}$ is the horizontal wind velocity. Here, the thickness equation is the conservation of volume for an incompressible fluid, where the volume is normalized by the cell area, which is constant in time. Additionally, note that our model does not employ a wetting and drying algorithm. There is an existing implementation of a wetting and drying algorithm for both RK4 and SplitLTS3, but not for SplitFB-LTS, so we opt to turn wetting and drying off for all methods. Applying such an algorithm would likely improve our models' comparisons to observations, but as our primary focus is the comparison of performance and accuracy between individual time-stepping methods, leaving wetting and drying off is sufficient for our purposes. In future work, we plan to implement a wetting and drying algorithm for SplitFB-LTS.

The LTS algorithms implemented in MPAS-Ocean use the operator splitting described in Section 3.2; for this hurricane model the terms treated as fast in (40) are

$$\begin{aligned}\Phi^{\mathrm{fast}}(\mathbf{u}, h) &= -g\nabla\left(\xi - \beta\xi\right)\\ \Psi^{\mathrm{fast}}(\mathbf{u}, h) &= -\nabla\cdot(h\mathbf{u})\,.\end{aligned} \tag{41}$$

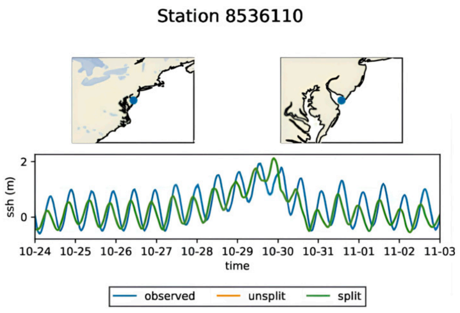


Fig. 6. The sea-surface height solution produced by the hurricane Sandy model using LTS3 both with and without the operator splitting described in Section 3.2. The unsplit solution is covered by the split solution, showing the split model produces a solution of the same quality.

That is, the external gravity wave subsystem is treated as fast. To show that this splitting does not degrade the quality of the model solution, we run the hurricane model using the unsplit implementation of LTS3 from the now outdated version of MPAS-Ocean used in Lilly et al. [15], and the current version (Fig. 6).

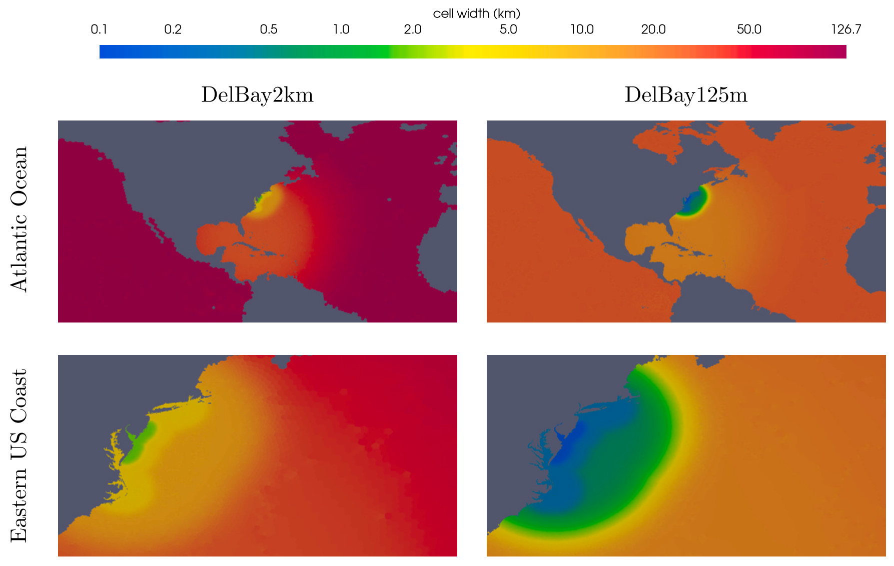
We consider two meshes, DelBay2km and DelBay125m, shown in Fig. 7 and described in Table 1. These meshes are similar to those by the same names from Lilly et al. [15], but have slight differences due to updates to mesh generation libraries. Both meshes highly resolve Delaware Bay, then smoothly vary in resolution out to a low global background resolution. These meshes were chosen to highly resolve the hurricane itself while controlling the overall number of cells in the global mesh, making the potential benefits of local time-stepping schemes obvious. The simulations are run for 24 simulated-days; starting on 10/10/2012 and ending on 11/03/2012.

Related to these meshes, there are two parameters which are important to understanding the performance of our LTS schemes. What we refer to as the *count ratio* is the ratio of the number of coarse cells to the number of fine cells in the mesh, i.e. a count ratio greater than one means that the mesh contains more cells using the coarse time-step than the fine time-step. The *resolution ratio* is the ratio of the cell width of the coarse cells to the cell width of the fine cells. In the case where there are cells of multiple resolutions in either region, as is the case in our meshes, we consider the smallest value of cell width in a given region as it is the smallest cell that restricts the size of the time-step admittable in that region. The higher the resolution ratio, the higher we expect M to be, the ratio between the fine and coarse time-step. See Lilly et al. [15] for further details about this kind of analysis, Fig. 6 and the surrounding discussion in particular.

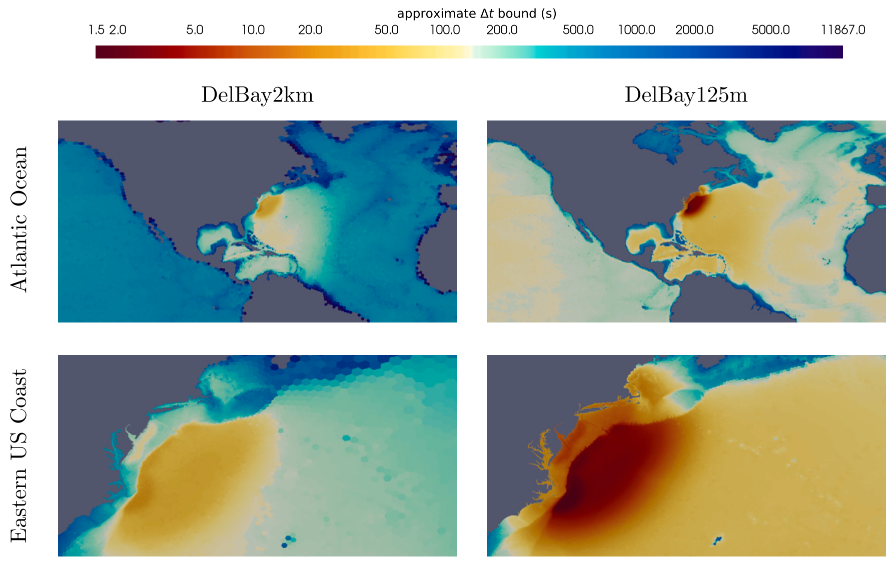
Both SplitLTS3 and SplitFB-LTS were configured to use the same configuration of the LTS regions, taking the fine region (the region where the fine time-step is used) to be around the Delaware coast and the coarse region (the region where the coarse time-step is used) to be the rest of the globe as pictured in Fig. 7a. One can note that the FB-LTS scheme uses more cells labeled as interface adjacent fine cells than LTS3; these are the cells denoted as $\mathcal{F}_P^5$ cells in Section 2.2. These cells are advanced with the same fine time-step as the rest of the fine region, they are only differentiated within the FB-LTS and LTS3 algorithms because these cells require a small number of additional computations in order to obtain needed data on the interface one and two regions during the coarse advancement step of both methods. In order to obtain the data needed to perform the interface one prediction step, we need to obtain uncorrected data on interface one at time t^{n+1}. FB-RK(3,2) takes three stages to obtain data at time t^{n+1}, while SSPRK3 produces a prediction for this data with its first stage; as a result the domain of dependence for interface one cells to obtain this needed data is larger for FB-LTS. One can also note that, as described in Section 2.2, the FB-LTS algorithm requires these extra computations on a decreasing subset of these interface adjacent fine cells as the method progresses through the coarse advancement step, starting with $\mathcal{F}_P^5$, then $\mathcal{F}_P^4$, then $\mathcal{F}_P^3$, et cetera. In order to decrease the complexity of the FB-LTS implementation in MPAS-Ocean we opt not to shrink this region and instead perform these extra calculations on all the interface adjacent cells pictured in Fig. 2. This has no effect on the numerical scheme itself as the unneeded data is simply thrown away, and the penalty to the computational performance of the implementation is negligible.

Further, note that the configuration of the LTS regions in Fig. 8 assumes that the radius of the tendency operators for both the velocity and the thickness is $r = 2$. If one were to add additional terms to the fast subsystem treated with LTS, the number of layers of interface cells could increase. For example, in TRiSK, the radius of a biharmonic turbulence closure ("del4") for the momentum equation is $r = 3$. Adding this to the fast subsystem would increase the number of layers of interface 1 and interface 2 cells to three, and the number of interface adjacent fine cells to 15.

The time-steps used on each mesh by each time-stepping scheme for performance tests are given in Table 1. These time-steps were obtained experimentally by running the model at increasing time-steps until it becomes unstable and selecting the largest time-steps for which it is stable. In the case of DelBay2km, the model is stable at the reported time-steps for the entire 24 simulated-day duration of the hurricane simulation. Because of the greatly increased computational cost of running the model on DelBay125m, the reported time-steps are obtained by running for one simulated-day starting from an initial condition given by the model state after three simulated days (10/13), which is when the tidal dynamics stabilize after spinning up from rest. When finding the fine and coarse time-steps in this way, there are two possible approaches; the fine and coarse time-steps must differ by an integer factor of M such that $\Delta t_{\text{fine}} = \frac{\Delta t_{\text{coarse}}}{M}$, so they can be obtained in different orders. One could find the largest time-step admittable on the coarse region of the mesh then find the smallest value of M that gives an admittable fine time-step, or first find the largest admittable fine time-step



(a) Cell width in kilometers for DelBay2km and DelBay125m. Each subplot shares the same color scale in log-space.



(b) The approximate bound on the time-step in seconds due to the CFL condition imposed by external gravity waves. Each cell is assigned a value in color space equal to $\nu^{\max}\frac{\Delta x}{\sqrt{gH}}$, the bound on Δt given by the CFL condition (1), assuming that $c = \sqrt{gH}$. As the maximal Courant number $\nu^{\max}$ varies depending on the chosen discretizations, we set it to 1 here. Each subplot shares the same color scale in log-space.

Fig. 7. The DelBay2km and DelBay125m meshes as described in Table 1 for the hurricane Sandy test case. Note that these are global meshes and that parts of the globe not pictured here use the background resolutions shown in Table 1.

then the largest value of M that gives an admittable coarse time-step. This has the result of the time-step in one region or the other not technically being maximal. For both DelBay2km and DelBay125m, we opt to first maximize the time-step in the fine region, then find the largest admittable value of M.

Looking at the CFL estimates given by Fig. 7b, we see good agreement between the approximated maximum time-steps and the experimentally obtained maximal time-steps from Table 1 on DelBay2km. On this mesh, our methods take time-steps between 20 s and 46 s in the fine region, and between 60 s and 138 s in the coarse region. On DelBay125m, we have similarly good agreement in the fine region, taking time-steps between 1 s and 2 s. However, in the coarse region, our LTS schemes are taking time-steps smaller

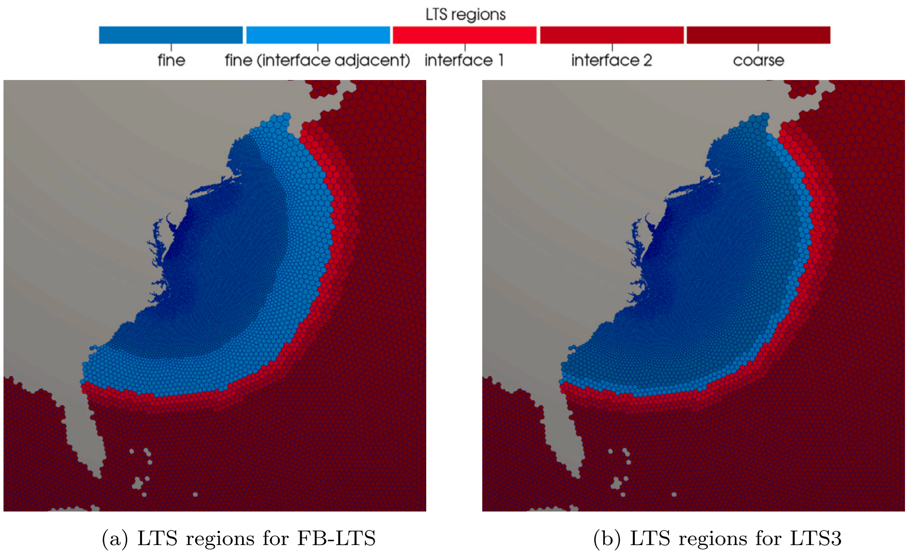


Fig. 8. Configurations for the LTS regions for FB-LTS and LTS3 as shown on DelBay2km. The LTS regions are configured similarly on DelBay125m.

than predicted by the CFL estimates. We should be taking time-steps on the order of 30 s in the coarse region, but coarse time-steps for both LTS schemes are 4 s instead. As discussed in Section 4.2, we find that this discrepancy is due to the operator splitting.

Further, the CFL estimates from Fig. 7b show how the restriction on the time-step depends on both cell size Δx and resting bottom depth H. In particular, note that the tightest bounds are not imposed by the smallest cells deep in Delaware bay, but instead by mid-sized cells on the continental shelf. This points to the potential value of using CFL estimates like these as criteria by which the fine and coarse regions are defined. While the LTS regions as configured here (Fig. 8) are effective, they are not necessarily optimal; it is likely that a more complex selection of the fine and coarse regions could be made that would further increase the speedup obtained by LTS. For instance, a configuration based on the CFL distribution shown in Fig. 7b. Such a configuration is subject of current work and will be explored in a future paper.

4.2. Computational performance

In order to demonstrate the computational performance of SplitFB-LTS, we run the hurricane Sandy model using RK4, SplitFB-RK(3,2), SplitLTS3, and SplitFB-LTS on both DelBay2km and DelBay125m. For each mesh, we run the model for a number of simulated-seconds equal to a common multiple of the time-steps used by each scheme. We measure the speedup by taking the ratio of the CPU-time for the slower scheme to that of the faster scheme, that is,

$$\text{speedup} = \frac{\text{slower scheme CPU-time}}{\text{faster scheme CPU-time}}\,. \tag{42}$$

For example, the speedup for SplitFB-LTS versus RK4 is given by

$$\text{speedup vs. RK4} = \frac{\text{RK4 CPU-time}}{\text{SplitFB-LTS CPU-time}}\,. \tag{43}$$

On DelBay2km, SplitFB-LTS outperforms RK4 by a factor of 10.08 and outperforms SplitLTS3 by a factor of 2.27, and on DelBay125m FB-SplitLTS outperforms RK4 by a factor of 5.13 and outperforms SplitLTS3 by a factor of 1.32 (Table 2). The speedups obtained on DelBay125m are less stark than those obtained on DelBay2km; this is explained in part by the CFL issue introduced by the splitting as discussed further below. However, one must also consider that the count ratio on DelBay125m is much smaller than that on DelBay2km. On DelBay125m, most of the mesh consists of fine cells, where the advantage of SplitFB-LTS versus RK4 is less than on the coarse cells. Because of this, it is expected that the speedup on DelBay125m would be less than that on DelBay2km.

Previous to the implementation of SplitLTS3 and SplitFB-LTS in the MPAS-Ocean code-base, RK4 scheme was the method of choice for single-layer configurations like that used in our hurricane model, so the speedup versus RK4 achieved by both LTS schemes is significant and points to the value of local time-stepping schemes and the operator splitting described in Section 3.2 for efficiency of ocean simulations. SplitFB-LTS further improves on the speedup achieved by SplitLTS3 by taking advantage of the CFL optimized scheme on which it is based, and it will be shown in Section 4.3 that this additional speedup does not incur a penalty to the quality of the model solution.

We can also compare the results from Table 2 to results from Table 2 from Lilly et al. [15]. On DelBay2km, the previous unsplit implementation of LTS3 only outperformed RK4 by a factor of 1.48, while on DelBay125m unsplit LTS3 was slower than RK4 by a factor of 1.16. One should keep in mind that there multiple factors involved in the difference in performance between these two cases; the new schemes benefit from less MPI communication overhead due to the improved domain decomposition paradigm described in

Table 1
Relevant parameters for each mesh and LTS scheme used in performance experiments. The *count ratio* is the ratio of the number of coarse cells to the number of fine cells in the mesh. The *resolution ratio* is the ratio of the cell width of the coarse cells to the cell width of the fine cells. For both meshes, the fine region includes Delaware Bay, the Delaware coast, and the Eastern US coast. The coarse region includes the Western Atlantic and the rest of the globe (global background).

	DelBay2km	**DelBay125m**
Grid Cell Width (km)		
Global background	120	30
Western Atlantic	30	15
Eastern US coast	10	0.625
Delaware coast	5	0.3125
Delaware Bay	2	0.125
Mesh Parameters		
Number of cells	58,141	4,617,372
Count ratio	1.92	0.12
Resolution ratio	15	120
Time-Steps		
RK4		
Δt_{global} (s)	34	2
SplitLTS3		
Δt_{fine} (s)	20	1
Δt_{coarse} (s)	60	4
M	3	4
SplitFB-LTS		
Δt_{fine} (s)	46	2
Δt_{coarse} (s)	138	4
M	3	2
Time-Step Ratios		
$\Delta t^{\text{FB-LTS}}_{\text{fine}} / \Delta t^{\text{RK4}}_{\text{global}}$	1.35	1.00
$\Delta t^{\text{FB-LTS}}_{\text{fine}} / \Delta t^{\text{LTS3}}_{\text{fine}}$	2.30	2.00
$\Delta t^{\text{FB-LTS}}_{\text{coarse}} / \Delta t^{\text{LTS3}}_{\text{coarse}}$	2.30	1.00

Table 2
CPU-time performance of RK4, SplitFB-RK(3,2), SplitLTS3, and SplitFB-LTS on DelBay2km and DelBay125m. Each reported time above is the average of individual runs, to account for conditions on the machine. The run-times were chosen as common multiples of the time-steps used be each scheme as reported in Table 1. On each mesh, the global time-steps used by SplitFB-RK(3,2) are equal to the fine time-steps used by SplitFB-LTS.

	DelBay2km	**DelBay125m**
run-time (hh:mm:ss)	06:31:00	00:10:00
number of MPI ranks	32	128
RK4 (s)	*26.07*	*186.02*
SplitFB-RK(3,2) (s)	*6.72*	*58.78*
speedup vs. RK4	3.88	3.16
SplitLTS3 (s)	*5.86*	*47.85*
speedup vs. RK4	4.45	3.89
speedup vs. SplitFB-RK(3,2)	1.15	1.23
SplitFB-LTS (s)	*2.59*	*36.26*
speedup vs. RK4	10.08	5.13
speedup vs. SplitFB-RK(3,2)	2.60	1.62
speedup vs. SplitLTS3	2.27	1.32

Section 3.1 and the operator splitting described in Section 3.2 in particular. Regardless, this shows the strong progression of efforts to increase the computational efficiency of these SWE solvers.

As one of the primary results, we take some additional time to break-down and understand the factor 10.08 speedup obtained by SplitFB-LTS versus RK4 on DelBay2km (Fig. 9). There are three distinct sources of speedup that are contributing to this result. First, moving from RK4 to (unsplit) FB-RK(3,2). Second, moving from FB-RK(3,2) to SplitFB-RK(3,2). Third, moving from SplitFB-RK(3,2) to SplitFB-LTS. Call these speedups A, B, and C respectively. We get an estimate for C from Table 2 as $C \approx 2.60$. We cannot get

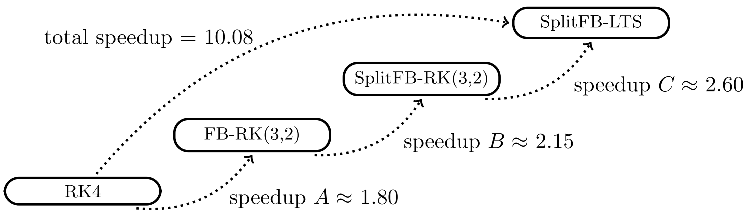


Fig. 9. A visualization of the sources of the speedup obtained by SplitFB-LTS versus RK4 on DelBay2km. The analysis performed to obtain the estimated speedup contributed by each step can be found in the text of Section 4.2.

direct experimental estimates for A and B since there is no unsplit implementation of FB-RK(3,2) in MPAS-Ocean. However, we can estimate both values through a simple analysis. First, we can estimate A by looking at the size of the time-steps used by RK4 and FB-RK(3,2) relative to the number of Runge-Kutta stages in each scheme. On DelBay2km, RK4 takes a global time-step of 34 s over four RK stages, so RK4 advances $34/4 = 8.5$ seconds per RK stage. Assuming that FB-RK(3,2) uses the fine time-step used by SplitFB-LTS, it uses a global time-step of 46 s over three RK stages, so it advances $46/3 \approx 15.33$ seconds per RK stage. Taking these two quantities together, we estimate that $A \approx 15.33/8.5 \approx 1.80$. Lastly, since we know that $ABC = 10.08$, we simply estimate $B \approx 10.08/AC \approx 2.15$. From this, we see that each of the three algorithmic developments considered here contribute significantly to the overall speedup obtained. Additionally, we give some supplementary analysis related to the speedup C and the relative computational cost of the slow and fast tendencies in Appendix B.

As noted in Section 3.2, we are also interested in how the operator splitting affects the CFL performance of our LTS schemes. On DelBay2km, the results suggest that the CFL performance of both SplitLTS3 and SplitFB-LTS is not affected by the splitting. That is, the restriction placed on the maximum admittable time-steps, as reported in Table 1, are enforced by the terms treated as fast. In this case SplitFB-LTS is taking time-steps 2.3 times larger than SplitLTS3 in both the fine and coarse regions, as we expect from Lilly et al. [16]. Contrast this with the time-steps used on DelBay125m; SplitFB-LTS outperforms SplitLTS3 in CFL efficiency by a factor of 2 in the fine region but both take a time-step of 4 s in the coarse region. This behavior is not consistent with what we expect to achieve, if we assume the time-steps are bound by the fast terms (Fig. 7b). Further, we would expect that the ratio M between the fine and coarse time-step was much larger. For DelBay2km the resolution ratio is 15, and $M = 3$. On DelBay125m with a resolution ratio of 120 and the same underlying model equations, we would expect to have that M is at least 3, if not significantly larger (in Lilly et al. [15], the authors found that $M = 24$ for unsplit LTS3 in this case).

Motivated by the desire to understand this unexpected CFL behavior, we ran additional tests that revealed that the problem was due to a global CFL restriction imposed by the slow terms. Running the model on DelBay125m, with both SplitLTS3 and SplitFB-LTS, we observed that any increase to the coarse time-step would cause the model to become unstable, with the instability manifesting in the fine region far from the interface regions, deep in Delaware Bay. The fact that increasing the coarse time-step causes instability in the fine region means that it is very unlikely that the instability is being caused by the fast terms, which are being advanced with LTS.

The slow terms are only calculated once per *coarse* time-step at time $t = t^n$, then that value is used to advance on the entire mesh. As a result, we encounter stability issues in the fine region coming from the slow terms. It is possible that this is due to the omission of the Coriolis term from the fast terms in the splitting as mentioned in Section 3.2; this will be a topic of future work.

A possible way to combat this limit on CFL performance imposed by the splitting would be to alter the algorithm so that, during the *Fine Advancement* step, the slow tendencies were evaluated once per fine-step. Instead of using the values of the slow tendencies as evaluated at time t^n for the entire routine, we could calculate the slow tendencies at times $t^{n,k}$ for $k = 0, \cdots M-1$ and use these values while advancing on fine cells. Doing this, the algorithm would effectively be treating the slow subsystem equally in both the fine and coarse regions. Exploration of this particular change to the operator splitting algorithm is outside the scope of this work, as its implementation in MPAS-Ocean would be particularly difficult due to the limitations of the framework described in Section 3. Alternatively, it is possible that adding the Coriolis term to the set of fast terms would be beneficial. To summarize, the speedup achieved by SplitFB-LTS versus RK4 on DelBay125m is significant, but the majority is necessarily coming from the splitting rather than from the local time-stepping because of the issue discussed above. If SplitFB-LTS were taking time-steps at least 24 times larger than RK4 in the coarse region, we would expect to see some additional speedup on DelBay125m due to LTS. Regardless, because the number of coarse cells in DelBay125m is small relative to the number of fine cells, it is likely that the speedup due to LTS would not be as significant as the speedup due to the splitting.

Finally, testing the strong parallel scaling of RK4, SplitLTS3, and SplitFB-LTS, we see the expected approximately linear strong scaling as the number of MPI ranks is increased (Fig. 10). For each of the three methods, one can notice a slight degradation in the scaling as we reach 32 ranks; this occurs because the number of cells per process (which in this case is approximately 1,800) is small enough that the communication between MPI ranks begins to dominate the simulation. This is expected behavior, and simply a signal that running the simulation on this mesh with more ranks would have diminishing returns.

4.3. Solution quality

Along with the greatly increased performance discussed in Section 4.2, SplitFB-LTS is able to produce SSH solutions of the same quality of both RK4 and SplitLTS3. We run the hurricane model using all three time-stepping schemes on DelBay2km for the full

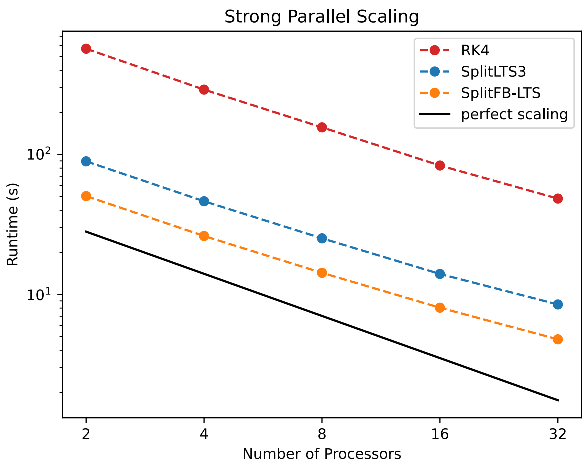
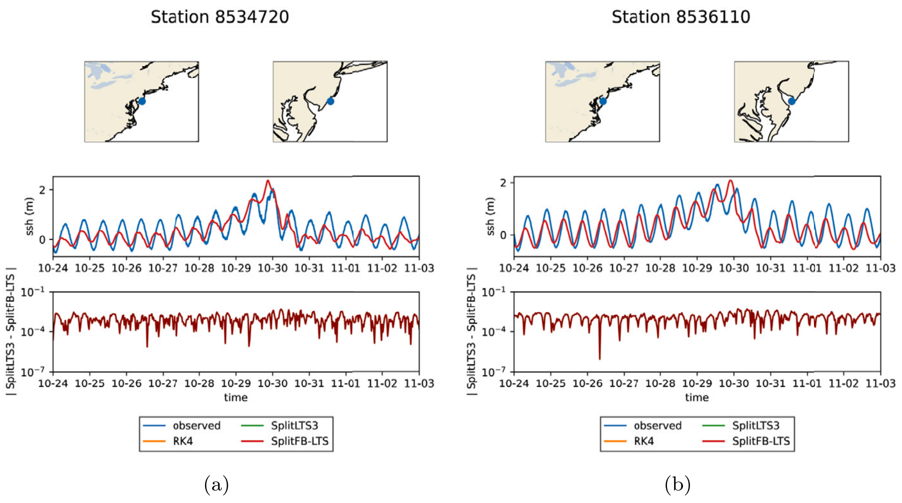


Fig. 10. Strong parallel scaling in the hurricane Sandy test case on DelBay2km, run for 12 simulated hours.



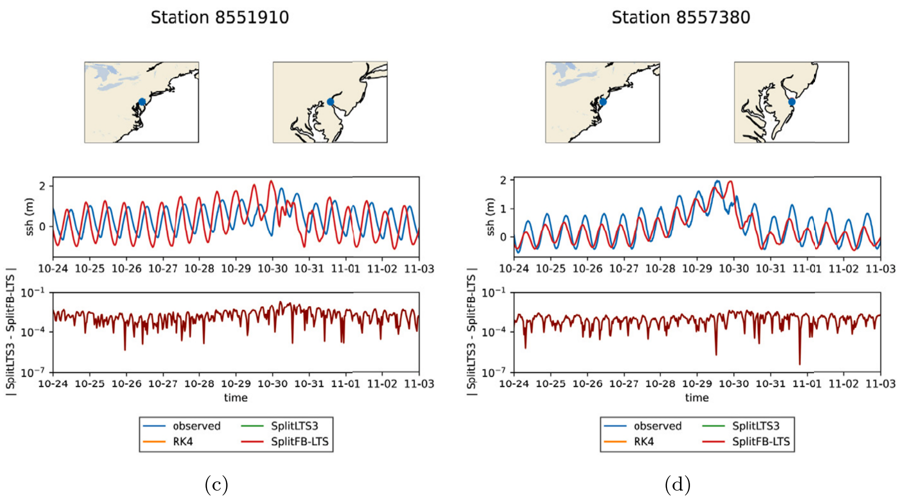


Fig. 11. SSH solutions from RK4, SplitLTS3, and SplitFB-LTS compared to observed tide gauge data on DelBay2km, using time steps of 30 s, 60 s, and 120 s respectively, with $M = 3$ for both LTS schemes. Note that The solution curves are directly on top of each other. The absolute difference between SplitLTS3 and SplitFB-LTS is shown on a log scale in the lower plots.

duration of the simulation and record the model SSH at the locations of tidal gauges in and around Delaware Bay. We also compare model solutions to observed data. The observed data are from NOAA's Center for Operational Oceanographic Products and Services (CO-OPS) gauges and are available at https://tidesandcurrents.noaa.gov/. We observe that the model solutions for each time-stepping scheme do not differ meaningfully. In particular, the difference between the SSH solutions produced by SplitLTS3 and SplitFB-LTS is, at most, on the order of centimeters (Fig. 11).

5. Conclusion

We have presented a new local time-stepping scheme (FB-LTS) for the shallow water equations based on the CFL optimized forward-backward Runge-Kutta schemes from Lilly et al. [16]. We have shown that the scheme gives exact conservation of mass and absolute vorticity when applied to a TRiSK discretization and performed numerical experiments that show that FB-LTS is a second order scheme everywhere, including on interface cells that allow communication between the fine and coarse regions. Implementing this method in MPAS-Ocean along with the operator splitting described in section 3.2, the SplitFB-LTS scheme produces solutions qualitatively equivalent to those produced by a SSPRK3 based local time-stepping scheme (SplitLTS3) and by the classical four-stage, fourth-order Runge-Kutta method (RK4). Further, these solutions are produced at a significantly reduced computational cost; SplitFB-LTS is up to 10.08 times faster (in terms of CPU-time) than RK4, and up to 2.27 times faster than SplitLTS3.

Moving forward, we are interested in adapting SplitFB-LTS (and/or FB-LTS) for use in a multi-layer ocean model that uses a barotropic-baroclinic splitting. In particular, SplitFB-LTS would be a strong choice for a barotropic solver, as the barotropic subsystem is similar to the SWEs, so we would expect to see performance benefits very similar to those reported in this work. As we continuously chase greater computational performance in large scale models, particularly those of Earth's climate, it is important to carefully consider the efficiency and efficacy of our methods. Our work here shows that SplitFB-LTS could significantly accelerate a layered model's barotropic solver at little cost to accuracy.

CRediT authorship contribution statement

Jeremy R. Lilly: Writing – review & editing, Writing – original draft, Visualization, Validation, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Giacomo Capodaglio:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Investigation. **Darren Engwirda:** Writing – review & editing, Methodology, Investigation, Conceptualization. **Robert L. Higdon:** Writing – review & editing, Methodology, Formal analysis. **Mark R. Petersen:** Supervision, Funding acquisition.

Data statement

The source code for E3SM (and MPAS-Ocean as a component) that implements SplitFB-LTS is available on GitHub.

E3SM: https://github.com/E3SM-Project/E3SM/tree/6b38b86

Similarly, the test case configuration libraries that implement the hurricane Sandy test case (Compass) and the external gravity wave test case (Polaris) for SplitFB-LTS are also available on GitHub.

Compass: https://github.com/MPAS-Dev/compass/tree/74edf14
Polaris: https://github.com/jeremy-lilly/polaris/tree/aquaplanet-external-gravity-wave

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jeremy Lilly reports financial support was provided by National Science Foundation. Giacomo Capodaglio reports financial support was provided by Los Alamos National Laboratory. Mark Petersen reports financial support was provided by Los Alamos National Laboratory. Darren Engwirda reports financial support was provided by Los Alamos National Laboratory. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

JRL was supported in part by the National Science Foundation (NSF) Mathematical Sciences Graduate Internship (MSGI). GC, MRP, and DE were supported by the Earth System Model Development program area of the U.S. DOE Office of Science, Office of Biological and Environmental Research as part of the multi-program, collaborative Integrated Coastal Modeling (ICoM) project.

Appendix A. Derivation of interface prediction coefficients

FB-RK(3,2) is a second order scheme, so we will derive second order predictor coefficients for use in the *Interface Prediction* step of FB-LTS. Assume that we already have data for $\mathbf{h}^n$ and $\mathbf{u}^n$, and uncorrected data $\tilde{\mathbf{h}}^{n+1/3}$, $\tilde{\mathbf{u}}^{n+1/3}$, $\tilde{\mathbf{h}}^{n+1/2}$, $\tilde{\mathbf{u}}^{n+1/2}$, $\tilde{\mathbf{h}}^{n+1}$, and $\tilde{\mathbf{u}}^{n+1}$ on interface one.

Start with the spatially discretized system in (5), and assume that both Φ_e and Ψ_i are Lipschitz for all $e \in C_E^{\text{IF-1}}$ and $i \in C_P^{\text{IF-1}}$. First, we need predictions for $\mathbf{h}^{n,k}$ for $k = 0, \cdots M$ and $\mathbf{u}^{n,k}$ for $k = 0, \cdots M-1$. Start with $\mathbf{h}^{n,k}$; take a Taylor series expansion centered at time $t = t^n$, writing terms out to get a second order approximation for $i \in C_P^{\text{IF-1}}$,

$$\begin{aligned} h_i\left(t^n + k\frac{\Delta t}{M}\right) &= h_i(t^n) + \left(t^n + k\frac{\Delta t}{M} - t^n\right)\tfrac{\partial h_i}{\partial t}(t^n) + \mathcal{O}\left((\Delta t)^2\right) \\ &= h_i(t^n) + k\frac{\Delta t}{M}\tfrac{\partial h_i}{\partial t}(t^n) + \mathcal{O}\left((\Delta t)^2\right). \end{aligned}$$

Next, we approximate $h_i(t^n)$ by h_i^n, which is a second order approximation (the order of FB-RK(3,2)). Then $\frac{\partial h_i}{\partial t}(t^n)$ can be approximated by Forward Euler (FE), $\frac{\partial h_i}{\partial t}(t^n) = \frac{\tilde{h}_i^{n+1} - h^n}{\Delta t} + \mathcal{O}\left((\Delta t)^1\right)$. Insert these into the above,

$$\begin{aligned} h_i\left(t^n + k\frac{\Delta t}{M}\right) &= \left(h_i^n + \mathcal{O}\left((\Delta t)^2\right)\right) + k\frac{\Delta t}{M}\left(\frac{\tilde{h}_i^{n+1} - h^n}{\Delta t} + \mathcal{O}\left((\Delta t)^1\right)\right) + \mathcal{O}\left((\Delta t)^2\right) \\ &= h_i^n + k\frac{\Delta t}{M}\frac{\tilde{h}_i^{n+1} - h_i^n}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right) \\ &= \frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right)h_i^n + \mathcal{O}\left((\Delta t)^2\right). \end{aligned}$$

Therefore, a second order approximation to $\mathbf{h}$ data at times $t^{n,k}$ for $i \in C_P^{\text{IF-1}}$ is given by

$$h_i^{n,k} = \frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right)h_i^n. \tag{A.1}$$

Similarly for $\mathbf{u}^{n,k}$, take a Taylor series expansion centered at t^n for $e \in C_E^{\text{IF-1}}$,

$$\begin{aligned} u_e\left(t^n + k\frac{\Delta t}{M}\right) &= u_e(t^n) + \left(t^n + k\frac{\Delta t}{M} - t^n\right)\tfrac{\partial u_e}{\partial t}(t^n) + \mathcal{O}\left((\Delta t)^2\right) \\ &= u_e(t^n) + k\frac{\Delta t}{M}\tfrac{\partial u_e}{\partial t}(t^n) + \mathcal{O}\left((\Delta t)^2\right). \end{aligned}$$

Approximate $u_e(t^n)$ by u_e^n, which is a second order approximation (the order of FB-RK(3,2)). We approximate $\frac{\partial u_e}{\partial t}(t^n)$ similarly to the above, $u_t(t^n) = \frac{\tilde{u}^{n+1} - u^n}{\Delta t} + \mathcal{O}\left((\Delta t)^1\right)$. Insert these approximations into the above to get

$$\begin{aligned} u_e\left(t^n + k\frac{\Delta t}{M}\right) &= \left(u_e^n + \mathcal{O}\left((\Delta t)^2\right)\right) + k\frac{\Delta t}{M}\left(\frac{\tilde{u}_e^{n+1} - u_e^n}{\Delta t} + \mathcal{O}\left((\Delta t)^1\right)\right) + \mathcal{O}\left((\Delta t)^2\right) \\ &= u_e^n + k\frac{\Delta t}{M}\frac{\tilde{u}_e^{n+1} - u_e^n}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right) \\ &= \frac{k}{M}\tilde{u}_e^{n+1} + \left(1 - \frac{k}{M}\right)u_e^n + \mathcal{O}\left((\Delta t)^2\right). \end{aligned}$$

A second order approximation to $\mathbf{u}$ data times $t^{n,k}$ for $e \in C_E^{\text{IF-1}}$ is given by

$$u_e^{n,k} = \frac{k}{M}\tilde{u}_e^{n+1} + \left(1 - \frac{k}{M}\right)u_e^n. \tag{A.2}$$

Next, we require predictions for $\bar{\mathbf{h}}^{n,k+1/3}$ and $\bar{\mathbf{u}}^{n,k+1/3}$ for $k = 0, \cdots, M-1$. Starting with $\bar{\mathbf{h}}^{n,k+1/3}$, we insert the above prediction $\mathbf{h}^{n,k}$, given by (A.1), into the first stage of FB-RK(3,2) with the fine time-step $\frac{\Delta t}{M}$ for $i \in C_P^{\text{IF-1}}$,

$$\bar{h}_i^{n,k+1/3} = h_i^{n,k} + \frac{\Delta t}{3M}\Psi_i\left(\mathbf{u}^{n,k}, \mathbf{h}^{n,k}\right).$$

We need a way to approximate $\Psi_i\left(u^{n,k}, h^{n,k}\right)$. Assuming sufficient smoothness of Ψ_i, we can use that $\Psi_i\left(u^{n,k}, h^{n,k}\right) = \Psi_i\left(u^n, h^n\right) + \mathcal{O}\left((\Delta t)^1\right)$. Insert this into the above and we get

$$\bar{h}_i^{n,k+1/3} = h_i^{n,k} + \frac{\Delta t}{3M}\left(\Psi_i\left(u^n, h^n\right) + \mathcal{O}\left((\Delta t)^1\right)\right)$$

$$= \left(\frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right) h_i^n + \mathcal{O}\left((\Delta t)^2\right) \right) + \frac{\Delta t}{3M}\Psi_i\left(\mathbf{u}^n, \mathbf{h}^n\right) + \mathcal{O}\left((\Delta t)^2\right)$$

$$= \frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right) h_i^n + \frac{\Delta t}{3M}\Psi_i\left(u^n, h^n\right) + \mathcal{O}\left((\Delta t)^2\right).$$

Then, we know that $\Psi_i\left(\mathbf{u}^n, \mathbf{h}^n\right) = \frac{3(\tilde{h}_i^{n+1/3} - h_i^n)}{\Delta t}$. Substitute this to get

$$\bar{h}_i^{n,k+1/3} = \frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right) h_i^n + \frac{\Delta t}{3M}\frac{3(\tilde{h}^{n+1/3} - h^n)}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right)$$

$$= \frac{k}{M}\tilde{h}_i^{n+1} + \frac{1}{M}\tilde{h}_i^{n+1/3} + \left(1 - \frac{k+1}{M}\right) h_i^n + \mathcal{O}\left((\Delta t)^2\right).$$

A second order approximation to first-stage $\mathbf{h}$ data at intermediate time levels for $i \in C_P^{\text{IF-1}}$ is given by

$$\bar{h}_i^{n,k+1/3} = \frac{k}{M}\tilde{h}_i^{n+1} + \frac{1}{M}\tilde{h}_i^{n+1/3} + \left(1 - \frac{k+1}{M}\right) h_i^n. \tag{A.3}$$

Proceed similarly for $\bar{\mathbf{u}}^{n,k+1/3}$, insert $\mathbf{u}^{n,k}$ into the first stage of FB-RK(3,2) with the fine time-step for $e \in C_E^{\text{IF-1}}$,

$$\bar{u}_e^{n,k+1/3} = u_e^{n,k} + \frac{\Delta t}{3M}\Phi_e\left(\mathbf{u}^{n,k}, \mathbf{h}^{*,k}\right).$$

We need to approximate $\Phi_e\left(\mathbf{u}^{n,k}, \mathbf{h}^{*,k}\right)$. We would like to do this by $\Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right) = \frac{3(\tilde{u}_e^{n+1/3} - u_e^n)}{\Delta t}$. One can show that $\left|\Phi_e\left(\mathbf{u}^{n,k}, \mathbf{h}^{*,k}\right) - \Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right)\right| \le \mathcal{O}\left((\Delta t)^1\right)$, and so $\Phi_e\left(\mathbf{u}^{n,k}, \mathbf{h}^{*,k}\right) = \Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right) + \mathcal{O}\left((\Delta t)^1\right)$. This follows from the assumption that Φ_e is Lipschitz, and we omit the details for reasons of space. Insert this approximation into the above and we get

$$\bar{u}_e^{n,k+1/3} = u_e^{n,k} + \frac{\Delta t}{3M}\left(\Phi_e\left(\mathbf{u}^n, \mathbf{h}^*\right) + \mathcal{O}\left((\Delta t)^1\right)\right)$$

$$= \left(\frac{k}{M}\tilde{u}^{n+1} + \left(1 - \frac{k}{M}\right) u^n + \mathcal{O}\left((\Delta t)^2\right)\right) + \frac{\Delta t}{3M}\frac{3(\tilde{u}_e^{n+1/3} - u_e^n)}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right)$$

$$= \frac{k}{M}\tilde{u}_e^{n+1} + \frac{1}{M}\tilde{u}_e^{n+1/3} + \left(1 - \frac{k+1}{M}\right) u_e^n + \mathcal{O}\left((\Delta t)^2\right).$$

A second order approximation to first-stage $\mathbf{u}$ data at intermediate time levels for $e \in C^{\text{IF-1}}$ is given by

$$\bar{u}_e^{n,k+1/3} = \frac{k}{M}\tilde{u}_e^{n+1} + \frac{1}{M}\tilde{u}_e^{n+1/3} + \left(1 - \frac{k+1}{M}\right) u_e^n. \tag{A.4}$$

Finally, we predictions for second-stage data $\bar{\mathbf{h}}^{n,k+1/2}$ and $\bar{\mathbf{u}}^{n,k+1/2}$ for $k = 0, \cdots, M-1$. Starting with $\bar{\mathbf{h}}^{n,k+1/3}$, we insert the above prediction $\mathbf{h}^{n,k}$ into the second stage of FB-RK(3,2) with the fine time-step $\frac{\Delta t}{M}$ for $i \in C_P^{\text{IF-1}}$,

$$\bar{h}_i^{n,k+1/2} = h_i^{n,k} + \frac{\Delta t}{2M}\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/3}, \bar{\mathbf{h}}^{n,k+1/3}\right).$$

We need a way to approximate $\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/3}, \bar{\mathbf{h}}^{n,k+1/3}\right)$. Again using the assumption that we have Lipschitz continuity, we can use that $\Psi_i\left(\bar{\mathbf{u}}^{n,k+1/3}, \bar{\mathbf{h}}^{n,k+1/3}\right) = \Psi_i\left(\tilde{\mathbf{u}}^{n+1/3}, \tilde{\mathbf{h}}^{n+1/3}\right) + \mathcal{O}\left((\Delta t)^1\right)$. Insert this into the above and we get

$$\bar{h}_i^{n,k+1/2} = h_i^{n,k} + \frac{\Delta t}{2M}\left(\Psi_i\left(\tilde{\mathbf{u}}^{n+1/3}, \tilde{\mathbf{h}}^{n+1/3}\right) + \mathcal{O}\left((\Delta t)^1\right)\right)$$

$$= \left(\frac{k}{M}\tilde{h}_i^{n+1} + \left(1 - \frac{k}{M}\right) h_i^n + \mathcal{O}\left((\Delta t)^2\right)\right) + \frac{\Delta t}{2M}\frac{2(\tilde{h}_i^{n+1/2} - h_i^n)}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right)$$

$$= \frac{k}{M}\tilde{h}_i^{n+1} + \frac{1}{M}\tilde{h}_i^{n+1/2} + \left(1 - \frac{k+1}{M}\right) h_i^n + \mathcal{O}\left((\Delta t)^2\right).$$

A second-order approximation to stage-two $\mathbf{h}$ data for $i \in C_P^{\text{IF-1}}$ is given by

$$\bar{h}_i^{n,k+1/2} = \frac{k}{M}\tilde{h}_i^{n+1} + \frac{1}{M}\tilde{h}_i^{n+1/2} + \left(1 - \frac{k+1}{M}\right) h_i^n. \tag{A.5}$$

For $\bar{u}_e^{n,k1/2}$ for $e \in C_E^{\text{IF-1}}$, proceed similarly:

$$\bar{u}_e^{n,k+1/2} = u_e^{n,k} + \frac{\Delta t}{2M}\Phi_e\left(\bar{\mathbf{u}}^{n,k+1/3}, \mathbf{h}^{**,k}\right).$$

As we did to obtain the prediction for $\bar{h}_i^{n,k+1/2}$, we can approximate $\Phi_e\left(\bar{\mathbf{u}}^{n,k+1/3}, \mathbf{h}^{**,k}\right) = \Phi_e\left(\tilde{\mathbf{u}}^{n+1/3}, \mathbf{h}^{**}\right) + \mathcal{O}\left((\Delta t)^1\right)$. Insert this into the above to get

$$\begin{aligned}
\bar{u}_e^{n,k+1/2} &= u_e^{n,k} + \frac{\Delta t}{2M}\left(\Phi_e\left(\tilde{\mathbf{u}}^{n+1/3}, \mathbf{h}^{**}\right) + \mathcal{O}\left((\Delta t)^1\right)\right) \\
&= \left(\frac{k}{M}\tilde{u}^{n+1} + \left(1 - \frac{k}{M}\right)u^n + \mathcal{O}\left((\Delta t)^2\right)\right) + \frac{\Delta t}{2M}\frac{2(\tilde{u}_e^{n+1/2} - u_e^n)}{\Delta t} + \mathcal{O}\left((\Delta t)^2\right) \\
&= \frac{k}{M}\tilde{u}_e^{n+1} + \frac{1}{M}\tilde{u}_e^{n+1/2} + \left(1 - \frac{k+1}{M}\right)u_e^n + \mathcal{O}\left((\Delta t)^2\right).
\end{aligned}$$

A second-order approximation to stage-two $\mathbf{u}$ data for $e \in C_E^{\text{IF-1}}$ is given by

$$\bar{u}_e^{n,k+1/2} = \frac{k}{M}\tilde{u}_e^{n+1} + \frac{1}{M}\tilde{u}_e^{n+1/2} + \left(1 - \frac{k+1}{M}\right)u_e^n. \tag{A.6}$$

Taking all this together, the interface one prediction step is given by (18).

Appendix B. Operator splitting influence on speedup with LTS

One may observe that the speedup C (Fig. 9) of SplitFB-LTS versus SplitFB-RK(3,2) cannot be exclusively from the addition of local time-stepping. In fact, there is an additional source of speedup in play, originating from the operator splitting and the fact that the slow terms are calculated *globally* once per *coarse* time-step, resulting in extra savings when taking a longer coarse time-step with LTS.

To see this, we will use DelBay2km along with SplitFB-LTS and SplitFB-RK(3,2) as an example. For simplicity, say that the count ratio is 2, so that 1/3 of the mesh uses the fine time-step and 2/3 uses the coarse time-step. Say that SplitFB-LTS uses a coarse time-step Δt and that $M = 3$. Then, SplitFB-RK(3,2) uses a global time-step of $\frac{\Delta t}{3}$. In order for both methods to advance from time $t = 0$ to time $t = \Delta t$,

- SplitFB-LTS calculates
  - the slow tendencies once, globally,
  - the fast terms $3 \cdot 3 = 9$ times in the fine region (3 RK stages with $M = 3$),
  - the fast terms 3 times in the coarse region.
- SplitFB-RK(3,2) calculates
  - the slow tendencies 3 times, globally,
  - the fast tendencies $3 \cdot 3 = 9$ times, globally.

Let S and F represent the computational cost of calculating the fast and the slow tendencies respectively. Then, we can calculate the theoretical speedup as

$$\begin{aligned}
\frac{\text{SplitFB-RK(3,2) cost}}{\text{SplitFB-LTS cost}} &= \frac{3S + 9F}{\frac{1}{3}(S + 9F) + \frac{2}{3}(S + 3F)} \\
&= \frac{3S + 9F}{S + 5F}.
\end{aligned} \tag{B.1}$$

If we were to say that there was no splitting, that all terms were treated as fast, then $S = 0$ and we get the theoretical speedup from LTS as $9/5 = 1.8$. This does not align with the speedup of 2.6 we get from Table 2. From (B.1) we see that as the cost S increases relative to the cost F, the speedup obtained by SplitFB-LTS increases, explaining the extra speedup.

In fact, we can use (B.1) to estimate the cost of the slow terms relative to that of the fast terms. Setting $S = cF$, we can solve

$$\frac{3(cF) + 9F}{cF + 5F} = 2.6 \tag{B.2}$$

to get that $c = 10$. That is, the slow terms are approximately ten times as computationally expensive as the fast terms.

Data availability

All code is open source on GitHub. Code to setup and run test cases is also open source and available on GitHub. Links to relevant repositories are given in the 'Data Statement' section.

References

[1] Y. Accad, C.L. Pekeris, H. Jeffreys, Solution of the tidal equations for the M2 and S2 tides in the world oceans from a knowledge of the tidal potential alone, Philos. Trans. R. Soc. Lond. Ser. A, Math. Phys. Sci. 290 (1978) 235–266, https://doi.org/10.1098/rsta.1978.0083.

[2] Akio Arakawa, V.R. Lamb, Computational design of the basic dynamical processes of the UCLA general circulation model, in: Julius Chang (Ed.), General Circulation Models of the Atmosphere, in: Methods in Computational Physics: Advances in Research and Applications, vol. 17, Elsevier, 1977, pp. 173–265.

[3] B.K. Arbic, M.H. Alford, J.K. Ansong, M.C. Buijsman, R.B. Ciotti, J.T. Farrar, R.W. Hallberg, C.E. Henze, C.N. Hill, C.A. Luecke, D. Menemenlis, E.J. Metzger, M. Muller, A.D. Nelson, B.C. Nelson, H.E. Ngodock, R.M. Ponte, J.G. Richman, A.C. Savage, R.B. Scott, J.F. Shriver, H.L. Simmons, I. Souopgui, P.G. Timko, A.J. Wallcraft, L. Zamudio, Z. Zhao, Primer on global internal tide and internal gravity wave continuum modeling in HYCOM and MITgcm, New Front. Oper. Oceanogr. (2018) 307–392.

[4] K.N. Barton, N. Pal, S.R. Brus, M.R. Petersen, B.K. Arbic, D. Engwirda, A.F. Roberts, J.J. Westerink, D. Wirasaet, M. Schindelegger, Global barotropic tide modeling using inline self-attraction and loading in MPAS-Ocean, J. Adv. Model. Earth Syst. 14 (2022) e2022MS003207, https://doi.org/10.1029/2022MS003207.

[5] G. Capodaglio, M. Petersen, Local time stepping for the shallow water equations in MPAS, J. Comput. Phys. 449 (2022) 110818, https://doi.org/10.1016/j.jcp.2021.110818.

[6] M.M. Francois, A. Sun, W.E. King, N.J. Henson, D. Tourret, C.A. Bronkhorst, N.N. Carlson, C.K. Newman, T. Haut, J. Bakosi, J.W. Gibbs, V. Livescu, S.A. Vander Wiel, A.J. Clarke, M.W. Schraad, T. Blacker, H. Lim, T. Rodgers, S. Owen, F. Abdeljawad, J. Madison, A.T. Anderson, J.L. Fattebert, R.M. Ferencz, N.E. Hodge, S.A. Khairallah, O. Walton, Modeling of additive manufacturing processes for metals: challenges and opportunities, Curr. Opin. Solid State Mater. Sci. 21 (2017) 198–206, https://doi.org/10.1016/j.cossms.2016.12.001.

[7] J.R. Garratt, Review of drag coefficients over oceans and continents, Mon. Weather Rev. 105 (1977) 915–929, https://doi.org/10.1175/1520-0493(1977)105<0915:RODCOO>2.0.CO;2.

[8] J.C. Golaz, L.P. Van Roekel, X. Zheng, A.F. Roberts, J.D. Wolfe, W. Lin, A.M. Bradley, Q. Tang, M.E. Maltrud, R.M. Forsyth, C. Zhang, T. Zhou, K. Zhang, C.S. Zender, M. Wu, H. Wang, A.K. Turner, B. Singh, J.H. Richter, Y. Qin, M.R. Petersen, A. Mametjanov, P.L. Ma, V.E. Larson, J. Krishna, N.D. Keen, N. Jeffery, E.C. Hunke, W.M. Hannah, O. Guba, B.M. Griffin, Y. Feng, D. Engwirda, A.V. Di Vittorio, C. Dang, L.M. Conlon, C.C.J. Chen, M.A. Brunke, G. Bisht, J.J. Benedict, X.S. Asay-Davis, Y. Zhang, M. Zhang, X. Zeng, S. Xie, P.J. Wolfram, T. Vo, M. Veneziani, T.K. Tesfa, S. Sreepathi, A.G. Salinger, J.E.J. Reeves Eyre, M.J. Prather, S. Mahajan, Q. Li, P.W. Jones, R.L. Jacob, G.W. Huebler, X. Huang, B.R. Hillman, B.E. Harrop, J.G. Foucar, Y. Fang, D.S. Comeau, P.M. Caldwell, T. Bartoletti, K. Balaguru, M.A. Taylor, R.B. McCoy, L.R. Leung, D.C. Bader, The DOE E3SM model version 2: overview of the physical model and initial model evaluation, J. Adv. Model. Earth Syst. 14 (2022) e2022MS003156, https://doi.org/10.1029/2022MS003156.

[9] R.L. Higdon, A two-level time-stepping method for layered ocean circulation models: further development and testing, J. Comput. Phys. 206 (2005) 463–504, https://doi.org/10.1016/j.jcp.2004.12.011.

[10] T.T.P. Hoang, W. Leng, L. Ju, Z. Wang, K. Pieper, Conservative explicit local time-stepping schemes for the shallow water equations, J. Comput. Phys. 382 (2019) 152–176, https://doi.org/10.1016/j.jcp.2019.01.006.

[11] M.J. Hoffman, M. Perego, S.F. Price, W.H. Lipscomb, T. Zhang, D. Jacobsen, I. Tezaur, A.G. Salinger, R. Tuminaro, L. Bertagna, MPAS-Albany Land Ice (MALI): a variable-resolution ice sheet model for Earth system modeling using Voronoi grids, Geosci. Model Dev. 11 (2018) 3747–3780, https://doi.org/10.5194/gmd-11-3747-2018.

[12] S.R. Jayne, L.C. St. Laurent, Parameterizing tidal dissipation over rough topography, Geophys. Res. Lett. 28 (2001) 811–814, https://doi.org/10.1029/2000GL012044.

[13] L. Ju, T. Ringler, M. Gunzburger, Voronoi tessellations and their application to climate and global modeling, in: P. Lauritzen, C. Jablonowski, M. Taylor, R. Nair (Eds.), Numerical Techniques for Global Atmospheric Models, in: Lecture Notes in Computational Science and Engineering, Springer, Berlin, Heidelberg, 2011, pp. 313–342.

[14] E. Larour, H. Seroussi, M. Morlighem, E. Rignot, Continental scale, high order, high spatial resolution, ice sheet modeling using the Ice Sheet System Model (ISSM), J. Geophys. Res., Earth Surf. 117 (2012), https://doi.org/10.1029/2011JF002140.

[15] J.R. Lilly, G. Capodaglio, M.R. Petersen, S.R. Brus, D. Engwirda, R.L. Higdon, Storm surge modeling as an application of local time-stepping in MPAS-Ocean, J. Adv. Model. Earth Syst. 15 (2023) e2022MS003327, https://doi.org/10.1029/2022MS003327.

[16] J.R. Lilly, D. Engwirda, G. Capodaglio, R.L. Higdon, M.R. Petersen, CFL optimized forward–backward Runge–Kutta schemes for the shallow-water equations, Mon. Weather Rev. 151 (2023) 3191–3208, https://doi.org/10.1175/MWR-D-23-0113.1.

[17] C.B. Marsh, J.W. Pomeroy, R.J. Spiteri, H.S. Wheater, A finite volume blowing snow model for use with variable resolution meshes, Water Resour. Res. 56 (2020) e2019WR025307, https://doi.org/10.1029/2019WR025307.

[18] A. Okabe, Spatial tessellations, in: International Encyclopedia of Geography, John Wiley & Sons, Ltd., 2017, pp. 1–11.

[19] M.R. Petersen, X.S. Asay-Davis, A.S. Berres, Q. Chen, N. Feige, M.J. Hoffman, D.W. Jacobsen, P.W. Jones, M.E. Maltrud, S.F. Price, T.D. Ringler, G.J. Streletz, A.K. Turner, L.P. Van Roekel, M. Veneziani, J.D. Wolfe, P.J. Wolfram, J.L. Woodring, An evaluation of the ocean and sea ice climate of E3SM using MPAS and interannual CORE-II forcing, J. Adv. Model. Earth Syst. 11 (2019) 1438–1458, https://doi.org/10.1029/2018MS001373.

[20] T. Ringler, M. Petersen, R.L. Higdon, D. Jacobsen, P.W. Jones, M. Maltrud, A multi-resolution approach to global ocean modeling, Ocean Model. 69 (2013) 211–232, https://doi.org/10.1016/j.ocemod.2013.04.010.

[21] T.D. Ringler, J. Thuburn, J.B. Klemp, W.C. Skamarock, A unified approach to energy conservation and potential vorticity dynamics for arbitrarily-structured C-grids, J. Comput. Phys. 229 (2010) 3065–3090, https://doi.org/10.1016/j.jcp.2009.12.007.

[22] L.J. Wicker, W.C. Skamarock, Time-splitting methods for elastic models using forward time schemes, Mon. Weather Rev. 130 (2002) 2088–2097, https://doi.org/10.1175/1520-0493(2002)130<2088:TSMFEM>2.0.CO;2.

[23] Z. Xu, A. Di Vittorio, Hydrological analysis in watersheds with a variable-resolution global climate model (VR-CESM), J. Hydrol. 601 (2021) 126646, https://doi.org/10.1016/j.jhydrol.2021.126646.

[24] Y. Zhou, Y. Zhang, J. Li, R. Yu, Z. Liu, Configuration and evaluation of a global unstructured mesh atmospheric model (GRIST-A20.9) based on the variable-resolution approach, Geosci. Model Dev. 13 (2020) 6325–6348, https://doi.org/10.5194/gmd-13-6325-2020.