JAMES | Journal of Advances in Modeling Earth Systems

RESEARCH ARTICLE
10.1029/2022MS003327

Key Points:

- Storm surge modeling of Hurricane Sandy around Delaware Bay is used to demonstrate variable-resolution ocean time-stepping methods
- Local time-stepping schemes are up to 35% faster, and produce solutions of comparable quality to higher-order globally uniform schemes

Correspondence to:

J. R. Lilly,
lillyj@oregonstate.edu

Citation:

Lilly, J. R., Capodaglio, G., Petersen, M. R., Brus, S. R., Engwirda, D., & Higdon, R. L. (2023). Storm surge modeling as an application of local time-stepping in MPAS-Ocean. *Journal of Advances in Modeling Earth Systems*, *15*, e2022MS003327. https://doi.org/10.1029/2022MS003327

Received 4 AUG 2022
Accepted 10 JAN 2023

Author Contributions:

Conceptualization: Jeremy R. Lilly, Giacomo Capodaglio, Mark R. Petersen, Darren Engwirda, Robert L. Higdon
Data curation: Jeremy R. Lilly, Steven R. Brus
Formal analysis: Jeremy R. Lilly, Giacomo Capodaglio
Funding acquisition: Jeremy R. Lilly, Giacomo Capodaglio, Mark R. Petersen, Darren Engwirda
Investigation: Jeremy R. Lilly, Giacomo Capodaglio
Methodology: Jeremy R. Lilly, Giacomo Capodaglio, Mark R. Petersen
Project Administration: Giacomo Capodaglio, Mark R. Petersen, Darren Engwirda
Resources: Giacomo Capodaglio, Mark R. Petersen, Steven R. Brus

Storm Surge Modeling as an Application of Local Time-Stepping in MPAS-Ocean

Jeremy R. Lilly[1], Giacomo Capodaglio[2], Mark R. Petersen[2], Steven R. Brus[3], Darren Engwirda[4], and Robert L. Higdon[1]

[1]Department of Mathematics, Oregon State University, Corvallis, OR, USA, [2]Computational Physics and Methods Group, Los Alamos National Laboratory, Los Alamos, NM, USA, [3]Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL, USA, [4]Fluid Dynamics and Solid Mechanics Group, Los Alamos National Laboratory, Los Alamos, NM, USA

Abstract This paper presents the first practical application of local time-stepping (LTS) schemes in the Model for Prediction Across Scales-Ocean (MPAS-O). We use LTS schemes in a single-layer, global ocean model that predicts the storm surge around the eastern coast of the United States during Hurricane Sandy. The variable-resolution meshes used are of unprecedentedly high resolution in MPAS-O, containing cells as small as 125 m wide in Delaware Bay. It is shown that a particular, third-order LTS scheme (LTS3) produces sea-surface height solutions that are of comparable quality to solutions produced by the classical four-stage, fourth-order Runge-Kutta method (RK4) with a uniform time step on the same meshes. Furthermore, LTS3 is up to 35% faster in the best cases considered, where the number of cells using the coarse time-step relative to those using the fine time-step is as low as 1:1. This shows that LTS schemes are viable for use in MPAS-O with the added benefit of substantially less computational cost. The results of these performance experiments inform us of the requirements for efficient mesh design and configuration of LTS regions for LTS schemes. In particular, we see that for LTS to be efficient on a given mesh, it is important to have enough cells using the coarse time-step relative to those using the fine time-step, typically at least 1:5 to see an increase in performance.

Plain Language Summary In many applications of modern ocean models, it is useful to partition the globe into cells of different sizes, depending on the level of accuracy desired in a given region. One uses smaller cells in regions where more accuracy is desired, and larger cells elsewhere in order to save on computational costs; such partitions are generally called variable-resolution meshes. A well known limitation of explicit time-stepping methods, used to advance the temporal state of the model, is that the largest step forward in time the model can take is limited by the size of the smallest cell in the mesh. Global time-stepping schemes use a given time-step, whose size is determined by the size of the smallest cell in the mesh, everywhere on the mesh. Local time-stepping (LTS) methods, allow us to select multiple time-steps based on the size of cells in a localized region. Here, we use LTS schemes to model the storm surge around the eastern US coast during Hurricane Sandy in 2012. We show that a particular LTS scheme produces sea-surface height predictions of comparable quality to those produced by a state-of-the-art global time-stepping method and that LTS is up to 35% faster in the best cases.

1. Introduction

Variable-resolution climate models have become increasingly popular in the past decade (e.g., Skamarock et al., 2012; Danilov et al., 2017; Korn, 2017), as they offer the ability to create high-resolution regions within global domains, with fine control over the extent and transitions in grid-cell size. It is well known that the size of the largest time-step that can be used by an explicit time-stepping scheme is bounded above by the size of the smallest cell in the mesh according to the Courant–Friedrichs–Lewy (CFL) condition. This restriction is of particular interest on meshes where the cell size varies greatly. To optimize the computational cost of running a model on a variable-resolution mesh, one would like to select a small time-step on regions of high resolution (regions defined by *small* cells), and a large time-step on regions of low resolution (defined by *large* cells). For simplicity, variable-resolution climate model components typically use global time-stepping schemes, where a uniform time-step is used on the entire computational domain. As a result, one is forced to use a small time-step that is restricted by the CFL condition influenced by the smallest cell in the mesh even on large cells that would admit a larger time-step in the absence of smaller cells. This approach results in unnecessary computational cost

Software: Jeremy R. Lilly, Giacomo Capodaglio, Steven R. Brus, Darren Engwirda
Supervision: Giacomo Capodaglio, Mark R. Petersen, Darren Engwirda, Robert L. Higdon
Validation: Jeremy R. Lilly, Giacomo Capodaglio, Mark R. Petersen
Visualization: Jeremy R. Lilly, Giacomo Capodaglio, Steven R. Brus, Darren Engwirda
Writing – original draft: Jeremy R. Lilly, Giacomo Capodaglio
Writing – review & editing: Jeremy R. Lilly, Giacomo Capodaglio, Mark R. Petersen, Steven R. Brus, Darren Engwirda, Robert L. Higdon

on low-resolution cells. Local time-stepping (LTS) schemes can combat this performance bottle-neck by allowing the selection of different time-steps on different regions of the mesh according to local CFL conditions instead of a single global one.

In this paper, we present the first practical application of LTS in the framework of the Model for Prediction Across Scales-Ocean (MPAS-O) (Ringler et al., 2013) by modeling the storm surge caused by Hurricane Sandy on the eastern coast of the United States. MPAS-O is a global ocean model being developed at Los Alamos National Laboratory (LANL) as a part of the Department of Energy's Energy Exascale Earth System Model (E3SM) (Golaz et al., 2019; Petersen et al., 2019). MPAS-O is a multi-layer, primitive equation global ocean model that uses the TRiSK scheme for spatial discretization (Ringler et al., 2010; Thuburn et al., 2009), which is a staggered Arakawa C-grid method (Arakawa & Lamb, 1977) defined on a Voronoi tessellation (Ju et al., 2011; Okabe, 2017). The vertical coordinates are treated with an Arbitrary Lagrangian-Eulerian (ALE) framework, as detailed in Petersen et al. (2015). A feature of MPAS-O that is of particular interest to this work is the ability to run global ocean simulations on unstructured meshes of variable resolution (Hoch et al., 2020), that is, computational cells of different sizes can be used on specific regions of the globe depending on the desired degree of spatial accuracy. For storm surge modeling and for the accurate simulation of coastal processes in general, variable resolution meshes with high resolution on the coast and low resolution in the deep ocean are the obvious choice to maximize computational performance while at the same time maintaining the accuracy of the physical predictions (Mandli & Dawson, 2014; Pringle et al., 2021).

Several LTS methods have been investigated in the literature for a broad span of applications, and a comprehensive overview of these methods is beyond the scope of this work. Examples of such applications include solving Maxwell's equations in Montseny et al. (2008) and the wave equation in Diaz and Grote (2009), and problems in computational aeroacoustics in Liu et al. (2010). In Trahan and Dawson (2012) and Dawson et al. (2013), a first-order LTS scheme has been used to solve the shallow-water equations and for storm surge modeling using a model similar to the one considered in this work. The analysis in Dawson et al. (2013) shows that their LTS scheme produces results comparable to that of a global second order scheme, in about half the computational time. A difference between Dawson et al. (2013) and the present work is that a discontinuous Galerkin discretization was considered in the former, whereas MPAS-O relies on the TRiSK scheme. Moreover, our choice of LTS scheme is not the same as in Dawson et al. (2013), but is the third order method developed in Hoang et al. (2019). The schemes in Hoang et al. (2019) are particularly relevant to the MPAS framework as they were developed specifically to solve the shallow water equations (SWEs) using TRiSK horizontal discretizations. These time-stepping schemes have been implemented by the authors in the shallow water core of MPAS-O and it has been shown that they can solve the SWEs up to 70% faster than higher-order global methods (Capodaglio & Petersen, 2022).

The focus of the current work is to explore the use of LTS in MPAS-O via a real-world application, with the goal of showing that LTS can produce virtually the same numerical solution as higher-order global methods in considerably less time. Specifically, we use LTS to model the storm surge in the region of Delaware Bay during Hurricane Sandy and consider meshes with regions of very high resolution, such as the one in Figure 1. This particular mesh contains cells with 2 km width in Delaware Bay and a global background resolution of 120 km, but the meshes that will be used throughout this work have unprecedentedly high resolution near the coast, with cell widths as small as 125 m on Delaware Bay. We are able to run at such high resolutions by using a single-layer ocean model, as opposed to the multi-layer model, with 60–80 layers, that is the default for MPAS-O. The choice of a single-layer model for this study was motivated by two reasons. First, the processes relevant to a storm surge model are essentially barotropic in nature, so a single layer is a good approximation of the physics involved. Second, the LTS schemes developed in Hoang et al. (2019) were not developed for a layered model and as such do not take into account vertical transport between layers; work is currently underway to adapt these schemes to a layered model.

The paper is structured as follows. We begin by describing the Hurricane Sandy test case and model configuration. We then give a brief background on the LTS schemes developed in Hoang et al. (2019) and a discussion of the inherent challenges of configuring a mesh for running with LTS. Next, we provide an in-depth description of a set of meshes of increasingly high resolution on which we run our Hurricane Sandy simulations. Finally, we compare the performance in terms of CPU-time of LTS scheme of order three in Hoang et al. (2019) (LTS3) to the classical fourth-order, four-stage Runge-Kutta method (RK4) and show that LTS3 can obtain a sea-surface height solution substantially faster than RK4, with minimal differences in the quality of the prediction.

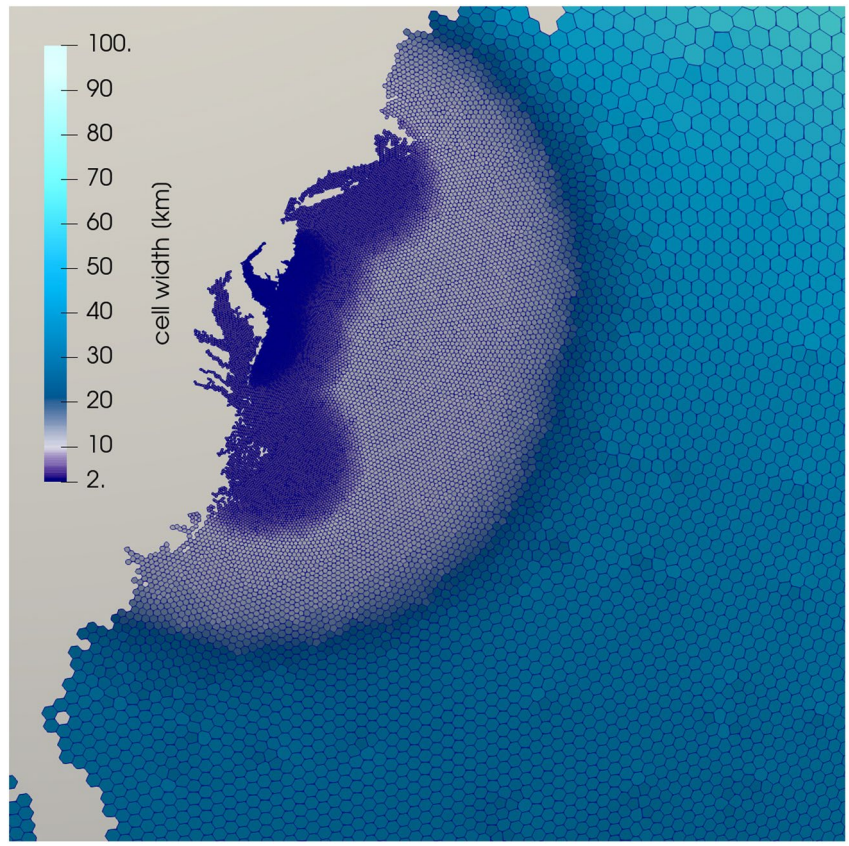


Figure 1. Model for Prediction Across Scales variable-resolution mesh on the East Coast of the US, with 2 km resolution in Delaware Bay.

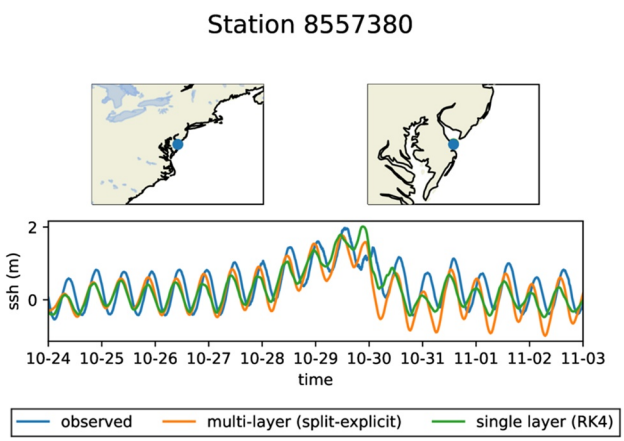


Figure 2. Sea-surface height during Hurricane Sandy as predicted by a multi-layer ocean model and single-layer model considered here compared to observed data on the DelBay2km mesh (see Table 1). Split-explicit refers to the first-order, explicit, split barotropic-baroclinic time-stepping scheme that is the method of choice for a multi-layer model in MPAS-O (Ringler et al., 2013).

2. Methods

2.1. Hurricane Sandy Test Case

The physical processes involved in the storm surge created by a hurricane are essentially barotropic in nature and can therefore be modeled by a single-layer ocean, the most important physical processes being forcing due to tidal cycles, surface wind stress, and atmospheric pressure (Mandli & Dawson, 2014; Pringle et al., 2021). Historically, MPAS-O has been used primarily for long-term (on the order of hundreds of years) climate modeling; such models require a multi-layered ocean that can resolve both barotropic and baroclinic motions. As such, it is uncommon for MPAS-O to be run in a single layer configuration and only recently was the ability to compute tidal forcing added to the model (Barton et al., 2022).

Figure 2 shows observed sea-surface height (SSH) at a particular tidal gauge versus time compared to the SSH predicted by MPAS-O in a multi-layer configuration and a single-layer configuration. In Figure 2, the multi-layered model uses a barotropic-baroclinic split-explicit time-stepping scheme (Higdon, 2005), and the single-layer model uses RK4. This figure shows that both the single-layer and multi-layer models include the relevant physical processes of tidal forcing, which produces semidiurnal oscillations, and the surface winds and atmospheric pressure, which produce the storm surge.

Previous work on LTS in the MPAS framework has been done exclusively in the shallow-water core of MPAS-O. In the single layer configuration considered here, the ocean core differs from the shallow-water core in having

realistic bathymetry, realistic coastlines, and additional forcing terms necessary to simulate the storm surge, all of which are absent from the shallow-water code base. The affinity between the shallow-water core of MPAS and the present model, and the need for high local resolution (generally with cells smaller than 1–5 km in width) for the accurate prediction of sea-surface height in coastal regions, make storm surge modeling a good real-world test-case for LTS.

The momentum and thickness equations for the Hurricane Sandy model are given by Equation 1. 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.

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\nabla \times \mathbf{u} + f\mathbf{k}) \times \mathbf{u} = & -\nabla K - \dfrac{1}{\rho_0}\nabla p^s - g\nabla(\eta - \eta_{EQ} - \beta\eta) \\ & -\chi \dfrac{C\mathbf{u}}{H} - C_D \dfrac{|\mathbf{u}|\mathbf{u}}{h} + C_W \dfrac{|\mathbf{u}_W - \mathbf{u}|(\mathbf{u}_W - \mathbf{u})}{h}, \\ \dfrac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{u}) = 0. \end{cases} \tag{1}$$

In these equations $\mathbf{u}$ is the horizontal 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, ρ_0 is the (constant) fluid density, p^s is the surface pressure, g is the gravitational constant, η is the sea-surface height perturbation, η_{EQ} is the sea-surface height perturbation due to equilibrium tidal forcing (Arbic et al., 2018), β is the self-attraction and loading coefficient (Accad et al., 1978), $\frac{C}{H}$ is a spatially varying internal tide dissipation coefficient (Jayne & St. Laurent, 2001), χ is a scalar tuning factor optimized for barotropic tides response (Barton et al., 2022). H is the resting depth of the ocean, h is the total ocean thickness such that $h = H + \eta$, C_D is the bottom drag coefficient, C_W is the wind stress coefficient, and $\mathbf{u}_W$ is the horizontal wind velocity.

The wind velocity $\mathbf{u}_W$ and the atmospheric pressure p^s are linearly interpolated from data observed at 1-hr increments between 10 October 2012 and 11 March 2012. Hurricane Sandy hit the Eastern US coast on 29 October 2012. The wind and atmospheric surface pressure data were obtained from the Climate Forecast System Version 2 (CFSv2) reanalysis product (Saha et al., 2014). Our bathymeteric data set is SRTM15+ (Tozer et al., 2019). The equilibrium tidal potential η_{EQ} is computed with the 8 major tidal constituents M_2, S_2, N_2, K_2, K_1, O_1, Q_1, and P_1.

When running the model, we start with initial conditions of zero for both $\mathbf{u}$ and η everywhere on the globe and let model spin-up for ten simulated days while the primary forcing is due to winds and tides. For further model configuration details, see the Data Availability Statement at the end of this manuscript.

The model equations are spatially discretized using a C-grid type finite volume method called the TRiSK scheme wherein the fluid thickness h is stored at cell centers and the normal component of the fluid velocity $\mathbf{u}$ is computed at cell edges. This method has been shown to conserve the total energy of a system and robustly simulates potential vorticity (Ringler et al., 2010; Thuburn et al., 2009). The current code configuration was chosen to show the performance improvements possible with LTS. Some processes, like wetting and drying cells and spatially varying bottom friction, were not included here because they are still under development. Because a wetting and drying scheme is not used here, we avoid the possibility of drying cells by setting the minimum value for the depth to two m. It is likely that omission of such a scheme has a negative effect on the accuracy of our model as compared to observational data. Similarly, it is likely that a working implementation of spatially varying bottom friction would improve comparisons to observed data. However, we are primarily concerned with the performance of LTS3 and its accuracy compared to that of well accepted global schemes, which are not affected by these omissions.

2.2. Time-Stepping Schemes

The particular LTS scheme used here is based on so-called, strong stability preserving Runge-Kutta (SSPRK) methods, which are explicit methods for solving systems of ODEs resulting from the spatial discretization of hyperbolic conservation laws. SSPRK methods satisfy the total variation diminishing (TVD) property, which means that given a sufficiently small time-step, the total spatial variation of an SSPRK solution will not increase over time, which implies that the solution will remain stable. For a more complete discussion of SSPRK methods, see Gottlieb and Shu (1998) and Gottlieb et al. (2001).

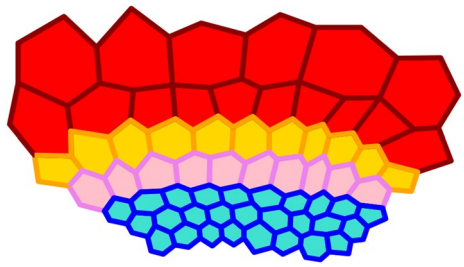


Figure 3. A mesh with cells labeled for local time-stepping. Red cells are coarse, yellow cells are interface 2, pink cells are interface 1, and blue are fine.

The LTS method used here is derived from a three-stage, third-order SSPRK method (SSPRK3) and is itself third-order; we refer to this method as LTS3. LTS3 was originally developed for use in the MPAS framework by Hoang et al. (2019), then implemented in the MPAS framework by Capodaglio and Petersen (2022). While a full review of the LTS3 scheme is outside the scope of this paper, and can be found in Hoang et al. (2019) and Capodaglio and Petersen (2022), we give a short description of the scheme for completeness. Consider Figure 3, which serves as a hypothetical mesh wherein there are four classes of cells. The red cells are referred to as the coarse cells, the yellow as interface 2 cells, the pink as interface 1 cells, and the blue as fine cells. The coarse, interface 1, and interface 2 cells all advance with the coarse time-step Δt_{coarse} and the fine cells advance with the fine time-step Δt_{fine} such that $\Delta t_{\text{fine}} = \frac{\Delta t_{\text{coarse}}}{M}$ for some positive integer M. For practical applications, the labels *fine* and *coarse* denote the regions by their cell size, but the algorithm simply advances two different time steps in the two regions, so the regions could have other cell sizes—for example, identically sized cells for testing purposes.

The LTS3 scheme proceeds as follows.

1. Starting from time t^n, advance the solution on the coarse, interface 1, and interface 2 cells with the first two stages of SSPRK3 using Δt_{coarse}.
2. Sub-cycle on the fine cells; advance to t^{n+1} by repeating all three stages of SSPRK3 M times using Δt_{fine}.
   - This requires that we know values for $\mathbf{u}$ and h at intermediate time-levels on interface 1 cells. Obtain predicted values for these with an appropriate prediction step using data from time t^n and the first two stages of SSPRK3 that were obtained in step 1.
3. Advance to time t^{n+1} on the coarse cells with the final stage of SSPRK3.
4. Correct the values for $\mathbf{u}$ and h at time t^{n+1} on interface 1 and interface 2 cells by accounting for fluxes coming from the fine cells during the sub-cycling in step 2.

For brevity, the above description omits several delicate parts of the scheme. First, note that in step 1, to advance with the second stage of SSPRK3 on the interface cells that border the fine region, we need the first stage SSPRK data on the three layers of fine cells adjacent to the interface region. Three layers are required here because the spatial discretization of the model equations mandates that a given cell's advancement depends on the three layers of adjacent cells. This data is not used to advance the fine cells themselves, however Δt_{coarse} is employed in this computation. Although we did not carry out an in-depth analysis to confirm this, it is likely that the interface-adjacent fine cells need to be large enough to admit the coarse time-step. In step 2, the prediction of values for $\mathbf{u}$ and h at intermediate time-levels on interface 1 cells is obtained via a truncated Taylor expansion (in t) centered at time t^n, where the unknown derivatives are estimated by known values of existing data from t^n and the first two stages of SSPRK3. We refer the reader to Hoang et al. (2019) for a full description of this step.

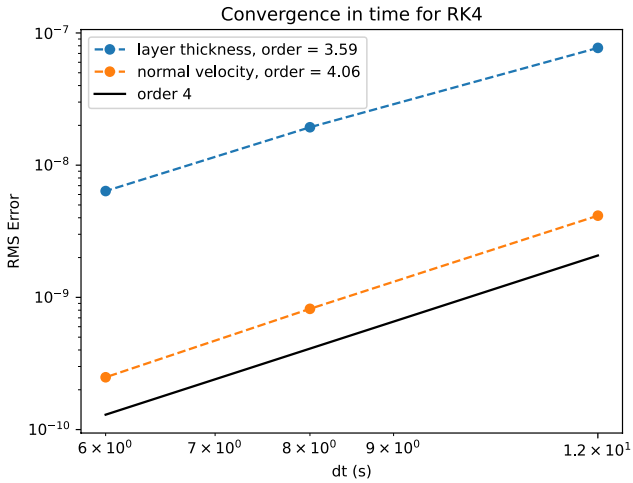
LTS3 is $\mathcal{O}\left((\Delta t)^3\right)$ everywhere, including on the interface layers. Figures 4a and 4b show the convergence in time for both $\mathbf{u}$ and h using RK4 and LTS3 respectively, on the spatially discretized versions of the model Equation 1. The root-mean-square (RMS) error is defined as

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

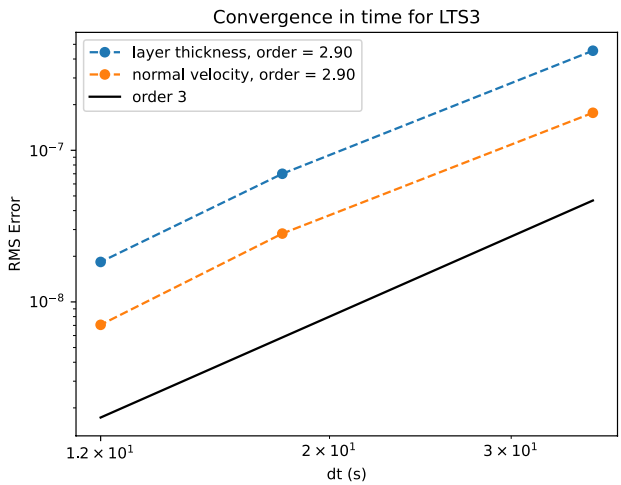
where $\{s_i\}_{i=1}^{N}$ is the discrete reference solution, $\{m_i\}_{i=1}^{N}$ is the discrete model solution, and N is the number of discretization points.

2.3. Meshes and LTS Configuration

Here we consider five different meshes; the relevant parameters for each are given in Table 1. Of particular interest is the physical distribution of regions of low and high resolutions. Each mesh is divided into five regions, each of which is populated by cells of a different size, with smooth transitions in resolution between regions. These regions are Delaware Bay, the area around the Delaware coast, the area around the eastern coast of the US, the western Atlantic ocean, and the rest of the globe. In the *Resolutions* section of Table 1, we give the cell width in kilometers for each of these five regions, for all meshes considered. In particular, note that the highest resolutions



(a)



(b)

Figure 4. Temporal convergence for layer thickness h and normal velocity $\mathbf{u}$ and using RK4 (a) and LTS3 (b) on the discretized version of the model Equation 1. Convergence tests were conducted on the DelBay2km mesh (see Table 1 and Figure 5). For LTS3, we used $M = 4$ and the values on the horizontal axis are the coarse time-steps. The errors were calculated against a reference solution generated by RK4 with $\Delta t = 0.1$. The run duration is 1 hr. The RMS error is as defined in Equation 2.

in these meshes range from 2 km down to 125 m and the ratio of the lowest to highest resolution ranges from 60 to 240. It should be noted that even our lowest resolution mesh using 2 km cells in Delaware Bay is of extremely high resolution for MPAS-O.

All meshes were created with JIGSAW (Engwirda, 2017), a library that can quickly generate high-quality variable-resolution meshes. Global meshes are designed by specifying the cell width distribution across the sphere (Figure 5), which is then passed into JIGSAW. The five regions of the mesh were used as a straightforward means of providing regional refinement in the Delaware Bay estuary within a global mesh. The Delaware Bay region is the highest level of refinement, while the Delaware coast, and eastern US coast regions provide a smooth transition to the coarser global resolution. The western Atlantic Ocean region uses a higher level of refinement than the global ocean to better resolve the hurricane wind and pressure fields, while the rest of the Earth has the coarsest resolution to reduce the total number of cells in the mesh.

2.3.1. Choice of Time-Steps

The time-steps used on each mesh and each LTS configuration for performance tests are given in Table 1. These time-steps were obtained experimentally and are the largest values that are admissible to guarantee stability of the model.

Because there are two different time-steps for the LTS scheme and these must differ by a factor of M such that $\Delta t_{\text{fine}} = \frac{\Delta t_{\text{coarse}}}{M}$, 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 gave an admittable fine time-step, or first find the largest admittable fine time-step 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.

There are advantages to both methods that depend on the mesh itself. For instance, on a mesh where there are sufficiently more coarse cells than fine cells, one might wish to first maximize the time-step in the coarse region and take a small penalty to the size of the fine time-step. In this paper however, we have opted to find admittable time-steps by first maximizing the fine time-step, then by finding the largest value of M that gives a valid coarse time-step with the intention on minimizing the work done in the fine region.

A primary concern of this work is to compare the efficiency, in terms of CPU-time, of LTS3 and RK4. One should note that RK4 uses one more Runge-Kutta stage per time-step than LTS3 and therefore has a lessened restriction on the size of the largest time-step it can use on high-resolution cells, for example, see Table 1 and note that on the mesh labeled DelBay2km, RK4 uses a global time-step of 30 s, while LTS3 uses a fine time-step of 18 s. Consider that, on the smallest cells, RK4 advances on average $\frac{30}{4} = 7.5$ seconds per stage while LTS3 advances on average $\frac{18}{3} = 6$ seconds per stage, which means that RK4 would be 25% faster if LTS3 were to use a global time-step. This puts LTS3 at an inherent disadvantage when comparing computational performance on these meshes. Nevertheless, we will show that LTS3 is still capable of considerably outperforming RK4 in almost all of our case studies. We have chosen to use RK4 as our point of comparison rather than a third order method because RK4 is the state-of-the-art scheme in MPAS-O for a single-layer model.

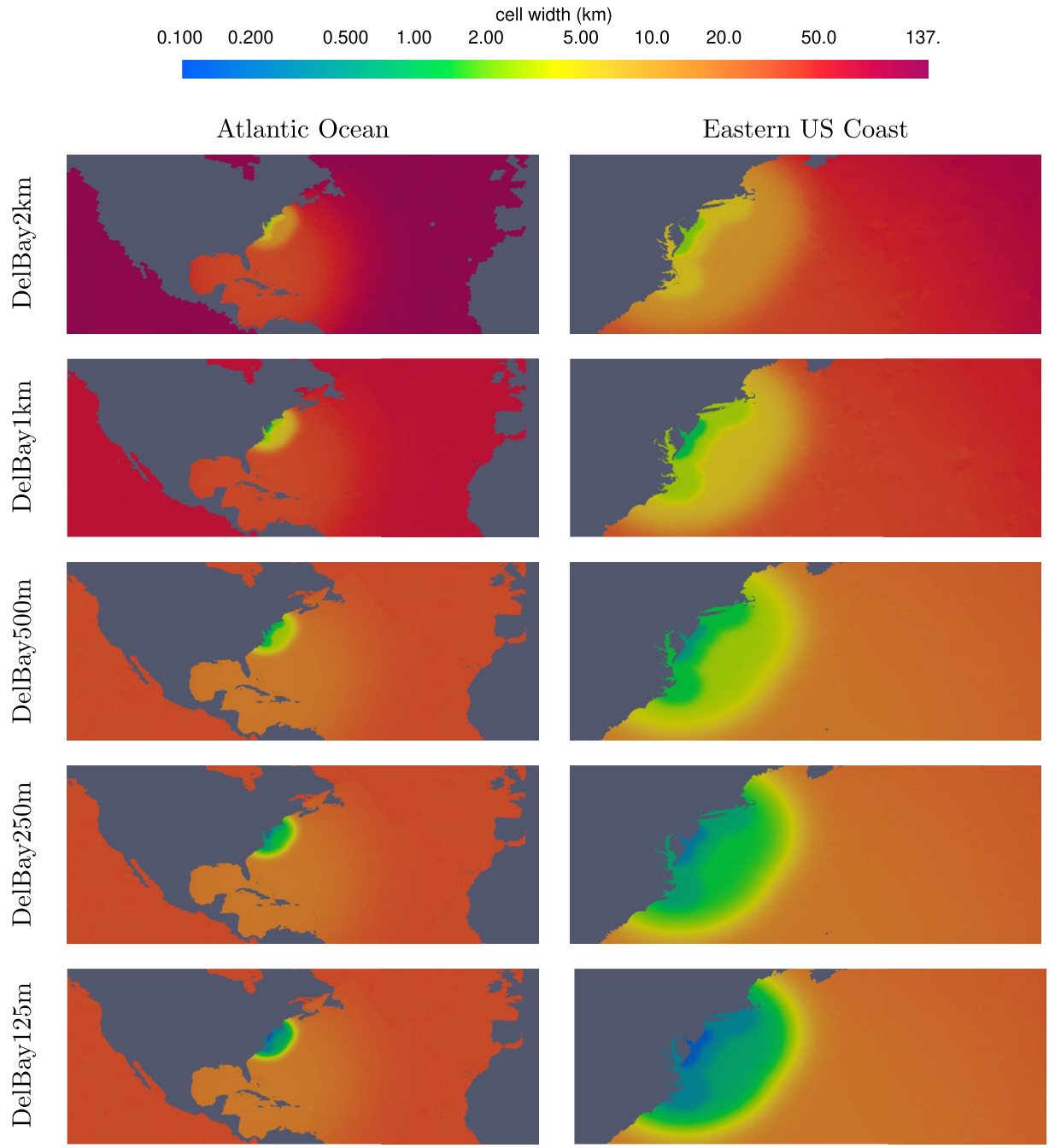


Figure 5. Global cell width in kilometers for each mesh described in Table 1. Each subplot shares the same color scale, which is in log-space, so differences in high-resolution areas can be easily distinguished.

2.3.2. LTS Mesh Parameters

When designing a mesh for use with LTS there are two parameters of particular interest, which we refer to as the *count ratio* and the *resolution ratio*. Both of these parameters depend on the cells where the fine time-step is used, referred to as the fine cells, and the cells where the coarse time-step is used, referred to as the coarse cells. Note that, in this discussion, the label of fine or coarse does not necessarily reference the size of a cell, 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 globe is called the coarse region.

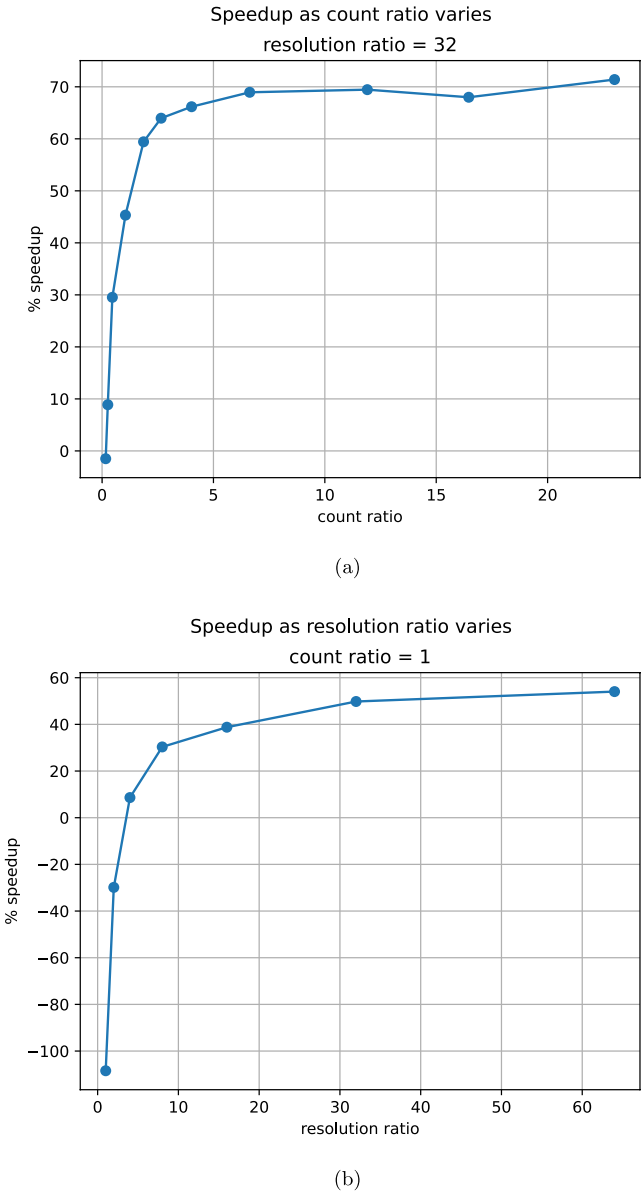


Figure 6. Speedup versus mesh design parameters: (a) varying count ratio while resolution ratio is held constant at 32; (b) varying resolution ratio as count ratio is held near one. In both cases, speedup increases and then levels off at 50%–70%. Speedup is calculated as in Equation 4.

The count ratio is the ratio of the number of cells in the coarse region to the number of cells in the fine region, that is,

$$\text{count ratio} = \frac{\text{number of coarse time-step cells}}{\text{number of fine time-step cells}}.$$

The resolution ratio is the ratio of the cell width of the coarse cells to the cell width of the fine cells, that is,

$$\text{resolution ratio} = \frac{\text{cell width of coarse time-step cells}}{\text{cell width of fine time-step cells}}.$$

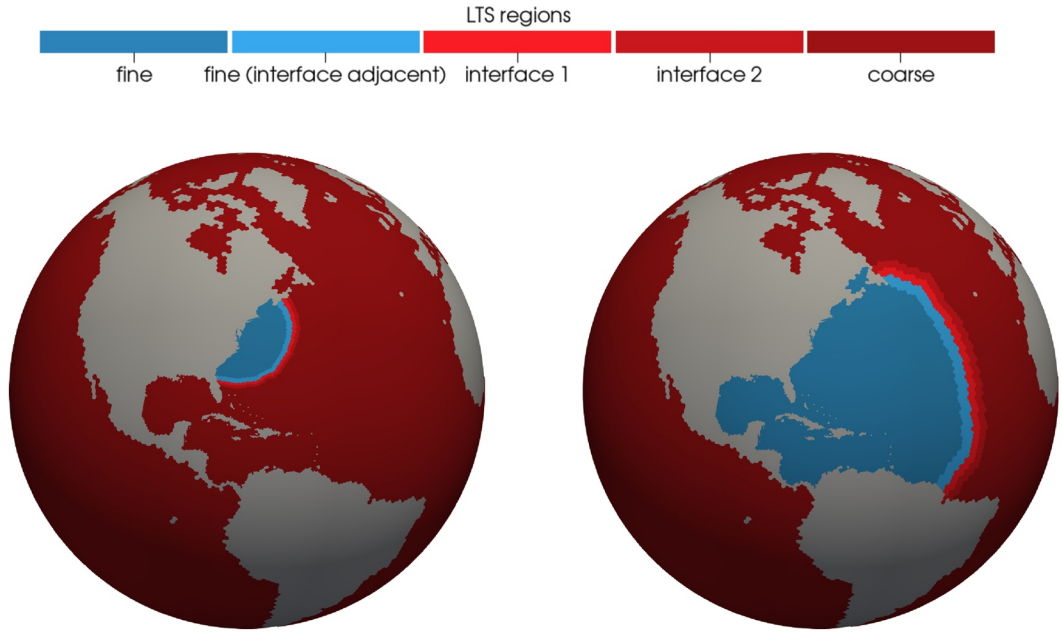
In the case where there are cells of multiple resolutions in either region, as is the case in our meshes, we consider the smallest value of cell width in a given region as it is the smallest cell that restricts the size of the time-step admittable in that region. For example, in DelBay2km with the EC configuration the smallest fine cells have a cell width of 2 km and the smallest coarse cells have a cell width of 30 km so we would say that the resolution ratio was 15.

These parameters are the primary drivers behind the performance of LTS. The higher the resolution ratio is, the larger the coarse time-step can be, allowing LTS to take significantly less time-steps on coarse cells compared to a global method, which must use a small time-step everywhere. Similarly, the higher the count ratio is, the more coarse cells there are in a mesh where LTS has to do less work than a global method. To see the importance of these parameters, consider the following idealized analysis. Consider a mesh consisting of n_c cells which use a coarse time-step Δt_{coarse} and n_f cells which use a fine time-step Δt_{fine}. Let M be such that $M = \frac{\Delta t_{\text{coarse}}}{\Delta t_{\text{fine}}}$. When advancing from one time-level to the next, an LTS scheme does an amount of work proportional to $n_c + Mn_f$, whereas a global scheme using Δt_{fine} everywhere does an amount of work proportional to $M(n_c + n_f)$. The theoretical maximum percentage speedup for LTS is then given by

$$\frac{M(n_c + n_f) - (n_c + Mn_f)}{M(n_c + n_f)} \cdot 100\,\% = \frac{M-1}{M}\frac{n_c}{n_c + n_f} \cdot 100\,\%. \quad (3)$$

Here, M is an analogue for the resolution ratio, and $\frac{n_c}{n_f}$ is the count ratio. Using this expression, we can see how both the resolution ratio and the count ratio effect theoretical speedup. As M increases, the positive contribution of the resolution ratio increases asymptotically toward 1. If we imagine holding the number of coarse cells n_c fixed, then as we decrease n_f, the quantity $\frac{n_c}{n_c+n_f}$ increases, that is, the larger the count ratio, the more positive the contribution of $\frac{n_c}{n_c+n_f}$ to the speedup. In general, we see that the best speedups can be achieved on meshes with fewer fine cells relative to the number of coarse cells. It was shown in Capodaglio and Petersen (2022) that LTS3 is able to achieve this theoretical maximum speedup in practice for a given mesh and configuration of the LTS regions.

In a preliminary analysis, we investigated how varying the count and resolution ratios affected the performance of LTS in an idealized test case; in the shallow-water core of MPAS-O, we created a test case following test case five from Williamson et al. (1992) and generated a series of meshes that had varying values for the count ratio and the resolution ratio. We produced the plots in Figure 6, which are relevant to the work done here for two reasons. First, the shallow-water equations used to generate this data are identical to those used in our Hurricane Sandy test case, except for additional forcing terms used here. That is, we can expect for the results shown in Figure 6 to be similar to the results we would get with the Hurricane Sandy test case. Second, this data clearly demonstrates the importance of both the count and resolution ratios when considering the performance of LTS.



(a) East US coast (EC) fine region (b) Western Atlantic (WA) fine region

Figure 7. The two configurations of the fine region used for DelBay meshes: the smaller East US Coast fine region (a) and the larger Western Atlantic fine region (b).

The metric by which we compare LTS3 and RK4 is the speedup, or the percentage value that LTS3 is faster than RK4. This is computed as

$$\text{speedup} = \frac{\text{time RK4} - \text{time LTS3}}{\text{time RK4}} \cdot 100\ \%. \tag{4}$$

Note that this metric is fundamentally different from the one given in Equation 3. The theoretical maximum speedup given by Equation 3 measures the performance gains made possible by using LTS versus not using LTS, that is, the benefits of using Δt_{coarse} on parts of the mesh, versus having to use Δt_{fine} everywhere. In contrast, the speedup in Equation 4 measures the gain in performance of a specific LTS scheme (LTS3) over a specific global scheme (RK4) that uses a global time-step that may not equal the fine time-step used by the LTS scheme. As such, it does not make sense to compare the experimentally obtained speedups from Equation 4 to the theoretical maximum values from Equation 3.

Figure 6 shows that as either the count ratio or the resolution ratio increases, the speedup increases, and that this increase is drastic close to zero. The curves in both plots eventually level off as the speedup becomes limited by the other parameter, which is fixed.

Figure 7 shows two configurations for the placement of the fine and coarse regions. The configuration in Figure 7a will be referred to as the EC configuration and the configuration in Figure 7b will be referred to as the WA configuration. In the EC configuration, the time-step restriction on the coarse cells comes from mid-sized cells in the western Atlantic, so the coarse cells away from the western Atlantic are forced to use a smaller time-step than is optimal. On the other hand, the WA configuration forces all western Atlantic cells to be inside the fine region so that all coarse cells are approximately the same size, which is the largest size among mesh cells, and so admit approximately the same large time-step. For example, note how in Table 1, the DelBay2km EC configuration uses a coarse time-step of 72 s, while the corresponding WA configuration uses a coarse time-step of 306 s. The trade-off in the WA configuration is that the fine region then contains more fine cells that are forced to use the fine time-step than the EC configuration. This trade-off between the count ratio and the resolution ratio is a significant part of our investigation and is discussed in Section 3.1.

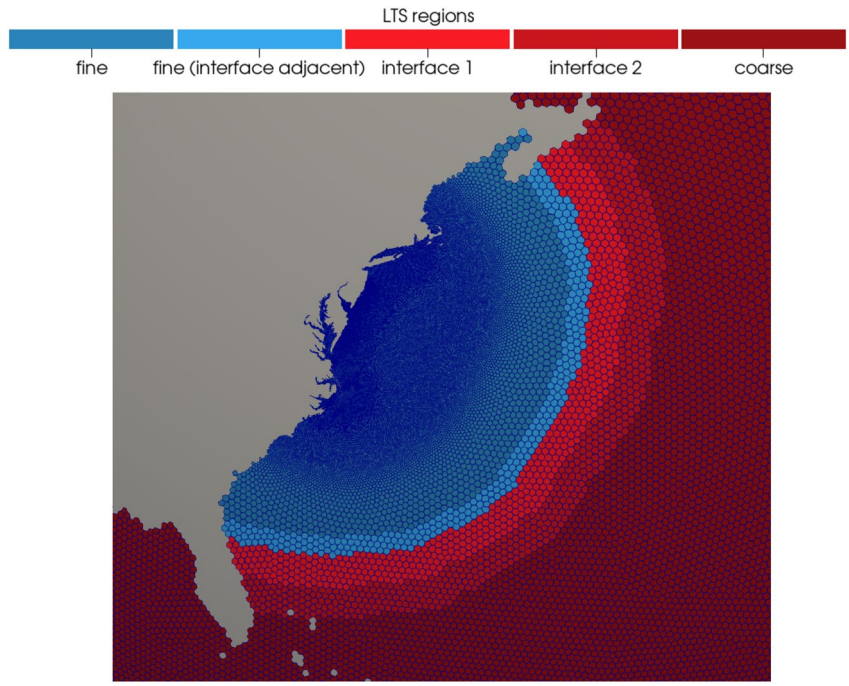


Figure 8. An example of a mesh needing additional layers of interface cells for load balancing in DelBay1km. Here, there are 10 interface layers in total—five layers of interface 1 cells, and five layers of interface 2 cells.

2.3.3. Load Balancing, Communication, and Additional Interface Layers

Performing the LTS3 algorithm in parallel requires careful consideration of load balancing. LTS requires that cells in a given mesh be sorted into different groups, some are coarse cells, some are fine cells, and some are interface cells. Each type of cell requires a different amount of work, so load balancing among MPI processes is non-trivial; an efficient way to address this has been a primary concern of Capodaglio and Petersen (2022). For effective load-balancing, each MPI rank is given three sets of cells, one set of fine cells, one set of coarse cells, and one set of interface cells. The different groups of cells are spread across each rank so that each process has an approximately equal number of each type. This helps to ensure that no process is idle while it waits for the others to finish work. An issue with this scheme is that in most mesh configurations, there are very few interface cells relative to the number of fine or coarse cells. As a result, when running with more than a few processes, there may not be enough interface cells in the mesh for an equal number to be spread across processes in a way that also respects the expense of MPI communication calls. To avoid this issue, additional interface layers are added to the mesh (see Figure 8). Because LTS3 is $\mathcal{O}\left((\Delta t)^3\right)$ everywhere on the mesh, these additional interface layers have no negative impact on the model's accuracy. Adding these additional interface layers allows us to insure that there are enough interface cells to spread across a given number of processes. To minimize storage requirements, certain computations unique to interface cells are performed during the sub-stepping (Capodaglio & Petersen, 2022). The addition of extra interface layers therefore incurs a non-zero computational cost, and so it is not viable to simply add an arbitrarily large number of interface cells just to achieve scalability. It is shown in Capodaglio and Petersen (2022) that, as a rule of thumb, each process should own at least 100 interface cells to achieve sufficient load-balancing. On each mesh and for each LTS fine region configuration, we have set a number of interface layers that ensures this requirement is met. The number of interface layers used in each case is reported in Table 1, where the given value refers to the number of cell layers for each interface, that is, interface 1 or interface 2. For example, the number of interface layers for DelBay2km in the EC configuration is reported as two, which means there are two layers of interface 1 cells, and two layers of interface 2 cells, for a total of four layers of interface cells.

An important feature of the implementation of LTS3, and in fact of any LTS scheme, is the ability to compute the model tendencies on arbitrary sub-regions of the mesh. For example, during the fine time-step sub-cycling, one needs to compute the tendencies *only* on the fine cells—any computations on the coarse cells during this step

Table 1
Relevant Parameters for Each Mesh and Local Time-Stepping Configuration Used in Performance Experiments

	DelBay2km	DelBay1km	DelBay500m	DelBay250m	DelBay125 m
Resolutions: Grid cell width (km)					
Global background	120	60	30	30	30
Western Atlantic	30	30	15	15	15
Eastern US coast	10	5	2.5	1.25	0.625
Delaware coast	5	2.5	1.25	0.625	0.3125
Delaware Bay	2	1	0.5	0.25	0.125
Mesh Parameters					
Number of cells	58,240	198,776	794,172	1,591,416	4,617,565
Number of MPI ranks	2	8	32	64	178
Thousands of cells per MPI rank	29	25	25	25	26
LTS Parameters					
Eastern US Coast (EC) Fine Region					
Number of interface layers	2	5	10	20	50
Count ratio	1.88	1.71	1.71	0.46	0.12
Resolution ratio	15	30	30	60	120
Δt_{RK4} (s)	30	15	7.5	3.75	1.875
Δt_{fine} (s)	18	8	4	2	1
Δt_{coarse} (s)	72	72	24	24	24
M	4	9	6	12	24
Western Atlantic (WA) Fine Region					
Number of interface layers	2	5	10	20	50
Count ratio	0.92	1.28	1.28	0.39	0.11
Resolution ratio	60	60	60	120	240
Δt_{RK4} (s)	30	15	7.5	3.75	1.875
Δt_{fine} (s)	18	8	4	2	1
Δt_{coarse} (s)	306	152	80	80	80
M	17	19	20	40	80

would be wasted, causing performance to suffer. The task of computing tendency terms on specific sub-regions of a parallel partition is a non-trivial task due to the interplay of the MPI-halo layers of the parallel partition with the cells and edges that define a given sub-region. As explained in Capodaglio and Petersen (2022), this can be done by using MPI memory blocks. Each MPI rank is associated with three memory blocks, each of which owns exactly one type of cell; fine, coarse, or interface. The halo cells of each block are allowed to (and in fact most often do) contain cells of a different type than those owned by the block. There are several advantages to using this approach: First, the right hand side terms can be computed only on the regions where the solution is advanced without any modification to the routines that carry out these computations, which could be extremely cumbersome in a complex model like MPAS-O. Second, the parallelism is automatically handled and the block halo coincides with the sub-region halo. Third, the block partitioning can be easily achieved using METIS (Karypis & Kumar, 1997) and some simple pre-processing Python scripts, as detailed in Capodaglio and Petersen (2022).

3. Results

In this section, we discuss the performance of LTS3 compared to RK4, in terms of computational time and accuracy of the sea-surface height predictions.

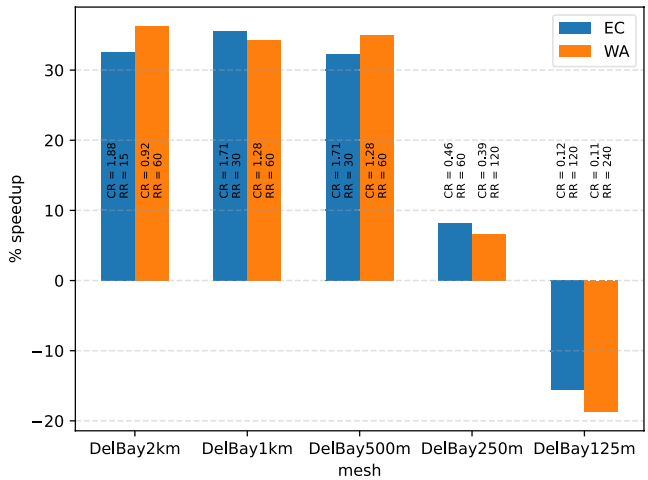


Figure 9. Speedup in terms of CPU-time obtained using LTS3 over RK4 in both the EC and WA configuration. Speedup is calculated as in Equation 4. CR is the count ratio and RR is the resolution ratio.

3.1. Computational Time

Figure 9 reports the performance differences in terms of CPU-time of RK4 and LTS3 in both the EC and WA configurations. To obtain this data, the model has been run using both RK4 and LTS3 for a set number of simulated seconds (referred to as the run-time) and the amount of time the model spends on time-stepping in both cases was recorded. To account for differences in network communication speeds, timings are averaged over five simulations. All simulations were run on the Badger cluster at Los Alamos National Laboratory, which is a 660 node compute cluster where each node has 128 GB of memory and contains two 2.10 GHz Intel Xeon E5-2695 v4 processors, each having 18 cores. Badger has a peak compute speed of 798 Tera-FLOPS per second.

Table 2 reports the specific results of each test case, including the values for the time integration timer in each case and the number of simulated seconds the model was run for. One can note that, on each mesh, the run-time is different, and is selected to be the smallest number of seconds that is divisible by the coarse time-step used in the WA configuration, the coarse time-step used in the EC configuration, and the time-step for RK4. This is to ensure that for each method and configuration, there is no case in which the model is actually advancing to a time further than intended, which would pollute the results.

In another storm surge focused application of LTS schemes, Dawson et al. (2013) are able to cut the computational time of their model in half using LTS. We note that these results are not directly comparable to those given here; whereas we compute speedup by comparing our LTS scheme to RK4, which uses a larger time-step than the fine time-step used by LTS, the speedups obtained by Dawson et al. (2013) are computed by comparing their LTS scheme to a global scheme using a time-step equal to the fine time-step used by LTS. As such, the results from Dawson et al. (2013) are comparable to the theoretical maximum speedup given by Equation 3, which LTS3 achieves (Capodaglio & Petersen, 2022).

It is well known that in parallel applications, communication between processes is more expensive than floating-point operations in terms of computational time, and that on a fixed mesh, past a certain number of processes, a time-stepping scheme will become dominated by communication between processes and will not scale efficiently. Because LTS3 requires that the computational domain be decomposed into different classes of cells that require different work-loads, LTS3 requires more communication than RK4; as the number of processes increases, communication overhead will have a stronger impact on LTS3 than on RK4. For a fair comparison between the two schemes, the runs from Table 2 were obtained with a number of processes for which communication did not become prevalent for either RK4 or LTS3.

For completeness, in Table 3, we give performance results at higher process counts where there are only 3,000 cells per process as opposed to more than 25,000 cells per process as in Figure 9 and Table 2. Here, we see that LTS3

Table 2
CPU-Time Performance of Fourth-Order Runge-Kutta Method and Third-Order LTS Scheme in Both the EC and WA Configuration

	DelBay2km	DelBay1km	DelBay500m	DelBay250m	DelBay125m
Run-time (s)	6,120	6,840	960	960	960
Number of MPI ranks	2	8	32	64	178
RK4 (s)	57.92	133.26	47.57	95.76	208.58
LTS3 EC (s)	39.10	85.94	32.24	87.98	241.19
Speedup	32.50	35.51	32.23	8.13	−15.63
LTS3 WA (s)	36.94	87.65	30.89	89.39	247.52
Speedup	36.23	34.23	35.05	6.65	−18.67

Note. Speedup is calculated as in Equation 4. The number of cells per process on each mesh is approximately between 25,000 and 29,000.

Table 3
CPU-Time Performance of RK4 and LTS3 at High Process Counts

	DelBay2km	DelBay1km	DelBay500m	DelBay250m	DelBay125m
Run-time (s)	6,120	6,840	960	960	960
Number of MPI ranks	16	64	256	512	1,536
RK4 (s)	9.57	20.73	5.96	12.05	23.35
LTS3 EC (s)	8.49	17.93	5.90	38.51	83.90
Speedup	11.31	13.50	1.00	−219.73	−259.36
LTS3 WA (s)	7.75	17.71	5.29	26.19	90.60
Speedup	19.07	14.56	11.23	−117.46	−288.05

Note. The number of cells per process on each mesh is approximately 3,000.

suffers from increased communication time and therefore speed-ups are less dramatic than in Table 2, although LTS3 still remains faster than RK4 even in this case, for all but one of the meshes it was faster on before. The increased communication time in this case is due to each processes not having enough work to do during each time-step, because this is a single layer model. In a layered model with n layers, there is approximately n times more work to be done per cell per time-step, and it has been shown in Capodaglio and Petersen (2022) that in a shallow water model with 100 layers, LTS can be run effectively with as little as 500 cells per process without communication becoming dominant.

From the results in Figure 9, there are a few things we learn about efficient mesh design and LTS configuration to achieve optimal performance with LTS. First, consider DelBay500 m in the WA configuration where we see a speedup of 35%. As noted in Section 2.3.2, we are particularly interested in the count ratio and resolution ratio of a mesh when considering the potential for good performance with LTS (see Figure 6). In DelBay500 m, the count ratio is 1.28 and the resolution ratio is 60, that is, there are 1.28 times as many coarse cells than fine cells, and the smallest fine cells are 60 times smaller than the smallest coarse cells. On the other hand, for the DelBay250 m in the WA configuration the speed up is 6.65%. Here, the count ratio is only 0.39 (meaning that there are approximately 2.56 times more fine cells than coarse cells) and the resolution ratio is 120.

The DelBay250m WA case has the advantage over the DelBay500m WA case in having a higher resolution ratio, but in turn has a much lower count ratio. This means that in the DelBay250m WA case, LTS3 spends the majority of its effort time-stepping on the fine region with a time-step that is smaller than the global time-step used by RK4, and even though the coarse time-step is much larger than RK4's global time-step, there are fewer cells where it is used. This points to the fact that, when designing a mesh with the desire to take advantage of the benefits of LTS, one needs to take into account the count ratio so that the mesh is not overloaded with fine time-step cells. In the case of DelBay125m in both the EC and WA configurations, when the count ratio is especially low (0.12 and 0.11 respectively), we see that LTS3 does not perform as well as RK4 in terms of CPU-time. In extreme cases such as this where the mesh is predominantly composed of high-resolution cells, the use of an LTS scheme does not make sense, and one might benefit from simply using a higher-order global method that admits a larger time-step on the mesh's smallest cells.

Another important consideration is the placement of the border between the fine and coarse regions that defines which cells use the fine time-step and which use the coarse time-step. In meshes with only two regions of differing resolutions there is no choice to make, but on meshes such as the ones used here, where there are more than two resolutions, there are multiple choices for the placement of the fine region. What this choice should be is not always clear. As can be seen in Figure 9, there are some cases where the EC configuration performs better and some where the WA configuration performs better. In the case of DelBay250m and DelBay125m, the performance in both configurations is not particularly good. However, that does not mean that LTS cannot perform well on these meshes. The LTS configurations chosen in this work are experimental and there may be other configurations not investigated here that could be used to improve performance on these meshes.

In summary, LTS schemes require that the user consider both the count and resolution ratios as they configure a mesh for LTS, but this extra time spent is easily worth the greater than 35% speedups that can be achieved when the configuration is done well. As a general rule, one's goal is to maximize both of these parameters. It is particularly

important that the count ratio not be too low. Figure 6a suggests that a count ratio of 0.2 (1:5) as a rule-of-thumb cut-off. The results in Figures 6b and 9 as well as Equation 3 suggest that increasing the resolution ratio has diminishing returns if it is detrimental to the count ratio. That is, if the vast majority of work is being done with the fine time-step, it does not matter if the coarse time-step is many times larger than the fine time-step. To give some general advice, the quality of which will doubtlessly vary across applications, it is best to first achieve a reasonable value for the count ratio even at the expense of the resolution ratio as long as the resolution ratio can be kept reason-

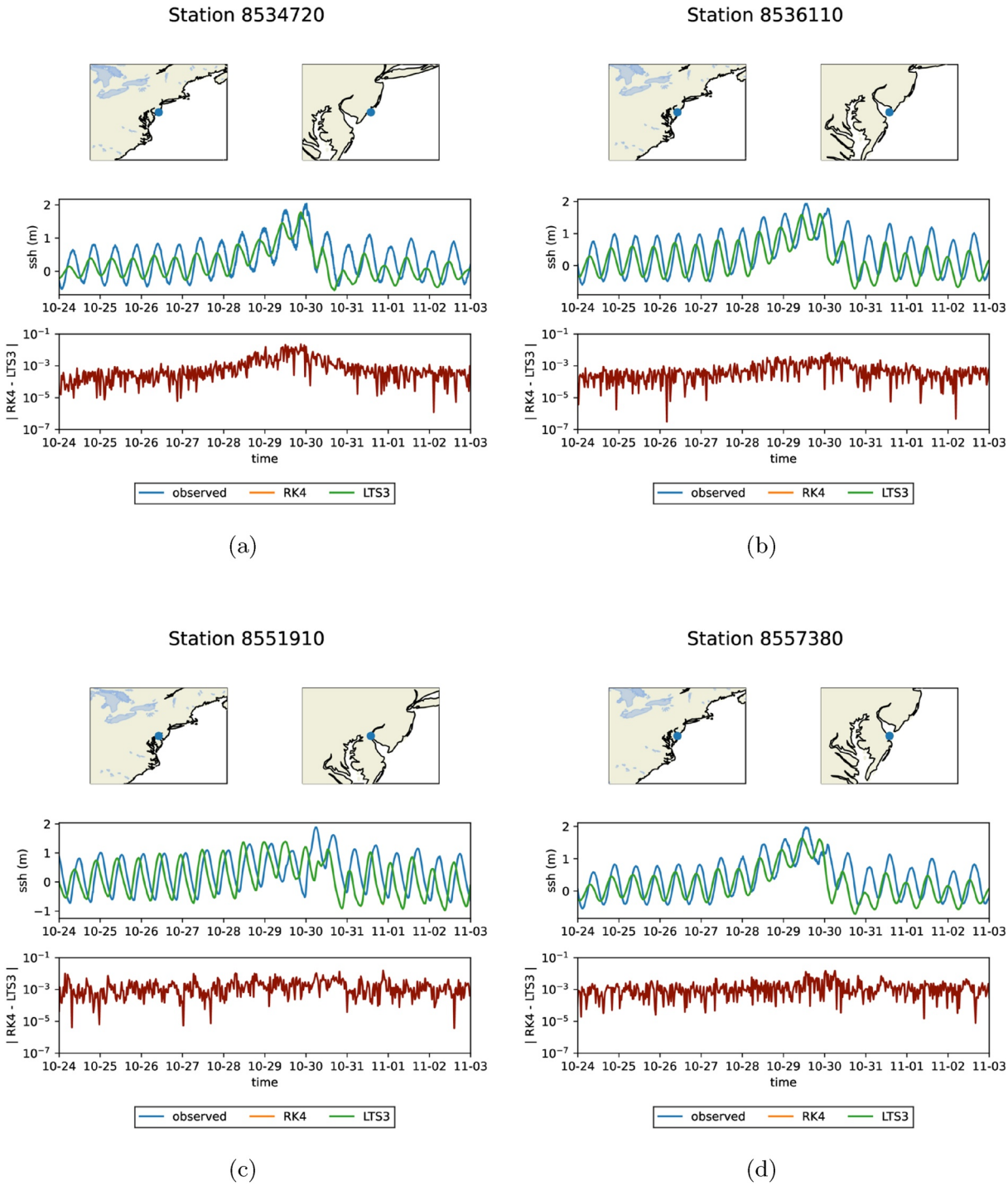


Figure 10. Sea-surface height solutions from RK4 and LTS3 compared to observed tide gauge data on DelBay125m. Note that the curve corresponding to LTS3 is covering the curve corresponding to RK4 in the upper plots. The absolute difference between the two schemes is shown on a log scale in the lower plots.

ably high. If this is not possible, on may wish to use a method that admits a larger time-step altogether like RK4; on a multi-resolution mesh predominately populated by small cells it does not make sense to use a LTS scheme.

3.2. Accuracy

Here we compare the sea-surface height (SSH) predicted by RK4 and LTS3 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/.

As noted in Section 2.1, there are a number of processes omitted from our model that are likely affecting the accuracy of the SSH predictions from our model as compared to observed data. Both wetting and drying cells and spatially varying bottom friction are not present in our model, as the implementations of these processes are under development. Because of these sources of error in our model, we expect that the choice of time-stepping scheme is not the dominate source of error, and so the SSH solutions produced by LTS3 and RK4 should agree closely. The results in Figure 10 verify this; the absolute differences between the LT3 and RK4 solutions are, at most, on the order of centimeters.

4. Conclusion

We have used a third order LTS scheme (LTS3) to model the storm surge of Hurricane Sandy in the Delaware Bay, using meshes of unprecedentedly high resolution for MPAS-O, and have shown that the solutions obtained are qualitatively comparable to those produced by the classical four stage, fourth order Runge-Kutta scheme (RK4). Furthermore, LTS3 produces these solutions in considerably less computational time than RK4, with speedups of up to 35%. This was the first real-world, practical application of LTS schemes in the framework of MPAS-O, and will be used to pave the way for further use of LTS within the MPAS framework for the accurate capture of coastal physical phenomena requiring high spatial resolution, without compromising the overall speed of the simulation. Moving forward, we are interested in using LTS as a tool to enhance the existing split-explicit solver that is the standard time-stepping scheme used in MPAS-O for a multi-layered ocean. LTS in its current state could be used within the split-explicit solver to solve the barotropic mode, while the baroclinic mode is handled as normal. We are also exploring the possibility of adapting LTS3 for use in a layered model, which would lead to the possibility of using LTS3 as a baroclinic solver as well.

Acknowledgments
JRL was supported by the U.S. Department of Energy (DOE), Office of Science, Office of Workforce Development for Teachers and Scientists, Office of Science Graduate Student Research (SCGSR) program. The SCGSR program is administered by the Oak Ridge Institute for Science and Education (ORISE) for the DOE. ORISE is managed by ORAU under contract number DE-SC0014664. GC, MRP, SRB, 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. This research used resources provided by the Los Alamos National Laboratory Institutional Computing Program, which is supported by the U.S. DOE National Nuclear Security Administration under Contract No. 89233218CNA000001. The authors would also like to thank Sergey Danilov and two anonymous reviewers for their insightful comments and suggestions.

Data Availability Statement

The source code for the LTS development branch of MPAS-O used here (Capodaglio et al., 2022) can be found on GitHub and Zenodo.
GitHub: https://github.com/jeremy-lilly/MPAS-Model/tree/4e1f5a3cfe78ef01afa07e61f0d46670fdb6c014
Zenodo: https://doi.org/10.5281/zenodo.6904061

The data generated for this paper (Lilly et al., 2022) are also publicly available on Zenodo. This includes the model output used to generate SSH plots, the log files from performance experiments, and an example run directory which includes all necessary data and configuration options to run the model.
Zenodo: https://doi.org/10.5281/zenodo.6908349

References

Accad, Y., Pekeris, C. L., & Jeffreys, H. (1978). Solution of the tidal equations for the M2 and S2 tides in the world oceans from a knowledge of the tidal potential alone. *Philosophical Transactions of the Royal Society of London–Series A: Mathematical and Physical Sciences*, *290*(1368), 235–266. https://doi.org/10.1098/rsta.1978.0083

Arakawa, A., & Lamb, V. R. (1977). Computational design of the basic dynamical processes of the UCLA general circulation model. *General circulation models of the atmosphere*, *17*, 173–265.

Arbic, B. K., Alford, M. H., Ansong, J. K., Buijsman, M. C., Ciotti, R. B., Farrar, J. T., et al. (2018). Primer on global internal tide and internal gravity wave continuum modeling in HYCOM and MITGCM. *New Frontiers in Operational Oceanography*, 307–392.

Barton, K. N., Pal, N., Brus, S. R., Petersen, M. R., Arbic, B. K., Engwirda, D., et al. (2022). Global barotropic tide modeling using inline self-attraction and loading in MPAS-ocean. *Journal of Advances in Modeling Earth Systems*, *14*(11). https://doi.org/10.1029/2022MS003207

Capodaglio, G., Lilly, J. R., & Petersen, M. R. (2022). MPAS-Model LTS Source Code (Commit: 4e1f5a3). Zenodo. https://doi.org/10.5281/zenodo.6904061

Capodaglio, G., & Petersen, M. R. (2022). Local time stepping for the shallow water equations in MPAS. *Journal of Computational Physics*, *449*, 110818. https://doi.org/10.1016/j.jcp.2021.110818

Danilov, S., Sidorenko, D., Wang, Q., & Jung, T. (2017). The finite-volume sea ice–ocean model (FESOM2). *Geoscientific Model Development*, *10*(2), 765–789. https://doi.org/10.5194/gmd-10-765-2017

Dawson, C., Trahan, C. J., Kubatko, E. J., & Westerink, J. J. (2013). A parallel local time stepping Runge–Kutta discontinuous Galerkin method with applications to coastal ocean modeling. *Computer Methods in Applied Mechanics and Engineering*, *259*, 154–165. https://doi.org/10.1016/j.cma.2013.03.015

Diaz, J., & Grote, M. J. (2009). Energy conserving explicit local time stepping for second-order wave equations. *SIAM Journal on Scientific Computing*, *31*(3), 1985–2014. https://doi.org/10.1137/070709414

Engwirda, D. (2017). JIGSAW-GEO (1.0): Locally orthogonal staggered unstructured grid generation for general circulation modelling on the sphere. *Geoscientific Model Development*, *10*(6), 2117–2140. https://doi.org/10.5194/gmd-10-2117-2017

Golaz, J.-C., Caldwell, P. M., Van Roekel, L. P., Petersen, M. R., Tang, Q., Wolfe, J. D., et al. (2019). The DOE E3SM coupled model version 1: Overview and evaluation at standard resolution. *Journal of Advances in Modeling Earth Systems*, *11*(7), 2089–2129. https://doi.org/10.1029/2018MS001603

Gottlieb, S., & Shu, C.-W. (1998). Total variation diminishing Runge-Kutta Schemes. *Mathematics of Computatio*n, *67*(221), 73–85. https://doi.org/10.1090/S0025-5718-98-00913-2

Gottlieb, S., Shu, C.-W., & Tadmor, E. (2001). Strong stability-preserving high-order time discretization methods. *SIAM Review*, *43*(1), 89–112. https://doi.org/10.1137/S003614450036757X

Higdon, R. L. (2005). A two-level time-stepping method for layered ocean circulation models: Further development and testing. *Journal of Computational Physics*, *206*(2), 463–504. https://doi.org/10.1016/j.jcp.2004.12.011

Hoang, T.-T.-P., Leng, W., Ju, L., Wang, Z., & Pieper, K. (2019). Conservative explicit local time-stepping schemes for the shallow water equations. *Journal of Computational Physics*, *382*, 152–176. https://doi.org/10.1016/j.jcp.2019.01.006

Hoch, K. E., Petersen, M. R., Brus, S. R., Engwirda, D., Roberts, A. F., Rosa, K. L., & Wolfram, P. J. (2020). MPAS-Ocean simulation quality for variable-resolution North American coastal meshes. *Journal of Advances in Modeling Earth Systems*, *12*(3), e2019MS001848. https://doi.org/10.1029/2019MS001848

Jayne, S. R., & St. Laurent, L. C. (2001). Parameterizing tidal dissipation over rough topography. *Geophysical Research Letters*, *28*(5), 811–814. https://doi.org/10.1029/2000GL012044

Ju, L., Ringler, T., & Gunzburger, M. (2011). 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* (pp. 313–342). Springer. https://doi.org/10.1007/978-3-642-11640-7_10

Karypis, G., & Kumar, V. (1997). METIS—a software package for partitioning unstructured graphs, partitioning meshes and computing fill-reducing ordering of sparse matrices.

Korn, P. (2017). Formulation of an unstructured grid model for global ocean dynamics. *Journal of Computational Physic*s, *339*, 525–552. https://doi.org/10.1016/j.jcp.2017.03.009

Lilly, J. R., Capodaglio, G., & Petersen, M. R. (2022). Data for 'Storm Surge Modeling as an Application of Local Time-stepping in MPAS-Ocean'. *Zenodo*. https://doi.org/10.5281/zenodo.6908349

Liu, L., Li, X., & Hu, F. Q. (2010). Nonuniform time-step Runge–Kutta discontinuous Galerkin method for computational aeroacoustics. *Journal of Computational Physics*, *229*(19), 6874–6897. https://doi.org/10.1016/j.jcp.2010.05.028

Mandli, K. T., & Dawson, C. N. (2014). Adaptive mesh refinement for storm surge. *Ocean Modelling*, *75*, 36–50. https://doi.org/10.1016/j.ocemod.2014.01.002

Montseny, E., Pernet, S., Ferriéres, X., & Cohen, G. (2008). Dissipative terms and local time-stepping improvements in a spatial high order discontinuous Galerkin scheme for the time-domain Maxwell's equations. *Journal of Computational Physics*, *227*(14), 6795–6820. https://doi.org/10.1016/j.jcp.2008.03.032

Okabe, A. (2017). Spatial tessellations. In *International encyclopedia of geography* (pp. 1–11). John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118786352.wbieg0601

Petersen, M. R., Asay-Davis, X. S., Berres, A. S., Chen, Q., Feige, N., Hoffman, M. J., et al. (2019). An evaluation of the ocean and sea ice climate of E3SM using MPAS and interannual CORE-II forcing. *Journal of Advances in Modeling Earth Systems*, *11*(5), 1438–1458. https://doi.org/10.1029/2018MS001373

Petersen, M. R., Jacobsen, D. W., Ringler, T. D., Hecht, M. W., & Maltrud, M. E. (2015). Evaluation of the arbitrary Lagrangian–Eulerian vertical coordinate method in the MPAS-Ocean model. *Ocean Modelling*, *86*, 93–113. https://doi.org/10.1016/j.ocemod.2014.12.004

Pringle, W. J., Wirasaet, D., Roberts, K. J., & Westerink, J. J. (2021). Global storm tide modeling with ADCIRC V55: Unstructured mesh design and performance. *Geoscientific Model Development*, *14*(2), 1125–1145. https://doi.org/10.5194/gmd-14-1125-2021

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

Ringler, T. D., Thuburn, J., Klemp, J. B., & Skamarock, W. C. (2010). A unified approach to energy conservation and potential vorticity dynamics for arbitrarily-structured C-grids. *Journal of Computational Physics*, *229*(9), 3065–3090. https://doi.org/10.1016/j.jcp.2009.12.007

Saha, S., Moorthi, S., Wu, X., Wang, J., Nadiga, S., Tripp, P., et al. (2014). The NCEP climate Forecast system version 2. *Journal of Climate*, *27*(6), 2185–2208. https://doi.org/10.1175/jcli-d-12-00823.1

Skamarock, W. C., Klemp, J. B., Duda, M. G., Fowler, L. D., Park, S.-H., & Ringler, T. D. (2012). A multiscale Nonhydrostatic atmospheric model using centroidal Voronoi tesselations and C-grid staggering. *Monthly Weather Review*, *140*(9), 3090–3105. https://doi.org/10.1175/MWR-D-11-00215.1

Thuburn, J., Ringler, T. D., Skamarock, W. C., & Klemp, J. B. (2009). Numerical representation of geostrophic modes on arbitrarily structured C-grids. *Journal of Computational Physics*, *228*(22), 8321–8335. https://doi.org/10.1016/j.jcp.2009.08.006

Tozer, B., Sandwell, D. T., Smith, W. H. F., Olson, C., Beale, J. R., & Wessel, P. (2019). Global bathymetry and topography at 15 arc sec: SRTM15+. *Earth and Space Science*, *6*(10), 1847–1864. https://doi.org/10.1029/2019EA000658

Trahan, C. J., & Dawson, C. (2012). Local time-stepping in Runge–Kutta discontinuous Galerkin finite element methods applied to the shallow-water equations. *Computer Methods in Applied Mechanics and Engineering*, *217–220*, 139–152. https://doi.org/10.1016/j.cma.2012.01.002

Williamson, D. L., Drake, J. B., Hack, J. J., Jakob, R., & Swarztrauber, P. N. (1992). A standard test set for numerical approximations to the shallow water equations in spherical geometry. *Journal of Computational Physics*, *102*(1), 211–224. https://doi.org/10.1016/S0021-9991(05)80016-6