Localized Exponential Time Differencing Method for Shallow Water Equations: Algorithms and Numerical Study

Xucheng Meng[1,3,4], Thi-Thao-Phuong Hoang[2], Zhu Wang[1], Lili Ju[1,*]

[1] *Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA.*
[2] *Department of Mathematics and Statistics, Auburn University, Auburn, AL 36849, USA.*
[3] *SUSTech International Center for Mathematics, Southern University of Science and Technology, Shenzhen, Guangdong 518055, China.*
[4] *Center for Mathematical studies, Advanced Institute of Natural Sciences, Beijing Normal University at Zhuhai, Zhuhai, Guangdong 519087, China.*

Abstract. In this paper, we investigate the performance of the exponential time differencing (ETD) method applied to the rotating shallow water equations. Comparing with explicit time stepping of the same order accuracy in time, the ETD algorithms could reduce the computational time in many cases by allowing the use of large time step sizes while still maintaining numerical stability. To accelerate the ETD simulations, we propose a localized approach that synthesizes the ETD method and overlapping domain decomposition. By dividing the original problem into many subdomain problems of smaller sizes and solving them locally, the proposed approach could speed up the calculation of matrix exponential vector products. Several standard test cases for shallow water equations of one or multiple layers are considered. The results show great potential of the localized ETD method for high-performance computing because each subdomain problem can be naturally solved in parallel at every time step.

Key words: Exponential time differencing, domain decomposition, rotating shallow water equations, finite volume discretization

1 Introduction

The exponential time differencing (ETD) method, as an exponential integrator-based method, has been developed for solving evolutionary partial differential equations of semi-linear or fully nonlinear types (see, for example, [1–8] and a thorough review [9]). Such a method is constructed on the basis of exponential integrators and variation-of-constants formula, and is known for its desirable numerical stability. Indeed, for stiff problems, a large time step size can be used in ETD, while tiny time step sizes are often required by explicit time stepping. In addition, differently from standard implicit time stepping, nonlinear solvers are not required in ETD. Therefore, the ETD method usually leads to significant computational savings comparing with other time-stepping algorithms. The major computational efforts in ETD schemes are spent in evaluating matrix exponential and vector products. Many algorithms have been proposed in order to evaluate matrix exponentials [10–15]. Some of them are designed for large sparse matrices, while the others only work for matrices of moderate size. We are interested in the former as the discrete system we considered in this paper is of large dimensions. The

*Corresponding author. *Email addresses:* mengx@mailbox.sc.edu (X. Meng), tzh0059@auburn.edu (T.-T.-P. Hoang), wangzhu@math.sc.edu (Z. Wang), ju@math.sc.edu (L. Ju)

first comprehensive package for evaluating large-scale matrix exponential vector products is EXPOKIT, which was developed by Sidje in [12]. The backbone of its sparse routines is Krylov subspace projection method such as the Arnoldi and Lanczos processes whose mathematical foundation was established in [16–18]. By projecting large matrices to smaller ones via the Krylov subspace approach, the corresponding matrix exponential becomes easier to compute. The other main idea in EXPOKIT is to adapt the time step size of the ETD simulations based on an error estimator developed in [17]. Combing the time stepping idea of EXPOKIT and the adaptivity of the dimension of the Krylov subspace [1], the phipm function algorithm was developed in [13]. This algorithm achieves a balance between the time stepping error and Krylov projection error by dynamically choosing the dimension of the Krylov subspace and the size of time stepping. But it was pointed out in [19] that the Arnoldi procedure still takes too much time, which made phipm less efficient than other semi-implicit predictor-corrector schemes for geophysical fluid dynamics problems. Thus, the Krylov subspace with the incomplete orthogonalization procedure (phipm/IOM2) was recently introduced, which has been successfully applied, in the context of exponential Rosenbrock integration methods, to the shallow water equations on the sphere [15]. Another solver, KIOPS, was proposed in [14] for calculating φ-functions in exponential integrators to allow efficient implementation of multi-stage exponential integrators.

To accelerate the exponential time integration, a different research direction is to take advantage of parallel and high-performance computing. A straightforward way is to perform the parallelization at the algebraic level. For instance, the parallel adaptive-Krylov exponential solver was proposed in [20], where the standard data-parallel approach is taken, that is, each vector is split across all the processors and MPI communication is used for performing the vector algebraic operations. However, since the matrix exponential is global and dense, this approach usually requires a high communication volume. A different way is to divide the problem into many subdomain problems of smaller size using domain decomposition (DD) so that they can be solved using exponential integrators in parallel. Based on the overlapping domain decomposition, a localized compact ETD algorithm was first introduced in [21] for extreme-scale phase field simulations. In that approach, subdomain problems are solved locally in parallel at each time step, and the data in overlapping regions is shared by neighboring subdomains; thus only a small volume of data needs to be transferred between the neighbor subdomains for time stepping. Numerical experiments on three-dimensional coarsening dynamics demonstrated great computational efficiency and excellent parallel scalability of this approach on supercomputers. The overlapping localized ETD was analyzed in [22] and in [23] for the time-dependent diffusion and semi-linear parabolic equations respectively, in which the convergence of the iterative solutions to fully discrete localized ETD solutions and to the exact semi-discrete solution was rigorously proved. A non-overlapping localized ETD was proposed and analyzed for diffusion problems in [24], where the convergence and exact mass conservation were demonstrated.

In this paper, we consider the rotating shallow water equations in the context of MPAS-Ocean [25], the ocean component of MPAS - Model for Prediction Across Scales. MPAS is a set of open-source software utilities jointly developed by National Center for Atmospheric Research and Los Alamos National Laboratory to model atmosphere, ocean and other earth-system components with application to climate, regional climate and weather studies. To obtain a global ocean model capable of resolving full physics and handling multiple resolutions within a single simulation, MPAS-Ocean utilizes the TRiSK scheme [26,27] - a C-grid staggering finite volume method - for spatial discretization on unstructured, multi-resolution meshes of the sphere. Such meshes are constructed by Spherical Centroidal Voronoi Tessellations (SCVTs) [28,29] whose dual meshes are Delaunay triangulations. In fact (as explained further

below), the TRiSK scheme is applicable to any conforming mesh composed of convex polygons that are locally-orthogonal, including latitude-longitude grids, dipole and tripole displaced pole grids, conformally mapped cubed sphere grids, Voronoi diagrams and Delaunay triangulations. The scheme possesses desirable properties for modeling oceanic and atmospheric flows, in particular, it supports steady-state nonlinear geostrophic balance, and allows for the conservation of mass and total energy and a robust simulation of potential vorticity dynamics. These properties are achieved mainly because of the construction of the flux interpolation scheme which maps a flux field on the primal mesh to a new flux field on the dual mesh in such a way that the divergence of the new flux on the dual mesh is an interpolation of the divergences on the neighboring primal mesh cells. Equivalently, the constructed flux is geometry-compatible, i.e. satisfying the "null-divergence" condition proposed and studied in [30–32] for hyperbolic conservation laws on manifolds to ensure stationary geostrophic modes. It should be noted that the flux reconstruction scheme in TRiSK is robust and works on any orthogonal grids, particularly on multi-resolution meshes by SCVTs as demonstrated in [33]. Therefore, the TRiSK scheme has been widely used for modeling both global ocean/atmosphere circulations and flow motions in coastal regions. The analysis of the scheme was presented in [26] and [27] for the linearized and nonlinear shallow water equations, respectively.

In this work, we focus on efficient time-stepping methods for the nonlinear shallow water equations discretized in space by the TRiSK scheme; specificially, we propose a localized ETD method that combines the ETD method with the overlapping domain decomposition. The ETD methods have been increasingly used as the time integration scheme for the shallow water equations on the sphere of earth [15, 19, 34, 37]. Among them, in [37], different ETD methods have been proposed and analyzed for the multilayer rotating shallow water equations with TRiSK discretization. These schemes are efficient in the sense that considerably larger time steps over an explicit integrator can be taken while stability and sufficient accuracy are still maintained; moreover, numerical results show that significant cost reductions are achieved with the ETD method over an explicit time discretization scheme (RK4). The conservation of mass of the ETD scheme in the framework of TRiSK discretization is also proved. Our goal here is to construct localized ETD methods which maintain all the desirable properties of the global ETD method while reducing the size of the problem with domain decomposition, thus further speed up the overall computation. Extensive numerical experiments are carried out to demonstrate the effectiveness of the localized ETD method, which achieves the desired accuracy as the global ETD method and shows great potential in parallel computing due to its natural scalability.

The rest of this paper is organized as follows. The mathematical model and spatial discretization for the shallow water equations are presented in Section 2. The ETD method is briefly reviewed in Section 3, and the localized ETD method is then proposed and discussed in Section 4. Various numerical experiments are presented in Section 5, followed by some concluding remarks given in Section 6.

2 Rotating shallow water equations and spatial discretization

The rotating shallow water equations (SWEs) have been widely used for modeling the atmospheric and oceanic flows, which can be seen as a simplification of the primitive equations obtained under the assumption of a small ratio of the vertical length scale (fluid thickness) to the horizontal one. The single-layer SWEs describe the motion of a thin layer of fluid with a uniform density and a free surface, lying on a rigid bed. The multilayer model considers the variance of density in the vertical direction and models the dynamics of several layers of fluids stacked on top of each other, which provides a more accurate vertical profile than the

single-layer model. Both models will be presented in the following. Furthermore, for geophysical flow problems, the Coriolis force is included in order to account for the rotating effect of the Earth. To better address the multi-scales of the geophysical flows, meshes with multiple resolutions are often applied. To this end, we choose the TRiSK scheme for spatial discretization [25–27,35], as it can handle meshes with variable resolutions and possesses desired properties of conservations.

2.1 Single-layer shallow water equations

Let Ω be the sphere of earth or a part of it. The single-layer rotating SWEs can be written in the following vector-invariant form:

$$\begin{cases} \dfrac{\partial h}{\partial t}+\nabla\cdot(h\boldsymbol{u})=0, & \text{in } \Omega\times(0,T), \\ \dfrac{\partial \boldsymbol{u}}{\partial t}+(f+\omega)\boldsymbol{k}\times\boldsymbol{u}+\nabla\left(\dfrac{|\boldsymbol{u}|^2}{2}+g(h+b)\right)=\boldsymbol{G}(h,\boldsymbol{u}), & \text{in } \Omega\times(0,T), \end{cases} \tag{2.1}$$

where h is the fluid thickness, $\boldsymbol{u}$ is the fluid horizontal velocity field on the earth's surface, $\omega=\boldsymbol{k}\cdot(\nabla\times\boldsymbol{u})$ is the relative vorticity with $\boldsymbol{k}$ the surface normal vector satisfying $\boldsymbol{k}\cdot\boldsymbol{u}=0$, and $\boldsymbol{G}$ is the additional stress or diffusion terms that must be determined on a case-to-case basis. Three parameters involved are the gravity acceleration g, the bottom topography b, and the Coriolis parameter $f=2\Omega_0\sin\theta$, where Ω_0 is the angular velocity of rotation and θ the latitude. The fluid absolute vorticity is $f+\omega$. By introducing the potential vorticity (PV)

$$q=\frac{f+\omega}{h}, \tag{2.2}$$

the rotating SWEs can be recast in the following form:

$$\begin{cases} \dfrac{\partial h}{\partial t}+\nabla\cdot(h\boldsymbol{u})=0, \\ \dfrac{\partial \boldsymbol{u}}{\partial t}+q(h\boldsymbol{u}^{\perp})+\nabla\left(\dfrac{|\boldsymbol{u}|^2}{2}+g(h+b)\right)=\boldsymbol{G}(h,\boldsymbol{u}), \end{cases} \tag{2.3}$$

where $\boldsymbol{u}^{\perp}=\boldsymbol{k}\times\boldsymbol{u}$ is the velocity rotated through a right angle, and $q(h\boldsymbol{u}^{\perp})$ is the thickness flux of PV perpendicular to the velocity field $\boldsymbol{u}$ and is referred to as the nonlinear Coriolis force [27].

2.2 Multilayer shallow water equations

The multilayer shallow water model describes the motion of a stack of fluids with distinct densities [36–38]. It is able to provide more accurate vertical profiles of the fluid velocity and depth, compared with the single-layer model. Assuming that there are L layers of fluids, we define ρ_l to be the density of the l-th layer that satisfies $\rho_{l-1}<\rho_l$, for $2\leq l\leq L$ (i.e. the density is increasing with water depth). Denote by h_l and $\boldsymbol{u}_l$ the fluid thickness and velocity of the l-th layer, and by $\boldsymbol{h}$ and $\boldsymbol{u}$ the vectors containing all the layer variables. Set the layer coordinates η_l as

$$\eta_l(\boldsymbol{h})=b+\sum_{i=l}^{L}h_i, \quad \text{for } l=1,\ldots,L, \text{ and } \eta_{L+1}=b, \tag{2.4}$$

so that the layers are separated based on their densities (also known as isopycnal contours). Then the governing equations for the fluid in layer l read:

$$\begin{cases} \dfrac{\partial h_l}{\partial t} + \nabla \cdot (h_l \boldsymbol{u}_l) = 0, \\ \dfrac{\partial \boldsymbol{u}_l}{\partial t} + q(h_l, \boldsymbol{u}_l) h_l \boldsymbol{u}_l^{\perp} + \nabla \left(\dfrac{|\boldsymbol{u}_l|^2}{2} + \dfrac{g p_l(\boldsymbol{h})}{\rho_l} \right) = \boldsymbol{G}_l(\boldsymbol{h}, \boldsymbol{u}), \end{cases} \tag{2.5}$$

where $\boldsymbol{G}_l$ is the additional stress or diffusion that will be specified in numerical experiments, and $p_l(\boldsymbol{h}) = \rho_l \eta_l(\boldsymbol{h}) + \sum_{i=1}^{l-1} \rho_i h_i$ is the dynamical pressure.

Remark 2.1. The rotating SWEs have been widely used for predicting tides and storm surge levels of the ocean, in which the term $\boldsymbol{G}$ in equation (2.3) or $\boldsymbol{G}_l$ in equation (2.5) plays an important role in making the prediction feasible. The choice of this term will be specified in Section 5 for each test case.

2.3 Spatial discretization by the TRiSK scheme

The TRiSK scheme is a C-grid staggering, mimetic finite volume/finite difference scheme for spatial discretization that preserves the desirable properties of the continuous equations, such as the conservation of mass, total energy and PV. The TRiSK scheme has been used for the horizontal discretization of the primitive equations in MPAS-Ocean. Here, we give a brief introduction of the scheme. For details, the reader is referred to [26, 27, 39].

The TRiSK scheme uses the spherical centroidal Voronoi tessellation (SCVT) [40] and its dual Delaunay triangulation as the primal mesh and dual mesh, respectively. As a C-grid staggering scheme, for the discrete quantities of the rotating SWEs (2.3) and the multilayer extension (2.5), the TRiSK scheme places the fluid thickness at the primal mesh cell centers, stores the normal component of the velocity at the primal cell edges, and puts the PV at the primal cell vertices (see Figure 1). Take the single-layer rotating SWEs for example, the semi-

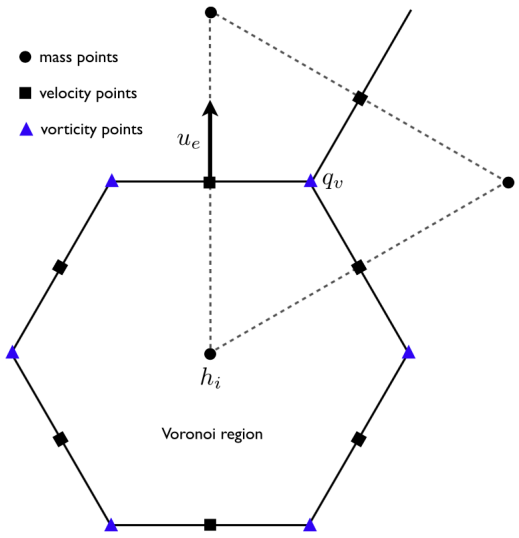

Figure 1: The staggering of variables for the thickness h, the normal component of the velocity $\boldsymbol{u}$ and the PV q in the TRiSK scheme [25].

discrete system after the spatial discretization is:

$$\begin{cases} \dfrac{\partial h_i}{\partial t} = -[\nabla \cdot F_e]_i, \\ \dfrac{\partial u_e}{\partial t} = -[\nabla (K_i + g(h_i + b_i))]_e - F_e^{\perp} \widehat{q}_e + \boldsymbol{G}(\mathfrak{h}, \mathfrak{u}), \end{cases} \tag{2.6}$$

where $h_i = \int_{P_i} h d\mathbf{x} / A_i$ is the cell average of fluid thickness h over the primal mesh cell P_i with A_i being the area of cell P_i and $\mathfrak{h} = (h_1(t), \ldots, h_{N_c}(t))$ with N_c being the number of primal cells. For the velocity, the unknowns are $u_e = \boldsymbol{u} \cdot \boldsymbol{n}_e$ representing the component of the horizontal velocity field in the direction normal to the primal cell edge e and $\mathfrak{u} = (u_1(t), \ldots, u_{N_e}(t))$ with N_e the number of primal edges. In addition, $F_e = \widehat{h}_e u_e$ is the fluid thickness flux per unit length in the direction of $\boldsymbol{n}_e$ with $\widehat{h}_e = [h]_{i \to e}$ denoting the average of h on primal edge e using its values h_i on neighboring primal cells; $\widehat{q}_e = [q]_{v \to e}$ an averaging of q on the primal edge e using its values q_v on primal vertices and $F_e^{\perp}$ the thickness flux perpendicular to F_e. The specific form of $F_e^{\perp}$ can be obtained based on the flux reconstruction operator in [26, 27].

The semi-discrete system (2.6) can be rewritten in the following general form:

$$\frac{d\boldsymbol{W}(\boldsymbol{t})}{dt} = \boldsymbol{F}(\boldsymbol{W}), \tag{2.7}$$

where $\boldsymbol{W} = (h_1(t), \ldots, h_{N_c}(t), u_1(t), \ldots, u_{N_e}(t))^T \in \mathbb{R}^{N_c + N_e}$ and the right-hand side vector $\boldsymbol{F}(\boldsymbol{W})$ is generally nonlinear. Next, we discretize (2.7) in time to obtain a fully discrete problem.

3 Exponential time differencing method for time discretization

For time integration, we make use of the exponential time differencing Runge-Kutta (ETD-RK) method, which is known for better stability over the explicit stepping methods. In particular, we illustrate the ETD-RK scheme of third order accuracy in detail as it will be used in our subsequent numerical experiments. Note that in [37], an ETD-RK type scheme was proposed using a static linearization constructed from the Hamiltonian formulation of the continuous equations evaluated at a reference state with zero velocity. However, the applicability of such a scheme is limited as the fixed linearization matrix is not reliable when the velocity field is far away from the zero, which is a common situation in many cases, such as the shallow water test case 5 (SWTC5) and test case 6 (SWTC6) in [41] (see Section 5). Therefore, the continuous linearization via computing the Jacobian matrix of the right hand side of system (2.7) is used in this paper.

3.1 Exponential time differencing Runge-Kutta

Consider a uniform partition of time interval $[0, T]$: $0 = t_0 < t_1 < \ldots < t_M = T$ with time step $\Delta t = T/M$. Denote by $\boldsymbol{W}_n$ the approximation of $\boldsymbol{W}(t_n)$ at the time t_n and $\boldsymbol{J}_n$ the Jacobian matrix of $\boldsymbol{F}(\boldsymbol{W})$ evaluated at $\boldsymbol{W}_n$, $\boldsymbol{J}_n = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{W}}(\boldsymbol{W}_n)$. The system (2.7) is then equivalent to

$$\frac{d\boldsymbol{W}(t)}{dt} = \boldsymbol{J}_n \boldsymbol{W}(t) + \boldsymbol{R}_n(\boldsymbol{W}(t)), \tag{3.1}$$

where $\boldsymbol{R}_n(\boldsymbol{W}(t))$ is the associated nonlinear remainder: $\boldsymbol{R}_n(\boldsymbol{W}) = \boldsymbol{F}(\boldsymbol{W}) - \boldsymbol{J}_n \boldsymbol{W}$. Multiplying (3.1) by the integrating factor $e^{-\boldsymbol{J}_n t}$, and then integrating the system from $t = t_n$ to $t = t_{n+1}$, we obtain

$$\boldsymbol{W}(t_{n+1}) = e^{\Delta t \boldsymbol{J}_n} \boldsymbol{W}(t_n) + e^{\Delta t \boldsymbol{J}_n} \int_0^{\Delta t} e^{-\tau \boldsymbol{J}_n} \boldsymbol{R}_n(\boldsymbol{W}(t_n + \tau)) d\tau. \tag{3.2}$$

The essence of ETD methods is to approximate the nonlinear term $\boldsymbol{R}_n(\boldsymbol{W}(t_n+\tau))$ appearing in the integrand of (3.2) for $\tau\in[0,\Delta t]$ [2]. General three-stage, third-order ETD-RK schemes were proposed in [42]. In our implementation, we use the following ETD-RK3 scheme:

$$\begin{aligned}
\boldsymbol{w}_{n,1} &= \boldsymbol{W}_n+\frac{1}{2}\Delta t\varphi_1(\frac{1}{2}\Delta t\boldsymbol{J}_n)\boldsymbol{F}(\boldsymbol{W}_n),\\
\boldsymbol{w}_{n,2} &= \boldsymbol{W}_n+\frac{2}{3}\Delta t\varphi_1(\frac{2}{3}\Delta t\boldsymbol{J}_n)\boldsymbol{F}(\boldsymbol{W}_n)+\frac{8}{9}\Delta t\varphi_2(\frac{2}{3}\Delta t\boldsymbol{J}_n)\Big(\boldsymbol{R}_n(\boldsymbol{w}_{n,1})-\boldsymbol{R}_n(\boldsymbol{W}_n)\Big),\\
\boldsymbol{W}_{n+1} &= \boldsymbol{W}_n+\Delta t\varphi_1(\Delta t\boldsymbol{J}_n)\boldsymbol{F}(\boldsymbol{W}_n)+\frac{3}{2}\Delta t\varphi_2(\Delta t\boldsymbol{J}_n)\Big(\boldsymbol{R}_n(\boldsymbol{w}_{n,2})-\boldsymbol{R}_n(\boldsymbol{W}_n)\Big),
\end{aligned}\tag{3.3}$$

where the two φ-functions are defined as

$$\varphi_1(z)=\frac{e^z-1}{z},\quad \varphi_2(z)=\frac{\varphi_1(z)-1}{z}=\frac{e^z-1-z}{z^2}.$$

It can be seen from (3.3) that the products of matrix exponential functions with vectors are required. Since the Jacobian matrix appearing in many practical problems is typically large and sparse, how to quickly evaluate the matrix exponential vector products is essential in the simulations. The Krylov subspace algorithms provide an efficient way to perform these calculations.

3.2 Krylov subspace algorithms

By projecting the matrix onto a subspace of small dimension, the Krylov subspace algorithms are able to evaluate the matrix exponential function in a more efficient way than the original problem. In addition, one doesn't need to form the matrix explicitly in this approach, because only the action of the matrix on single vectors is required.

The Krylov subspace of dimension m for a matrix $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ and a vector $\boldsymbol{b}\in\mathbb{R}^n$ is defined as

$$K_m(\boldsymbol{A},\boldsymbol{b})=\text{span}\{\boldsymbol{b},\boldsymbol{A}\boldsymbol{b},\ldots,\boldsymbol{A}^{m-1}\boldsymbol{b}\},\tag{3.4}$$

where $m\ll n$. Applying the Arnoldi process leads to the orthonormal basis $\{\boldsymbol{v}_i\}_{i=1}^m$ of $K_m(\boldsymbol{A},\boldsymbol{b})$: for $j=1,2,\ldots,m$,

$$\begin{aligned}
\widetilde{\boldsymbol{v}}_{j+1} &= \boldsymbol{A}\boldsymbol{v}_j-\sum_{i=1}^{j}h_{i,j}\boldsymbol{v}_i, && h_{i,j}=(\boldsymbol{v}_i,\boldsymbol{A}\boldsymbol{v}_j),\quad \text{for } i=1,2,\ldots,j,\\
\boldsymbol{v}_{j+1} &= \frac{1}{h_{j+1,j}}\widetilde{\boldsymbol{v}}_{j+1}, && h_{j+1,j}=\|\widetilde{\boldsymbol{v}}_{j+1}\|,
\end{aligned}$$

where $\boldsymbol{v}_1=\boldsymbol{b}/\|\boldsymbol{b}\|$, $(\cdot,\cdot)$ denotes the Euclidean inner product in $\mathbb{R}^n$, and $\|\cdot\|=\sqrt{(\cdot,\cdot)}$ is the associated induced norm. Let $\boldsymbol{V}_m=[\boldsymbol{v}_1,\ldots,\boldsymbol{v}_m]$ be the $n\times m$ matrix formed by the orthonormal basis vectors $\{\boldsymbol{v}_i\}_{i=1}^m$, and $\boldsymbol{H}_m$ the $m\times m$ upper Hessenberg matrix consisting of the coefficients $h_{i,j}$ obtained from the Arnoldi process. We then have the following relation:

$$\boldsymbol{A}\boldsymbol{V}_m=\boldsymbol{V}_m\boldsymbol{H}_m+h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^T,\tag{3.5}$$

where $\boldsymbol{e}_m=(0,\ldots,0,1)^T\in\mathbb{R}^m$ is the m-th canonical basis vector of $\mathbb{R}^m$. It follows from the above relation that the Hessenberg matrix $\boldsymbol{H}_m=\boldsymbol{V}_m^T\boldsymbol{A}\boldsymbol{V}_m$, which means that $\boldsymbol{H}_m$ is the projection of $\boldsymbol{A}$ onto the Krylov subspace with respect to the orthonormal basis $\{\boldsymbol{v}_i\}_{i=1}^m$. Finally, the

approximation of matrix exponential vector products, $\varphi_s(\Delta t\boldsymbol{A})\boldsymbol{b}$ for an integer $s\geq 0$, can be given as in [13] by

$$\varphi_s(\Delta t\boldsymbol{A})\boldsymbol{b}\approx \boldsymbol{V}_m\varphi_s(\Delta t\boldsymbol{V}_m^T\boldsymbol{A}\boldsymbol{V}_m)\boldsymbol{V}_m^T\boldsymbol{b}=\|\boldsymbol{b}\|\boldsymbol{V}_m\varphi_s(\Delta t\boldsymbol{H}_m)\boldsymbol{e}_1, \tag{3.6}$$

where $\boldsymbol{e}_1$ is the first canonical basis vector of $\mathbb{R}^m$. Because the size of $\boldsymbol{H}_m$ is small, the evaluation $\varphi_s(\Delta t\boldsymbol{H}_m)\boldsymbol{e}_1$ is computationally cheap and can be computed using, for instance, a dense Padé approximation, see [12] for the details.

However, it has been observed that the Arnoldi process has taken a big part of computational efforts in numerical simulations [19]. To further reduce computational cost, the incomplete orthogonalization method (IOM) has been used instead of the standard Arnoldi process. The IOM was originally proposed in [43] to compute the eigenvalues of large unsymmetric matrices, and was recently applied to perform an efficient exponential time integration for the advection-diffusion equation [44] and for the shallow water equations on the sphere [15, 19]. We follow [15] and use the IOM with an orthogonalization of length 2 (IOM2): for $j=1,2,\ldots,m$,

$$\begin{aligned}
&\widetilde{\boldsymbol{v}}_{j+1}=\boldsymbol{A}\boldsymbol{v}_j-\sum_{i=\max(1,j-2)}^{j}h_{i,j}\boldsymbol{v}_i, && h_{i,j}=(\boldsymbol{v}_i,\boldsymbol{A}\boldsymbol{v}_j),\quad \text{for } i=\max(1,j-2),\ldots,j,\\
&\boldsymbol{v}_{j+1}=\frac{1}{h_{j+1,j}}\widetilde{\boldsymbol{v}}_{j+1}, && h_{j+1,j}=\|\widetilde{\boldsymbol{v}}_{j+1}\|.
\end{aligned}$$

It is worthwhile to note that $\{\boldsymbol{v}_i\}_{i=1}^m$ are only orthonormal locally, *i.e.*,

$$(\boldsymbol{v}_i,\boldsymbol{v}_j)=\delta_{ij},\quad \textit{for } |i-j|\leq 2.$$

where δ_{ij} is the Kronecker delta. The phipm/IOM2 solver developed in [19] is based on this idea and the adaptivity of Krylov subspaces.

4 Localized exponential time differencing method

In this section, we propose a localized ETD (LETD) method with natural scalability, that synthesizes the overlapping spatial domain decomposition and the ETD time integration.

For a given domain Ω, we construct an overlapping decomposition of Ω by first partitioning it into K non-overlapping subdomains $\{\widetilde{\Omega}_k\}_{k=1}^K$, namely the control regions, as shown in Figure 2 for the case of two subdomains (extension to the multiple subdomains is straightforward). In practice, mesh partitioning tools, such as METIS [45], can be used to generate such a non-overlapping partition of Ω. Then the overlapping subdomains Ω_k are obtained by extending $\widetilde{\Omega}_k$ to its neighbors by a fixed distance Δ_k (or a certain number of layers of the mesh) and we write $\Omega_k=\widetilde{\Omega}_k\cup B_k$, where B_k is called the associated "buffer zone". In the following, we present the proposed LETD algorithm for the single-layer SWEs. The method, however, can be applied directly to the multilayer case, and will be tested numerically in Section 5.

The subdomain problems find the subdomain solutions $h_{(k)}$ and $\boldsymbol{u}_{(k)}$ at t_{n+1} on Ω_k, for $k=1,\ldots,K$, given the solution at t_n and appropriate boundary conditions on $\partial\Omega$, such that

$$\begin{cases}
\dfrac{\partial h_{(k)}}{\partial t}+\nabla\cdot(h_{(k)}\boldsymbol{u}_{(k)})=0, & \text{in } \Omega_k\times(t_n,t_{n+1}),\\
\dfrac{\partial \boldsymbol{u}_{(k)}}{\partial t}+q(h_{(k)}\boldsymbol{u}_{(k)}^{\perp})+\nabla\left(\dfrac{|\boldsymbol{u}_{(k)}|^2}{2}+g(h_{(k)}+b)\right)=\boldsymbol{G}(h_{(k)},\boldsymbol{u}_{(k)}), & \text{in } \Omega_k\times(t_n,t_{n+1}),\\
h_{(k)}(\boldsymbol{x},t_n)=h(\boldsymbol{x},t_n),\boldsymbol{u}_{(k)}(\boldsymbol{x},t_n)=\boldsymbol{u}(\boldsymbol{x},t_n), & \text{in } \Omega_k,
\end{cases} \tag{4.1}$$

where $h(\boldsymbol{x},t)$ and $\boldsymbol{u}(\boldsymbol{x},t)$ denote the global solution constructed from the subdomain solutions:

$$(h(\boldsymbol{x},t),\boldsymbol{u}(\boldsymbol{x},t)) = (h_{(k)}(\boldsymbol{x},t),\boldsymbol{u}_{(k)}(\boldsymbol{x},t)), \quad \text{if } \boldsymbol{x}\in\widetilde{\Omega}_k. \tag{4.2}$$

Since the information in a hyperbolic system is propagated along the characteristics, we assume the following relation between the size of buffer zone Δ_k and the time step size $\Delta t = t_{n+1}-t_n$:

$$\Lambda_{\max,n}\Delta t < \min_{k\in\{1,\dots,K\}} \Delta_k, \tag{4.3}$$

in order to make sure that the subdomain problems provide the same approximation as the global scheme inside their control regions, where $\Lambda_{\max,n} = \Lambda_{\max}(h(\cdot,t_n),\boldsymbol{u}(\cdot,t_n))$ is the largest wave propagation speed of the system at t_n. Under this condition, the solution at time level t_{n+1} in the control domain $\widetilde{\Omega}_k$ can be completely determined by the solution at the time level $t = t_n$ in its computational domain Ω_k. See Figure 2 for an illustration for the case of two subdomains using the wave propagation with a constant right-going speed. In such a situation, it is obvious that the domain of dependence of the solution in $\widetilde{\Omega}_k$ at $t = t_{n+1}$ is part of its computational domain Ω_k if condition (4.3) is satisfied.

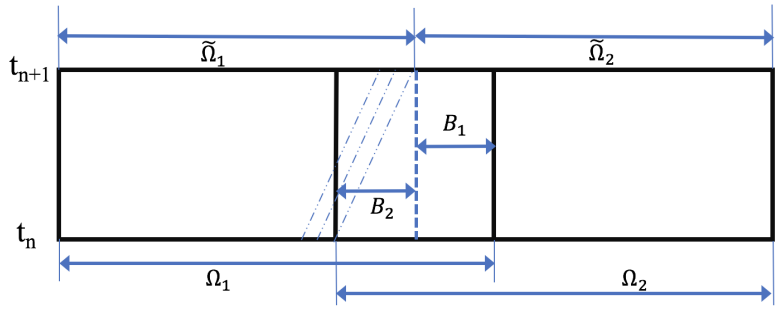

Figure 2: A decomposition of two overlapping subdomains in one dimension.

The localized ETD method is to solve subdomain problems at each time step using the TRiSK scheme for the spatial discretization and the ETD-RK schemes for time integration. Following [37], we define the characteristics time step size Δt_C by $\Delta t_C = 1/\lambda_{\max,n}$, where $\lambda_{\max,n}$ is the largest (in absolute magnitude) discrete wave propagation speed of the fully discrete system at t_n. Let $\Delta t_{CFL} = c\Delta t_c$ with the Courant number $c<1$. For numerical simulations, we enforce the discrete version of the constraint (4.3):

$$\Delta t < \min_{k=1,\dots,K} \Delta_k \cdot \Delta t_{CFL}. \tag{4.4}$$

Under this condition, the LETD algorithm reads as follows: for $n=0,\dots,M-1$,

(1) Given the numerical solution $\boldsymbol{W}_n$ of (3.1) at t_n;

(2) Set $\boldsymbol{W}_{(k),n} = \boldsymbol{W}_n|_{\Omega_k}$, for $k=1,\dots,K$;

(3) Use the TRiSK scheme for the spatial discretization and evolve the semi-discrete system with an ETD-RK scheme locally in each Ω_k to $t=t_{n+1}$ with the time step size Δt satisfying (4.4);

(4) Update $\boldsymbol{W}_{n+1}$ by $\boldsymbol{W}_{n+1}|_{\widetilde{\Omega}_k} = \boldsymbol{W}_{(k),n+1}|_{\widetilde{\Omega}_k}$, for $k=1,\dots,K$.

Remark 4.1. The solution to the semi-discrete system (2.6) satisfies conservation of total energy by the construction of the TRiSK scheme (for more details, see [27]). Thus application of the

ETD or LETD methods to (2.6) results in a numerical scheme in which the total energy is conserved within time truncation errors. For the conservation of mass, it has been proved in [37] that the ETD solution of the semi-discrete system (2.6) conserves mass exactly. In the case of the LETD method with overlapping subdomains, because of the local matrix exponentials, the LETD solutions in the overlapping region shared by different subdomains may not be exactly the same. For instance, for the two subdomains Ω_1 and Ω_2 as in Figure 2, $\boldsymbol{W}_{(1),n+1}|_{\Omega_1\cap\Omega_2}$ may not exactly match $\boldsymbol{W}_{(2),n+1}|_{\Omega_1\cap\Omega_2}$, though the difference is very small and in the order of time truncation errors. Note that the same behavior has been observed for diffusion equations in [22, 24]). However, as we shall numerically verify in Section 5, the LETD method has the same accuracy as the global ETD method, and it inherits all desirable conservation properties of the TRiSK scheme, namely, conservation of mass, total energy and PV for long time horizons.

5 Numerical experiments

The goal of this section is twofold: first, we investigate the numerical behavior of the ETD-RK method (in particular, the ETD-RK3 scheme (3.3) is used) on single-layer and multilayer SWEs, and compare its performance with the standard RK method; second, we test the LETD method and demonstrate its efficiency over its global counterpart in the sequential setting. Note that we do not investigate the corresponding performance of the LETD method in parallel implementation in this paper and would like to leave it as future work. We shall consider three test cases: the test case of simulating ocean mesoscale activity (SOMA) [46], the shallow water test case 5 (SWTC5) and test case 6 (SWTC6) from the standard shallow-water test case suite in Williamson et al. [41]. We use the phipm/IOM2 solver [19] as presented in Subsection 3.2 for matrix exponential vector products, where we set the initial dimension of Krylov subspace to be $m=30$, and the maximum Krylov subspace dimension $m_{\max}=100$. We compute the relative errors between the numerical solution and the reference solution as:

$$\mathcal{E}_h=\frac{\max_{j=1,\dots,N_c}|\mathfrak{h}_n(j)-\mathfrak{h}_n^{ref}(j)|}{\max_{j=1,\dots,N_c}|\mathfrak{h}_n^{ref}(j)|},\quad \mathcal{E}_{\boldsymbol{u}}=\frac{\max_{j=1,\dots,N_e}|\mathfrak{u}_n(j)-\mathfrak{u}_n^{ref}(j)|}{\max_{j=1,\dots,N_e}|\mathfrak{u}_n^{ref}(j)|}, \tag{5.1}$$

where $\mathfrak{h}_n^{ref}=\left(h_{1,n}^{ref},\dots,h_{N_c,n}^{ref}\right)$, and $\mathfrak{u}_n^{ref}=\left(u_{1,n}^{ref},\dots,u_{N_e,n}^{ref}\right)$ denote the reference solution at time t_n obtained by RK4 with a small time step size $\Delta t_{ref}=0.001\Delta t_C$. The physical parameters used in the test cases are the radius of earth $R=6.37122\times10^6$ m, the gravity acceleration $g=9.80616$ m$\cdot$ s^{-2}, and the angular velocity of earth $\Omega_0=7.292\times10^{-5}$ s^{-1}. All tests are implemented on Dell Precision 5530 with Intel (R) Xeon (R) E-2176M CPU @ 2.70 GHz and 32 Gb memory.

5.1 Single-layer rotating SWEs

Three test cases are considered for studying the proposed algorithms on the single-layer rotating SWEs: 1) a simplified version of the SOMA test case with a double-gyre circulation, where the same geophysical domain and bathymetry are used, but neither wind stress nor bottom friction is applied; 2) two test cases of flow on the sphere of earth - the SWTC5 of zonal flow over an isolated mountain and the SWTC6 of Rossby-Haurwitz wave.

5.1.1 The SOMA test case

The same geometrical setting as in [37, 46] is considered: the spatial domain is a circular basin centered at the point $\boldsymbol{x}_c$ of latitude $\theta_c=35^\circ$ and longitude $\alpha_c=0^\circ$ with radius 1250 km, lying on the surface of earth. The fluid depth in the basin varies from 2500 m at the center to 100 m on the coastal shelf. The initial fluid depth is denoted by $h_0=-b+\eta_0$, where $b<0$ represents

the bottom topography and $\eta_0 = \bar{\eta} e^{-(x-x_c)^2/(2\sigma^2)}$ is a Gaussian-type function with $\bar{\eta} = 2$ m and $\sigma = 200$ km. The initial velocity is chosen as $\boldsymbol{u}_0 = \frac{g}{f(\boldsymbol{x}_c)} \boldsymbol{k} \times \nabla \eta_0$, where $\boldsymbol{k}$ is the local vertical unit vector. With such a choice of initial conditions, it can be shown that geostrophic balance holds, that is, initial velocity satisfies $\nabla \cdot \boldsymbol{u}_0 = 0$ and the pressure gradient $g \nabla \eta_0$ balances the Coriolis force $f \boldsymbol{k} \times \boldsymbol{u}_0$. The velocity field and initial sea surface height η_0 are shown in Figure 3.

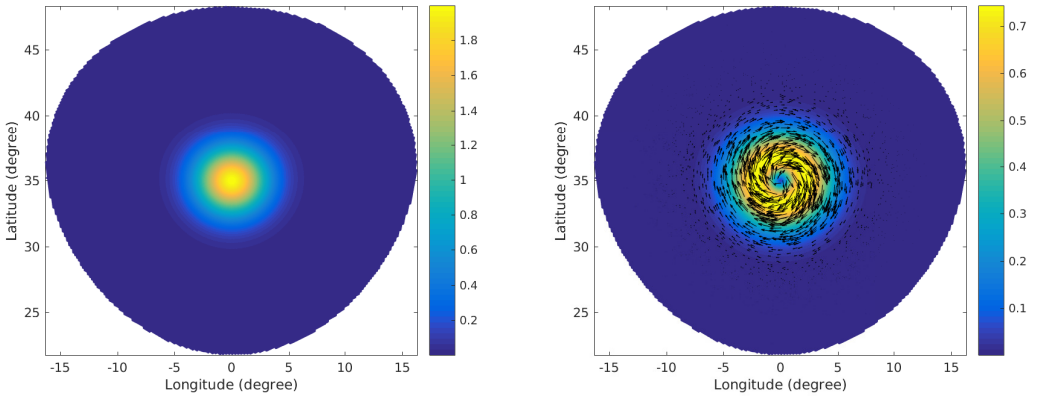

Figure 3: Initial conditions for the SOMA test: the sea surface height measured in meters (left); the velocity field measured in meters/second (right).

Remark 5.1. To evolve the semi-linear system (2.7) obtained by the TRiSK scheme, the normal component of the initial velocity evaluated at the center of every primal edge e is required. In this test case, the normal velocity at the center of edge e is $u_{0,e} = \boldsymbol{u}_0 \cdot \boldsymbol{n}_e = \frac{g}{f(\boldsymbol{x}_c)} (\boldsymbol{k} \times \nabla \eta_0) \cdot \boldsymbol{n}_e = -\frac{g}{f(\boldsymbol{x}_c)} (\boldsymbol{k} \times \boldsymbol{n}_e) \cdot \nabla \eta_0 = -\frac{g}{f(\boldsymbol{x}_c)} \nabla \eta_0 \cdot \boldsymbol{t}_e$, where $\boldsymbol{t}_e = \boldsymbol{k} \times \boldsymbol{n}_e$ is the unit tangential vector along edge e, and the function η_0 is evaluated at the center of edge e. Therefore, the term $\nabla \eta_0 \cdot \boldsymbol{t}_e$ is the tangential derivative of η_0 evaluated at the center of edge e, which can be simply approximated by using the finite difference scheme. This technique is also used in other test cases in this section.

The ETD-RK3 (3.3) is used for time integration. During the simulations, we only update the Jacobian matrix once every 10 time steps in order to improve the efficiency. Three quasi-uniform meshes of different resolutions are considered in our simulations:

1) 32 km resolution with 8,521 cells, 25,898 edges, and 17,378 vertices;

2) 16 km resolution with 30,217 cells, 91,285 edges and 61,069 vertices;

3) 8 km resolution with 120,953 cells, 364,124 edges and 243,172 vertices.

The numerical errors for ETD-RK3 on three meshes subject to different time step sizes are listed in Table 1, and the corresponding simulation time is listed in Table 2. It is observed that (i) as the time step size decreases successively, the ETD-RK3 converges at the optimal rate on all the meshes; (ii) as the time step size increases, the total CPU time decreases as fewer steps are needed for completing simulations. However, for large time step sizes, the CPU time per step increases significantly since more inner steps are used in matrix exponential evaluation; consequently, the total CPU time may not monotonically decrease.

To further analyze the performance of the ETD-RK3 scheme, we compare it with the standard RK3 scheme. In Figure 4, we plot the numerical errors in h versus the total simulation time based on the data in Tables 1-2 together with the results obtained by RK3 simulations

Δt	32 km				16 km				8 km			
Δt_{CFL}	$\mathcal{E}_h$		$\mathcal{E}_u$		$\mathcal{E}_h$		$\mathcal{E}_u$		$\mathcal{E}_h$		$\mathcal{E}_u$	
160	1.03e-07	-	8.50e-05	-	8.46e-08	-	4.25e-05	-	9.60e-09	-	7.19e-06	-
80	5.74e-08	[0.84]	4.90e-05	[0.80]	1.39e-08	[2.61]	9.12e-06	[2.22]	1.75e-09	[2.46]	1.20e-06	[2.59]
40	8.65e-09	[2.73]	8.13e-06	[2.59]	1.84e-09	[2.92]	1.59e-06	[2.52]	1.78e-10	[3.30]	1.53e-07	[2.96]
20	8.10e-10	[3.42]	6.61e-07	[3.62]	2.07e-10	[3.15]	1.88e-07	[3.08]	2.39e-11	[2.90]	1.66e-08	[3.21]
10	6.43e-11	[3.66]	6.71e-08	[3.30]	1.89e-11	[3.45]	2.04e-08	[3.20]	2.24e-12	[3.41]	1.44e-09	[3.53]
5	5.75e-12	[3.48]	5.93e-09	[3.50]	1.63e-12	[3.54]	1.45e-09	[3.81]	2.04e-13	[3.46]	1.27e-10	[3.50]

Table 1: (SOMA) The errors of fluid thickness and velocity obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$ on meshes with different resolutions at Day 1.

Δt	32 km		16 km		8 km	
Δt_{CFL}	time/step	total time	time/step	total time	time/step	total time
160	0.43	5.98	3.50	101.49	14.52	842.32
80	0.20	5.55	1.52	86.70	8.19	949.62
40	0.15	8.11	0.85	95.58	3.94	909.46
20	0.15	16.27	0.77	173.43	3.56	1642.59
10	0.15	31.94	0.77	347.59	3.70	3412.09
5	0.15	63.85	0.77	696.82	3.71	6849.96

Table 2: (SOMA) The average CPU time for each time stepping and total CPU time obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$ during one day's simulation. The simulation time is measured in seconds.

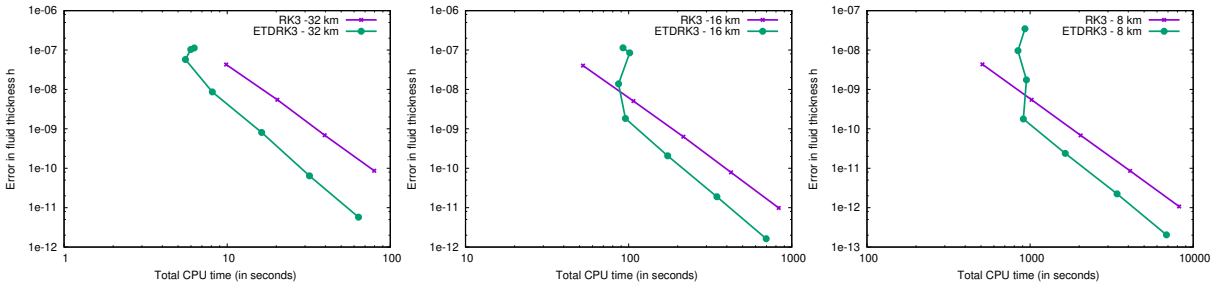

Figure 4: (SOMA) Errors in fluid thickness h versus total CPU time for the one-day simulations on meshes with 32 km resolution (left), 16 km resolution (middle) and 8 km resolution (right).

over the same time interval. We see that for the mesh with 32 km resolution, the ETD-RK3 is more efficient than RK3; for the mesh with 16 km resolution, when the error is larger than 10^{-8}, RK3 is more efficient than ETD-RK3, whereas for an error threshold less than 10^{-8}, ETD-RK3 becomes more efficient; for the mesh with 8 km resolution, when the error threshold is less than 8×10^{-10} or so, ETD-RK3 is more efficient than RK3. In addition, as the mesh becomes finer, the computational time for ETD-RK3 increases quickly due to the evaluation of matrix exponential vector products becomes more expensive.

This observation motivates the study of localized ETD-RK schemes. We first divide the spatial domain using METIS. Figure 5 provides an illustration for partitioning the mesh with a 32 km resolution into 2, 4 and 8 non-overlapping subdomains, which represent the control regions of subdomains. We then add a buffer zone including 10 layers of neighboring cells to each control region, which yields the computational domain in each subdomain problem.

During one step of the simulations, all the subdomain problems are solved by ETD-RK3 individually with time step $\Delta t = 10\Delta t_{CFL}$, then information on the overlapping zone is exchanged, and the global solution is advanced to next time level. We consider different numbers of sub-

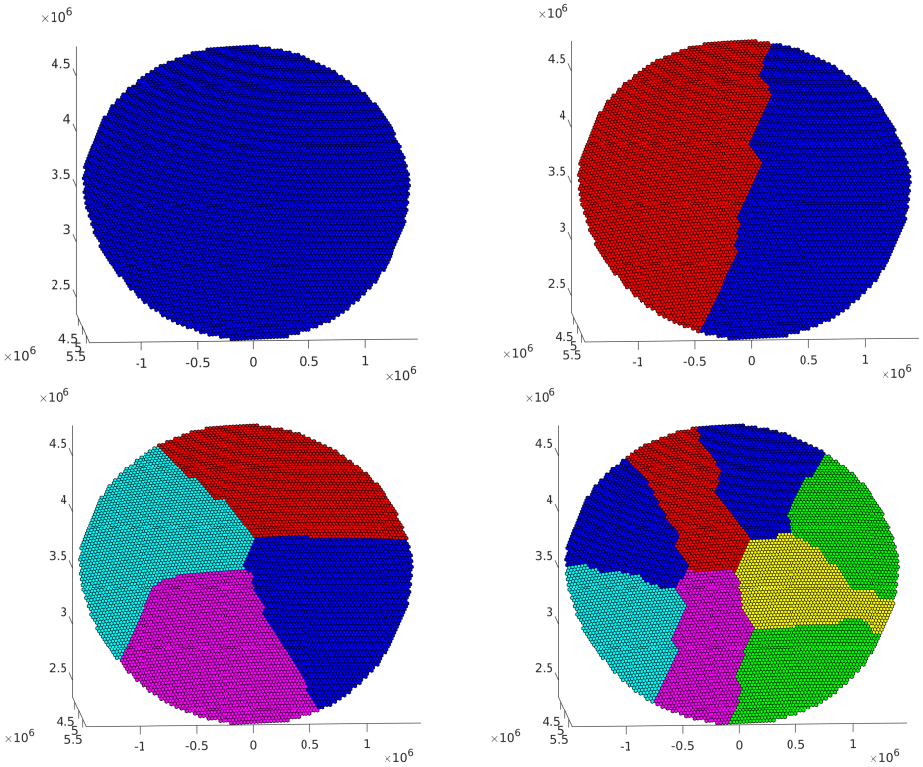

Figure 5: The primal mesh of 8521 cells and subdomains decomposed by METIS. Top-left: the computational domain; top-right: two subdomains; bottom-left: four subdomains; bottom-right: eight subdomains.

domains, and compute the errors in fluid thickness and velocity on the three meshes after a one-day simulation as shown in Table 3. The associated CPU time is displayed in Table 4. Obviously, when the number of subdomains equals one, the LETD-RK3 coincides with the global ETD-RK3 scheme. It can be seen from Table 3 that the numerical accuracy of LETD-RK3 is identical to its global counterpart. This is expected as the size of buffer zone and time step size satisfy the relation (4.4) in our computational setting. The benefit of LETD-RK schemes can be deduced from Table 4. On the mesh of the 32 km resolution, the total CPU time consumed by LETD-RK3 with 32 subdomains is about six times of that of global ETD-RK3 scheme, while on the mesh of the 8 km resolution, the total CPU time of LETD-RK3 using 32 subdomains is almost identical to that of global ETD-RK3 scheme. Note that these results are obtained with sequential computing but the subdomain problems can be solved naturally in parallel, thus we expect that the CPU time could be significantly reduced when the LETD-RK schemes are combined with parallel computing, especially for meshes with fine resolutions.

Finally, we consider 15-day-long simulations. Here, we only present the results obtained on the 16-km-resolution mesh because the behaviors of the schemes on other meshes are similar. The numerical solutions at $T=15$ days obtained by ETD-RK3 are shown in Figure 6. The LETD-RK3 with 8 subdomains achieves the same results. The evolution of the relative changes in total energy and mass obtained both schemes is plotted in Figure 7, which shows that both schemes have good conservation properties.

No. of subdomains	32 km		16 km		8 km	
	$\mathcal{E}_h$	$\mathcal{E}_u$	$\mathcal{E}_h$	$\mathcal{E}_u$	$\mathcal{E}_h$	$\mathcal{E}_u$
1	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09
2	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09
4	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09
8	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09
16	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09
32	6.4298e-11	6.7118e-08	1.8907e-11	2.0411e-08	2.2431e-12	1.4377e-09

Table 3: (SOMA) The errors of fluid thickness and velocity by the LETD-RK3 scheme on meshes with different resolutions and different number of subdomains at Day 1.

No. of subdomains	32 km		16 km		8 km	
	time/step	total time	time/step	total time	time/step	total time
1	0.15	31.94	0.77	347.59	3.70	3412.09
2	0.17	37.12	0.74	335.76	2.63	2429.60
4	0.21	46.55	0.80	362.21	2.84	2622.96
8	0.33	72.86	0.94	422.73	3.17	2924.95
16	0.48	106.08	1.15	519.51	3.69	3403.89
32	0.83	181.36	1.50	679.98	4.28	3954.93

Table 4: (SOMA) The CPU time for each time stepping and total CPU time of LETD-RK3 during a one day's simulation on meshes with different resolutions and different number of subdomains. The simulation time is measured in seconds.

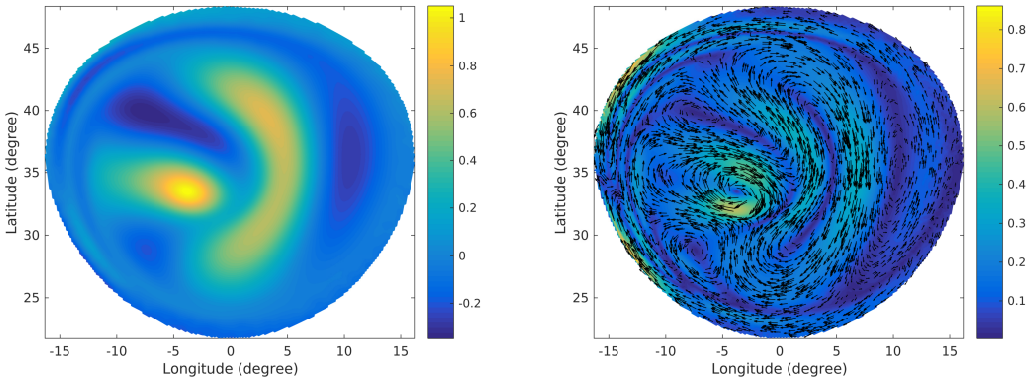

Figure 6: (SOMA) Snapshots of sea surface height with unit of m (left) and the velocity with unit of m $\cdot$ s^{-1} (right) at $T=15$ days using ETD-RK3 with time step $\Delta t = 10\Delta t_{CFL}$.

5.1.2 The SWTC5: Zonal flow over an isolated mountain

We now consider the SWTC5 in [27,41] on the single-layer configuration. The physical domain is the whole sphere of earth, and the initial state consists of a zonal flow impinging on an isolated mountain located around the longitude $\alpha_c = 3\pi/2$ and latitude $\theta_c = \pi/6$. The height of the mountain is given by

$$h_s = h_{s_0}(1 - r/a), \tag{5.2}$$

where $h_{s_0} = 2000$ m, $a = \pi/9$, $r^2 = \min\{a^2, (\alpha-\alpha_c)^2 + (\theta-\theta_c)^2\}$, with α and θ being longitude and latitude, respectively. The initial horizontal velocity in the longitudinal and latitudinal

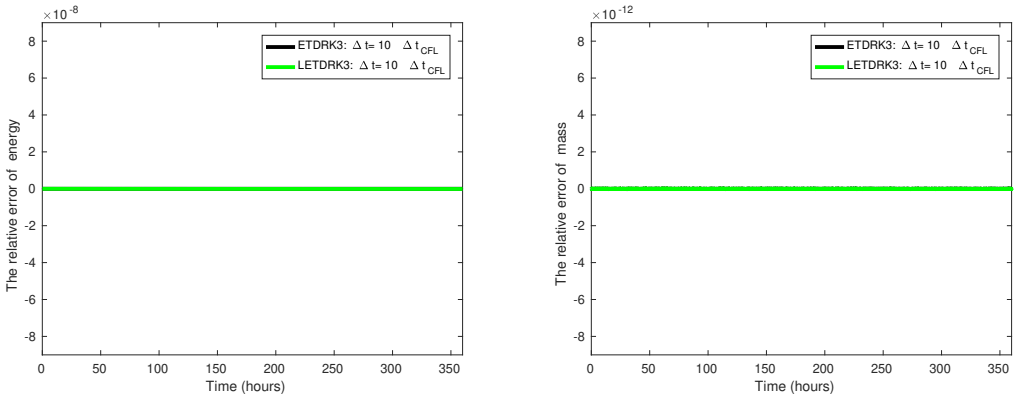

Figure 7: (SOMA) Evolution of the relative changes in total energy (left) and mass (right) for T = 15 days by ETD-RK3 and LETD-RK3 using the time step size $\Delta t = 10\Delta t_{CFL}$.

directions is $(u,v) = (\tilde{u}_0 \cos\theta, 0)$, where $\tilde{u}_0 = 20\ m/s$. Note that the initial horizontal velocity $\boldsymbol{u}_0$ can also be defined by the stream function, i.e., $\boldsymbol{u}_0 = \boldsymbol{k} \times \psi$, where $\psi = -R\tilde{u}_0 \sin\theta$ is the stream function, see [41]. The initial fluid thickness is

$$h = h_0 - h_{s_0} - \frac{1}{g}\left(R\Omega_0\tilde{u}_0 + \frac{\tilde{u}_0^2}{2}\right)\sin^2\theta, \tag{5.3}$$

where $R = 6371.22$ km is the radius of earth, and $h_0 = 5960$ m. The bottom topography b and the initial surface height $dh = h + b$ are shown in Figure 8.

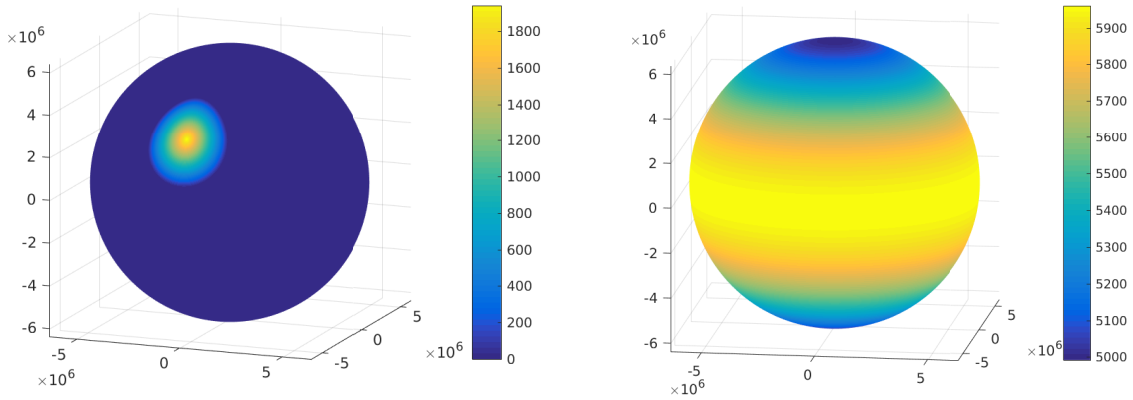

Figure 8: (SWTC5) The bottom topography b (left), and the initial surface height $dh = h + b$ (right).

Two resolutions are considered for the quasi-uniform mesh used in the simulations:

1) 60 km resolution with 163,842 cells, 491,520 edges and 327,680 vertices;

2) 30 km resolution with 655,362 cells, 1,966,080 edges and 1,310,720 vertices.

The numerical errors obtained by using the ETD-RK3 scheme on two different meshes at $T = 1$ day are listed in Table 5, which shows that the ETD-RK3 scheme achieves the expected accuracy in time. The associated CPU time is presented in Table 6. As for the SOMA test case, we compare the performance of the ETD-RK3 and standard RK3 schemes. Figure 9 displays

the errors in fluid thickness versus the total simulation time according to the data in Tables 5-6 together with the RK3 results for a one-day simulation. It is observed that for an error threshold less than 10^{-5}, the ETD-RK3 is more efficient than RK3.

$\frac{\Delta t}{\Delta t_{CFL}}$	60 km				30 km			
	$\mathcal{E}_h$		$\mathcal{E}_u$		$\mathcal{E}_h$		$\mathcal{E}_u$	
160	4.45e-04	-	2.15e-03	-	1.80e-04	-	1.89e-03	-
80	1.82e-04	[1.29]	1.53e-03	[0.49]	1.19e-04	[0.59]	1.05e-03	[0.85]
40	6.31e-05	[1.53]	5.47e-04	[1.48]	3.64e-05	[1.71]	3.05e-04	[1.78]
20	1.05e-05	[2.59]	1.01e-04	[2.43]	4.72e-06	[2.95]	4.20e-05	[2.86]
10	8.69e-07	[3.59]	7.92e-06	[3.68]	3.49e-07	[3.76]	3.31e-06	[3.66]
5	6.19e-08	[3.81]	6.25e-07	[3.66]	2.56e-08	[3.77]	2.47e-07	[3.74]

Table 5: (SWTC5) The errors in fluid thickness and velocity obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$ on meshes with two different resolutions at Day 1.

$\frac{\Delta t}{\Delta t_{CFL}}$	60 km		30 km	
	time/step	total time	time/step	total time
160	15.79	236.84	80.23	2406.95
80	5.82	168.77	31.91	1882.59
40	3.39	193.24	19.00	2223.15
20	2.93	334.25	16.45	3850.44
10	3.00	680.90	16.60	7752.87
5	2.91	1317.53	16.44	15338.13

Table 6: (SWTC5) The average CPU times for each time stepping and total CPU time during a one day's simulation obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$. The simulation time is measured in seconds.

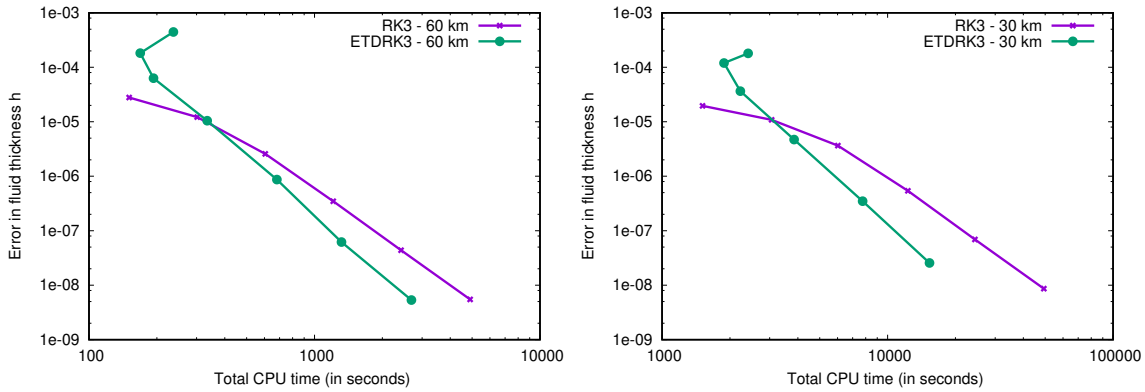

Figure 9: (SWTC5) The error in fluid thickness vs. the total CPU time for one-day simulations on meshes with 60 km resolution (left) and 30 km resolution (right).

Next we investigate the performance of the LETD-RK3 scheme when the time step $\Delta t = 10\Delta t_{CFL}$ and the size of buffer zone is 10. The LETD-RK3 approximation errors in fluid thickness and velocity on two different mesh resolutions at $T = 1$ day are listed in Table 7 where different number of subdomains are used. The results indicate that the LETD-RK3 is able to

provide the same accurate results as the global ETD-RK3. Moreover, the corresponding simulation time presented in Table 8 supports the same observation as in the SOMA testcase, that is, the simulation time could be significantly saved if the LETD-RK3 is implemented in parallel, especially on high-resolution meshes.

No. of subdomains	60 km		30 km	
	$\mathcal{E}_h$	$\mathcal{E}_u$	$\mathcal{E}_h$	$\mathcal{E}_u$
1	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06
2	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06
4	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06
8	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06
16	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06
32	8.6921e-07	7.9254e-06	3.4875e-07	3.3109e-06

Table 7: (SWTC5) The errors of fluid thickness and velocity with the LETD-RK3 scheme on meshes with different resolutions and different number of subdomains at Day 1.

No. of subdomains	60 km		30 km	
	time/step	total time	time/step	total time
1	3.00	680.90	16.60	7752.87
2	2.88	654.82	15.77	7363.72
4	2.85	647.00	14.13	6600.15
8	3.21	729.03	13.40	6256.18
16	3.54	804.89	13.55	6329.23
32	4.10	931.67	15.27	7132.86

Table 8: (SWTC5) The average CPU time for each time stepping and total CPU time of LETD-RK3 during a one day's simulation on meshes with different resolutions and different number of subdomains. The simulation time is measured in seconds.

Finally, we run 15-day-long simulations using the ETD-RK3 and the LETD-RK3 with 8 subdomains, respectively. For the mesh of 60 km resolution, the surface height snapshot at day 15 is shown in Figure 10, together with the time evolution of relative changes in total energy and mass. It is seen that both ETD-RK3 and LETD-RK3 schemes are able to conserve the total energy and mass along the time.

5.1.3 The SWTC6: Rossby-Haurwitz wave

In this subsection, we are concerned with the numerical simulation of zonal wavenumber 4 Rossby-Haurwitz wave which admits the formation of Rossby wave. The physical domain is the sphere of earth, and the two meshes presented in the SWTC5 test case are used. A detailed description of this test case can be found in [41], and we refer to [47–49] for an exhaustive discussion on the physical and numerical phenomena of Rossby-Haurwitz waves. The reader is referred to [32] and the references therein for the use of a high-order upwind finite volume scheme to simulate the equatorial Rossby waves on a planar domain.

The initial velocity field $\boldsymbol{u}_0$ is non-divergent and takes the form of $\boldsymbol{u}_0 = \boldsymbol{k} \times \nabla \psi$ [41,49], where $\boldsymbol{k}$ is the local unit vertical vector, and ψ is the stream function defined by

$$\psi = \omega_0 R^2 (\cos^4\theta \cos 4\alpha - 1) \sin\theta. \tag{5.4}$$

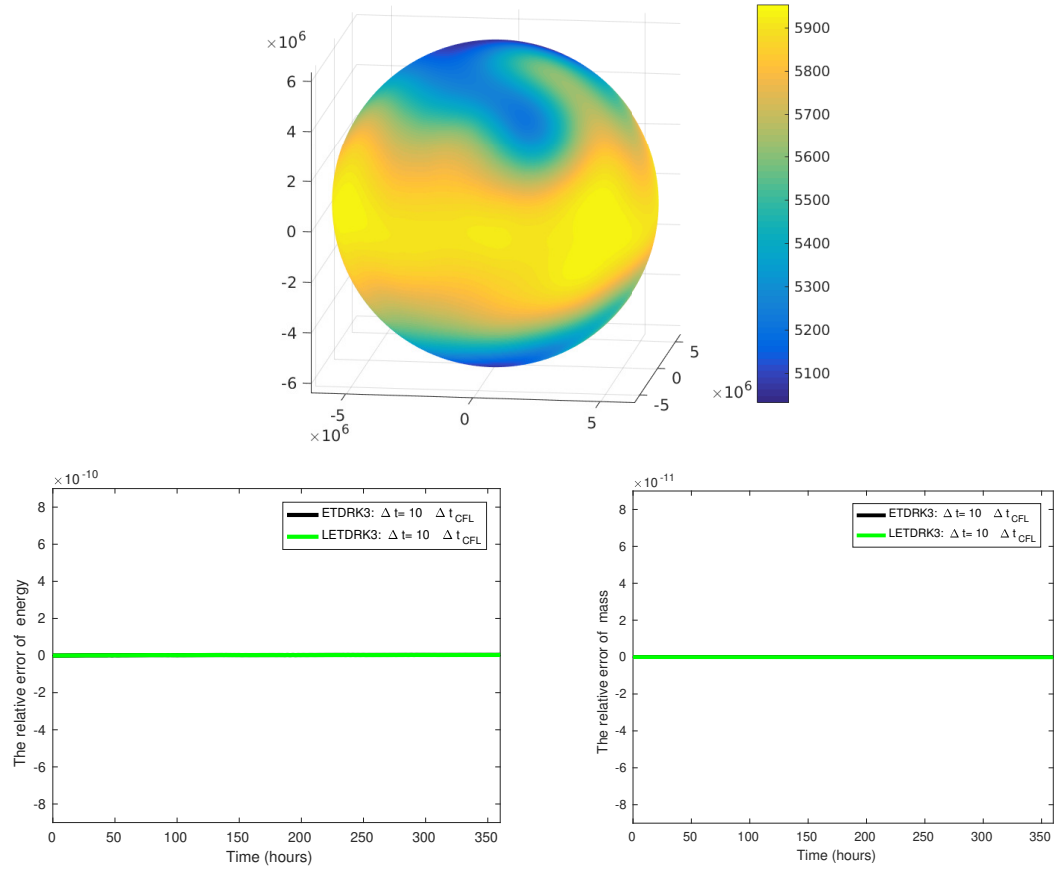

Figure 10: (SWTC5) Fluid height at day $T=15$ on the mesh with the 60 km resolution (top); time evolution of relative changes in total energy (bottom-left) and mass (bottom-right) in the ETD-RK3 and LETD-RK3 simulations using time step size $\Delta t = 10\Delta t_{CFL}$.

Furthermore, the initial fluid thickness is given by

$$h = h_0 + (R^2/g)(A(\theta) + B(\theta)\cos 4\alpha + C(\theta)\cos 8\alpha), \tag{5.5}$$

where

$$A(\theta) = (\omega_0(\omega_0+2\Omega_0)/2)\cos^2\theta + (\omega_0^2/4)\cos^6\theta(5\cos^4\theta + 26\cos^2\theta - 32),$$
$$B(\theta) = (\omega_0(\omega_0+\Omega_0)/15)(26-25\cos^2\theta)\cos^4\theta, \quad C(\theta) = (\omega_0^2/4)(5\cos^2\theta - 6)\cos^8\theta,$$

with R being the radius of earth, Ω_0 the angular velocity of earth, θ and α the latitude and longitude, respectively. The other two parameters are $\omega_0 = 7.848\times 10^{-6}$ s^{-1} and $h_0 = 8000$ m. In this test case, the bottom topography b is flat, and is set to be $b = 0$ m. In Figure 11 (left), we show the contour plot of the initial fluid thickness on mesh resolution of 60 km in the latitude-longitude coordinate system. It can be seen from the figure that the initial thickness field is a perfect four-wavelength zonal pattern. We also present the contour plot of fluid thickness obtained from ETD-RK3 scheme with $\Delta t = 10\Delta t_{CFL}$ on day 15 in Figure 11 (right). It can be observed that the simulation is stable, and the wave propagates steadily eastwards and its initial structure of the wave number four could be maintained with only minor vacillations after a 15-day simulation.

The numerical errors obtained by using the ETD-RK3 scheme on two different meshes at $T = 1$ day are listed in Table 9, which shows that the ETD-RK3 scheme achieves the expected

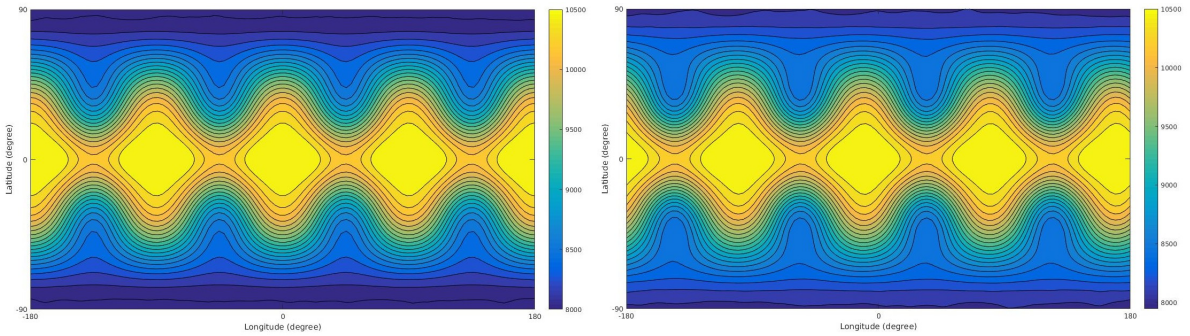

Figure 11: (Rossby-Haurwitz) Contour plot of the initial fluid thickness (left), and its contour plot on day 15 (right) obtained by ETD-RK3 scheme with $\Delta t = 10\Delta t_{CFL}$.

accuracy in time. The associated CPU time is presented in Table 10. As for the two previous test cases, we compare the performance of the ETD-RK3 and standard RK3 schemes. Figure 12 displays the errors in fluid thickness versus the total simulation time according to the data in Tables 9-10 together with the RK3 results for a one-day simulation. It is observed that for the 60-km-resolution mesh with an error threshold less than 6×10^{-6}, and for the 30-km-resolution mesh with a threshold less than 1.0×10^{-4}, the ETD-RK3 is more efficient than RK3.

$\frac{\Delta t}{\Delta t_{CFL}}$	60 km				30 km			
	$\mathcal{E}_h$		$\mathcal{E}_u$		$\mathcal{E}_h$		$\mathcal{E}_u$	
160	1.80e-04	-	5.25e-04	-	2.78e-04	-	5.67e-04	-
80	1.16e-04	[0.64]	6.22e-04	[-]	1.92e-04	[0.53]	4.67e-04	[0.28]
40	1.09e-04	[0.09]	3.97e-04	[0.65]	1.09e-04	[0.82]	3.67e-04	[0.35]
20	4.31e-05	[1.33]	1.42e-04	[1.48]	4.06e-05	[1.42]	1.05e-04	[1.80]
10	3.76e-06	[3.52]	1.76e-05	[3.00]	4.80e-06	[3.08]	8.42e-06	[3.64]
5	2.61e-07	[3.85]	1.30e-06	[3.75]	3.65e-07	[3.72]	6.31e-07	[3.74]

Table 9: (Rossby-Haurwitz) The errors in fluid thickness and velocity obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$ on meshes with two different resolutions at Day 1.

$\frac{\Delta t}{\Delta t_{CFL}}$	60 km		30 km	
	time/step	total time	time/step	total time
160	6.72	194.92	38.25	2256.75
80	4.20	239.18	22.13	2611.04
40	3.58	408.46	17.07	4011.69
20	3.20	729.45	16.81	7884.24
10	3.21	1460.47	16.86	15798.29
5	3.28	2983.39	17.09	32011.47

Table 10: (Rossby-Haurwitz) The average CPU times for each time stepping and total CPU time during a one day's simulation obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 5\Delta t_{CFL}$. The simulation time is measured in seconds.

Next we investigate the performance of the LETD-RK3 scheme when the time step $\Delta t = 10\Delta t_{CFL}$ and the size of buffer zone is 10. The LETD-RK3 approximation errors in fluid thickness and velocity on two different mesh resolutions at $T = 1$ day are listed in Table 11 where

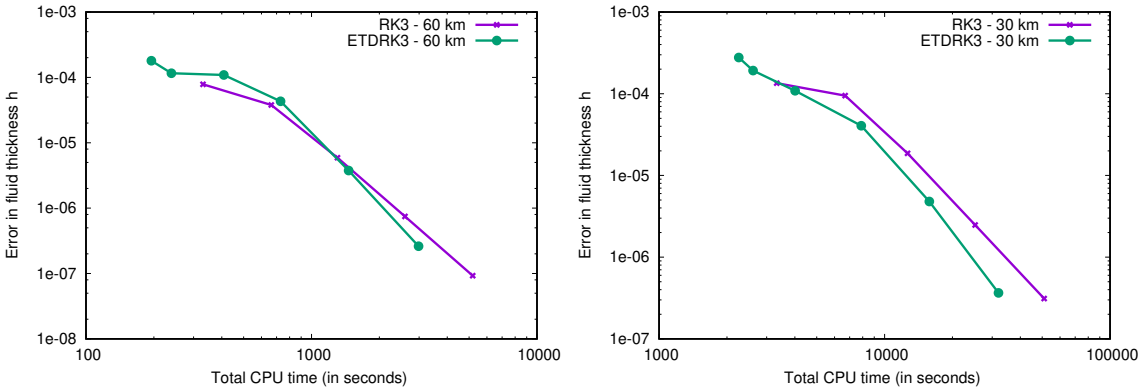

Figure 12: (Rossby-Haurwitz) The error in fluid thickness vs. the total CPU time for one-day simulations on meshes with 60 km resolution (left) and 30 km resolution (right).

different number of subdomains are used. The results indicate that the LETD-RK3 is able to provide the same accurate results as the global ETD-RK3. Moreover, the corresponding simulation time presented in Table 12 supports the same observation as in the last two test cases, that is, the simulation time could be significantly reduced if the LETD-RK3 is implemented in parallel, especially when a high-resolution mesh is used.

No. of	60 km		30 km	
subdomains	$\mathcal{E}_h$	$\mathcal{E}_u$	$\mathcal{E}_h$	$\mathcal{E}_u$
1	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06
2	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06
4	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06
8	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06
16	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06
32	3.7584e-06	1.7605e-05	4.8029e-06	8.4171e-06

Table 11: (Rossby-Haurwitz) The errors of fluid thickness and velocity with the LETD-RK3 scheme on meshes with different resolutions and different number of subdomains at Day 1.

To verify that the ETD-RK3 and LETD-RK3 can conserve both the mass and total energy, we run 15-day-long simulations using these two schemes with time step size $\Delta t = 10\Delta t_{CFL}$, and with 8 subdomains for LETD-RK3. We show the time evolution of relative changes in total energy and mass on the 60-km-resolution mesh in Figure 13. It is seen that both ETD-RK3 and LETD-RK3 schemes are able to conserve the total energy and mass along the time.

5.2 Multilayer SWEs for the SOMA test case

Next, we investigate the numerical performance of the global and localized ETD-RK3 schemes on a three-layer shallow water model for the SOMA test case. The same geometrical domain is considered. The initial interfaces of the three-layer configuration locate at η_1^0=0 km, η_2^0=$-25/3$ km, and η_3^0=$-50/3$ km, and the layer densities are set as ρ=(1025,1027,1028) kg/m^3. In order to make the model more realistic for the ocean modeling, the forcing and bi-harmonic smoothing terms are introduced [37, 46]:

(1) Surface wind stress $f_w = \frac{\tau_w}{\rho_1 h_1}$, which is added to the top layer only;

No. of	60 km		30 km	
subdomains	time/step	total time	time/step	total time
1	3.21	1460.47	16.86	15798.29
2	3.11	1413.58	15.68	14689.70
4	3.10	1408.75	14.55	13629.56
8	3.55	1614.23	13.65	12792.73
16	4.31	1961.10	13.78	12915.68
32	5.18	2355.03	15.68	14690.31

Table 12: (Rossby-Haurwitz) The average CPU time for each time stepping and total CPU time of LETD-RK3 during a one day's simulation on meshes with different resolutions and different number of subdomains. The simulation time is measured in seconds.

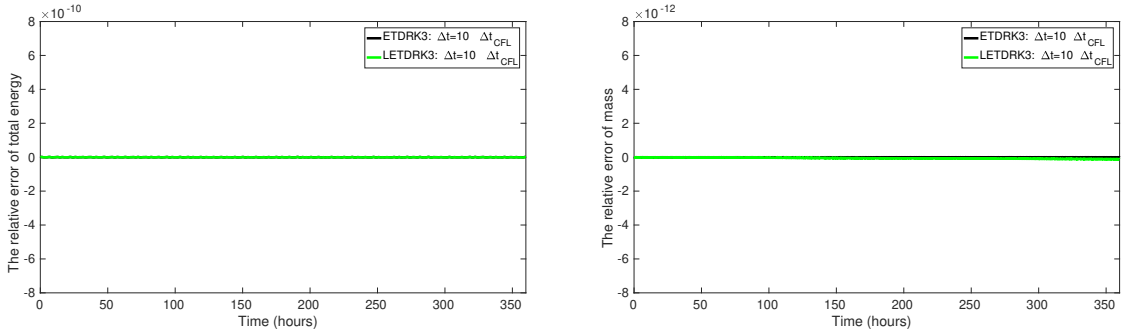

Figure 13: (Rossby-Haurwitz) Time evolution of relative changes in total energy (left), and mass (right) in the ETD-RK3 and LETD-RK3 simulations with $\Delta t = 10\Delta t_{CFL}$.

(2) Bottom friction $\boldsymbol{f}_b = -c_b \frac{|\boldsymbol{u}_b|\boldsymbol{u}_b}{h_b}$ represents the interaction between the flow and the bottom topography, which appears in the bottom layer only. Here, $\boldsymbol{u}_b$ is the velocity of bottom layer and we choose $c_b = 1.0 \times 10^{-3}$ following [37, 46].

(3) Artificial diffusion $f_d = -\epsilon\Delta^2\boldsymbol{u}_k$, which is introduced to every layer in order to overcome the accumulation of turbulent energy in long term simulations.

The same grids are used in all the layers. Two resolutions, 32 km and 16 km, are considered in the test. The total numbers of DOFs are triple of the corresponding single layer cases. As in the single-layer case, we first test the (global) ETD-RK3 scheme, and only update the Jacobian once every 10 time steps. The numerical errors on two meshes are listed in Table 13, and the associated simulation times are recorded in Table 14. The numerical results show that the ETD-RK3 scheme achieves the expected accuracy in time.

We further compare its performance with the standard RK3 scheme. In Figure 14, we plot the error in fluid thickness versus the total simulation time according to Tables 13-14 and the results obtained by RK3 for the one-day simulations. It is seen that for the same error threshold, ETD-RK3 is more efficient than RK3; and the total simulation time of ETD-RK3 increases when mesh becomes finer. Hence, we next investigate the performance of LETD-RK3 scheme with the time step $\Delta t = 10\Delta t_{CFL}$ and the size of buffer zone to be 10.

The LETD-RK3 approximation errors in fluid thickness and velocity on these two different meshes at $T = 1$ day are listed in Table 15 in which the number of subdomains varies. It shows that the number of subdomains does not affect the accuracy of the LETD-RK3. Based on the simulation time listed in Table 16, we can deduce that the LETD-RK scheme would lead to great efficiency if implemented in parallel.

$\Delta t/\Delta t_{CFL}$	32 km				16 km			
	$\mathcal{E}_h$		$\mathcal{E}_u$		$\mathcal{E}_h$		$\mathcal{E}_u$	
160	6.19e-09	-	3.95e-05	-	1.88e-09	-	9.38e-06	-
80	1.17e-09	[2.41]	1.56e-05	[1.34]	2.05e-10	[3.20]	2.46e-06	[1.93]
40	2.70e-10	[2.11]	2.40e-06	[2.70]	2.95e-11	[2.80]	3.07e-07	[3.01]
20	2.76e-11	[3.29]	2.92e-07	[3.04]	4.14e-12	[2.83]	4.08e-08	[2.91]
10	3.72e-12	[2.89]	3.81e-08	[2.94]	5.10e-13	[3.02]	5.19e-09	[2.98]

Table 13: (Three layers SOMA) The errors in fluid thickness and velocity obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 10\Delta t_{CFL}$ on two meshes with different resolutions at Day 1.

$\Delta t/\Delta t_{CFL}$	32 km		16 km	
	time/step	total time	time/step	total time
160	1.64	19.72	9.02	270.46
80	0.97	22.36	4.70	277.13
40	0.67	30.75	3.06	360.51
20	0.68	62.87	2.98	701.41
10	0.67	122.38	2.93	1377.93

Table 14: (Three layers SOMA) The average CPU time for each time stepping and total CPU time obtained by ETD-RK3 with time steps varying from $\Delta t = 160\Delta t_{CFL}$ to $\Delta t = 10\Delta t_{CFL}$ during a one day's simulation. The simulation time is measured in seconds.

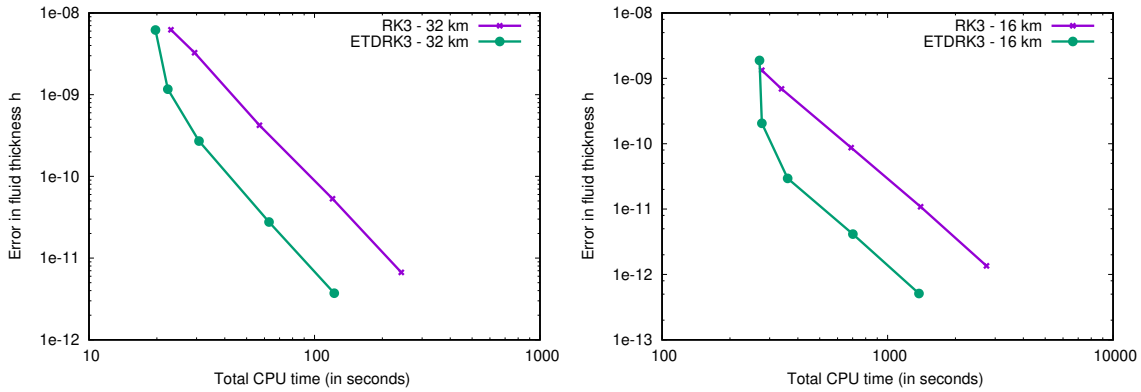

Figure 14: (Three layers SOMA) The errors in fluid thickness h as a function of total CPU time for a variety of time step sizes at $T = 1$ day. Left: 32 km resolution; right: 16 km resolution.

Finally, we consider 15-day-long simulations: the snapshots of sea surface height and the velocity at $T = 15$ days in the top layer using the ETD-RK3 scheme on the 16-km-resolution mesh with time step size $\Delta t = 10\Delta t_{CFL}$ are shown in Figure 15 and the time evolution of the relative change in mass during the 15-day simulations using the ETD-RK3 and the LETD-RK3 schemes with 8 subdomains is displayed in Figure 16. In this test case, the total energy is not conserved any more due to the appearance of additional forcing terms. But the ETD-RK3 and LETD-RK3 schemes are able to preserve the total mass.

No. of subdomains	32 km		16 km	
	$\mathcal{E}_h$	$\mathcal{E}_u$	$\mathcal{E}_h$	$\mathcal{E}_u$
1	3.7166e-12	3.8106e-08	5.0997e-13	5.1863e-09
2	3.7170e-12	3.8107e-08	5.1009e-13	5.1879e-09
4	3.7167e-12	3.8109e-08	5.0971e-13	5.1850e-09
8	3.7172e-12	3.8108e-08	5.1010e-13	5.1856e-09
16	3.7170e-12	3.8110e-08	5.0965e-13	5.1887e-09
32	3.7170e-12	3.8108e-08	5.1013e-13	5.1894e-09

Table 15: (Three layers SOMA) The errors of fluid thickness and velocity with the LETD-RK3 scheme on meshes with different resolutions and different number of subdomains at Day 1.

No. of subdomains	32 km		16 km	
	time/step	total time	time/step	total time
1	0.67	122.34	2.93	1377.93
2	0.68	124.99	2.40	1130.12
4	0.87	159.48	2.66	1250.76
8	1.23	225.90	3.15	1479.82
16	1.64	300.90	3.79	1781.78
32	2.17	397.04	4.98	2342.92

Table 16: (Three layers SOMA) The average CPU time for each time stepping and total CPU time of LETD-RK3 during a one day's simulation on meshes with different resolutions and different number of subdomains. The simulation time is measured in seconds.

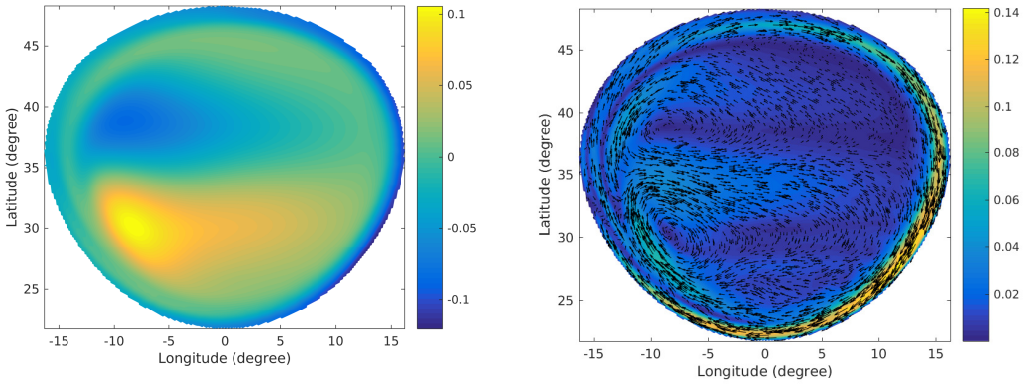

Figure 15: (Three layers SOMA) Snapshots of sea surface height (left), and the velocity (right) at $T=15$ days using ETD-RK3 with time step $\Delta t = 10\Delta t_{CFL}$ on the 16-km-resolution mesh.

6 Conclusions

The ETD-RK and LETD-RK methods have been investigated and compared in this paper for simulating the rotating shallow water equations of one layer or multiple layers. Comparing with the standard RK scheme of the same order, the ETD-RK scheme is more efficient when error thresholds are small. To further speed up the ETD simulations, we have developed a localized ETD method; different from the global ETD method, this approach first solves subdomain problems of smaller sizes using the ETD method for time integration, and then ex-

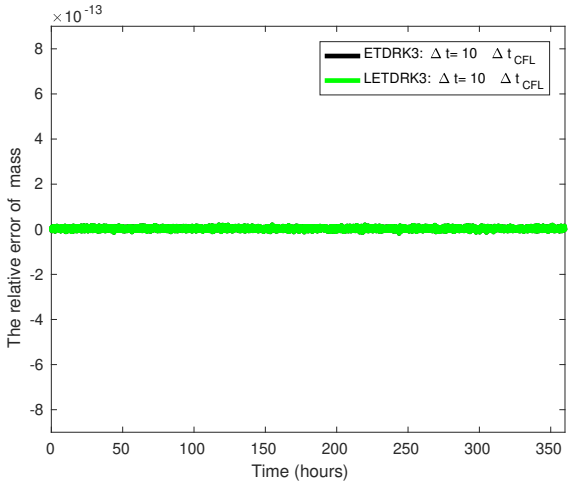

Figure 16: (Three layers SOMA) Evolution of the relative changes in total mass using the ETD-RK3 and LETD-RK3 schemes with time step $\Delta t = 10\Delta t_{CFL}$ on the 16-km-resolution mesh.

changes information in the overlapping areas. Numerical results show that the localized ETD can achieve the same accuracy and efficiency as the global ETD, but has the great potential to be efficiently implemented in parallel computing due to its natural scalability, which will be the focus of our future study.

Acknowledgments

This work was partially supported by U.S. Department of Energy through the grants DE-SC0016540 and DE-SC0020270, U.S. National Science Foundation through the grant DMS-1912626, Office of the Vice President for Research at the University of South Carolina through an ASPIRE grant, and Natural Science Foundation of China through the grant 11871454.

References

[1] M. Hochbruck, C. Lubich, H. Selhofer, Exponential integrators for large systems of differential equations, SIAM Journal on Scientific Computing 19 (5) (1998) 1552–1574.
[2] S. Cox, P. Matthews, Exponential time differencing for stiff systems, Journal of Computational Physics 176 (2) (2002) 430–455.
[3] Q. Du, W. Zhu, Analysis and applications of the exponential time differencing schemes and their contour integration modifications, BIT Numerical Mathematics 45 (2) (2005) 307–328.
[4] S. Krogstad, Generalized integrating factor methods for stiff PDEs, Journal of Computational Physics 203 (1) (2005) 72–88.
[5] M. Tokman, Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods, Journal of Computational Physics 213 (2) (2006) 748–776.
[6] M. Hochbruck, A. Ostermann, J. Schweitzer, Exponential Rosenbrock-type methods, SIAM Journal on Numerical Analysis 47 (1) (2009) 786–803.
[7] L. Ju, J. Zhang, L. Zhu, Q. Du, Fast explicit integration factor methods for semilinear parabolic equations, Journal of Scientific Computing 62 (2015) 431–455.
[8] L. Ju, J. Zhang, Q. Du, Fast and accurate algorithms for simulating coarsening dynamics of Cahn-Hilliard equations, Computational Materials Science 108 (2015) 272– 282.
[9] M. Hochbruck, A. Ostermann, Exponential integrators, Acta Numerica 19 (2010) 209–286.
[10] C. Moler, C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, SIAM Review 20 (4) (1978) 801–836.

[11] C. Moler, C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM Review 45 (1) (2003) 3–49.
[12] R. B. Sidje, Expokit: A software package for computing matrix exponentials, ACM Transactions on Mathematical Software 24 (1) (1998) 130–156.
[13] J. Niesen, W. M. Wright, Algorithm 919: A Krylov subspace algorithm for evaluating the ϕ-functions appearing in exponential integrators, ACM Transactions on Mathematical Software (TOMS) 38 (3) (2012) 22.
[14] S. Gaudreault, G. Rainwater, M. Tokman, KIOPS: A fast adaptive Krylov subspace solver for exponential integrators, Journal of Computational Physics 372 (2018) 236–255.
[15] V. T. Luan, J. A. Pudykiewicz, D. R. Reynolds, Further development of efficient and accurate time integration schemes for meteorological models, Journal of Computational Physics 376 (2019) 817–837.
[16] E. Gallopoulos, Y. Saad, Efficient solution of parabolic equations by Krylov approximation methods, SIAM Journal on Scientific and Statistical Computing 13 (5) (1992) 1236–1264.
[17] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM Journal on Numerical Analysis 29 (1) (1992) 209–228.
[18] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM Journal on Numerical Analysis 34 (5) (1997) 1911–1925.
[19] S. Gaudreault, J. A. Pudykiewicz, An efficient exponential time integration method for the numerical solution of the shallow water equations on the sphere, Journal of Computational Physics 322 (2016) 827–848.
[20] J. Loffeld, M. Tokman, Implementation of parallel adaptive-Krylov exponential solvers for stiff problems, SIAM Journal on Scientific Computing 36 (5) (2014) C591–C616.
[21] J. Zhang, C. Zhou, Y. Wang, L. Ju, Q. Du, X. Chi, D. Xu, D. Chen, Y. Liu, Z. Liu, Extreme-scale phase field simulations of coarsening dynamics on the Sunway Taihulight supercomputer, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'16), 2016, pp. 4:1–4:12.
[22] T.-T.-P. Hoang, L. Ju, Z. Wang, Overlapping localized exponential time differencing methods for diffusion problems, Communications in Mathematical Sciences 16 (6) (2018) 1531–1555.
[23] X. Li, L. Ju, T.-T.-P. Hoang, Overlapping domain decomposition based exponential time differencing methods for semilinear parabolic equations, BIT: Numerical Mathematics, to appear.
[24] T.-T.-P. Hoang, L. Ju, Z. Wang, Nonoverlapping localized exponential time differencing methods for diffusion problems, Journal of Scientific Computing 82 #37 (2020). https://doi.org/10.1007/s10915-020-01136-w.
[25] T. Ringler, M. Petersen, R. L. Higdon, D. Jacobsen, P. W. Jones, M. Maltrud, A multi-resolution approach to global ocean modeling, Ocean Modelling 69 (2013) 211–232.
[26] J. Thuburn, T. Ringler, W. Skamarock, J. Klemp, Numerical representation of geostrophic modes on arbitrarily structured C-grids, Journal of Computational Physics 228 (22) (2009) 8321–8335.
[27] 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, Journal of Computational Physics 229 (9) (2010) 3065–3090.
[28] Q. Du, V. Faber, M. Gunzburger, Centroidal voronoi tessellations: Applications and algorithms, SIAM Review 41 (4) (1999) 637–676.
[29] Q. Du, M. D. Gunzburger, L. Ju, Constrained centroidal voronoi tessellations for surfaces, SIAM Journal on Scientific Computing 24 (5) (2003) 1488–1506.
[30] M. Ben-Artzi, P. G. LeFloch, Well-posedness theory for geometry-compatible hyperbolic conservation laws on manifolds, Annales De Linstitut Henri Poincare Non Linear Analysis 24 (6) (2007) 989–1008.
[31] M. Ben-Artzi, J. Falcovitz, P. G. LeFloch, Hyperbolic conservation laws on the sphere. A geometry-compatible finite volume scheme, Journal of Computational Physics 228 (16) (2009) 5650–5668.
[32] A. Beljadid, A. Mohammadian, H. M. Qiblawey, An unstructured finite volume method for large-scale shallow flows using the fourth-order Adams scheme, Computers & Fluids 88 (2013) 579–589.
[33] T. D. Ringler, D. Jacobsen, M. Gunzburger, L. Ju, M. Duda, W. Skamarock, Exploring a multiresolution modeling approach within the shallow-water equations, Monthly Weather Review 139 (11) (2011) 3348–3368.
[34] C. Clancy, J. A. Pudykiewicz, On the use of exponential time integration methods in atmospheric

models, Tellus A: Dynamic Meteorology and Oceanography 65 (1) (2013) 20898.
[35] M. R. Petersen, D. W. Jacobsen, T. D. Ringler, M. W. Hecht, M. E. Maltrud, Evaluation of the arbitrary Lagrangian–Eulerian vertical coordinate method in the MPAS-Ocean model, Ocean Modelling 86 (2015) 93–113.
[36] Q. Chen, T. Ringler, P. R. Gent, Extending a potential vorticity transport eddy closure to include a spatially-varying coefficient, Computers & Mathematics with Applications 71 (11) (2016) 2206–2217.
[37] K. Pieper, K. C. Sockwell, M. Gunzburger, Exponential time differencing for mimetic multilayer ocean models, Journal of Computational Physics 398 (2019) 108900.
[38] S. Sari, T. Rowan, M. Seaid, F. Benkhaldoun, Simulation of three-dimensional free-surface flows using two-dimensional multilayer shallow water equations, Communications in Computational Physics 27 (2020) 1413–1442.
[39] J. Thuburn, C. J. Cotter, A framework for mimetic discretization of the rotating shallow-water equations on arbitrary polygonal grids, SIAM Journal on Scientific Computing 34 (3) (2012) B203–B225.
[40] L. Ju, T. Ringler, M. Gunzburger, Voronoi tessellations and their application to climate and global modeling, in: Numerical techniques for global atmospheric models, Springer, 2010, pp. 313–342.
[41] D. L. Williamson, J. B. Drake, J. J. Hack, R. Jakob, P. N. Swarztrauber, A standard test set for numerical approximations to the shallow water equations in spherical geometry, Journal of Computational Physics 102 (1) (1992) 211–224.
[42] M. Hochbruck, A. Ostermann, Explicit exponential Runge–Kutta methods for semilinear parabolic problems, SIAM Journal on Numerical Analysis 43 (3) (2005) 1069–1090.
[43] Y. Saad, Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices, Linear algebra and its applications 34 (1980) 269–295.
[44] A. Koskela, Approximating the matrix exponential of an advection-diffusion operator using the incomplete orthogonalization method, in: Numerical Mathematics and Advanced Applications-ENUMATH 2013, Springer, 2015, pp. 345–353.
[45] G. Karypis, V. Kumar, A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices, University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN.
[46] P. J. Wolfram, T. D. Ringler, M. E. Maltrud, D. W. Jacobsen, M. R. Petersen, Diagnosing isopycnal diffusivity in an eddying, idealized midlatitude ocean basin via lagrangian, in situ, global, high-performance particle tracking (light), Journal of Physical Oceanography 45 (8) (2015) 2114–2133.
[47] J. Thuburn, Y. Li, Numerical simulation of Rossby-Haurwitz waves, Tellus A 52 (2) (2000) 181 – 189.
[48] R. K. Smith, D. G. Dritschel, Revisiting the Rossby-Haurwitz wave test case with contour advection, Journal of Computational Physics 217 (2) (2006) 473–484.
[49] J.-G. Li, Shallow-water equations on a spherical multiple-cell grid, Quarterly Journal of the Royal Meteorological Society 144 (2018) 1–12.