Exponential Integrators for the Shallow Water Equations

Matthieu Brachet, Laurent Debreu, Christopher Eldred

EXPONENTIAL INTEGRATORS FOR THE SHALLOW WATER EQUATIONS

M. BRACHET[†,1], L. DEBREU[†,2] AND C. ELDRED[†,3]

Abstract. The time integration scheme is probably one of the most fundamental choice in the development of an ocean model. In this paper, we investigate several time integration schemes when applied to the shallow water equations. These set of equations is accurate enough when modelling a small depth ocean and is also relevant to study as it is the one solved for the barotropic (i.e. vertically averaged) component of a three dimensional ocean model. We analysed different schemes for the shallow water equations linearised around $(\bar{h}, 0)$. This simplified model give a good idea of difficulties occurring when applying a time integrator. Explicit schemes are accurate but the time step is constraint by the Courant-Friedrichs-Lewy stability condition. Implicit schemes can be unconditionally stable but not very accurate. In this article we propose a detailed comparison of such classical schemes with exponential integrators. The accuracy and the computational costs are analysed in different configurations..

1. Introduction

The shallow water equations are used to model the fluid's movements. The system is derived from Euler equations assuming a small thickness of fluid submitted to gravity force. In a one-dimensional framework, the unknowns are h, the fluid thickness, and u, the horizontal velocity. The system is expressed:

$$\begin{cases} \dfrac{\partial h}{\partial t} + \dfrac{\partial}{\partial x}\left(hu\right) = 0 \\ \dfrac{\partial u}{\partial t} + \dfrac{\partial}{\partial x}\left(\dfrac{1}{2}u^2 + gh\right) = f \end{cases} \tag{1.1}$$

for all $x \in [0, d]$, $t \geq 0$ and the initial data $(u_0(x), h_0(x))$. Equations (1.1) are closed with appropriate boundary conditions, which we consider here to be periodic. More precisely, for all $(t, x) \in \mathbb{R}^+ \times \mathbb{R}$, the functions h and u satisfy

$$u(t, x + d) = u(t, x) \text{ and } h(t, x + d) = h(t, x). \tag{1.2}$$

The function $f : (t, x) \in \mathbb{R}^+ \times [0, d] \mapsto f(t, x) \in \mathbb{R}$ represents the external interactions (e.g. bottom friction or wind).

The shallow water model is widely used in computational fluid dynamics. Among others, we can mention

- *Ocean model*: under the small thickness assumption but also in three dimensional "primitive" equations model where the fast (barotropic) component is solved apart of the 3D equations using a shallow water system forced by a depth averaged right hand side.
- *Bedload transport*: to model the sediment transport, shallow water equations are coupled with Exner equation. In this system, a topography is moving and the bedload velocity is smaller than fluid velocity [10].
- *Atmosphere model*: shallow water equations on a rotated sphere is the simplest atmosphere model of interest. It is considered as a first step in the study a numerical solver for geophysical fluids [38].

Function of the application, it can be important to obtain a solution as soon as possible by considering an efficient solver even if results are not accurate. In this case, we consider a low order accurate scheme but the computational cost to reach final time is low. The stability and the computational cost are the main properties of interest. This is the case in ocean modelling for example. On other application, the accuracy is essential and high order schemes are necessary. This is the topic of bedload transport models or tide model. Of course, the accuracy is impacted by the time step Δt. If the fluid is moving slowly, a large time step give us a good accuracy and A-stable schemes could be competitive. If the movement

Date: February 14, 2020.

Key words and phrases. Shallow water equations, time stepping, exponential integrators, finite difference, Krylov methods.

are fast, phenomenon can not be captured by a scheme with large time step and an explicit solver can be an interesting method. Furthermore, the study can be extend to acoustic waves. Considering this issues, the time integrator is crucial.

Explicit schemes like the explicit Runge-Kutta and the Forward-Backward schemes discussed later are accurate and the computation cost per time step is small. However, they are limited by a Courant-Friedrichs-Lewy Condition (CFL) stability condition. (see [23] for a review on the different stability conditions of an ocean model):

$$\frac{c_g \Delta t}{\Delta x} \leq \mathrm{C}^{\mathrm{ste}} \tag{1.3}$$

where Δt and Δx are the temporal and spacial discretization steps. c_g is a characteristic velocity which would correspond in the shallow water model to $c_g = \sqrt{gH}$ where H is the mean total depth [20, 34]. The computational cost per time step is small but a large number of time step can be necessary to reach the final time. Forward-Backward schemes are considered in [34] and Explicit Runge-Kutta integrators in [11, 37] among others.

An alternative is to use implicit schemes (see IMEX schemes in [21, 39]). This set of time integrators are A-stable and potentially allows to consider much greater time steps than explicit schemes. The price to pay is of the solution of a linear or non-linear system. Another drawback of implicit schemes is linked to the accuracy. Indeed, great time step can deteriorate the accuracy, in particular if the solution changes rapidly. The accuracy is also impacted by dispersion and possibly dissipation errors.

More recently, exponential integrators gain in interest to solve partial differential equations

$$\frac{dX}{dt} = LX + \mathcal{N}(X). \tag{1.4}$$

A review of this schemes are available in [17] in a general framework. The main property of this class of integrator is to solve exactly linear equations. A consequence is the A-stability: schemes are not constrained by a CFL condition. Dissipation and dispersion properties depend only on the space discretization. The main issue is the computation of matrix exponential involving in the methods. This can be done using a Krylov method [31].

The most famous class of methods are the Exponential Runge-Kutta Integrators (ERK) rely on the formula

$$X(t) = \exp(tL)\, X(0) + \int_0^t \exp((t-\tau)L)\, \mathcal{N}(X(\tau))\, d\tau. \tag{1.5}$$

They have been used to solve shallow water equations on a rotating sphere in the context of model the atmosphere in [6, 9, 15]. They highlight the accuracy of exponential Runge-Kutta integrators. Methods are as accurate as explicit Runge-Kutta methods with larger time steps. However, the computational cost is still larger than using Implicit time scheme with same time step [14]. In this articles, the accuracy is measured on the classical set of test cases [38].

In [7], ERK integrators are considered to solve tracer equations appearing in ocean modeling. These time schemes allows to consider larger time step and maintain a good accuracy. Two methods are compared to compute the exponential part. The first is based on squalling and squaring with a Padé approximation. It is the quicker solution because the exponential matrices are compute once for all but have to be save. In the ocean model interest, matrices are full and large, then it is impossible to save it. The second way is to compute exponential matrices at each time step using Krylov methods. It is more expensive but matrices have not to be save. Article [29] suggests to consider exponential integrators to solve multi-layer shallow water equations. The space discretization is done with mimetic elements then matrices related to the linear part are skew-symmetric and exponential are computed using skew-Lanczos methods. To obtain a computational cost smaller than RK4, it is necessary to accept a larger error than RK4's error.

All the previous articles are considering ERK integrators. They are impacted by the choice of quadrature in the integral part of (1.5). Therefore, we consider the change of variable $V(t) = \exp((t^n - t)L)X(t)$ in (1.4) and use an explicit scheme on the V equation. By this method, we avoid quadrature methods in the integral. These exponential integrators are called Linearly Exact Runge-Kutta integrators (LERK). LERK are used in [28] to solve the Korteweg-de Vries equation. Energy conservation properties are studied by [2] on a set of PDEs with damping/driving forces. It is easy to build a high order accurate LERK integrator avoiding quadrature in (1.5). Unfortunately, a lot of exponential of matrices are involving that make LERK expensive.

To reduce the computational cost, we consider a splitting between linear and non linear part. For example, the Lie's splitting is rely on two steps: compute X^* the solution of the linear equation at time $t^n + \Delta t$ with X^n the initial condition:

$$\begin{cases} \dfrac{dX}{dt} = LX \\ X(0) = X^n. \end{cases} \tag{1.6}$$

The solver is denoted $\mathcal{S}_{\mathrm{l},\Delta t}$. The second step is to solve the non linear part starting from X^*:

$$\begin{cases} \dfrac{dX}{dt} = \mathcal{N}(X) \\ X(0) = X^*. \end{cases} \tag{1.7}$$

The non linear part solver is denoted $\mathcal{S}_{\mathrm{nl},\Delta t}$. Each equation is solved with a highly accurate scheme but the final solution suffers from splitting errors. A more accurate method is the Strang's splitting. Integrators rely to splitting are used in [19] to capture shock created by shallow water equations.

In this article, we are mainly interested by the influence of the forcing function f in shallow water equations (1.1). A good insight of the difficulties occurring is obtained by linearising equations (1.1) around a steady state $(\bar{h}, \bar{u} = 0)$ where $\bar{h}$ is a constant. The new system of equations, called Linearized Shallow Water equations (LSWE), is given by

$$\begin{cases} \dfrac{\partial h'}{\partial t} + \bar{h}\dfrac{\partial u'}{\partial x} = 0 \\ \dfrac{\partial u'}{\partial t} + g\dfrac{\partial h'}{\partial x} = f \end{cases} \tag{1.8}$$

wherein (h', u') is a small perturbation of the constant state $(\bar{h}, \bar{u} = 0)$. This system is obtained by replacing (h, u) by $(\bar{h} + h', 0 + u')$ in (1.1) and neglecting the terms of order 2. The characteristic velocity of (1.8) is $c = \sqrt{g\bar{h}}$. In the following, we will no longer write $'$ to simplify notations.

The purpose of this paper is to compare various time schemes for linearised shallow water equations. This includes Runge-Kutta explicit schemes, implicit $\theta-$ scheme and exponential integrators. The stability constrains associated to each of this scheme is first recalled. Then the accuracy is studied and we are particularly interested in how the frequency of the forcing term f affects the implicit and exponential schemes. In addition since the global behaviour of the scheme is also dependent on the spatial discretization we study both second and fourth order spatial schemes, both of them on a staggered grid.

This paper is organize as follow. In section 2, we review two space discretizations on a staggered grid. They are second and fourth order accurate and we consider the phase error due to the space discretization. The section 3 is devoted to the Krylov subspace method used to compute matrix functions. A stopping criterion is studied. Various time schemes are considered to solve linearized shallow water equations. Beside stability and accuracy issues, we analyze dissipation and dispersion properties considering the space discretization in section 4. The numerical schemes are programmed and tested on different test cases in section 5. Its allow to confirm the properties and to consider the computational cost issues.

2. Second and fourth order centered spatial scheme on a staggered grid

Equations (1.8) are solved using the method of lines. The first step is the discretization in space. We consider two centered finite difference operators on a staggered grid, a one-dimensional version of the C-grid in the Arakawa classification (see [1]). These are second (C2) and fourth (C4) order accurate. On the one hand, the C4 scheme is more accurate than C2 but on the other hand, the spectral radius of the fourth order operator is greater than the second order operator. This last point has implication on stability and computational cost after time integration.

Two grids are considered for the space discretization of (1.8): one for h and an other one for u. The h-grid is an offset of the u-grid. Define two sets of points: $(x_i)_{0\leq i\leq N} \in \mathbb{R}^N$ and $(x_{i+1/2})_{0\leq i\leq N-1} \in \mathbb{R}^{N-1}$ with

$$\begin{cases} x_i := i\Delta x \\ x_{i+1/2} := x_i + \dfrac{\Delta x}{2} = \left(i + \dfrac{1}{2}\right)\Delta x, \end{cases} \tag{2.1}$$

the mesh size is $\Delta x = d/N$, $N \in \mathbb{N}^*$ (see Fig. 1). It is a one dimensional C-grid in the Arakawa classification [1]. It allows a better representation of characteristic waves for shallow water equations

than a non-staggered A-grid (see [3]) . The velocity h is estimated on $(x_i)_{0\leq i\leq N-1}$ and u is computed on $(x_{i+1/2})_{0\leq i\leq N-1}$:

$$\begin{cases} h(\cdot, x_i) \approx \mathfrak{h}_i \text{ with } 0 \leq i \leq N-1, \\ u(\cdot, x_{i+1/2}) \approx \mathfrak{u}_{i+1/2} \text{ with } 0 \leq i \leq N-1. \end{cases} \tag{2.2}$$

At this step, the vectors $\mathfrak{u} \in \mathbb{R}^N$ and $\mathfrak{h} \in \mathbb{R}^N$ are function of time.

FIGURE 1. Staggered grid representing $(x_i)_{0\leq i\leq N} \in \mathbb{R}^N$ and $(x_{i+1/2})_{0\leq i\leq N-1} \in \mathbb{R}^N$.

We review the finite difference operators (FD) second and fourth order accurate used to compute each space partial derivative.

- *FD Operators accurate order 2 (C2):*

$$\begin{cases} \dfrac{\partial u}{\partial x}_i \approx \delta_x \mathfrak{u}_i := \dfrac{\mathfrak{u}_{i+1/2} - \mathfrak{u}_{i-1/2}}{\Delta x} \\ \dfrac{\partial h}{\partial x}_{i+1/2} \approx \delta_x \mathfrak{h}_{i+1/2} := \dfrac{\mathfrak{h}_{i+1} - \mathfrak{h}_i}{\Delta x} \end{cases} \tag{2.3}$$

- *FD Operators accurate order 4 (C4):*

$$\begin{cases} \dfrac{\partial u}{\partial x}_i \approx \delta_x \mathfrak{u}_i := \dfrac{9}{8}\dfrac{\mathfrak{u}_{i+1/2} - \mathfrak{u}_{i-1/2}}{\Delta x} - \dfrac{1}{8}\dfrac{\mathfrak{u}_{i+3/2} - \mathfrak{u}_{i-3/2}}{3\Delta x} \\ \dfrac{\partial h}{\partial x}_{i+1/2} \approx \delta_x \mathfrak{h}_{i+1/2} := \dfrac{9}{8}\dfrac{\mathfrak{h}_{i+1} - \mathfrak{h}_i}{\Delta x} - \dfrac{1}{8}\dfrac{\mathfrak{h}_{i+2} - \mathfrak{h}_{i-1}}{3\Delta x} \end{cases} \tag{2.4}$$

The phase error induced by the space discretization is extracted from λ_x the eigenvalues of the difference operators δ_x . A simple Fourier analysis give us the phase error:

$$e_\Phi := \frac{\arg \exp\left(\lambda_x(\theta)\right)}{\theta} \tag{2.5}$$

with the normalized wave number $\theta = k\Delta x$. Values of e_Φ in C2 and C4 cases are given in Table 1 and plotted in Figure 2. As expected, higher order space operator C4 lead to a better accuracy than C2 operator.

Space operator	**Phase error**
C2	$\dfrac{\sin\theta}{\theta}$
C4	$\dfrac{(13-\cos\theta)\sin(\theta/2)}{\theta/2}$

TABLE 1. Numerical phase error for centered second and fourth order approximations on staggered grid. The normalized wave number is $\theta = k\Delta x$.

3. Krylov method for matrix exponential: algorithm and stopping criterion

In the case of exponential integrator for (LSWE), we must be able to compute a matrix product function $\varphi_k(A)b$ (see Appendix). We solve this kind of problem using Krylov methods [16, 32]. The method is based on project A onto a Krylov subspace using Arnoldi's algorithm and compute $\varphi_k(A)b$ in this subspace.

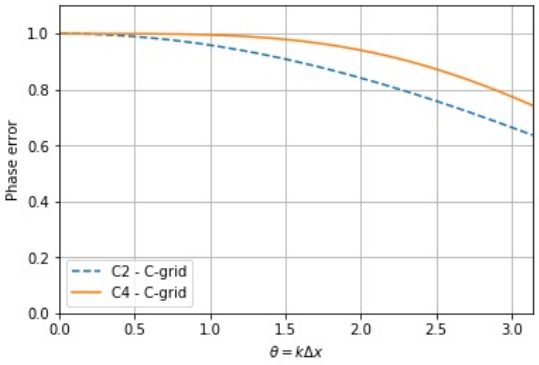

FIGURE 2. Numerical phase error for centered second and fourth order approximations on staggered grid. The normalized wave number is $\theta = k\Delta x$.

3.1. **Arnoldi's method for Krylov subspaces.** Instead of compute $\varphi_k(A)b$ directly, we consider the operation in a Krylov subspace $\mathcal{K}_m(A,b)$ define by

$$\mathcal{K}_m(A,b) = \text{Span}(b, Ab, A^2b, \cdots, A^{m-1}b). \tag{3.1}$$

Using Arnoldi's method, we build $H_m \in \mathbb{M}_m(\mathbb{R})$ the projection of A into $\mathcal{K}_m(A,b)$ and $V_m \in \mathbb{M}_{N,m}(\mathbb{R})$ an orthonormal basis of $\mathcal{K}_m(A,b)$. The method is detailed in Algorithm 1. The coefficients of H_m are $(h_{i,j})_{1\leq i,j\leq m}$ and the columns of V_m are the vectors $(v_i)_{1\leq i\leq m}$.

Algorithm 1 : Arnoldi Gram-Schmidt

Let $v_1 = b/\|b\|_2$
for $j = 1, \cdots, m$ **do**
  $w_j = Av_j$,
  for $i = 1, \cdots, j$ **do**
    $h_{i,j} = v_i^T \cdot w_j$,
    $w_j = w_j - h_{i,j}v_i$,
  end for
  $h_{j+1,j} = \|w_j\|_2$
  if $h_{j+1,j} = 0$, **then**
    break
  else $v_{j+1} = w_j/h_{j+1,j}$.
end for

The matrices H_m and V_m satisfy the following equalities [32]:

$$\begin{cases} AV_m = V_mH_m + w_me_m^T \\ V_m^TAV_m = H_m. \end{cases} \tag{3.2}$$

They will be used to compute matrix functions.

3.2. **Computing matrix function product.** As mentioned in [29], exponential integrators implies to be able to compute efficiently product $\varphi_k(A)b$ with A a matrix and b a vector. Approximating this product is based on Krylov method (see [16, 27]).

The first step is to built H_m and V_m related to the Krylov subspace $\mathcal{K}_m(A,b)$ using Algorithm 1. Then, the product $\varphi_k(A)b$ is approached by

$$\varphi_k(A)b \approx x_m = \|b\|_2V_m\varphi_k(H_m)e_1. \tag{3.3}$$

The global algorithm is given by:

Algorithm 2 : Krylov methods for $\varphi_k(A)$b

1: **for** $m = 1, \cdots$, until convergence **do**
2: Build H_m and V_m linked to $\mathcal{K}_m(A, b)$ using Algorithm 1,
3: $x_m = \|b\|_2 V_m \varphi_k(H_m) e_1$.
4: **end for**

The second line of Algorithm 2 should be considered as an update of matrices H_m and V_m to avoid redundant calculations.

Furthermore, $\varphi_k(H_m)$ is computed using Padé approximate. The error between Padé approximate and exact value of $\varphi_k(H_m)$ is linked to the norm of H_m, it is convenient to use scaling and squaring methods (see [26, 35]).

A stopping criterion is given in the exponential case (i.e. $k = 0$) in article [31]. We adapt it in a more general case. Let s_m^k be the error between $\varphi_k(A)b$ and the Krylov approximation:

$$s_m^k = \varphi_k(A)b - \|b\|_2 V_m \varphi_k(H_m) e_1. \tag{3.4}$$

Then, s_m^k satisfies the following lemma.

Lemma 3.1. Let m and k positive integers, $A \in \mathbb{M}_N(\mathbb{R})$ and $b \in \mathbb{R}^N$. Matrices H_m and V_m are computed using Algorithm 1. The following equality is satisfied:

$$s_m^k = \|b\|_2 h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 + A s_m^{k+1}. \tag{3.5}$$

Proof. We proceed by successive equalities:

$$\begin{aligned}
\frac{1}{\|b\|_2}\varphi_k(A)b &= \left(A\varphi_{k+1}(A) + \frac{1}{k!}\mathrm{Id}\right) v_1 \text{ by definition of } \varphi_k \\
&= \frac{1}{k!}v_1 + A\left[V_m \varphi_{k+1}(H_m)e_1 + \frac{1}{\|b\|_2} s_m^{k+1}\right] \text{ by definition of } s_m^k \\
&= \frac{1}{k!}v_1 + \left(V_m H_m + h_{m+1,m} v_{m+1} e_m^T\right)\varphi_{k+1}(H_m)e_1 + \frac{1}{\|b\|_2} A s_m^{k+1} \text{ using (3.2)}, \\
&= V_m\left[\frac{1}{k!}\mathrm{Id} + H_m \varphi_{k+1}(H_m)\right] e_1 + h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 + \frac{1}{\|b\|_2} A s_m^{k+1} \\
&= V_m \varphi_k(H_m) e_1 + h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 + \frac{1}{\|b\|_2} A s_m^{k+1} \text{ by definition of } \varphi_k.
\end{aligned}$$

It concludes the proof. □

Using this lemma, we can expressed the error s_m^k as a serie.

Proposition 3.2. The error s_m^k is expressed by

$$s_m^k = \|b\|_2 h_{m+1,m}\left(\sum_{i=k+1}^{\infty} A^{i-k+1} v_{m+1} e_m^T \varphi_i(H_m)\right) e_1. \tag{3.6}$$

Proof. We proof by induction that

$$s_m^k = \|b\|_2 h_{m+1,m}\left(\sum_{i=0}^{K} A^i v_{m+1} e_m^T \varphi_{k+1+i}(H_m)\right) e_1 + A^{K+1} s_m^{K+k+1}. \tag{3.7}$$

Furthermore, $\left(A^{K+1} s_m^{K+k+1}\right)_K$ converge to 0. Indeed, for all non negative value of x, the inequality

$$0 \leq \varphi_k(x) \leq \frac{\exp(x)}{k!} \tag{3.8}$$

is satisfied. Then, $|s_m^{k+K}|$ can be bounded. There exists C independent of K such that

$$\|s_m^{k+K}\|_2 \leq \frac{C}{(k+K)!}. \tag{3.9}$$

Remark that $\dfrac{C}{(k+K)!}$ tends toward 0, then the serie is convergent. □

As suggested in [31], the norm of the first term of this serie:

$$\|b\|_2 \cdot |h_{m+1,m}| \|v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1\|_2 \tag{3.10}$$

give us an idea of the error. Then, the stopping criterion we use is to continue the algorithm while $\|b\|_2 \cdot \|h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1\|_2$ is greater than a given tolerance. It is fixed to 10^{-12} in our experiment. In a practical point of view, to avoid computing two functions: $\varphi_k(H_m)$ (for the approximation) and $\varphi_{k+1}(H_m)$ (for the stopping criterion), we use the equality (A.11). In [31], author suggests to consider more terms of the series for the stopping criterion but it does not seem necessary in our numerical experiments.

On Figures 3 and 4, we plot the relative error and the value of the stopping criterion

$$\|b\|_2 \cdot \|h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1\|_2 \tag{3.11}$$

at each step of Krylov iteration to compute $\varphi_1(\Delta t L)b$ where L is defined by

$$LX = \begin{bmatrix} -\bar{h}\delta_x \mathfrak{u} \\ -g\delta_x \mathfrak{h} \end{bmatrix}, \tag{3.12}$$

$X = [\mathfrak{h}, \mathfrak{u}]^T \in \mathbb{R}^{2N}$. The vector $b \in \mathbb{R}^{2N}$ is randomized. Behaviours were similar for other functions φ_k tested. The stopping criterion is close to the relative error whatever the Courant number chosen. It is generally a little bit more greater. This stopping criterion is satisfactory in the experiments done. The number of iterations needed increase with the Courant number. C4 operators is slower to converge than C2. Furthemore, a large number of iterations is needed before the error goes down.

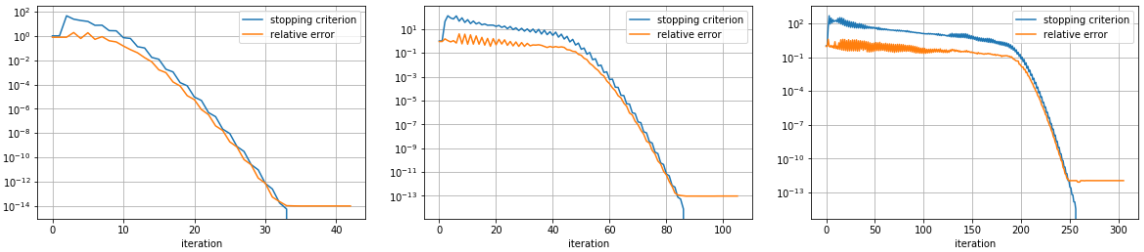

FIGURE 3. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There is $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel), 25 (center panel) and 100 (right panel). We consider the C2 space scheme.

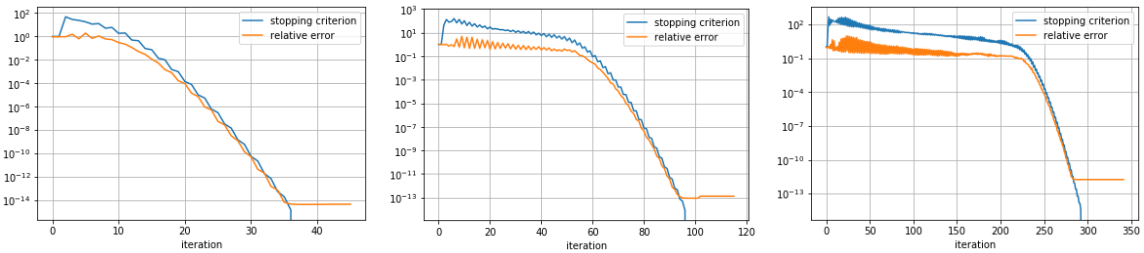

FIGURE 4. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There is $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel), 25 (center panel) and 100 (right panel). We consider the C4 space scheme.

4. Properties of numerical schemes

In this article, we consider various time schemes and space discretization to solve LSWE (1.8). The time schemes studied are given in Annexe A. Three sets of time integrators are discussed in this article. Explicit time schemes are cheaper but suffer from a CFL constraint. Exponential integrators and θ-schemes considered are A-stable, the time step Δt can be as large as desired. The space discretizations used are second order accurate C2 and fourth order C4. They are remind in section 2. Notation used in the following are given in Table 2.

Stability is one of the main property about a numerical scheme. A stable time scheme keep errors limited. Beside this, we also consider dissipation and dispersion properties. The dissipation is the ability to a scheme to keep extremums and the dispersion indicates how the numerical velocity of a simple wave is close to the theoretical velocity. We study dissipation and dispersion properties of numerical schemes.

Spatial discretization:	
C2	Centered order 2 scheme on C grid.
C4	Centered order 4 scheme on C grid.
Time integrators:	
FB	Forward-Backward.
RK(n)	Runge-Kutta integrator order (n).
IE	Implicit Euler.
CN	Crank-Nicholson.
THETA	θ-scheme with $\theta = 0.51$.
ERK(n)	Exponential RK Integrator order (n) for autonomous system.
ERK(n)c	Exponential RK Integrator order (n) for non-autonomous system.
LERK(n)	Linearly Exact Runge-Kutta (n)-th order accurate.
S(p)ERK(n)	plitting (p)-th order accurate coupled with RK(n).
subS(p)ERK(n)	Splitting (p)-th order accurate coupled with RK(n) with substeps.

Table 2. Notations used throughout the paper.

Consider a one step time scheme to solve (1.8). There exist a matrix $G \in \mathbb{M}_{2N}(\mathbb{R})$ such that:

$$X^{n+1} = GX^n, \tag{4.1}$$

where $X^n = [\mathfrak{h}^n, \mathfrak{u}^n]^T \in \mathbb{R}^{2N}$. For example, G is given by :

- RK3:

$$G = 1 + \Delta tL + \frac{1}{2}(\Delta tL)^2 + \frac{1}{6}(\Delta tL)^3, \tag{4.2}$$

- RK4:

$$G = 1 + \Delta tL + \frac{1}{2}(\Delta tL)^2 + \frac{1}{6}(\Delta tL)^3 + \frac{1}{24}(\Delta tL)^4, \tag{4.3}$$

- θ-scheme:

$$G = (\mathrm{Id} - \theta\Delta tL)^{-1} \cdot (\mathrm{Id} + (1-\theta)\Delta tL), \tag{4.4}$$

- Exponential Integrator:

$$G = \exp(\Delta tL). \tag{4.5}$$

Matrix L is defined by (3.12). A time scheme is stable if and only if $\|X^{n+1}\|_2 \leq \|X^n\|_2$, or equivalently $\mathrm{Sp}(G) \subset \mathcal{D}(0,1)$ in $\mathbb{C}$. In Table 3, we summarize the maximum Courant number $c\Delta t/\Delta x$ allowed to satisfy stability using different explicit schemes. The RK4 scheme has the larger stability limit while the RK3 scheme puts the stronger stability constraint.

The stability constraint has to be compare to the computational cost, mainly given by the right hand side evaluations of (1.8). In Table 4, we compute the maximum Courant number divided by the number of right hand side evaluations. This give a more precise idea of the computational cost to reach a given time using an explicit integrator. Greater the value is, cheaper the method is. Forward-Backward scheme is the cheaper among the explicit time schemes but also the less accurate. The Runge Kutta integrator RK4 is more accurate and less expensive than RK3. Implicit time schemes and exponential integrators are stable whatever the value $c\Delta t/\Delta x > 0$.

Scheme	RK4	RK3	FB
C2	$\sqrt{2} \approx 1.41$	$\frac{\sqrt{3}}{2} \approx 0.86$	1
C4	$48 \cdot \sqrt{\frac{283}{443749}} \approx 1.21$	$12 \cdot \sqrt{\frac{1698}{443749}} \approx 0.74$	$24 \cdot \sqrt{\frac{566}{443749}} \approx 0.86$

TABLE 3. Stability criterion for centered spatial discretization of second and four order schemes (C2 or C4) and for different explicit time schemes. To ensure stability, the Courant number $c\Delta t/\Delta x$ (where $c = \sqrt{g\bar{h}}$) must be smaller than the given value.

Scheme	RK4	RK3	FB
C2	$\frac{\sqrt{2}}{4} \approx 0.35$	$\frac{\sqrt{3}}{6} \approx 0.29$	1
C4	$12 \cdot \sqrt{\frac{283}{443749}} \approx 0.30$	$4 \cdot \sqrt{\frac{1698}{443749}} \approx 0.25$	$24 \cdot \sqrt{\frac{566}{443749}} \approx 0.86$

TABLE 4. Maximum Courant number $c\Delta t/\Delta x$ divided by the number of right hand side evaluation and for C2 and C4 spatial discretisations and for different explicit time scheme.

The dissipation and dispersion curves are plotted for different Courant number $c\Delta t/\Delta x$ in Figure 5 for the second order scheme (C2) and Figure 6 for the fourth order scheme (C4). To keep an extremum value at a given frequency $k\Delta x$, dissipation curve has to be close to one at this value. A smaller value means the extremum are attenuated, a greater value highlight an instability.

The velocity of a frequency is accurately obtain by the scheme is the value associate on the dissipation map is close to one. The shape of the numerical solution is bad if the dispersion is too poor.

The dissipation and dispersion errors of implicit schemes (with $\theta > 0.5$) increase with the Courant number. The Crank-Nicholson scheme ($\theta = 0.5$) does not dissipate but disperse (not plotted). In practice, dissipation can be needed to attenuate parasitic waves corresponding to ± 1 mode, that is why we can use θ-scheme with $\theta = 0.51$. On the contrary, exponential integrators do not dissipate. An interesting properties of exponential integrators is that dissipation and dispersion properties do not depend on the Courant number. High order accuracy space discretisation C4 permit to have better dissipation and dispersion properties than low order accuracy C2.

5. Numerical results

To assess the performances of each scheme, we consider various test cases for the linearized shallow water equations (1.8). Both the accuracy and stability are evaluated. Exponential integrators and implicit time schemes are A-stable allowing to use of large time steps but, while the computation of $\varphi_k(A)b$ implies a possibly large number of calls to the linear part, the implicit methods require the solution of a linear system. This here done thanks to a conjugate gradient method. Notations used for each scheme are summarized in Table 2.

Consider (1.8) initialized with (h_0, u_0). At each time $t \geq 0$, h is given by

$$\begin{aligned} h(t,x) =& \frac{1}{2}\sqrt{\frac{\bar{h}}{g}}\left[u_0(x-ct) - u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) + h_0(x+ct))\right] + ... \\ &... + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) - f(\tau, x+c(t-\tau))\right) d\tau, \end{aligned}$$

and the velocity u is

$$\begin{aligned} u(t,x) =& \frac{1}{2}\left[u_0(x-ct) + u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) - h_0(x+ct))\right] + ... \\ &... + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) + f(\tau, x+c(t-\tau))\right) d\tau \end{aligned}$$

with the parameter $c = \sqrt{g\bar{h}}$.

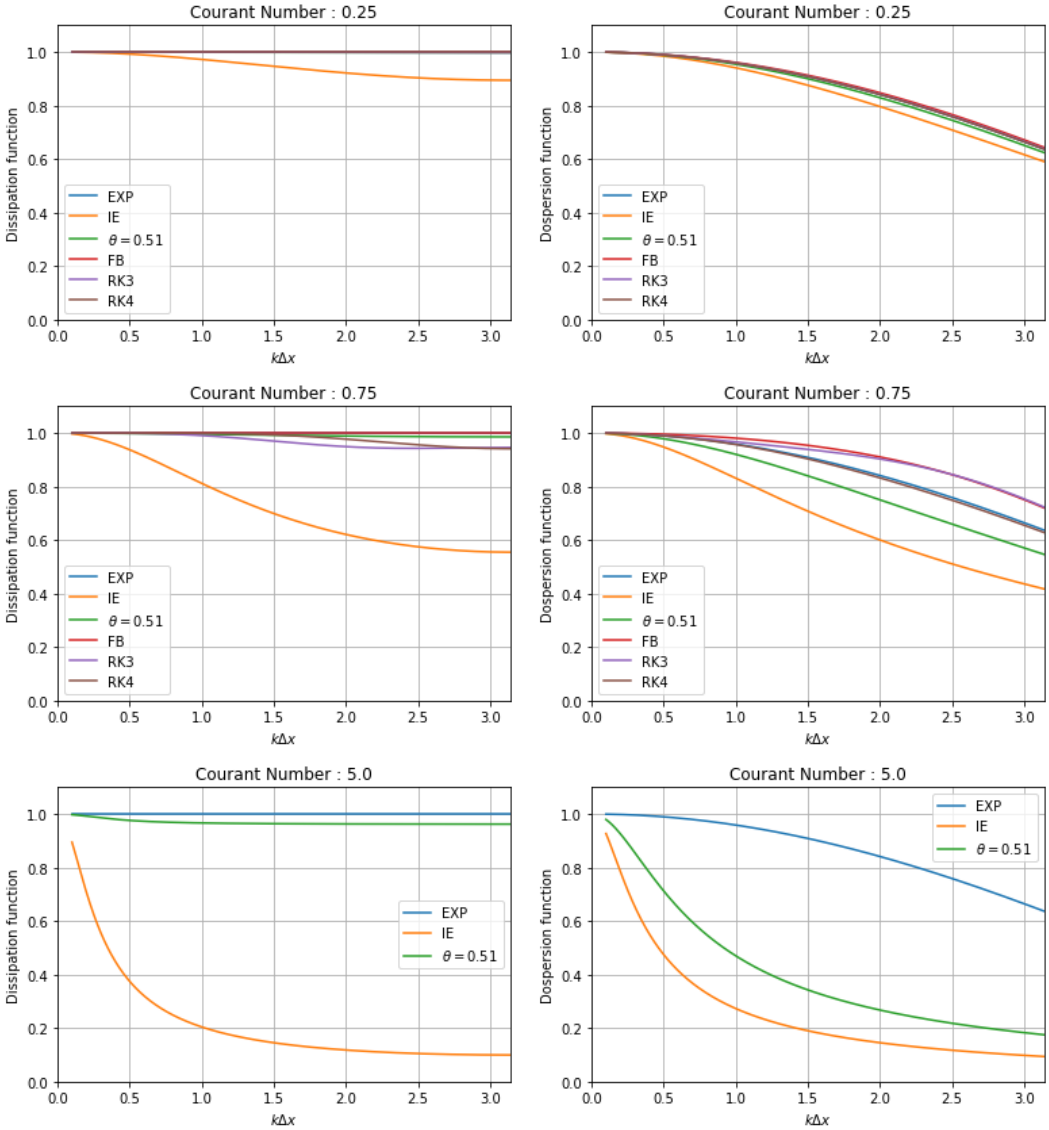

FIGURE 5. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C2 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered : 0.25, 0.75 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

Three source terms f will be considered. The first is $f = 0$ resulting in a simple homogeneous linear equation. Obviously, exponential integrators are exact in this case and the only error comes from the spatial discretization. The accuracy of the explicit schemes is a function of their orders. Finally implicit integrators may suffer from poor dissipation and dispersion properties when used in combination with a large time step.
In the second test case, the forcing function f is non zero but depends only on time. Its frequency is given by a parameter ω. Large ω values imply rapid variations of the exact solution. In this context, the choice of the time step is crucial to maintain an accurate numerical solution.
In the third test case f depends both on time and space. These test case give us, among other, an idea of the splitting error which was invisible in the second test case. It is also used to evaluate the link between accuracy and computational cost.

In all test cases, we consider a domain length $d = 500000$ meters and the gravity parameter $g = 9.81\text{m} \cdot \text{s}^{-2}$. Initialization (h_0, u_0) and sources terms f will be specified in the following. Two cases are considered: shallow ocean ($\bar{h} = 100\text{m}$) and deep ocean ($\bar{h} = 4000\text{m}$). The change of deep impacts the

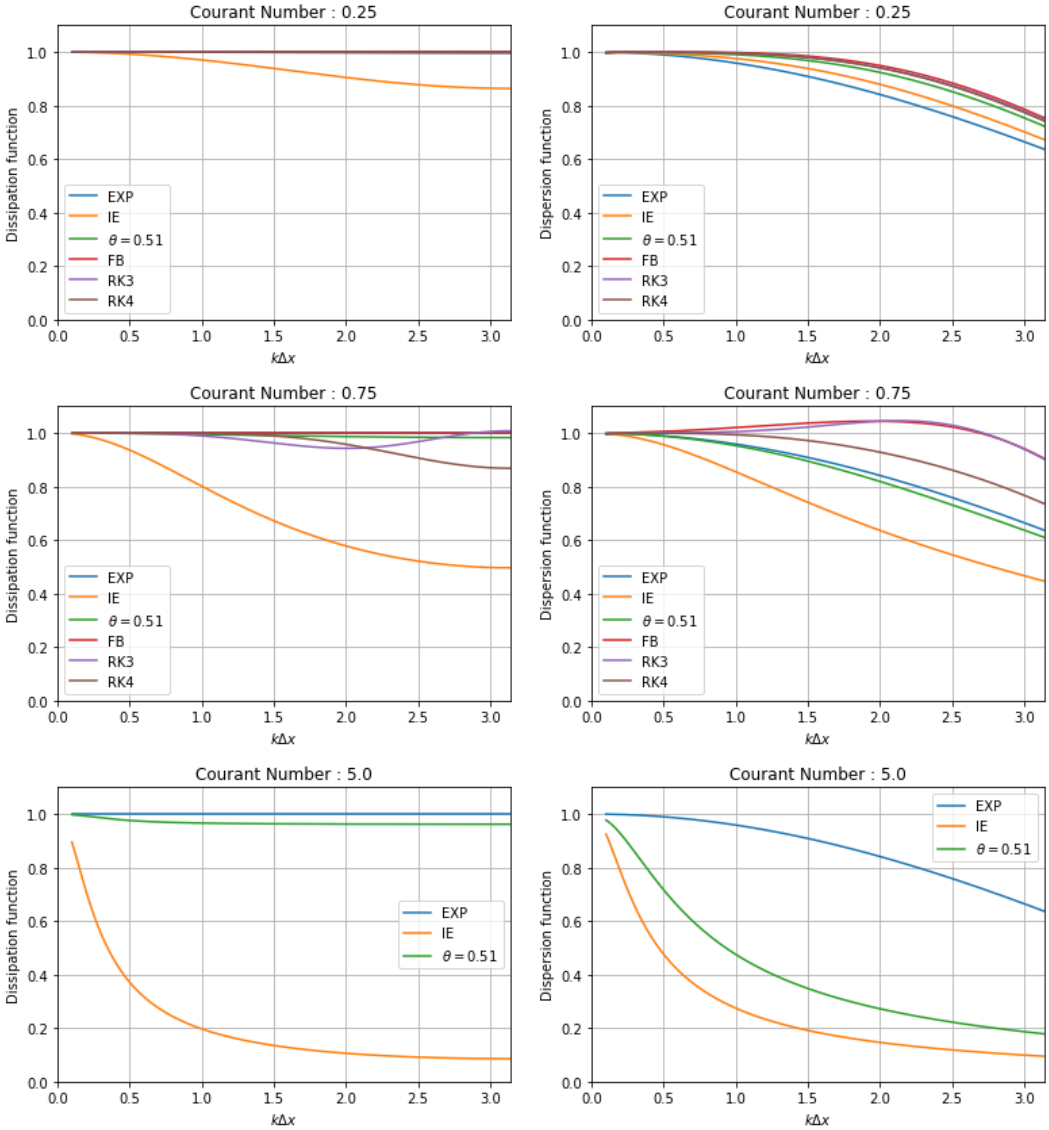

FIGURE 6. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C4 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered : 0.25, 0.75 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

value of the propagation speed c and thus of the Courant number. The ratio between the two thickness is $4000/100 = 40$, it results into a ratio on the Courant numbers $\sqrt{4000}/\sqrt{100} \approx 6.32$.

5.1. **Linear Case.** Consider (1.8) without source term. More precisely, f is

$$f(t,x) = 0 \text{ for all } x \in [0,d],\ t \geq 0. \tag{5.1}$$

Then, (1.8) is a linear autonomous PDE.

In our experiment, the initial data is

$$\begin{cases} h_0(x) = H \cdot \exp\left(-\left(\dfrac{x - 0.5 \cdot d}{\sigma}\right)^2\right) \\ u_0(x) = 0 \end{cases} \tag{5.2}$$

where $\sigma = d/10$ and $H = 1\text{m}$. Functions h_0 and u_0 are chosen such that we do not obtain an eigenvalue of L. If X^n is an eigenvalue of L, then the Krylov method to compute $\exp(\Delta t L)X^n$ converges into a

single iteration. These have no reason to happen in practice. At each time $t \geq 0$, the solution is

$$\begin{cases} h(t,x) = \dfrac{1}{2}\left(h_0(x-ct) + h_0(x+ct)\right) \\ u(t,x) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left(u_0(x-ct) - u_0(x+ct)\right) \end{cases} \tag{5.3}$$

where $c = \sqrt{g\bar{h}}$.

Visualisation of dissipation/dispersion errors of the unconditionally stable schemes. On Figure 7, we plot the water depth h at times $t = 1$ hours, $t = 10$ hours and $t = 40$ hours computed using different time schemes and with a fourth order (C4) spatial discretization and $N = 500$ grid points. The ocean is shallow ($\bar{h} = 100$ meters). In this high resolution case, the solution is spatially well resolved and most of the errors comes from the time stepping algorithm. The time step is $\Delta t = 300$s and corresponds to a Courant number $c\Delta t/\Delta x = 9.39$. We observe the dissipation of the solution computed with Backward Euler and the dispersion using Crank-Nicholson. The curves associated to θ-scheme ($\theta = 0.51$) is distorted too. As expected, exponential integrators are accurate, the numerical solution is visually not distinguishable to the exact solution. Similar conclusions are obtained using a second order accurate scheme (C2).

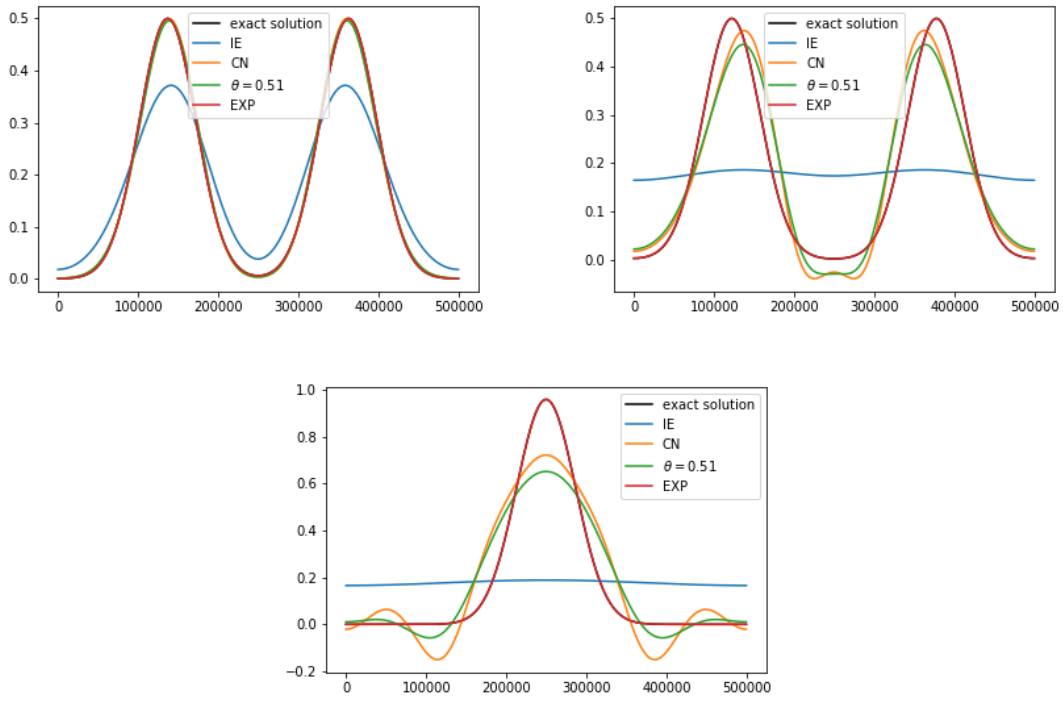

FIGURE 7. Numerical solutions h computed using C4 space operators with $N = 500$ grid points and $\bar{h} = 100$ meters. Different time schemes are considered with $\Delta t = 300$ secs ($c\Delta t/\Delta x = 9.39$). Top left panel: solution at $t = 1$hour. Top right panel: solution at $t = 10$hour. Bottom panel: solution at $t = 40$hour.

Accuracy / Computational cost. We here compare the error on h and u for the different time schemes, and the associated computational costs. These costs are assumed to be proportional to the number of calls of the linear part. For the exponential integrators and implicit schemes, these calls arise within the matrix-vector products of the Krylov and conjugate gradient methods.

We compute the relative errors on h by formula

$$\frac{\|h(t^n,\cdot) - \mathfrak{h}^n\|_2}{1 + \|h(t^n,\cdot)\|_2} \tag{5.4}$$

and on u by

$$\frac{\|u(t^n,\cdot)-\mathfrak{u}^n\|_2}{1+\|u(t^n,\cdot)\|_2}. \tag{5.5}$$

This errors are considered to avoid division by a very small number when $\|h(t^n,\cdot)\|_2$ or $\|u(t^n,\cdot)\|_2$ are close to zero. Only one time step is considered with explicit time schemes. They are close to their maximum allowed values: it minimizes the number of total calls required to achieve the final time of integration. The time steps Δt considered for explicit time schemes are summarized in Table 5.

	Shallow - C2	**Shallow - C4**	**Deep - C2**	**Deep - C4**
FB	31.9	27.3	5.0	4.3
RK3	27.6	23.6	4.3	3.7
RK4	45.1	38.7	7.1	6.1

TABLE 5. Time steps Δt considered for explicit time FB, RK3 and RK4. With this values, there are $N = 500$ grid points corresponding to $\Delta x = 1000$ meters. These time steps are close to the maximum value allowed to ensure stability.

Implicit time schemes and exponential integrators are A-stable, there are no constraint on the time step. We consider various values summarized in Table 6.

Δt	100	600	1200	3600	7200
Shallow ocean	3.05	18.31	36.61	109.84	219.68
Deep ocean	19.30	115.79	231.57	697.71	1384.42

TABLE 6. Courant numbers $c\Delta t/\Delta x$ related to time steps in case of shallow ocean ($\bar{h} = 100$ meters) and deep ocean ($\bar{h} = 4000$ meters). The space discretization parameter $\Delta x = 1000$ meters which corresponds to $N = 500$.

Results are plotted in Figure 8 using second order space discretization C2, and in Figure 9 for the fourth order accurate operators C4. As expected, exponential integrator is accurate in the two cases. Observed errors corresponds to the spacial error. Implicit time schemes are less accurate, this is due to the poor dissipation and dispersion properties. For the shallow ocean and implicit time schemes, we remark that the smallest time steps give the most accurate result. Unfortunately, the error increase quickly. Exponential integrator and implicit time schemes are cheaper than explicit time schemes with larger Courant numbers. With the maximum allowed time step, RK3 and RK4 are less accurate than exponential integrator but costly. Exponential integrator is cheaper than other time schemes and more accurate. Using the same time step Δt, exponential integrators are cheaper and more accurate than implicit scheme.

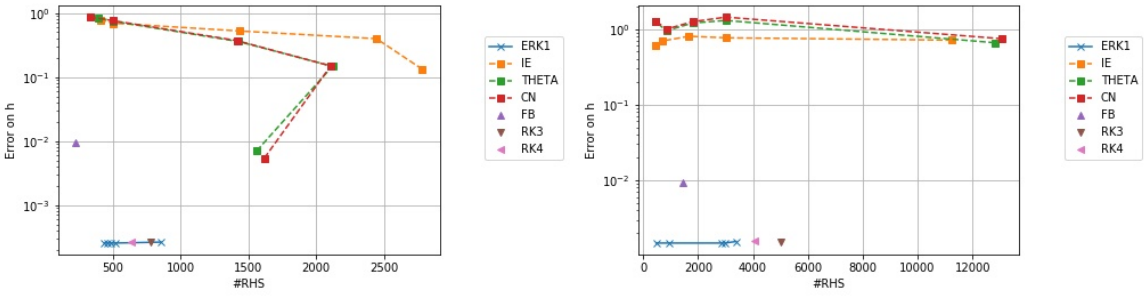

FIGURE 8. Linear test case. Relative error on h as a function of the number of right hand side (RHS) evaluations (which is itself given by the choice of the time step). The final time is $T = 7200$secs. On the left panel, we consider the shallow ocean $\bar{h} = 100$ meters, on the right panel, the deep ocean with $\bar{h} = 4000$ meters. The space scheme is C2 with $N = 500$ grid points.

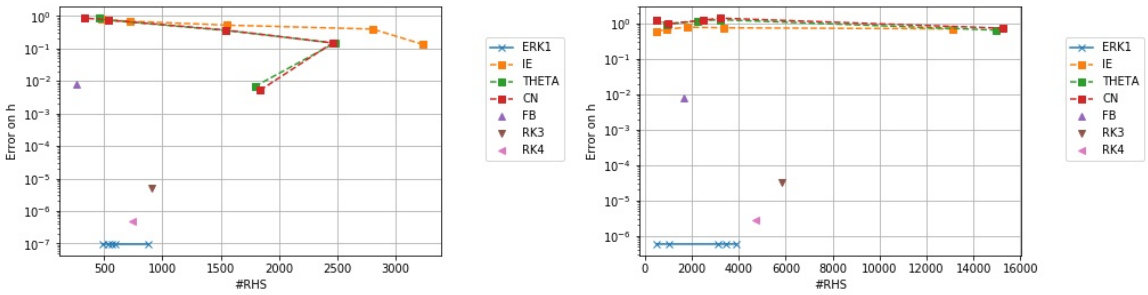

FIGURE 9. Linear test case. Relative error on h. The final time is $T = 7200$secs. On the left panel, we consider the shallow ocean $\bar{h} = 100$ meters, on the right panel, the deep ocean with $\bar{h} = 4000$ meters. The space scheme is C4 with $N = 500$ grid points.

5.2. **Time dependant forcing test case.** Even if the linear case give us an idea of results about efficiency, it is not sufficiently challenging. Indeed, exponential integrators solve exactly linear differential equations. Then, the observed error in the previous test is due to the space discretization C2 or C4.

We consider equation (1.8) initialized with $h_{|t=0} = h_0$ and $u_{|t=0} = 0$ on a periodic grid given by (5.2). To avoid exact solution, we consider $f \neq 0$, time dependant. Here, f is given by $f(t) = K \cdot \sin(\omega t)$, $K = 1 \cdot 10^{-5}\mathrm{m} \cdot \mathrm{s}^{-2}$. It does not depend of x. The solution is

$$\begin{cases} h(x,t) = \dfrac{1}{2}\left[h_0(x-ct) + h_0(x+ct)\right] \\ u(x,t) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left[h_0(x-ct) - h_0(x+ct)\right] + \dfrac{K}{\omega}\left(1-\cos\omega t\right). \end{cases} \tag{5.6}$$

In this test case, exponential integrators are not exact and a time errors will be apparent.

Influence of the frequency ω. The source term f is T_p-periodic with $T_p = 2\pi/\omega$. Large values of ω generates quick change. Given a time step Δt, it is more difficult for a time scheme to capture the solution. On Figure 10, we plot the modified relative errors (5.4) and (5.5) according to ω in the case of shallow ocean. The time step is $\Delta t = 600$ seconds.

The error on h does not depend on ω because the source function does not impact h equation. On contrary, the error on u increase with ω. It was attempt due to the quick change in the source term f. Large time steps, by comparison to the frequency ω, generate large errors whatever the time scheme considered. The phenomenon is less visible with C2 because of the spatial error but it is also present.

Accuracy/Computational cost. Even if a time integrator is accurate, error has to be compare to computational cost. Figure 11 (resp. Figure 12) contains the plots of modified errors related to the number of right hand side calls to reach the final time using C2 (resp. C4) with $\omega = 10^{-4}\mathrm{s}^{-1}$. For explicit time stepping, we consider values Δt given in Table 5. These reduces the computational cost to the minimum. Greater time steps are considered for A-stable time integrators.

On Figure 11, the space error is larger than time error in most of the time integrators due to the second order space accuracy (C2). Implicit time schemes have large errors whatever the time step considered. LERK1 and ERK1c are the less accurate exponential integrators. These are the only ones whose time error is visible. The FB scheme is less costly than the other but has also a poor accuracy. FB is more accurate than anticipated on u in deep ocean. It is due to error compensations. Exponential integrators with larger time steps are accurate and cheaper than RK3 and RK4.

Figure 12 allows to assess to the time scheme accuracy by using fourth order accurate space operators C4. Thus, the space error is small with the same grid. ERK1c, LERK and S1ERK4 are less costly than Runge-Kutta time schemes when we use the largest time steps. Substepped splitting exponential integrators subS1ERK4 and subS2ERK1 are particularly accurate, whatever the time step used because source function depends only on time. Indeed, in this particular context, splitting exponential integrators are fourth order accurate because error terms are zero when $[0, \mathfrak{f}(t)]^T \in \mathbb{R}^{2N}$ is in the kernel of L. It happens for all $t \geq 0$ in this test case. Furthermore, the linear part is exactly solved by exponential, the non linear part is accurately solved using RK4 with a small time step and splitting error is zero. In a realistic model, there is no reason why f does not depend of space.

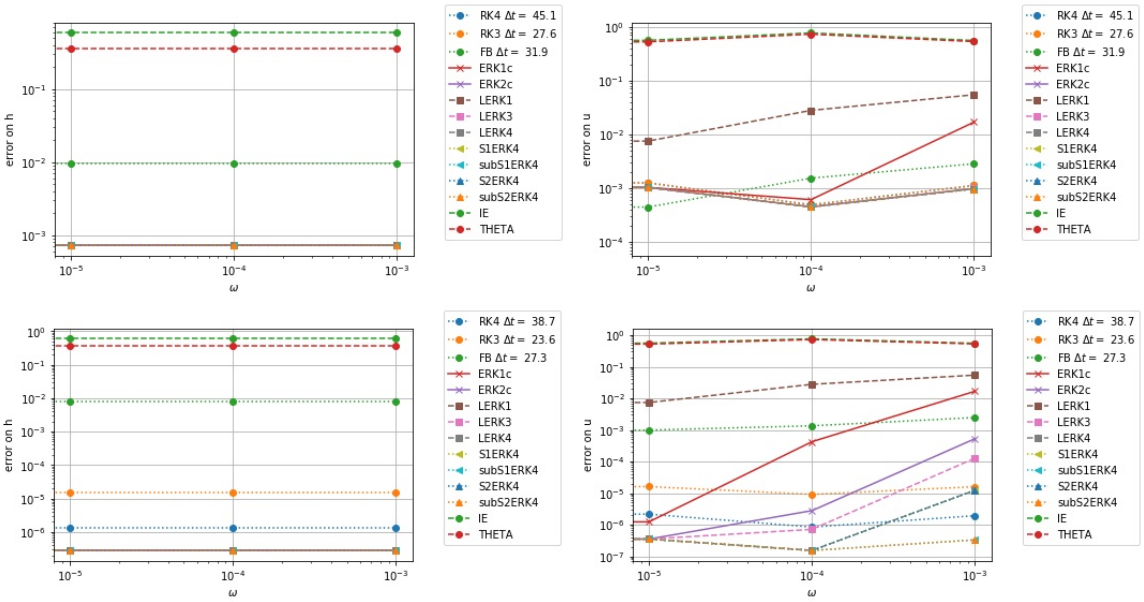

FIGURE 10. Time source term test case. We plot the relative error on h and u using different time scheme and values of ω. Top panel: C2 space scheme, Bottom panel: C4 space scheme. The parameters are $\bar{h} = 100$ meters, $t_{\max} = 2$hours and $N = 500$. A-stable time schemes are using $\Delta t = 600$ seconds. Explicit time schemes are using time steps close to the maximum allowed (see Table 5).

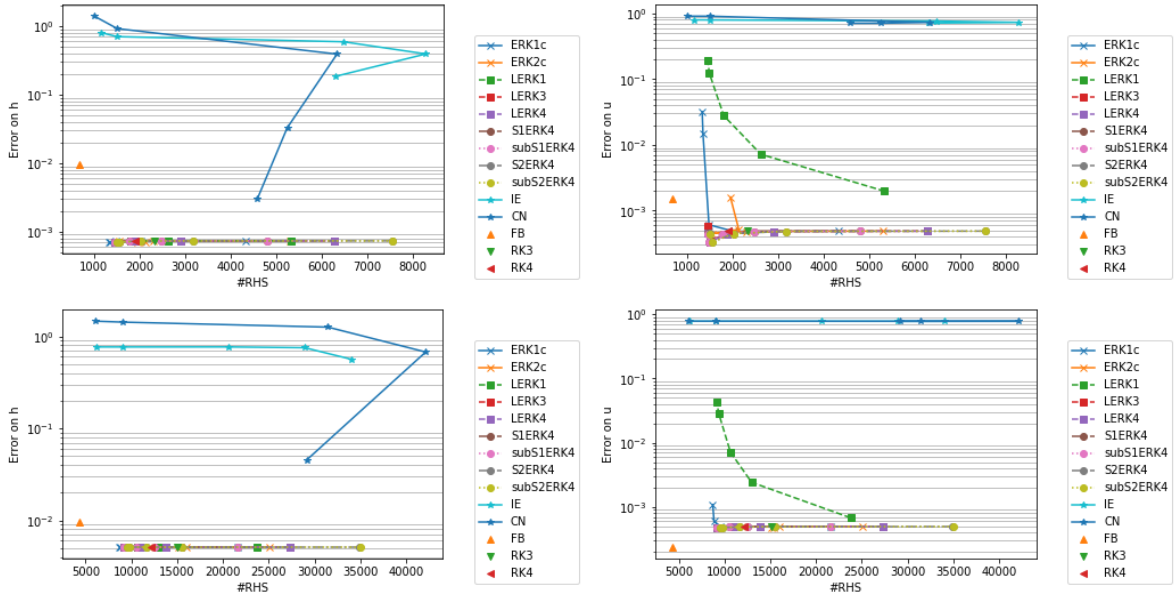

FIGURE 11. Time source term test case. Modified error related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$ meters) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$ meters) with $\Delta t \in \{10, 50, 150, 600, 900\}$ for A-stable schemes. For both, there are $N = 500$ grid points and C2 space discretization is used.

For both case (C2 and C4), LERK3 and LERK4 are cheaper than expected. It happens because b is often in the kernel of L when we compute $\exp(\Delta L)b$ during time iterations. Then, Krylov algorithm converges into a single iteration. This is due to the independence of f in space. The next test case will avoid this situation.

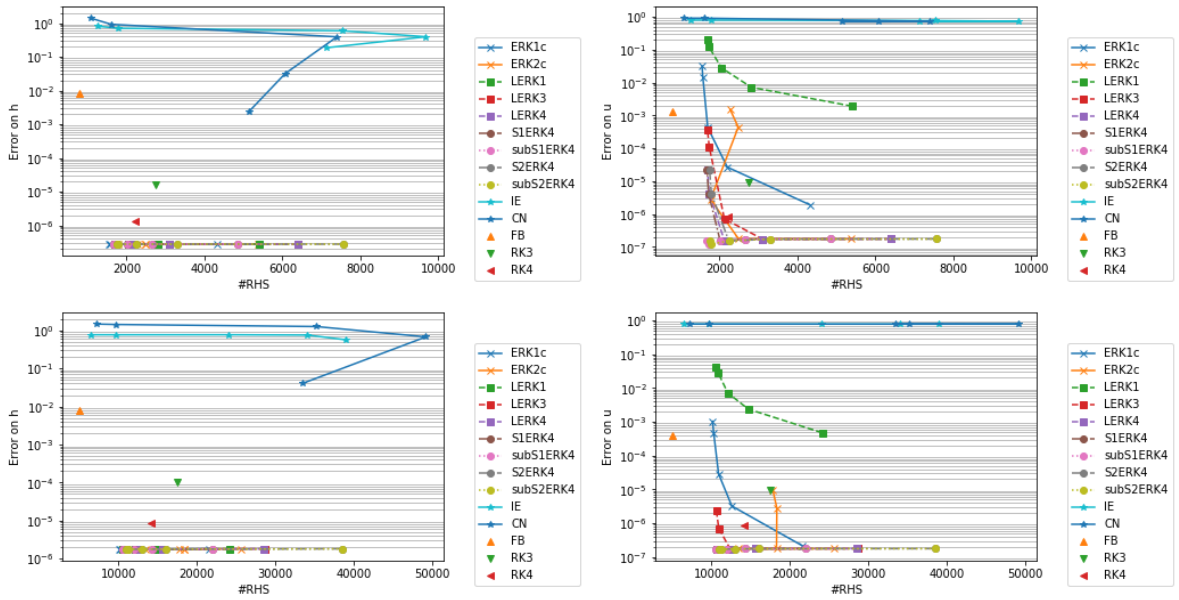

FIGURE 12. Time source term test case. Modified error related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$ meters) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$ meters) with $\Delta t \in \{10, 50, 150, 600, 900\}$ for A-stable schemes. For both, there are $N = 500$ grid points and C4 space discretization is used.

5.3. **Time and space dependant forcing test case.** In the previous section, the substepped splitting schemes subS1ERK4 and subS2ERK4 were very accurate. Furthermore, LERK3 and LERK4 are less costly than expected. This was because the source term f does not depend on space. In practice, there is no reason for this to happen. To avoid this idealized case, we consider a test in which f depends on time and space (i.e. $f = f(t, x)$):

$$f(t, x) = K \cdot \sin \omega t \cos kx. \tag{5.7}$$

We initialize (1.8) with (h_0, u_0) given by (5.2). At each time $t \geq 0$ and for all $x \in \mathbb{R}$, the height h is

$$h(x,t) = \frac{1}{2}\left(h_0(x - ct) + h_0(x + ct)\right) + + \sqrt{\frac{\bar{h}}{g}} K \frac{\omega \sin(kct) - ck \sin(\omega t)}{\omega^2 - c^2 k^2} \sin(kx), \tag{5.8}$$

and the velocity u is

$$u(x,t) = \frac{1}{2}\sqrt{\frac{g}{\bar{h}}}\left(h_0(x - ct) - h_0(x + ct)\right) + K\omega \frac{\cos(kct) - \cos(\omega t)}{\omega^2 - c^2 k^2} \cos(kx). \tag{5.9}$$

where $c = \sqrt{g\bar{h}}$. We choose $k = 4\pi/d \approx 2.51 \cdot 10^{-5}\mathrm{m}^{-1}$ to ensure periodicity and well resolution in space. The temporal frequency is $\omega = 10^{-4}\mathrm{s}^{-1}$. It corresponds to a time period $T_p = 2\pi/\omega \approx 62832$s. The parameter K is $K = 1 \cdot 10^{-5}\mathrm{m} \cdot \mathrm{s}^{-2}$.

Visualisation of h and error. The height h is plotted using different time schemes on left plots on Figure 13 using $N = 500$ grid points and C4 space discretization in case of shallow ocean. Results are less accurate with C2 but analyze is similar. The time step is $\Delta t = 600$s for A-stable time integrators. It corresponds to $c\Delta t/\Delta x \approx 18.79$. The time steps using explicit schemes are $\Delta t_{\mathrm{RK4}} = 30$s ($c\Delta t/\Delta x \approx 0.93$) and $\Delta t_{\mathrm{FB}} = 20$s ($c\Delta t/\Delta x \approx 0.63$). The right plots in Figure 13 are the relative errors

$$\frac{\mathfrak{h}_j^n - h(t^n, x_j)}{\max_j |h(t^n, x_j)|} \tag{5.10}$$

where $\mathfrak{h}_j^n$ is the numerical solution on x_j at time t^n and $h(t^n)$ the exact solution at the same time. We do not plot the errors for IE, CN and θ-scheme ($\theta = 0.51$) because they are visibly not accurate.

The time stepping ERK2c is the more accurate exponential integrator due to the third order accuracy. Implicit schemes have poor accuracy due to their dissipation and dispersion properties. Splitting time

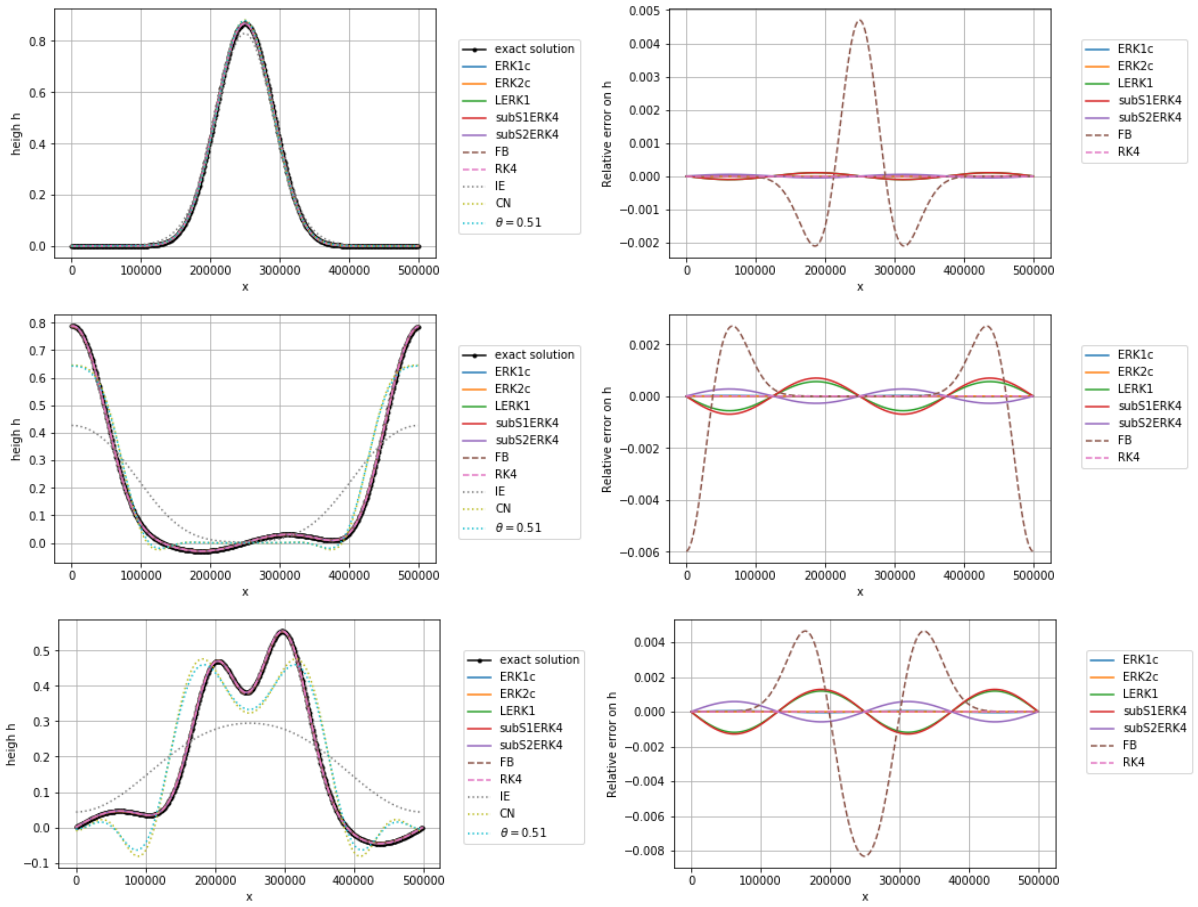

FIGURE 13. Space time source term test case. Numerical solution h computed using C4 space operators with $N = 500$ grid points and $\bar{h} = 100$ meters. Different time schemes are considered with $\Delta t = 600$ secs. First line panels: solution and relative error on h at $t = 10$ minutes. Second line panels: solution and relative error on h at $t = 2$ hours. Third line panels: solution and relative error on h at $t = 4$ hours.

schemes subS1ERK4 and subS2ERK4 are not as accurate than in the previous test case. These is due to the splitting error. It is observable in this test case because f depends on space in contrast to the previous test case. Obviously, RK4 is more accurate than ERK1c but also more expansive.

Computational cost. In Table 7, we give the number of calls of the linear part to reach $t = 2$ hours using C4 space operators and different time configurations. The computational cost is assumed to be proportional to the number of calls of the linear part. Values of explicit time schemes, implicit time schemes and exponential integrators are compared.

Greater the time step is, cheaper the integrators are. Implicit integrators with small time steps Δt are the more expansive. For exponential integrators, the computational cost is linked to the number of exponential functions occurring. Indeed, LERK3 and LERK4 are the more expansive exponential integrators. ERK1c and LERK1 are the cheaper integrators due to the only one exponential functions computed at each time step. To compare with explicit time schemes, 744 linear parts are computed to reach the final time with RK4 ($\Delta t = 38.7$ seconds, $c\Delta t/\Delta x = 1.21$), and 263 linear parts are computed with FB ($\Delta t = 27.3$ seconds, $c\Delta t/\Delta x = 0.85$). FB time integrator is the cheaper time scheme considered on this test cases.

Accuracy/Computational cost. As in the previous test cases, we consider the error related to the number of calls of the linear part. Errors on h and u are computed with formulae (5.4) and (5.5). Results are plotted in Figure 14 (C2) and Figure 15 (C4). The time steps for explicit integrators are given in Table 5.

	$\mathbf{\Delta t = 300}$s $\mathbf{c\Delta t/\Delta x = 9.39}$	$\mathbf{\Delta t = 600}$s $\mathbf{c\Delta t/\Delta x = 18.79}$	$\mathbf{\Delta t = 3600}$s $\mathbf{c\Delta t/\Delta x = 112.75}$
IE	3933	2805	672
CN	2538	2457	533
$\theta = 0.51$	2564	2478	533
ERK1c	597	555	499
ERK2c	1090	963	615
LERK1	772	683	573
LERK3	1924	1557	769
LERK4	1924	1557	769
subS1ERK4	748	671	571
subS2ERK4	912	748	596
RK4 ($\Delta t = 30$s)	960	960	960
RK4 ($\Delta t = 38.7$s)	744	744	744
FB ($\Delta t = 20$s)	360	360	360
FB ($\Delta t = 27.3$s)	263	263	263

TABLE 7. Space time source term test case. Number of calls of the linear part to reach $t = 2$ hours using C4 space operators with $N = 500$ grid points and $\bar{h} = 100$ meters.

For smallest time steps, exponential integrators are more accurate than Runge-Kutta time stepping. High order exponential integrators (LERK3 and LERK4) are also more expansive. ERK1c and ERK2c are accurate whatever the time step considered. ERK1c is cheaper than RK3 and RK4 for the four largest values of Δt. As expected, ERK2c is more expansive than ERK1c. LERK are both expansive and imprecise, even if the order of accuracy are largest. Splitting exponential integrators are not as accurate than in the previous test case, these was attempt due to the splitting error. Furthermore, they are as expansive than ERK1c. FB is more accurate on u than expected with the second order accurate scheme C2. It is due to compensation of errors. In all cases, FB is the cheaper time scheme. In this experiment, ERK1c is a good compromise between accuracy and computational cost while implicit time schemes are imprecise and costly.

6. Conclusion

In this article, we analyze a large set of time schemes to solve LSWE (1.8).

- *Explicit time schemes:* Runge-Kutta methods accurate order 3 and 4 and Forward-Backward scheme. This integrators are cheaper by time step but a large number of time iterations is needed to reach final time.
- *Implicit time schemes:* in the set of θ-schemes. In particular, we consider the case $\theta = 0.51$, $\theta = 1$ (Backward Euler) and $\theta = 0.5$ (Crank-Nicholson). They are unconditionally stable, the time step is chosen as larger as needed. Unfortunately, implicit time schemes suffer dramaticaly from poor dissipation and dispersion properties.
- *Exponential Integrators:* Exponential Runge-Kutta integrators, Linearly Exact Integrators, Splitting scheme are exact for linear autonomous equations. They are unconditionally stable. Dissipation and dispersion properties depend only on the space discretization.

The space discretization is done on a staggered grid and two order of accuracy are considered. Two space operators are considered: second order accurate (C2) and fourth order accurate (C4). Fourth order accurate scheme is obviously more accurate but using it, time integrators are more costly. Explicit time schemes have more restrictive conditions using C4 than C2. Furthermore, Krylov methods and conjuguate gradient converge more slowly with high order accurate space discretisation.

Due to the poor dissipation and dispersion properties, implicit solver suffer from weak accuracy. Numerical solution become flat or distort, in particular with great Courant number. Exponential integrators have better dissipation and dispersion properties. It particular, this properties do not depend on Courant number. For this reason, at the same Courant number, exponential integrators are always more accurate than implicit integrators in our experiments. Whatever the time integrator considered, large Courant number are not appropriate to compute accurately a solution that changes quickly.

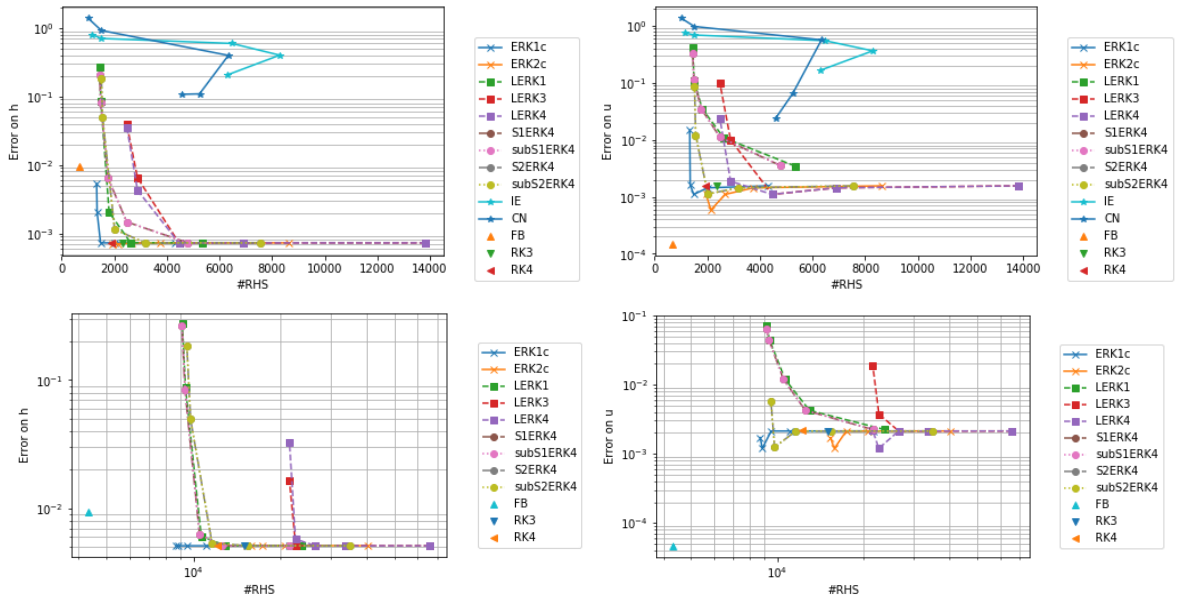

FIGURE 14. Space time source term test case. Modified error on h and u related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$) with $\Delta t \in \{10, 50, 150, 600, 900\}$ for A-stable schemes. For both, there are $N = 500$ grid points and C2 space discretization is used.

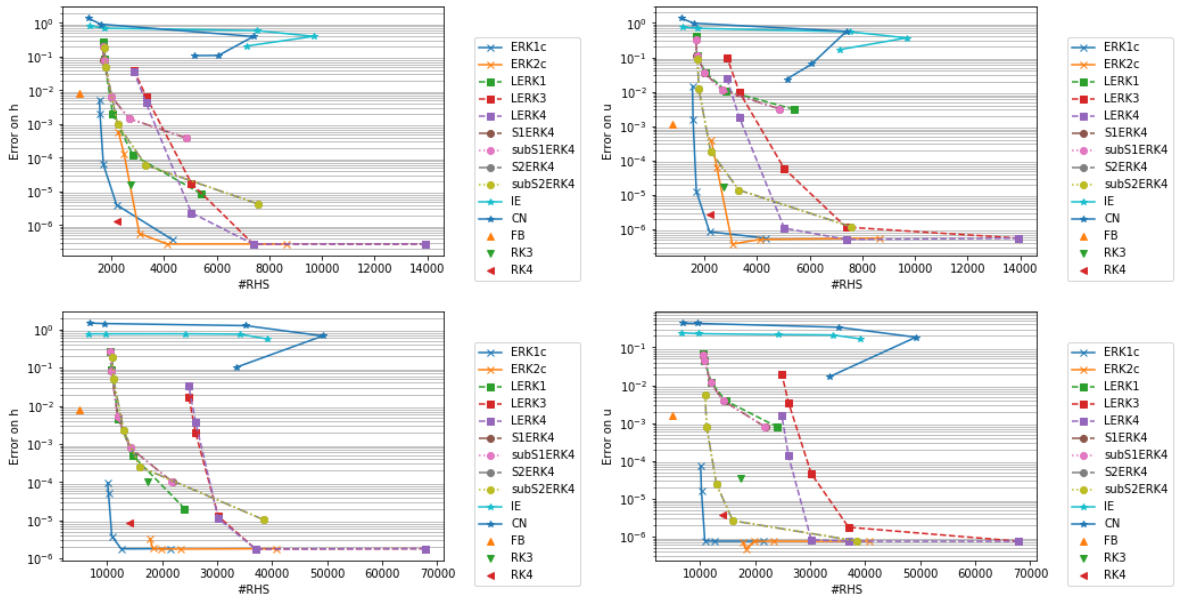

FIGURE 15. Space time source term test case. Modified error on h and u related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$) with $\Delta t \in \{10, 50, 150, 600, 900\}$ for A-stable schemes. For both, there are $N = 500$ grid points and C4 space discretization is used.

We analyze various exponential integrators. Linearly exact Runge-Kutta integrators are the more accurate time scheme considered. However they are expansive. Exponential Runge-Kutta integrators and splitting exponential integrators are cheaper. Splitting exponential integrators suffer dramatically to

the splitting error. Exponential Runge Kutta integrators are a good compromise between accuracy and computational cost.

Exponential integrators creates an opportunity to perform an accurate solver for shallow water equation using larger time steps than explicit integrators. The main disadvantage (but not least) of exponential integrators is the computational cost. There exist a large panel of methods allowing to reduce the computational cost in implicit time integrators to solve system. We hope to use the same kind of method to reduce the computational cost of exponential integrators. Among others we mention: parallelism (see [13, 33]), preconditioning (in the context of Lanczos methods [12, 36] or [8]) and domain decomposition [4, 24]. The application of this kind of method to exponential integrators will be the topic of future works.

Appendix A. Time integrators

We consider the ordinary differential equation

$$\frac{dX}{dt} = \mathcal{F}(X,t). \tag{A.1}$$

In some case, equation are autonomous: $\mathcal{F}$ does not depend of time, $\mathcal{F}(X,t) = \mathcal{F}(X)$. Equation (A.1) can be split between linear and non linear part:

$$\mathcal{F}(X,t) = LX + \mathcal{N}(X,t). \tag{A.2}$$

In the context of (1.8), the function $\mathcal{N}(X,t)$ does not depend on X, more precisely:

$$\frac{dX}{dt} = LX + b(t). \tag{A.3}$$

Single step time integration is to compute $X^{n+1} \approx X(t^{n+1})$ starting from $X^n \approx X(t^n)$. The values $(t^n)_n$ are define by $t^n = n\Delta t$, where $\Delta t > 0$ is the time step.

In the following, we recall some time integrators.

A.1. Explicit Time Schemes.

Explicit Runge Kutta integrators are widely used to solve differential equation. In this section, we consider two Runge-Kutta methods respectively order 3 and 4 and a Forward Backward scheme.

A.1.1. *The Ralston's third order integrator.*

The Ralston's third order integrator [30] is a third order accurate explicit Runge-Kutta integrator. It is given by

$$\begin{cases} K_1 = \mathcal{F}\left(X^n, t^n\right) \\ K_2 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{F}\left(X^n + \dfrac{3\Delta t}{4} K_2, t^n + \dfrac{3\Delta t}{4}\right) \\ X^{n+1} = X^n + \dfrac{\Delta t}{9}\left(2K_1 + 3K_2 + 4K_3\right). \end{cases} \tag{A.4}$$

At each time step, $\mathcal{F}$ is called three times.

A.1.2. *Fourth order Runge-Kutta integrator.*

An alternative to RK3 is the fourth order accurate Explicite Runge-Kutta integrators (RK4). It is given by the following steps:

$$\begin{cases} K_1 = \mathcal{F}\left(X^n, t^n\right) \\ K_2 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{F}\left(X^n + \Delta t K_3, t^n + \Delta t\right) \\ X^{n+1} = X^n + \dfrac{\Delta t}{6}\left(K_1 + 2K_2 + 2K_3 + K_4\right). \end{cases} \tag{A.5}$$

This time scheme is more accurate than RK3 but the right hand side part is called one time more.

A.1.3. *Forward-Backward time scheme.* The Forward-Backward (FB) time scheme is a well known time integrator for Linearized Shallow Water equation (1.8) (see [34] and references therein). It has a small computational cost per iteration: the right hand side is evaluated only once per time iteration while being second order accurate in time. This scheme is given by

$$\begin{cases} \mathfrak{h}_{i+1/2}^{n+1} = \mathfrak{h}_{i+1/2}^{n} - \Delta t \bar{h} \delta_x u_{i+1/2}^{n} \\ \mathfrak{u}_{i}^{n+1} = \mathfrak{u}_{i}^{n} - \Delta t g \delta_x u_{i}^{n+1} \end{cases} \tag{A.6}$$

where δ_x is a finite difference operators for the first order space derivative (C2 or C4).

A.2. **Implicit Time Scheme.** Implicit time schemes are an alternative those that are explicit. They allow us to use greater time steps (without CFL restriction) what is impossible with an explicit scheme. An example of one step implicit integrator is the θ-scheme given by

$$\frac{X^{n+1} - X^n}{\Delta t} = \theta \mathcal{F}(X^{n+1}, t^{n+1}) + (1-\theta)\mathcal{F}(X^n, t^n) \tag{A.7}$$

where $\theta \in [0,1]$. A soon as $\theta \neq 0$, an equation must be solved at each time step, this can be done using a Newton's algorithm or a linear solver in case of linear equation. In the context of (A.3), the scheme is

$$(\mathrm{Id} - \theta \Delta t L) X^{n+1} = X^n - \Delta t \left(\theta b(t^{n+1}) + (1-\theta) b(t^n) + (1-\theta) L X^n \right). \tag{A.8}$$

The θ-scheme is first order accurate if $\theta \neq 1/2$ and second order accurate if $\theta = 1/2$. In the case $\theta = 0$ (resp. $\theta = 1$), it corresponds to an Forward Euler scheme (resp. Backward Euler). The Crank-Nicholson scheme corresponds to $\theta = 1/2$ and for this value of θ, the scheme is neutral (no dissipation). In the numerical experiments, we will also consider the case $\theta = 0.51$. There are no stability criterion while $\theta \geq 1/2$.

A.3. **Exponential Integrators.** Exponential Integrators are an alternative to implicit scheme to solve stiff problems and keep stability. These allow to overcome the CFL condition as for implicit schemes but without the damping of high frequencies. We refer to [17] to review. In this section, we remind ideas of exponential integrators.

We start by consider an autonomous system:

$$\frac{dX}{dt} = \mathcal{F}(X). \tag{A.9}$$

By separating linear and non linear part $\mathcal{F}(X) = LX + \mathcal{N}(X)$. The function $X : t \mapsto X(t)$ satisfies

$$X(t^n + \Delta t) = \exp(\Delta t L) X(t^n) + \int_0^{t^n} \exp((\Delta t - \tau) L) \mathcal{N}(X(t^n + \tau)) \, d\tau \tag{A.10}$$

Various quadrature rules on the integral give us different exponential integrators. These depend of matrix functions of the set $(\varphi_k)_k$ define by

$$\begin{cases} \varphi_{k+1}(z) = \dfrac{\varphi_k(z) - \varphi_k(0)}{z} \\ \varphi_k(0) = 1/k! \\ \varphi_0(z) = \exp(z). \end{cases} \tag{A.11}$$

Exponential Integrators are A-stable and exact for linear equations. In the following, we present different kind of exponential integrators.

A.3.1. *Exponential Euler Integrator.* The simplest numerical method is to consider $\mathcal{N}(X(t^n + \tau)) \approx \mathcal{N}(X(t^n))$ in the integral part of (A.10). It leads to the Exponential Euler Integrator (ERK1):

$$X^{n+1} = \exp(\Delta t L) X^n + \Delta t \varphi_1(\Delta t L) \mathcal{N}(X^n) \tag{A.12}$$

with X^0 given by the initial condition and $X^n \approx X(t^n)$. The function φ_1 is $\varphi_1(z) = \frac{e^z - 1}{z}$ and extends on $z = 0$ by $\varphi_1(0) = 1$. The ERK1 exponential operator is first order accurate. If L contains the full linear part (see Rosenbrock exponential integrators [17])

$$L = L_n = \mathrm{Jac}_{X^n} \mathcal{F} \tag{A.13}$$

it is second order accurate. Here, $\mathrm{Jac}_{X^n} \mathcal{F}$ the Jacobian matrix. Considering (A.11), ERK1 can be written

$$X^{n+1} = X^n + \Delta t \varphi_1(\Delta t L) \mathcal{F}(X^n). \tag{A.14}$$

These allow to compute only one matrix function instead of two in (A.12).

A.3.2. *Second order Runge-Kutta Exponential Integrator.* A more accurate scheme is ERK2:

$$\begin{cases} a^n = X^n + \Delta t \varphi_1(\Delta t L)\mathcal{F}(X^n) \\ X^{n+1} = X^n + 2\Delta t \varphi_3\left(\Delta t L\right)\left(\mathcal{N}(a^n) - \mathcal{N}(X^n)\right). \end{cases} \tag{A.15}$$

The function φ_3 is

$$\begin{cases} \varphi_3(z) = \dfrac{e^z - 1 - z - z^2/2}{z^3} \text{ if } z \neq 0 \\ \varphi_3(0) = 1/6. \end{cases} \tag{A.16}$$

Two matrix functions are required in (A.15). ERK2 is second order accurate in general but if L is given by (A.13), it is third order accurate.

A.3.3. *Non autonomous system.* In case of non autonomous equation, the previous exponential integrators must be adapted. The time scheme (A.14) can be used to solve this family of problems assuming $\mathcal{N}\left(X(t^n+\tau), t^n+\tau\right) \approx \mathcal{N}\left(X(t^n), t^n\right)$ but parasitic waves are occurring and accuracy is poor.

In [18], a modification of (A.14) and (A.15) is described to solve autonomous problems. Then ERK1 is modified into

$$X^{n+1} = X^n + \Delta t \mathcal{F}\left(X^n, t^n\right) + \Delta t^2 \varphi_2\left(\Delta t L\right)\left[L\mathcal{F}(X^n, t^n) + \frac{\partial \mathcal{F}}{\partial t}(X^n, t^n)\right] \tag{A.17}$$

with $\varphi_2(z) = (e^z - 1 - z)/z^2$ and $\varphi_2(0) = 1/2$. The time scheme (A.15) become

$$\begin{cases} a^n = X^n + \Delta t \mathcal{F}\left(X^n, t^n\right) + \Delta t^2 \varphi_2\left(\Delta t L\right)\left[L\mathcal{F}(X^n, t^n) + \dfrac{\partial \mathcal{F}}{\partial t}(X^n, t^n)\right] \\ X^{n+1} = a^n + 2\Delta t \varphi_3(\Delta t L)\left[\mathcal{F}(a^n, t^n + \Delta t) - \mathcal{F}(X^n, t^n) - L(a^n - X^n) - \Delta t \dfrac{\partial \mathcal{F}}{\partial t}(X^n, t^n)\right]. \end{cases} \tag{A.18}$$

We denote this scheme ERK1c and ERK2c. The schemes (A.17) and (A.18) are respectively accurate order 1 and 2, $A-stable$ and exact if $\mathcal{F}(X,t) = LX$. If L contains a full linear part as in a Rosenbrock exponential integrator (see (A.13)), ERK1c is second order accurate and ERK2c is third order accurate. This is our case to solve (1.8).

A.4. **Linearly Exact Runge-Kutta methods.** Linearly Exact Runge-Kutta methods, also called Integrating Factor methods, were generalized Runge-Kutta process describe first in [22]. A review of these methods is given in [5, 25] and reference therein. In the following, we call these methods LERK for Linearly Exact Runge-Kutta methods.

The idea of LERK is to consider the change of variable

$$V(t) = \exp((t^n - t)L)X(t) \tag{A.19}$$

in equation (A.3). Then V is solution of

$$V'(t) = \exp\left((t^n - t)L\right)\mathcal{N}\left(\exp\left((t - t^n)L\right)(t), t\right) \tag{A.20}$$

where $\mathcal{N}(X,t) = \mathcal{F}(X,t) - LX$, in our case $\mathcal{N}(X,t) = b(t)$. At this step, it is sufficient to apply an explicit time scheme on (A.20). If we choose $L = 0$, LERK coincide with an explicit time scheme. Different examples of LERK are given in the following subsections.

A.4.1. *Linearly Exact Runge Kutta order 1.* A LERK integrator first order accurate is obtained using Forward Euler scheme. The Forward Euler scheme is not stable when apply directly on (1.8) but the exponential part stabilize the scheme. It is given by the formula

$$X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \Delta t \mathcal{N}(X^n, t^n)\right]. \tag{A.21}$$

This scheme is called LERK1. In the case of (A.3), it is written:

$$X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \Delta t b(t^n)\right]. \tag{A.22}$$

A.4.2. *Linearly Exact RK3.* Applying the RK3 time scheme (A.4) on (A.20), we obtain a third order LERK integrator:

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_1\right), t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{3\Delta t}{4}L\right)\mathcal{N}\left(\exp\left(\frac{3\Delta t}{4}L\right)\left(X^n + \frac{\Delta t}{2}K_2\right), t^n + \frac{3\Delta t}{4}\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{9}(2K_1 + 3K_2 + 4K_3)\right]. \end{cases} \tag{A.23}$$

This scheme will be called LERK3 in the following. In the case of (A.3), LERK3 take the following form:

$$\begin{cases} K_1 = b(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right) b\left(t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{3\Delta t}{4}L\right) b\left(t^n + \frac{3\Delta t}{4}\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{9}(2K_1 + 3K_2 + 4K_3)\right]. \end{cases} \tag{A.24}$$

ERK3 contains three exponential of matrices while ERK2c contains 2 matrix functions φ_1 and φ_3. As consequence, LERK3 is more expansive than ERK2c.

A.4.3. *Linearly Exact RK4.* Considering the fourth order Runge-Kutta (A.5) apply on (A.20), we obtain a fourth order accurate LERK time stepping. This scheme will be called LERK4 and is given by:

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_1\right), t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_2\right), t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\mathcal{N}\left(\exp\left(\Delta t L\right)\left(X^n + \Delta t K_3\right), t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{A.25}$$

In this time integrator, seven exponential of matrices are called at each time step. It is the more expansive time integrator considered here. In the case of (A.3) in which $\mathcal{N}(X,t) = b(t)$, LERK4 is less expansive and is simplified into:

$$\begin{cases} K_1 = b(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right) b\left(t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\left(t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 4K_2 + K_4)\right] \end{cases} \tag{A.26}$$

because $K_2 = K_4$. LERK4 is a fourth order scheme.

For non linear equations, LERK4 can be very expansive. To avoid redundant computation, it is possible to compute $E = \exp\left(-\Delta t/2L\right)$ one times for all and consider power of E but we need to save the complete matrix. We do not consider this option in this article.

A.5. **Splitting Exponential Integrators.** A possibility to reduce computational cost is to consider a splitting scheme. By separating linear part and non linear part, we consider two equations to be solve : a linear equation (we done by $\mathcal{S}_{l,\Delta t}$ the solver) which can be solve using an exponential integrator and a non linear equation (denoting the solver $\mathcal{S}_{nl,\Delta t}$) which can be solved using an explicit time scheme. The stability is ensure by the exponential part (exact in time) and the accuracy is ensure by the explicit

scheme. By using this method an error of splitting occurs then the choice of the splitting is crucial (see [19] and reference therein). Two splitting are considered here : Lie and Strang. We use an RK4 scheme for non linear part and exponential integrator for linear part.

A.5.1. *Lie splitting.* The Lie splitting is based on first solve the non linear equation and second consider the previous result as an initial condition for the linear equation. The method can be represent by $\mathcal{S}_{nl,\Delta t} \circ \mathcal{S}_{l,\Delta t}$ The scheme is then :

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^n + \Delta t K_3, t^n + \Delta t\right) \\ X^* = X^n + \dfrac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ X^{n+1} = \exp\left(\Delta t L\right) X^*. \end{cases} \tag{A.27}$$

The non linear equation is solved with a fourth order accurate method and the linear equation is exactly solved. However, due to the splitting, the final scheme is first order accurate and one exponential function is used at each time step. If the source term in (1.8) does not depend of space, the splitting scheme is 4-th order accurate because X^* is in the kernel of L. We will note this time scheme S1ERK4. Subsets can be considered in the RK4 part to solve the non-linear part. Denote subS1ERK4 this variant of the method. The substeps time step $\Delta t_{\text{substep}}$ is choose to be egual to maximum time step allowed for RK4 :

$$\frac{c\Delta t_{\text{substep}}}{\Delta x} \leq K_{\text{RK4}} \tag{A.28}$$

where K_{RK4} is the RK4 stability constraint (see Table 3). Care about Δt has to be a multiple of $\Delta t_{\text{substep}}$.

subS1ERK4 suffer from splitting error as S1ERK4. A more accurate scheme is based on Strang Splitting.

A.5.2. *Strang splitting.* The Strang splitting is based on a split of the linear equation into two linear equation. The method can be represent by $\mathcal{S}_{l,\Delta t/2} \circ \mathcal{S}_{nl,\Delta t} \circ \mathcal{S}_{l,\Delta t/2}$. The scheme is

$$\begin{cases} X^* = \exp\left(\dfrac{\Delta t}{2} L\right) X^n \\ K_1 = \mathcal{N}(X^*, t^n) \\ K_2 = \mathcal{N}\left(X^* + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^* + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^* + \Delta t K_3, t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\dfrac{\Delta t}{2} L\right) \left[X^* + \dfrac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{A.29}$$

We call this scheme S2ERK4. subS2ERK4 is the splitting scheme S2ERK4 with subsetps (and $\Delta t_{\text{substep}}$ egual to maximum time step allowed for RK4). It is the same idea as subS1ERK4. Two exponential matrices are used at each time step. As in the Lie splitting S1ERK4 and subS1ERK4, the time scheme is 4-th order accurate if source function f does not depend in space. It is second order accurate in general.

References

[1] A. Arakawa. Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I. *J. Comput. Phys.*, 1(1):119–143, 1966.

[2] A. Bhatt and B. E. Moore. Exponential integrators preserving local conservation laws of PDEs with time-dependent damping/driving forces. *J. of Comput. and App. Math.*, 352:341–351, 2019.

[3] E. Blayo. Compact finite difference schemes for ocean models: 1. ocean waves. *J. Comput. Phys.*, 164(2):241–257, 2000.

[4] L. Bonaventura. Local Exponential Methods: a domain decomposition approach to exponential time integration of PDEs. *arXiv preprint arXiv:1505.02248*, 2015.

[5] J. P. Boyd. *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

[6] M. Brachet and J.-P. Croisille. Numerical simulation of propagation problems on the sphere with a compact scheme. *HAL*, 2019.

[7] S. Calandrini, K. Pieper, and M. Gunzburger. Exponential Time Differencing for the Tracer Equations Appearing in Primitive Equation Ocean Models. *arXiv preprint arXiv:1910.02189*, 2019.

[8] P. Castillo and Y. Saad. Preconditioning the matrix exponential operator with applications. *J. of Sci. Comput.*, 13(3):275–302, 1998.

[9] C. Clancy and J. A. Pudykiewicz. On the use of exponential time integration methods in atmospheric models. *Tellus A*, 65(1):20898, 2013.

[10] S. Cordier, M. H. Le, and T. M. De Luna. Bedload transport in shallow water models: Why splitting (may) fail, how hyperbolicity (can) help. *Advances in Water Resources*, 34(8):980–989, 2011.

[11] J. Demange, L. Debreu, P. Marchesiello, F. Lemarié, E. Blayo, and C. Eldred. Stability analysis of split-explicit free surface ocean models: implication of the depth-independent barotropic mode approximation. *J. Comput. Phys.*, 398:108875, 2019.

[12] M. E Farquhar, T. J Moroney, Q. Yang, and I. W Turner. GPU accelerated algorithms for computing matrix function vector products with applications to exponential integrators and fractional diffusion. *SIAM J. Scientific Computing*, 38(3):C127–C149, 2016.

[13] M. J Gander and S. Güttel. PARAEXP: A parallel integrator for linear initial-value problems. *SIAM J. on Sci. Comput.*, 35(2):C123–C142, 2013.

[14] F. Garcia, L. Bonaventura, M. Net, and J. Sánchez. Exponential versus IMEX high-order time integrators for thermal convection in rotating spherical shells. *J. of Comput. Phys.*, 264:41–54, 2014.

[15] S. Gaudreault and J. A. Pudykiewicz. An efficient exponential time integration method for the numerical solution of the shallow water equations on the sphere. *J. Comput. Phys.*, 322:827–848, 2016.

[16] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Num. Anal.*, 34(5):1911–1925, 1997.

[17] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[18] M. Hochbruck, A. Ostermann, and J. Schweitzer. Exponential Rosenbrock-type methods. *SIAM Journal on Numerical Analysis*, 47(1):786–803, 2009.

[19] R. Holdahl, H. Holden, and K.-A. Lie. Unconditionally stable splitting methods for the shallow water equations. *BIT Num. Math.*, 39(3):451–472, 1999.

[20] V. M Kamenkovich and D. A. Nechaev. On the time-splitting scheme used in the Princeton Ocean Model. *Journal of Computational Physics*, 228(8):2874–2905, 2009.

[21] S. Kang, F. X. Giraldo, and T. Bui-Thanh. IMEX HDG-DG: a coupled implicit hybridized discontinuous Galerkin (HDG) and explicit discontinuous Galerkin (DG) approach for shallow water systems. *arXiv preprint arXiv:1711.02751*, 2017.

[22] J. D. Lawson. Generalized Runge-Kutta processes for stable systems with large Lipschitz constants. *SIAM J. on Num. Analysis*, 4(3):372–380, 1967.

[23] F. Lemarié, L. Debreu, G. Madec, J. Demange, J.-M. Molines, and M. Honnorat. Stability constraints for oceanic numerical models: implications for the formulation of time and space discretizations. *Ocean Modelling*, 92:124–148, 2015.

[24] S. Liang, J. Zhang, X.-Z. Liu, X.-D. Hu, and W. Yuan. Domain decomposition based exponential time differencing method for fluid dynamics problems with smooth solutions. *Computers & Fluids*, 194:104307, 2019.

[25] G. V. Minchev and W. Wright. A review of exponential integrators for first order semi-linear problems. Technical report, 2005.

[26] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM review*, 20(4):801–836, 1978.

[27] J. Niesen and W. M. Wright. Algorithm 919: A Krylov subspace algorithm for evaluating the ϕ-functions appearing in exponential integrators. *ACM Transac. Math. Soft. (TOMS)*, 38(3):22, 2012.

[28] A. Ostermann and C. Su. A Lawson-type exponential integrator for the Korteweg-de Vries equation. *arXiv preprint arXiv:1807.04570*, 2018.

[29] K. Pieper, K. C. Sockwell, and M. Gunzburger. Exponential time differencing for mimetic multilayer ocean models. *arXiv preprint arXiv:1901.08116*, 2019.

[30] A. Ralston. Runge-Kutta methods with minimum error bounds. *Mathematics of computation*, 16(80):431–437, 1962.

[31] Y. Saad. Analysis of some Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 29(1):209–228, 1992.

[32] Y. Saad. *Iterative methods for sparse linear systems*, volume 82. SIAM, 2003.

[33] M. Schreiber, N. Schaeffer, and R. Loft. Exponential integrators with parallel-in-time rational approximations for the shallow-water equations on the rotating sphere. *Parallel Comput.*, 2019.

[34] A. F. Shchepetkin and J. C. McWilliams. The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean modelling*, 9(4):347–404, 2005.

[35] B. Skaflestad and W. M. Wright. The scaling and modified squaring method for matrix functions related to the exponential. *App. Num. Math.*, 59(3-4):783–799, 2009.

[36] J. Van Den Eshof and M. Hochbruck. Preconditioning Lanczos approximations to the matrix exponential. *SIAM Journal on Scientific Computing*, 27(4):1438–1457, 2006.

[37] Walters, R. A. and Lane, E. M. and Hanert, E. Useful time-stepping methods for the Coriolis term in a shallow water model. *Ocean Modelling*, 28(1-3):66–74, 2009.
[38] D. L. Williamson, J. B. Drake, J. J. Hack, R. Jakob, and P. N. Swarztrauber. A standard test set for numerical approximations to the shallow water equations in spherical geometry. *J. Comput. Phys.*, 102(1):211–224, 1992.
[39] H. Zakerzadeh. Asymptotic analysis of the RS-IMEX scheme for the shallow water equations in one space dimension. *ESAIM: Mathematical Modelling and Numerical Analysis*, 53(3):893–924, 2019.

†Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP∗, LJK, 38000 Grenoble, France
E-mail address: [1]matthieu.brachet@inria.fr, [2]laurent.debreu@inria.fr, [3]christopher.eldred@inria.fr,