Comparaison of Exponential integrators and traditional time integration schemes for the Shallow Water equations

Matthieu Brachet, Laurent Debreu, Christopher Eldred

COMPARAISON OF EXPONENTIAL INTEGRATORS AND TRADITIONAL TIME INTEGRATION SCHEMES 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 choices 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 for the modeling of a shallow 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 analyze different time stepping algorithms for the linearized shallow water equations. High order explicit schemes are accurate but the time step is constraint by the Courant-Friedrichs-Lewy stability condition. Implicit schemes can be unconditionally stable but, in practice lacks of accuracy when used with large time steps. In this article we propose a detailed comparison of such classical schemes with exponential integrators. The accuracy and the computational costs are analyzed in different configurations.

1. Introduction

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

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

for all $x \in [0,d]$, $t \geq 0$ and the initial data are $(u_0(x), h_0(x))$. Equations (1.1) are closed with appropriate boundary conditions, which we consider here to be periodic. The function $f : (t,x) \in \mathbb{R}^+ \times [0,d] \mapsto f(t,x) \in \mathbb{R}$ represents terms other than advection or gravity (e.g. bottom friction or wind). g is the gravity.

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 the depth average of the 3D right hand side.
- *Bedload transport*: to model the sediment transport, shallow water equations are coupled with Exner equation. In this system, the topography is moving and the bedload velocity is smaller than fluid velocity [9].
- *Atmosphere model*: shallow water system on a rotated sphere is the simplest atmosphere model of interest. It is considered as a first step in the study of a numerical solver for geophysical fluids [34].

The desired degree of accuracy of the numerical solution of (1.1) is dependent on the application and impacts the choice of temporal and spatial discretization schemes. One may prefer a numerical solver with a low computational cost even if the accuracy is poor. This is the case for example for large scale (e.g. climate) oceanic applications. Operational (and thus time to solution) constraints can also flavor the use of such schemes. For other applications, like bedload transport or tidal model, the accuracy is essential and high order schemes are necessary.

In this paper, we study various time integration schemes for the solution of the shallow water equations. Explicit schemes, like the Runge-Kutta and the Forward-Backward schemes discussed later, have a low computation cost per time step since they rely on a few evaluations of the right hand side. However, they

Date: April 8, 2020.

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

are limited by a Courant-Friedrichs-Lewy (CFL) stability condition (see [20] for a review on the different stability conditions of an ocean model):

$$\frac{c_g \Delta t}{\Delta x} \leq \mathrm{C}^{\mathrm{ste}}, \tag{1.2}$$

where Δt and Δx are the temporal and spatial discretization steps. $\frac{c_g \Delta t}{\Delta x}$ is the Courant number with c_g is a characteristic velocity which corresponds in the shallow water model to $c_g = \sqrt{gH}$ where H is the mean total depth [18, 31]. In the context of ocean models, Forward-Backward schemes are considered in [31] and Runge-Kutta integrators in [10, 33] among others.

An alternative is to use implicit schemes, the simplest one being the Euler Backward scheme. This family 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, large time steps, more precisely the use of large Courant numbers, can reduce the accuracy which translates into dispersion and possibly dissipation errors.

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

$$\frac{dX}{dt} = \mathcal{F}(X, t). \tag{1.3}$$

Function $\mathcal{F}$ is split into linear and non linear part : $\mathcal{F}(X,t) = LX + \mathcal{N}(X,t)$. Linear part L can be fixed once for all or updated using the Jacobian at the current time $L = \mathrm{Jac}_{t^n}\mathcal{F}$. The latter is the Rosenbrock configuration.

A review of these schemes is available in [15] in a general framework. The main property of this class of integrators is to solve exactly linear equations, i.e. when $\mathcal{N} = 0$ in (A.1). A consequence is the A-stability: schemes are not constrained by a CFL condition. Dissipation and dispersion properties depend only on the spatial discretization. The main issue is the need for the computation of matrix exponential. Advances in the application of Krylov methods [28] for performing this computation has bring more interest to these methods.

- The most common class of methods are the Exponential Runge-Kutta integrators (ERK) relying on the formula

$$X(t) = \exp(tL)\, X(0) + \int_0^t \exp((t-\tau)L)\, \mathcal{N}(X(\tau))\, d\tau, \tag{1.4}$$

  to solve autonomous PDEs (i.e. when the right hand side $\mathcal{N}$ does not depend explicitly on time). The order of the ERK integrators depends on the quadrature rule to compute the right hand side of (1.4) (see A.3). They have been used to solve shallow water equations on a rotating sphere in the context of atmosphere modeling in [6, 8, 13]. Methods are as accurate as explicit Runge-Kutta methods but allow the use of much larger time steps. In all these three articles, the accuracy is analyzed on the classical set of test cases [34] for which there are no time dependent analytical solution available. Although the results are encouraging, they do not allow to measure accuracy and analyze errors.

  In [12], ERK integrators are compared to implicit time schemes on a three-dimensional Boussinesq thermal convection equations. These methods are found more accurate than implicit but also more expansive.

  These integrators have also been considered in [7] to solve tracer equations in an ocean model. Two methods are compared to compute the exponential part. The first is based on squalling and squaring with a Padé approximation. It is the faster method but matrices exponential have to be stored. The second way is to compute exponential matrices at each time step using Krylov methods. In that case, the exponentials do not have to be stored and can be updated when necessary at the price of a larger computational cost per time step.

  In [26], it is suggested to consider exponential integrators to solve multi-layer shallow water equations. The space discretization is done with mimetic elements leading to skew-symmetric matrices and exponential functions are computed thanks skew-Lanczos methods. To obtain a computational cost smaller than a fourth order Runge Kutta scheme (RK4) it seems however necessary to consider dramatically larger time steps Δt and to accept a larger error than RK4.

  ERK integrators give a way to solve autonomous equation, we denote ERKc (see A.3 and [16])

the ERK integrators corrected for the solution of non-autonomous systems. Thanks to this, the order of accuracy of ERK is kept for non autonomous PDEs.

- To avoid quadrature in the integral part of (1.4), we can proceed to the following change of variable:

$$V(t) = \exp((t^n - t)L)X(t) \tag{1.5}$$

in (1.3) and use an explicit scheme on the V equation given by

$$V'(t) = \exp((t^n - t)L)\mathcal{N}\left(\exp((t - t^n)L)V(t), t\right). \tag{1.6}$$

By this method, we avoid quadrature methods for the integral. These exponential integrators are called Linearly Exact Runge-Kutta integrators (LERK). They are used in [25] to solve the Korteweg-de Vries equation. The results show that the expected order of accuracy is attained but the computational cost is not considered.
Energy conservation properties are studied in [2] on a set of PDEs with damping/driving forces. LERK schemes are shown to be still accurate on non-linear PDEs such that the Korteweg-de Vries equation. The error is numerically investigated.
It is easy to build a high order accurate LERK integrator by considering a Runge-Kutta method on (1.6). Unfortunately, the computation of a lot of matrix exponential are then required and LERK integrators are expected to be expensive.

- To reduce the computational cost, it is possible to consider a splitting between linear and non linear parts. As an example, the Lie's splitting consists of two steps. First, compute X^* the solution of the linear equation at time $t^n + \Delta t$ with X^n as initial condition:

$$\begin{cases} \dfrac{dX}{dt} = LX \\ X(0) = X^n, \end{cases} \tag{1.7}$$

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

$$\begin{cases} \dfrac{dX}{dt} = \mathcal{N}(X, t) \\ X(0) = X^*, \end{cases} \tag{1.8}$$

thanks to an explicit scheme like the RK4 scheme. The solver for the non linear part is denoted $\mathcal{S}_{\mathrm{nl},\Delta t}$ and $X^{n+1} = \mathcal{S}_{\mathrm{nl},\Delta t}X^* = \mathcal{S}_{\mathrm{nl},\Delta t} \circ \mathcal{S}_{\mathrm{l},\Delta t}X^n$. Only one exponential is required for the computation of $\mathcal{S}_{\mathrm{l},\Delta t}$. Each equation is solved with a highly accurate scheme. The Lie's splitting will be denoted S1ERK4. To increase accuracy on the non-linear equation, especially when the non-linear term we consider N_s sub-steps on $\mathcal{S}_{\mathrm{nl},\Delta t}$. The Lie's splitting become $\left(\mathcal{S}_{\mathrm{nl},\Delta t/N_s}\right)^{N_s} \circ \mathcal{S}_{\mathrm{l},\Delta t}$ (subS1ERK4). Due to splitting errors, the resulting schemes is only first order accurate.
More accurate methods are based on the Strang's splitting given by relations $\mathcal{S}_{\mathrm{l},\Delta t/2} \circ \mathcal{S}_{\mathrm{nl},\Delta t} \circ \mathcal{S}_{\mathrm{l},\Delta t/2}$ (denoted by S2ERK4) and $\mathcal{S}_{\mathrm{l},\Delta t/2} \circ \left(\mathcal{S}_{\mathrm{nl},\Delta t/N_s}\right)^{N_s} \circ \mathcal{S}_{\mathrm{l},\Delta t/2}$ (denoted by subS2ERK4 for the substeps version). These schemes are second order accurate and require the computations of two matrices exponential.
Integrators relying on splitting methods have been used in [17] to capture shocks appearing in shallow water equations, however exponential methods were not used.
This class of exponential integrators is thus expected to be cheaper than other high order unsplit exponential integrators but splitting errors should reduce the accuracy.

Properties of the exponential integrators considered are summarized in Table 1. The first value is the order accuracy and the second one is the number of required exponential computations.
Two cases are distinguish: the left column corresponds to a non linear part $\mathcal{N}(X, t)$ depending on X. ERKc methods benefit from an updated linear part thanks the Jacobian and gain one order of accuracy without increasing the computational cost. LERK integrators are high order accurate but suffer from the number of exponential matrices occuring (5 exponentials for LERK3 while it is 7 for LERK4). LERK3 is more expansive than ERK2c while the order of accuracy is the same in Rosenbrock case. Splitting methods have a computational cost corresponding to their accuracy: n-th order methods have n exponential to compute per time step. Splitting errors are the limiting factor on accuracy.
The right column is devoted to equation in which the non linear term is $\mathcal{N}(X, t) = \mathcal{N}(t)$. In this

configuration, ERKc automatically benefit from the advantages of Rosenbrock case and are more accurate with the same number of exponential to compute.
LERK methods are less expansive than in the first column due to simplifications occurring in algorithm. Furthermore, if $\mathcal{N}(t)$ is in the kernel of L, some exponential are computed easily and it reduces significantly the computational cost. It does not impact the accuracy. Splitting integrators gain in accuracy when $\mathcal{N}(t) \in \mathrm{Ker}(L)$ because splitting errors are zero but the number of exponential is the same than other configurations.

	$\mathbf{X}'(\mathbf{t}) = \mathcal{F}(\mathbf{X}, \mathbf{t})$		$\mathbf{X}'(\mathbf{t}) = \mathbf{L}\mathbf{X} + \mathcal{N}(\mathbf{t})$	
	fixed linear part L	Rosenbrock $L = \mathrm{Jac}_n\mathcal{F}$	$\mathcal{N}(t) \notin \mathrm{Ker}(L)$	$\mathcal{N}(t) \in \mathrm{Ker}(L)$
ERK1c	1/1	2/1	2/1	2/1
ERK2c	2/2	3/2	3/2	3/2
LERK1	1/1	1/1	1/1	1/1
LERK3	3/5	3/5	3/3	3/1
LERK4	4/7	4/7	4/3	4/1
S1ERK4	1/1	1/1	1/1	4/1
subS1ERK4	1/1	1/1	1/1	4/1
S2ERK4	2/2	2/2	2/2	4/2
subS2ERK4	2/2	2/2	2/2	4/2

TABLE 1. Properties of exponential integrators: order of accuracy/number of exponential to compute. ERK are a specific case of ERKc for autonomous PDEs. The computation cost is the same but it is necessary to compute time derivative on the right hand side.

In the following, we will consider linearized equations around a steady state $(\bar{h}, \bar{u} = 0)$ where $\bar{h}$ is a positive constant height of fluid. The new system, called Linearized Shallow Water equations (LSWE), is given by

$$\begin{cases} \dfrac{\partial h'}{\partial t} + \bar{h}\dfrac{\partial u'}{\partial x} = 0 \\ \dfrac{\partial u'}{\partial t} + g\dfrac{\partial h'}{\partial x} = f(t,x) \end{cases} \tag{1.9}$$

wherein (h', u') is a small perturbation of the resting steady state $(\bar{h}, 0)$. The characteristic velocity of (1.9) is $c = \sqrt{g\bar{h}}$. In the following, we will drop the $'$ to simplify notations.

The purpose of this paper is to compare various time schemes for linearized shallow water equations. This includes Runge-Kutta explicit schemes, implicit $\theta-$ schemes and exponential integrators. The stability constraints 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 in (1.9) affects the time integration schemes. In addition since the global behavior of the scheme is also dependent on the spatial discretization, we study second and fourth order spatial schemes, both of them on a staggered grid.

This paper is organized as follows. In section 2, we present the spatial discretizations on a staggered grid and recall their properties, in particular in terms of phase errors. The section 3 is devoted to the Krylov subspace method used to compute matrix functions. The different time integration schemes are then considered. Beside stability and accuracy issues, we analyze, in section 4, spatio-temporal dissipation and dispersion properties according to the spatial discretizations. The numerical schemes are implemented and tested on different cases, according to the specification of the forcing term $f(t,x)$, in section 5.

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

Equations (1.9) 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]).
They 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 will have implications on stability and thus on the computational cost of the spatio-temporal schemes.

Two staggered grids are considered for the space discretization of (1.9): one for h and another one for u. The h-grid is an offset of the u-grid. We define two sets of points: $(x_i)_{0\leq i\leq N-1} \in \mathbb{R}^N$ and $(x_{i+1/2})_{0\leq i\leq N-1} \in \mathbb{R}^N$ 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 Figure 1). Centered schemes discretized on this staggered grid are known to have a smaller phase error than their equivalent on 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, vectors $\mathfrak{u} \in \mathbb{R}^N$ and $\mathfrak{h} \in \mathbb{R}^N$ are functions of time.

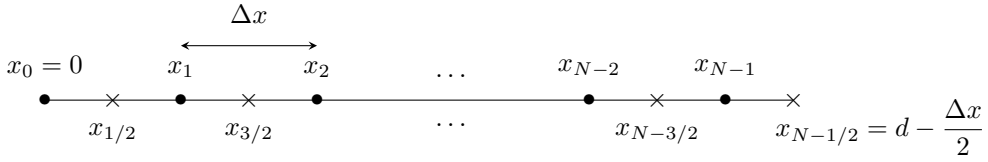

FIGURE 1. Staggered grid representing $(x_i)_{0\leq i\leq N-1} \in \mathbb{R}^N$ and $(x_{i+1/2})_{0\leq i\leq N-1} \in \mathbb{R}^N$.

The expressions of the second and fourth order finite difference (FD) operators are given below:

- *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 for the C2 and C4 schemes are given in Table 2 and plotted on Figure 2. As expected, higher order space operator C4 leads 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 2. Numerical phase error for centered second and fourth order approximations on staggered grid. The normalized wave number is $\theta = k\Delta x$.

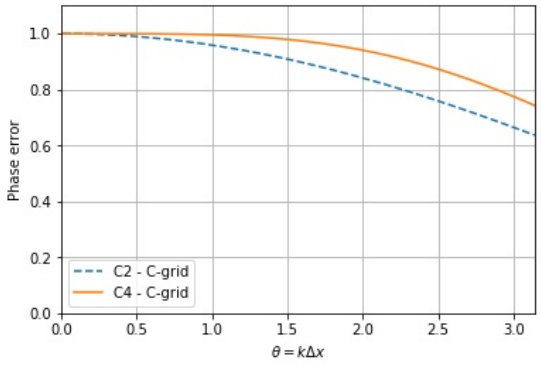

FIGURE 2. Numerical phase error for centered second and fourth order approximations on staggered grid. The normalized wave number is $\theta = k\Delta x$.

3. 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 A.3). We recall some φ_k used in our numerical schemes:

$$\begin{cases} \varphi_0(z) = \exp(z) \\ \varphi_1(z) = \dfrac{\exp(z)-1}{z} \\ \varphi_2(z) = \dfrac{\exp(z)-1-z}{z^2}, \end{cases} \tag{3.1}$$

functions are extended in 0 by $1/k!$.
The problem is solved thanks to Krylov methods [14, 24, 29]. It is based on the projection of the matrix A onto a Krylov subspace using Arnoldi's algorithm and compute $\varphi_k(A)b$ in this subspace.

3.1. Arnoldi's method for Krylov subspaces.

Krylov subspaces $\mathcal{K}_m(A,b)$ are defined by

$$\mathcal{K}_m(A,b) = \text{Span}(b, Ab, A^2b, \cdots, A^{m-1}b). \tag{3.2}$$

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 $\mathcal{K}_m(A,b)$. n is the size of A. H_m and V_m are built using 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

Matrices H_m and V_m satisfy the following equalitiy [29]:

$$V_m^T A V_m = H_m. \tag{3.3}$$

3.2. Computing matrix function product.

As mentioned in [26], exponential integrators imply to be able to compute efficiently product $\varphi_k(A)b$ with A a matrix and b a vector.

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

$$\varphi_k(A)b \approx x_m = \|b\|_2 V_m \varphi_k(H_m) e_1. \tag{3.4}$$

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**

Line 2 in 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 a Padé approximant. The error between the Padé approximant and exact value of $\varphi_k(H_m)$ is linked to the norm of H_m, and thus scaling and squaring methods are used (see [23, 32]).

A stopping criterion is given in the exponential case (i.e. $k = 0$) in [28]. It is adapted and used for a general φ_k in [24]. Following ideas given by these articles, we remind the error formula

$$s_m^k = \|b\|_2 h_{m+1,m} \left(\sum_{i=k+1}^{\infty} A^{i-k+1} v_{m+1} e_m^T \varphi_i(H_m) \right) e_1 \tag{3.5}$$

where s_m^k is the difference between exact formula and Krylov approximate:

$$s_m^k = \varphi_k(A)b - x_m. \tag{3.6}$$

The norm of the first term of this series

$$\|b\|_2 \cdot |h_{m+1,m}| \| v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 \|_2 \tag{3.7}$$

provides a good error estimate which we will use as a stopping criterion in our experiments. The tolerance is fixed to 10^{-12} in our experiment. On 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 [28], author suggests to consider more terms of the series for the stopping criterion but it seems to be not necessary in our numerical experiments.

To assess the relevancy of the stopping criterion, we plot the relative error and the value of the stopping criterion

$$\|b\|_2 \cdot \|h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1\|_2 \tag{3.8}$$

at each step of a Krylov iteration which computes $\varphi_1(\Delta t L)b$ where L is defined by

$$LX = \begin{bmatrix} -\bar{h}\delta_x \mathfrak{u} \\ -g\delta_x \mathfrak{h} \end{bmatrix}, \tag{3.9}$$

$X = [\mathfrak{h}, \mathfrak{u}]^T \in \mathbb{R}^{2N}$. The vector $b \in \mathbb{R}^{2N}$ is randomized to contain all frequencies. Results are given on Figures 3 and 4. Behaviors were similar for other functions φ_k tested.
As expected, the value of the stopping criterion is close to the relative error whatever the Courant number chosen. It is generally slightly larger. As can be seen of Figures 3 and 4, the number of iterations increases with the Courant number. This was expected due to the Theorem 4.3 in [28]. Larger Courant number increases the spectral radius and the *a priori* error bound. C4 operator is slower to converge than C2 for the same reason. Furthermore, there is a threshold after which the error drops exponentially. That is why the change of the tolerance does not impact significantly the computational cost.

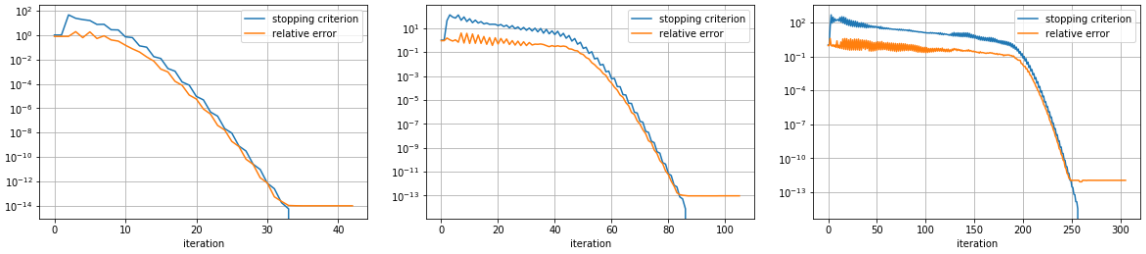

FIGURE 3. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There is $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel),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

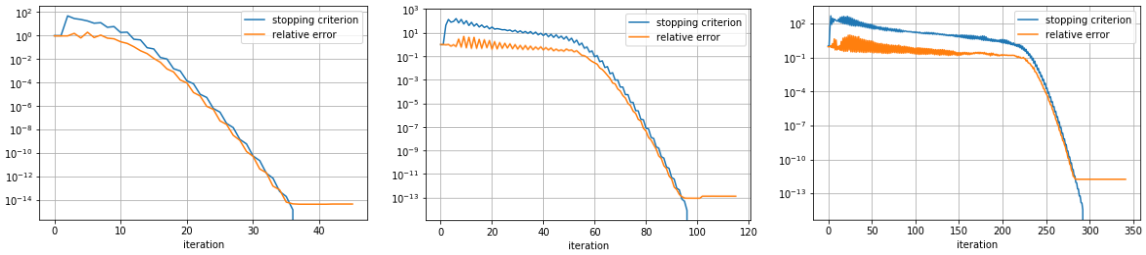

FIGURE 4. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There is $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel),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

4. PROPERTIES OF NUMERICAL SCHEMES

We here consider various time schemes and space discretization operators to solve LSWE (1.9). The details of the time schemes are given in Appendix A. Notations used are given in Table 3.

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

TABLE 3. Space and time numerical schemes used in this paper.

The stability condition is one of the main property of a numerical scheme. Consider a one step time scheme to solve (1.9). There exists 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, for the some of schemes presented in Appendix A, matrix 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 (3.9).
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 4, we summarize the maximum Courant numbers $c\Delta t/\Delta x$ allowed to maintain stability using different explicit schemes. The RK4 scheme has the larger stability limit while the RK3 scheme puts the stronger stability constraint.
Whatever the time scheme considered, the second order discretization allows for a time step larger than the one for the fourht order discretization, the ratio between the two is about 1/0.86.

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 4. Stability criterion for centered spatial discretization of second and fourth 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.

The stability constraint has to be associated to the computational cost, mainly given by the number of right hand side evaluations of (1.9). In Table 5, we compute the maximum Courant number divided by the number of right hand side evaluations. Greater the value is, cheaper the method is. Forward-Backward scheme is the cheapest among the explicit time schemes but also the less accurate. The Runge Kutta integrator RK4 is more accurate and less expensive than RK3.

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

The spatio-temporal dissipation and dispersion errors 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). Implicit time schemes and exponential integrators are unconditionally stable and thus allows for Courant number much larger.
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 it is generally preferable to use θ scheme with a value of θ slightly larger than 0.5. We use $\theta = 0.51$ here. Conversely, exponential integrators do not dissipate. FB does not dissipate (when stable). Note that the FB / C2 scheme is exact (no dissipation and no dispersion errors) when the Courant number

is equal to 1, the largest value of allowed Courant number. This error cancellation does not occur for the FB/C4 scheme. Exponential integrators dissipation and dispersion properties do not depend on the Courant number.

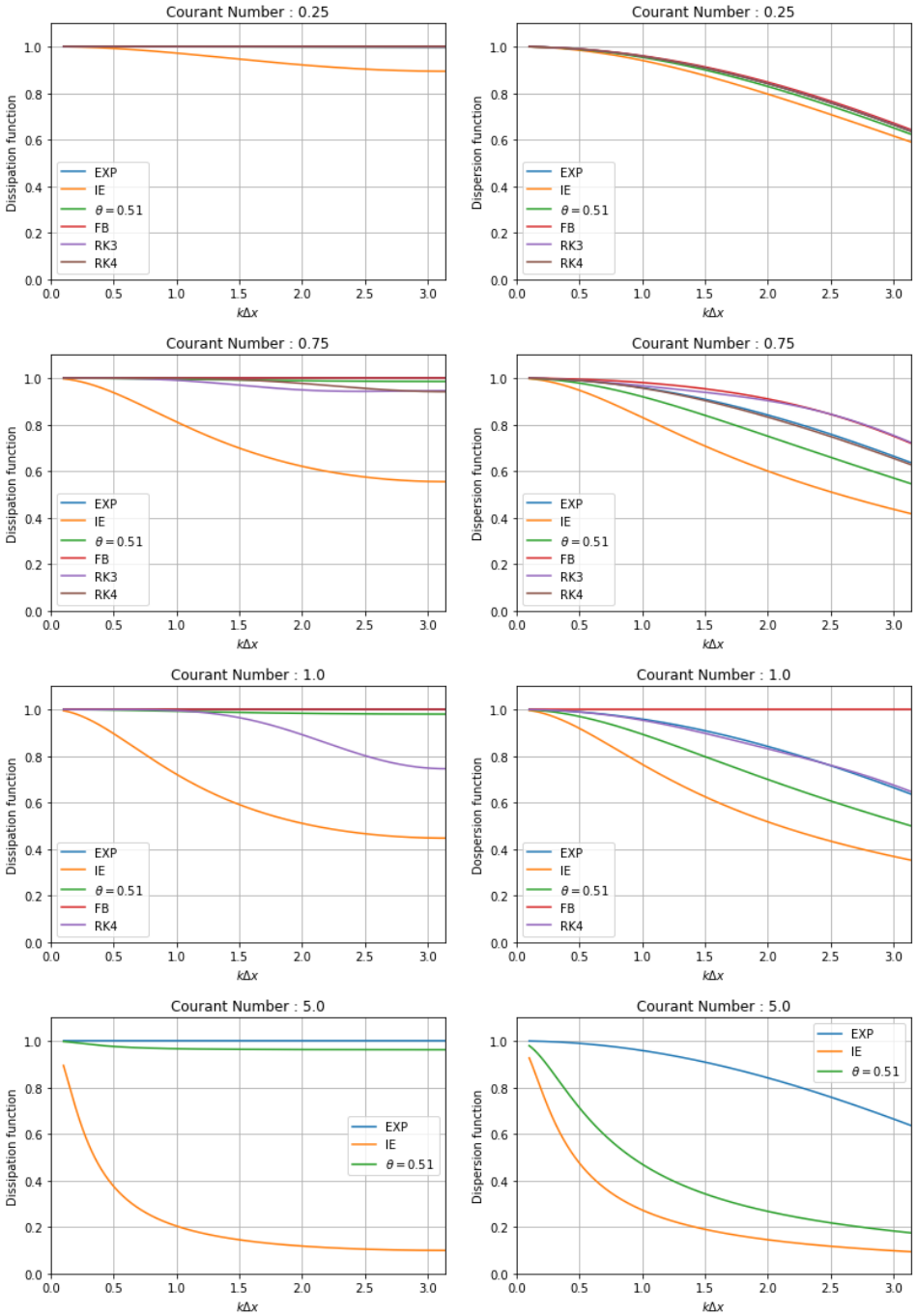

FIGURE 5. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C2 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered: 0.25, 0.75, 1 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

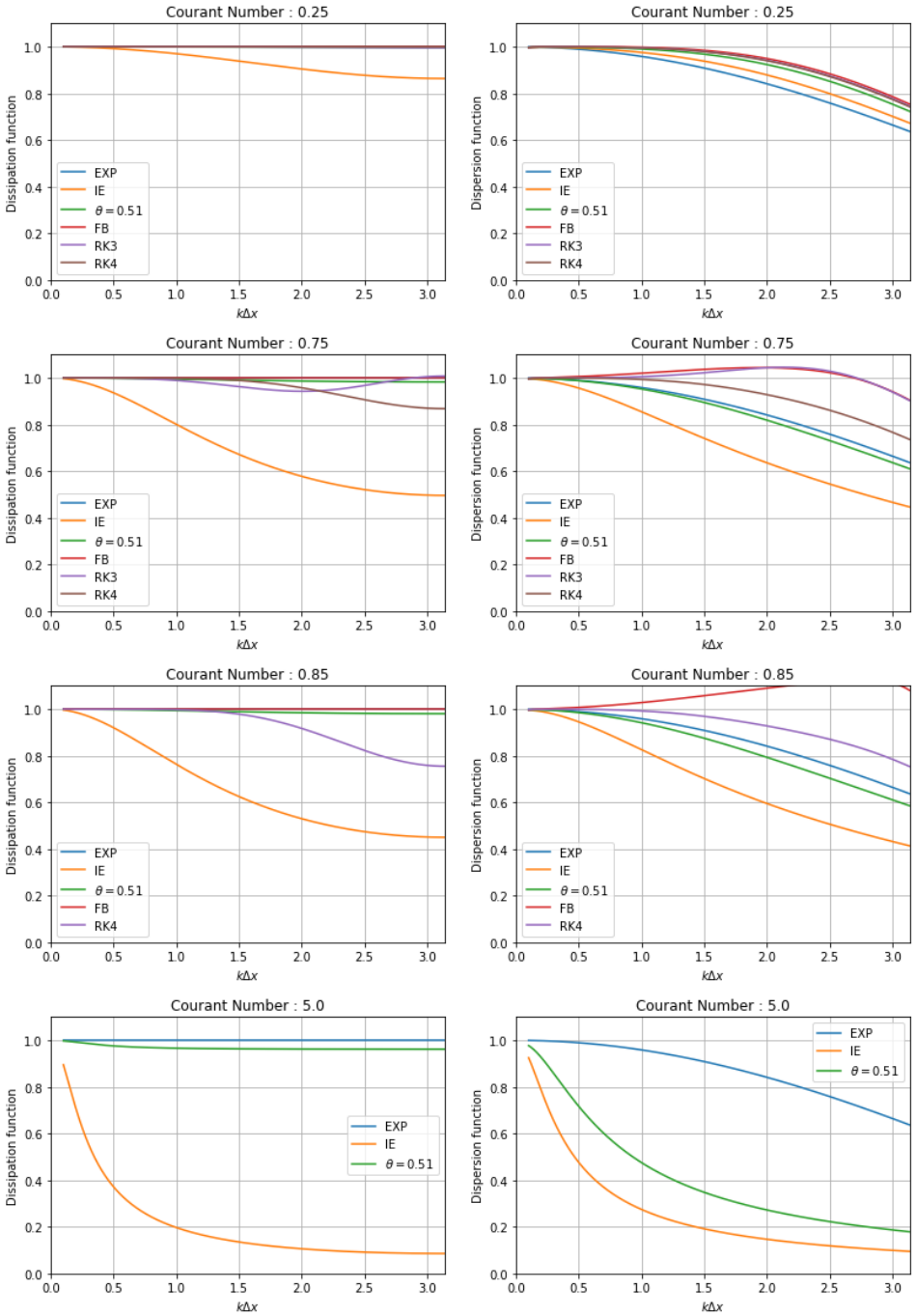

FIGURE 6. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C4 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered: 0.25, 0.75, 0.85 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

5. NUMERICAL RESULTS

To assess the performances of each scheme, we consider several test cases for the linearized shallow water equations (1.9). Both the accuracy and stability are evaluated. As mentioned before, exponential integrators and implicit time schemes are A-stable allowing to use of large time steps. Implicit methods require the solution of a linear system which is here achieved thanks to a conjugate gradient method.

Consider (1.9) initialized with $(h_0(x), u_0(x))$. The analytic solution is well known whatever the function f. The height h is given by the formula

$$h(t,x) = \frac{1}{2}\sqrt{\frac{\bar{h}}{g}}\left[u_0(x-ct) - u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) + h_0(x+ct))\right] + ... \\ ... + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) - f(\tau, x+c(t-\tau))\right) d\tau \tag{5.1}$$

and the velocity u is

$$u(t,x) = \frac{1}{2}\left[u_0(x-ct) + u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) - h_0(x+ct))\right] + ... \\ ... + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) + f(\tau, x+c(t-\tau))\right) d\tau \tag{5.2}$$

with the characteristic velocity $c = \sqrt{g\bar{h}}$.
Three source functions 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 Δt is crucial to maintain an accurate numerical solution. The source function $\mathcal{N}(t)$ is in the kernel of L then some integrators gain. LERK integrators have a reduce computational cost because only one exponential has to be compute at each time step. Splitting methods benefit from a splitting error equal to zero and become fourth order accurate.
In the third test case, f depends both on time and space. This test case give us, among other, an idea of the splitting error which is not present in the second test case. It is also used to evaluate the link between accuracy and computational cost in a more general case. Additionally, LERK integrators are more expansive because $\mathcal{N}(t) \notin \mathrm{Ker}(L)$ if f depends on space (see Appendix A.3).

In all test cases, we consider a domain length $d = 500000$ meters and the gravity parameter $g = 9.81\mathrm{m}\cdot\mathrm{s}^{-2}$. Initialization (h_0, u_0) are given by the formulae

$$\begin{cases} h_0(x) = h_m \cdot \exp\left(-\left(\dfrac{x - 0.5\cdot d}{\sigma}\right)^2\right) \\ u_0(x) = 0 \end{cases} \tag{5.3}$$

where $\sigma = d/10$ and $h_m = 1\mathrm{m}$ is the maximum of h_0.
Functions h_0 and u_0 are chosen such that the numerical solution does not correspond to an eigenvalue of L. Indeed, if X^n is an eigenvalue of L, then the Krylov method to compute $\varphi_k(\Delta t L)X^n$ converges into a single iteration.
The sources terms f will be specified in the following.
Two cases are considered: shallow ocean ($\bar{h} = 100\mathrm{m}$) and deep ocean ($\bar{h} = 4000\mathrm{m}$). The change of deep impacts the value of the propagation speed c and thus of the Courant number. The ratio between the two thicknesses is $4000/100 = 40$, it results into a ratio of the Courant numbers $\sqrt{4000}/\sqrt{100} \approx 6.32$.
Relative errors are computed at time t^n on h using

$$e_h = \sqrt{\frac{d\cdot T}{N\cdot N_t}\cdot\frac{\displaystyle\sum_{n=1}^{N_t}\|\mathfrak{h}^n - h(t^n)\|^2_{L^2([0,d])}}{\|h\|^2_{L^2([0,T_f]\times[0,d])}}} \tag{5.4}$$

and on u by

$$e_u = \sqrt{\frac{d \cdot T}{N \cdot N_t} \cdot \frac{\sum_{n=1}^{N_t} \|\mathfrak{u}^n - u(t^n)\|^2_{L^2([0,d])}}{\|u\|^2_{L^2([0,T_f]\times[0,d])}}} \tag{5.5}$$

where $\mathfrak{h}$ (resp. $\mathfrak{u}$) is the numerical approximate of h (resp. u). N is the number of grid points and N_t is the number of time iteration to reach final time $T = N_t \Delta t$.

5.1. **Linear Case.** Consider (1.9) without source term. More precisely, f is

$$f(t,x) = 0 \text{ for all } x \in [0,d],\ t \geq 0. \tag{5.6}$$

Then, (1.9) is a linear autonomous equation. There are only space discretisation error when exponential integrators are used.
At each time $t \geq 0$, the solution is

$$\begin{cases} h(t,x) = \dfrac{1}{2}\left(h_0(x-ct) + h_0(x+ct)\right) \\ u(t,x) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left(u_0(x-ct) - u_0(x+ct)\right) \end{cases} \tag{5.7}$$

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

Visualisation of dissipation/dispersion errors of the unconditionally stable schemes. On Figure 7, we plot the numerical solution h at times $t = 1$ hour, $t = 10$ hours and $t = 40$ hours. The ocean is shallow ($\bar{h} = 100$ meters). Space operator is fourth order accurate and there are $N = 500$ grid points leading to $\Delta x = 1000$m. The time step is $\Delta t = 300$s and corresponds to a Courant number $c\Delta t/\Delta x = 9.39$. In this high resolution case, the solution is spatially well resolved and most of the errors comes from the time stepping algorithm.

We observe the dissipation of the solution computed with Backward Euler and the dispersion using Crank-Nicholson. The curves associated to θ-scheme ($\theta = 0.51$) show less dispersion error than the Crank-Nicholson scheme at the price of a slightly larger dissipation. As expected, exponential integrators are accurate, the numerical solution is visually not distinguishable from the exact solution. Similar conclusions are obtained using a second order accurate scheme (C2).

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 right hand side evaluations. For exponential integrators and implicit schemes, these calls arise within the matrix-vector products of the Krylov and conjugate gradient methods.
For each explicit schemes, only one time step size is considered. It corresponds to a value 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 6.

	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 6. Time steps Δt considered for explicit time integrators FB, RK3 and RK4. 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.

For the implicit and exponential integrators, which are unconditionally stable, we consider different values given in Table 7, along with the corresponding Courant numbers.
Results are plotted on Figure 8 using second order space discretization C2, and on Figure 9 for the fourth order accurate operator C4. What is plotted are the relative errors as a function of the number of right hand side evaluations.

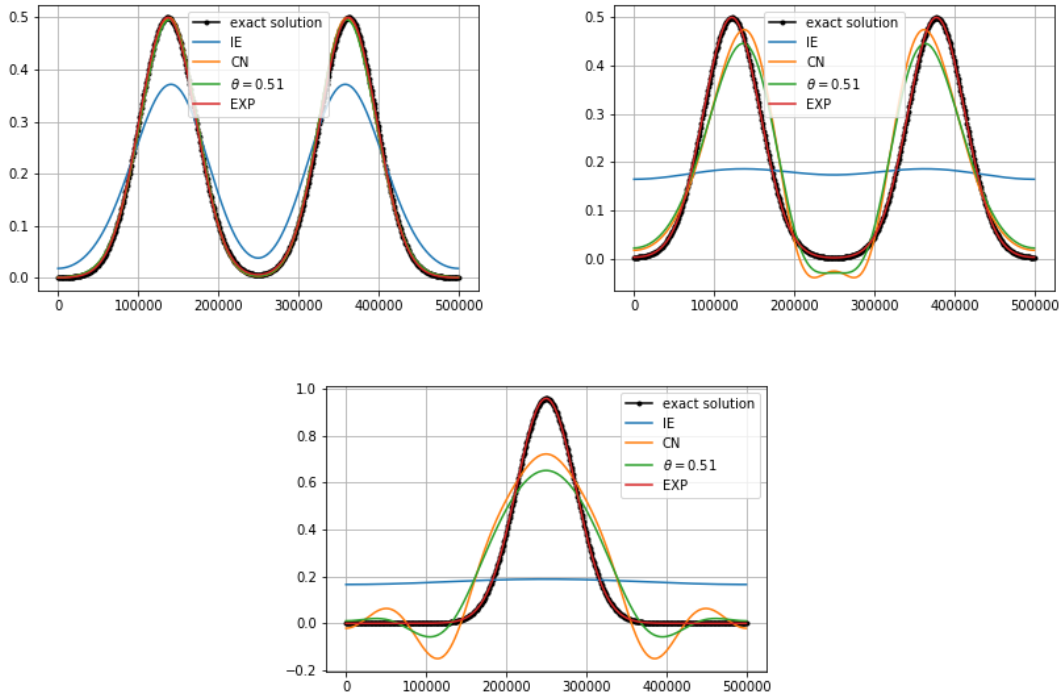

FIGURE 7. Linear test case. Numerical solutions $\mathfrak{h}$ computed using C4 space operators with $N = 500$ grid points and $\bar{h} = 100$ meters. We consider $\Delta t = 300$ secs ($c\Delta t/\Delta x = 9.39$). Top left panel: solution at $t = 1$hour. Top right panel: solution at $t = 10$hour.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

Δt	100	600	1200	3600	5400
Shallow ocean	3.05	18.31	36.61	109.84	164.76
Deep ocean	19.30	115.79	231.57	697.71	1038.15

TABLE 7. Time steps Δt and associated 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).

As mentioned before, the explicit Forward Backward scheme takes advantage of very good dissipation and dispersion properties when Courant number is close to 1 using C2 (see Figure 5). This is not true using C4 operators for which only limited accuracy is achieved. But its cost, in comparison with RK3 and RK4 schemes, is much lower. The Forward Backward scheme is thus a good alternative, when compared to any other schemes (either explicit or implicit) when the accuracy is not the factor limiting the choice of the time step.

As expected, the implicit schemes (Implicit Euler and θ-scheme) lead to low accuracy, except when the Courant number is moderated (e.g. for the shallow ocean and for a time step of $\Delta t = 100$). They can only be used when accuracy is not a required property.

The exponential integrator (here the ERK1 scheme) is the most accurate scheme since they are no time integration errors in this linear case. It is also the cheapest one for the deep ocean case where the stability condition of explicit schemes is more restrictive.

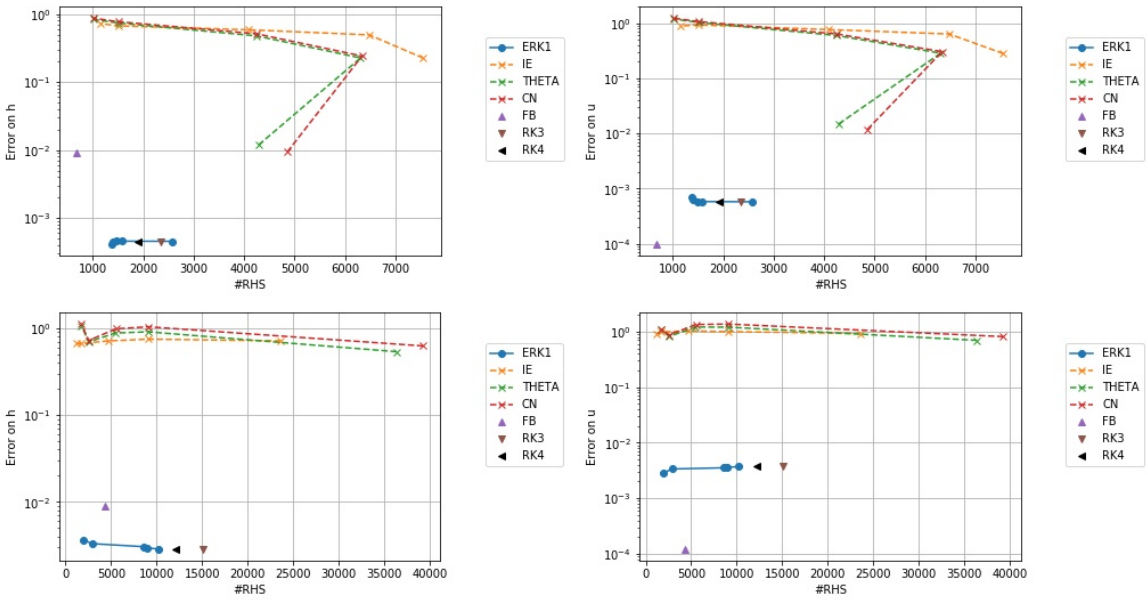

FIGURE 8. Linear test case. Relative error as a function of the number of right hand side (RHS) evaluations (which is itself given by the choice of the time step). The final time is $T = 6$ hours. On the left panel, we consider the shallow ocean $\bar{h} = 100$ meters; on the right panel, the deep ocean with $\bar{h} = 4000$ meters. The space scheme is C2 with $N = 500$ grid points.

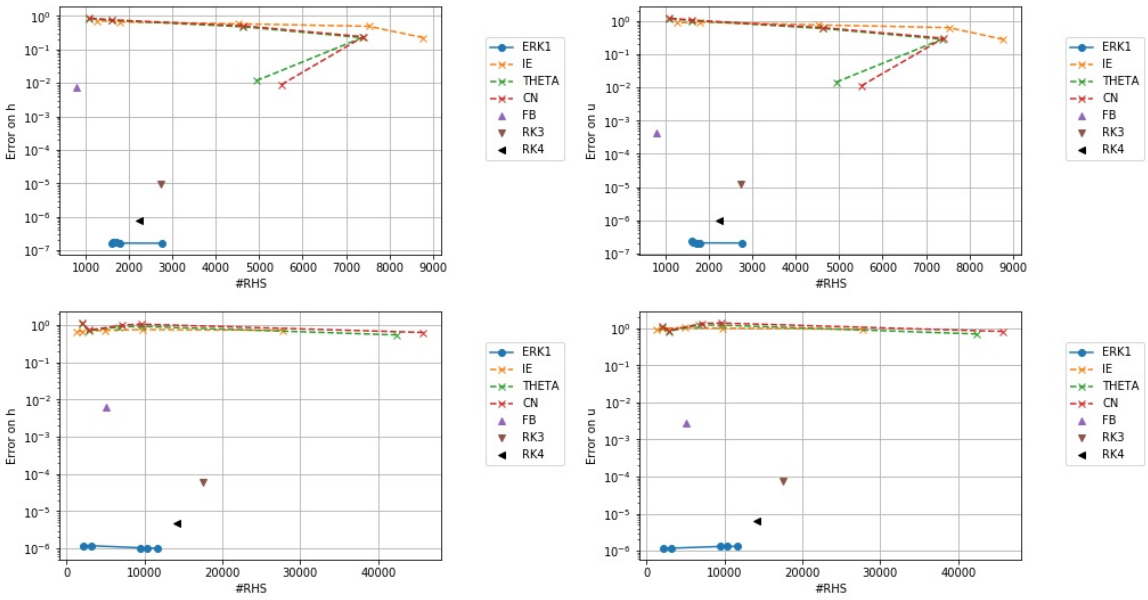

FIGURE 9. Linear test case. Relative error as a function of the number of right hand side (RHS) evaluations (which is itself given by the choice of the time step). The final time is $T = 6$ hours. On the left panel, we consider the shallow ocean $\bar{h} = 100$ meters; on the right panel, the deep ocean with $\bar{h} = 4000$ meters. The space scheme is C4 with $N = 500$ grid points.

Computational cost. To compare its computational cost, the numbers of RHS evaluations using ERK1 with different ocean thickness and operators are given in Table 8.
ERK1 is 1.07 times less costly using C2 instead of C4 in the shallow ocean case and 1.14 times in the deep ocean case. The number of RHS is is assumed to be proportional to the computational cost. With

	Shallow ocan	**Deep ocean**
C2	2746	11649
C4	2584	10204

TABLE 8. Linear test case. Number of calls to reach $t = 6$ hours with ERK1. The numerical parameters avec $N = 500$ grid points and $\Delta t = 100$seconds.

the same change (C2 instead of C4), keeping the same numerical and physical configuration, explicit time schemes are 1.16 less costly due to change in CFL conditions. Then, ERK1 is as impacted as explicit time schemes.
Deep ocean instead of shallow ocean leads a Courant number $\sqrt{4000/100} \approx 6.32$ times greater. Explicit schemes have thus to reduce their time step by a corresponding factor, then they are 6.32 more expansive considering the number of RHS to reach final time. ERK1/C2 in the deep ocean case has 4.24 times more RHS than for shallow ocean. The ratio is 3.93 for ERK1/C4. Thus the increase of the fluid thickness, and thus of the Courant number, has less impact on the ERK1 scheme than explicit time schemes.
As attempt, increase the Courant number increase the computational cost of each exponential. Use C4 instead of C2 has the same effect. High Courant number and high order of accuracy lead to a larger computational cost whatever the time scheme. This increase is smaller with ERK1 than explicit time schemes.

On this test case, we conclude that exponential integrators are the most efficient time integration schemes when accuracy is required. It is obviously related to the fact the integration formula is exact in the linear case since no quadrature is involved. Implicit time schemes are not precise due to their dissipation and dissipation properties.

In the next paragraphs, we evaluate the impact of having a non zero forcing term in the shallow water model. In that case, the exponential integrators will have time discretization errors.

5.2. **Time dependant forcing test case.** We consider here the time dependent forcing case: 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. According to (5.1) and (5.2), the exact solution is

$$\begin{cases} h(t,x) = \dfrac{1}{2}\left[h_0(x-ct) + h_0(x+ct)\right] \\ u(t,x) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left[h_0(x-ct) - h_0(x+ct)\right] + \dfrac{K}{\omega}\left(1-\cos\omega t\right). \end{cases} \tag{5.8}$$

In this test case, exponential integrators are not exact anymore.

Dependency of the error as a function of the frequency. The source term f is T_p-periodic with $T_p = 2\pi/\omega$. On Figure 10, we plot the values $e_h(\omega)/e_h(\omega = 0)$ and $e_u(\omega)/e_u(\omega = 0)$ in the case of shallow ocean for the second and fourth order spatial discretizations. $e_h(\omega)$ (resp. $e_u(\omega)$) is given by (5.4) (resp. (5.5)). It represents the variation of error in comparison with the error without a source term ($\omega = 0$) as in the previous subsection.
For all the unconditionally stable schemes, the time step is here fixed to $\Delta t = 600$ seconds and corresponds to a Courant number equal to 18.31. With this choice of time step, the computational costs of the exponential integrators are smaller or equal than those using RK3 or RK4. Explicit time schemes time steps Δt are given in Table 6. They reduce the computational cost to the minimum.
The error on h does not depend significantly on ω because the source function does not impact the exact $h(x,t)$ solution. On the contrary, it strongly impacts the velocity $u(x,t)$. Errors are related to the temporal resolution $T_p/\Delta t$: the number of time steps per period. Its value, for the different frequencies, is given in Table 9.
For a frequency less or equal to $\omega = 10^{-4}\mathrm{s}^{-1}$, the number of time steps per period is large. For the case $\omega = 10^{-3}\mathrm{s}^{-1}$, the number of time step is reduced to 10 and this will obviously have an impact on the results accuracy. Due to the smaller used time steps, the temporal resolution is significantly higher for the explicit integrators which should not be impacted by the temporal variation of the forcing term.
As seen on Figure 10 and looking at the fourth order C4 spatial scheme (bottom panel), low order exponential integrators (LERK1 and ERK1c) are strongly impacted by increased frequencies. This is due to the low precision on the source term. Schemes with higher order of accuracy (ERK2c, LERK3 and LERK4) significantly improve accuracy even if the error is still increasing with the frequency.

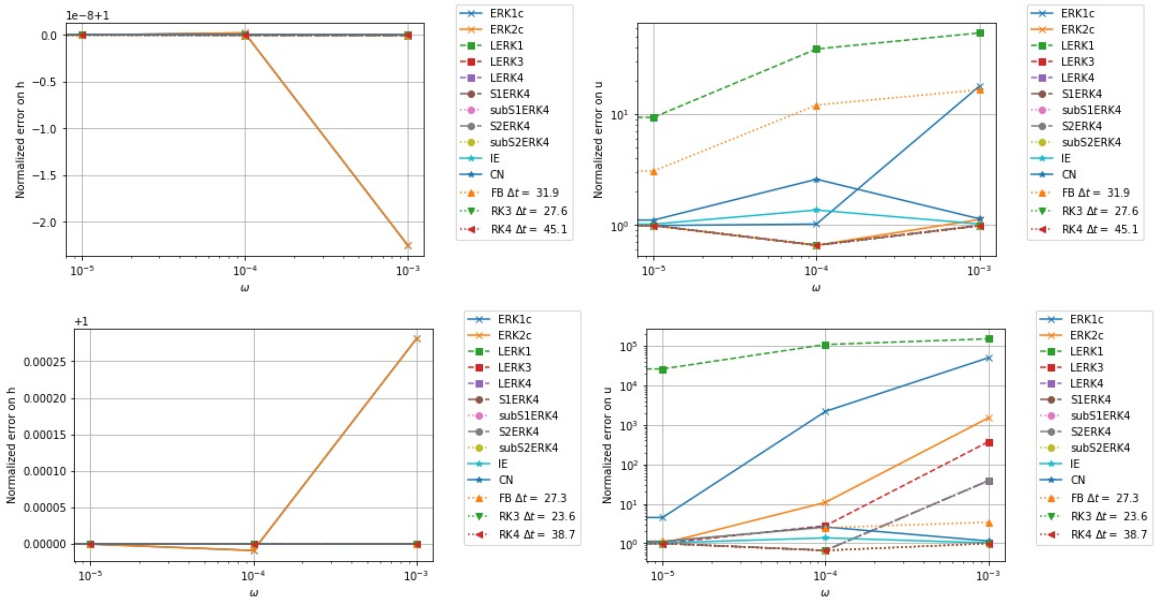

FIGURE 10. Time dependent source term test case. We plot the relative error on h and u using different time schemes and values of ω. Top panel: C2 spatial scheme, Bottom panel: C4 spatial scheme. Parameters are $\bar{h} = 100$ meters, $t_{\max} = 2$hours and $N = 500$. A-stable time schemes are using $\Delta t = 600$ seconds. Explicit time schemes are using time steps close to the maximum allowed (see Table 6).

Frequency ω	**$T_p/\Delta t$**
10^{-5}	1047.2
10^{-4}	104.7
10^{-3}	10.5

TABLE 9. Time dependent source term test case. Temporal resolution $T_p/\Delta t$ with $T_p = 2\pi/\omega$. The time step is $\Delta t = 600$ seconds.

Splitting time schemes (S1ERK4, subS1ERK4, S2ERK4 and subS2ERK4) are fourth order accurate. Indeed, in this experiment, the source term f does not here depend on space then there is no splitting error (see Appendix A.3). Then, splitting time schemes are all fourth order accurate and values $e_h(\omega)/e_h(0)$ and $e_u(\omega)/e_u(0)$ are practically the same due to small errors.
Implicit time schemes (IE and CN) are not sensitive to the frequencies ω, but this just translates the fact that even when ω is zero, the error is already large (see previous paragraph).
Explicit time schemes (FB, RK3 and RK4) are, as expected, accurate whatever the frequency.
These conclusions are less visible with the second order C2 scheme due to larger, dominating, spatial errors but they are still true.

Accuracy/Computational cost. To analyze the computational cost versus the errors, we consider Figures 11 and 12). It contains plots of relative errors related to the number of right hand side evaluations. The chosen frequency is here $\omega = 10^{-4}\text{s}^{-1}$. Greater time steps are considered for A-stable time integrators. For shallow ocean, the time steps are $\Delta t \in \{40, 150, 600, 3600, 5400\}$ while they are $\Delta t \in \{10, 50, 150, 600, 900\}$ if ocean is deep. We consider smaller time step with deep ocean to keep the Courant numbers in the same order.
Results, with the second order C2 scheme, are plotted in Figure 11. Spatial errors are larger than time errors as in the linear case. Methods with a low order of accuracy (ERKc and LERK1) are the only ones whose time error is visible. We recall that splitting integrators are fourth order accurate on this test case. Exponential integrators, when used with larger time steps Δt, are as accurate and cheaper than RK3 and RK4.
Figure 12 shows the results obtained with the fourth order accurate C4 scheme. Here, the spatial error

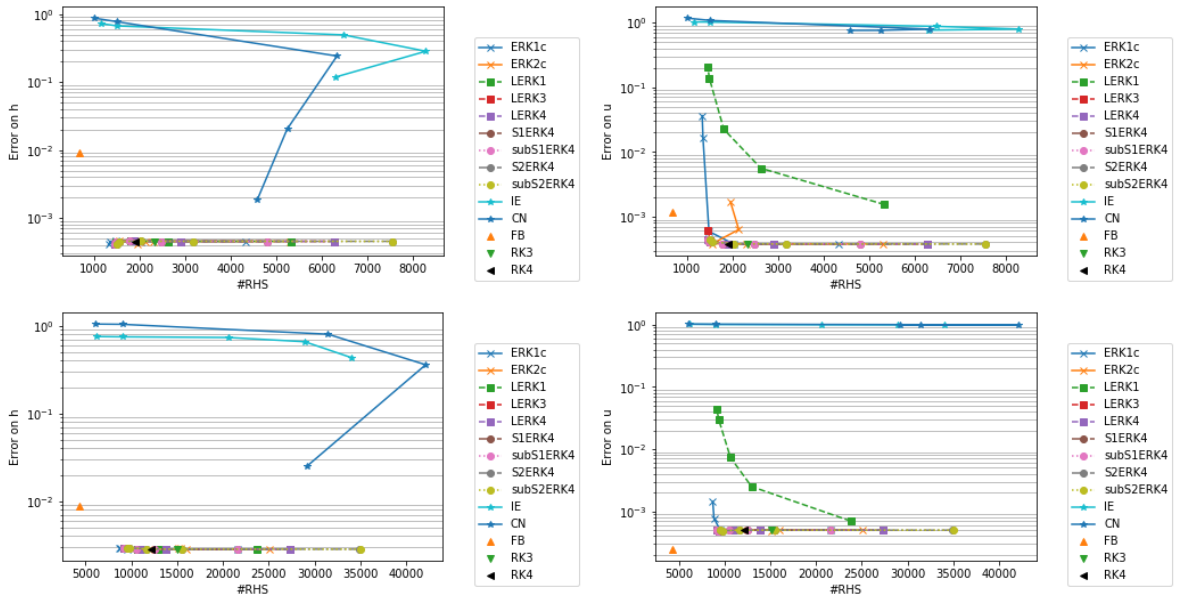

FIGURE 11. Time source term test case. Relative errors related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$ meters) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$ meters) with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

is reduced.
On the computational cost side, ERK1c, LERK and S1ERK4, when used with large time steps, are less costly than Runge-Kutta schemes. Substepped splitting exponential integrators subS1ERK4 and subS2ERK1 are particularly accurate, whatever the time step used because source function f depends only on time and there are no splitting errors (see previous section and Appendix A.3). Thus the time splitting allows the capture the temporal variation of the source term without penalizing the global accuracy. While the source term is accurately solved by the RK4 scheme using small time steps, the linear part is exactly solved by exponential.

For both case (C2 and C4), LERK3 and LERK4 are cheaper than expected. It happens because $\mathcal{N}(t)$ is in the kernel of L. It reduces the computational cost when we compute $\exp(\alpha \Delta t L)\mathcal{N}(t)$ -$\alpha \in \mathbb{R}$) during time iterations. Krylov algorithm converges into a single iteration. This is a consequence of the independence of f in space.

As it was anticipated in the linear case, exponential integrators are accurate for small frequencies ω. They can be cheaper than Runge-Kutta when large Courant numbers are taken.
Splitting and linearly exact exponential integrators are advantaged because source function f does not depend on space. The next section considers the case of a spatio-temporal varying source term.

5.3. **Time and space dependant forcing test case.** The preceding time dependent only forcing case flavored the use of splitting exponential integrators schemes (SERK4 ans subSERK4) and of high order accurate LERK. As we pointed it out, this was the consequence of the particularly of this case: there is no splitting error when the source term depends on time only and in addition the computational cost of the LERK schemes is greatly reduced.
In a more realistic context, we consider here the case when f depends both on time and space:

$$f(t, x) = K \cdot \sin \omega t \cos kx. \tag{5.9}$$

We initialize (1.9) with (h_0, u_0) given by (5.3). At each time $t \geq 0$ and for all $x \in \mathbb{R}$, the exact water height h and velocity u are given by:

$$h(t, x) = \frac{1}{2}\left(h_0(x - ct) + h_0(x + ct)\right) + \sqrt{\frac{\bar{h}}{g}} K \frac{\omega \sin(kct) - ck \sin(\omega t)}{\omega^2 - c^2 k^2} \sin(kx), \tag{5.10}$$

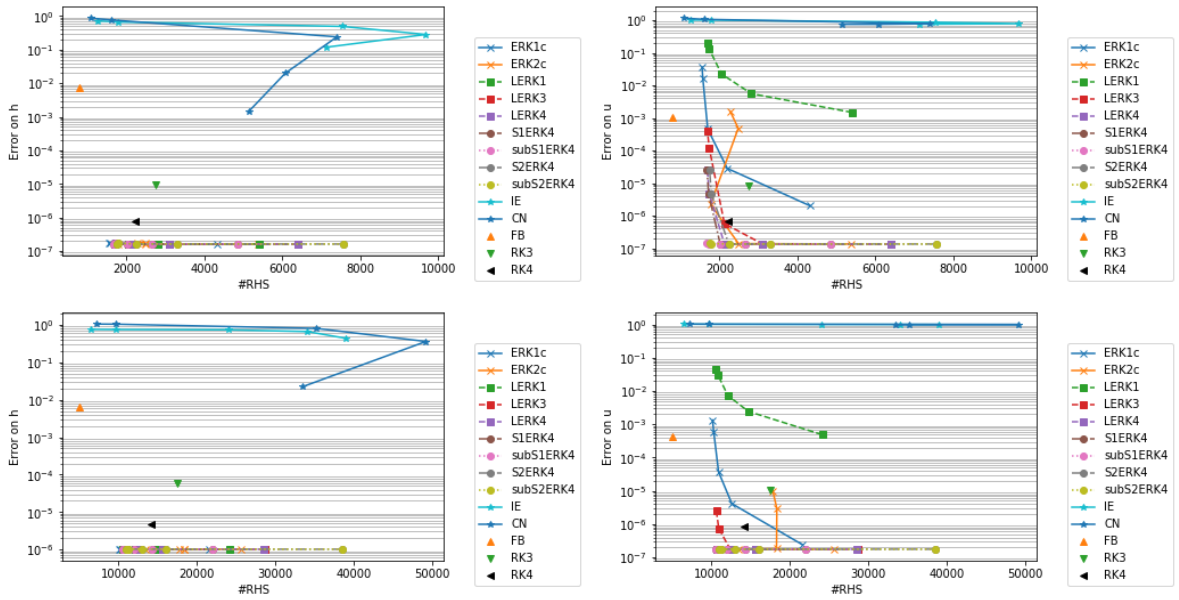

FIGURE 12. Time source term test case. Relative errors related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$ meters) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$ meters) with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

and

$$u(t,x) = \frac{1}{2}\sqrt{\frac{g}{\bar{h}}}\left(h_0(x-ct) - h_0(x+ct)\right) + K\omega\frac{\cos(kct) - \cos(\omega t)}{\omega^2 - c^2k^2}\cos(kx). \tag{5.11}$$

We choose $k = 4\pi/d \approx 2.51 \cdot 10^{-5}\mathrm{m}^{-1}$ and $\omega = 10^{-4}\mathrm{s}^{-1}$. It corresponds to a period $T_p = 2\pi/\omega \approx 62832$s. The constant K is set to $K = 1 \cdot 10^{-5}\mathrm{m} \cdot \mathrm{s}^{-2}$.

In our experiment, there are $N = 500$ grid points. It leads to $\Delta x = 1000$km and solution are well resolved in space. The time steps will be specified in the following.

Visualization of h and error. The numerical solution h and the relative error are plotted using different time schemes on Figure 13 using $N = 500$ grid points and C4 space discretization in the shallow ocean case. Results are less accurate with C2 but the analysis is similar. The time step is still $\Delta t = 600$s for unconditionally stable schemes, corresponding to a Courant number of $c\Delta t/\Delta x \approx 18.79$. The time steps used on explicit schemes are $\Delta t_{\mathrm{RK4}} = 30$s ($c\Delta t/\Delta x \approx 0.93$) and $\Delta t_{\mathrm{FB}} = 20$s ($c\Delta t/\Delta x \approx 0.63$). Thanks to this time steps used for explicit methods, source term f is well resolved in time and in space due to high resolution ($N = 500$ grid points).
The relative errors on Figure 13 are computed using the following formula:

$$\frac{\mathfrak{h}_j^n - h(t^n, x_j)}{\max_j |h(t^n, x_j)|}. \tag{5.12}$$

As in the previous cases, the accuracy of implicit time schemes is low, either due do dissipation or dispersion errors. Figure 13 does not include relative errors for the implicit schemes (IE, CN and θ-scheme ($\theta = 0.51$)) since they are order of magnitude larger than for the other schemes.
The FB scheme is, as before, the less accurate among the explicit schemes. RK3 and RK4 have a very good accuracy which is not affected by the spatio-temporal variability of the source term.
All the exponential integrators are also accurate except LERK1 and splitting methods due to their low order of accuracy and/or their splitting errors. ERK1c is the less accurate exponential integrators among LERK3, LERK4 and ERKc. However, it is more accurate than splitting methods and LERK1.

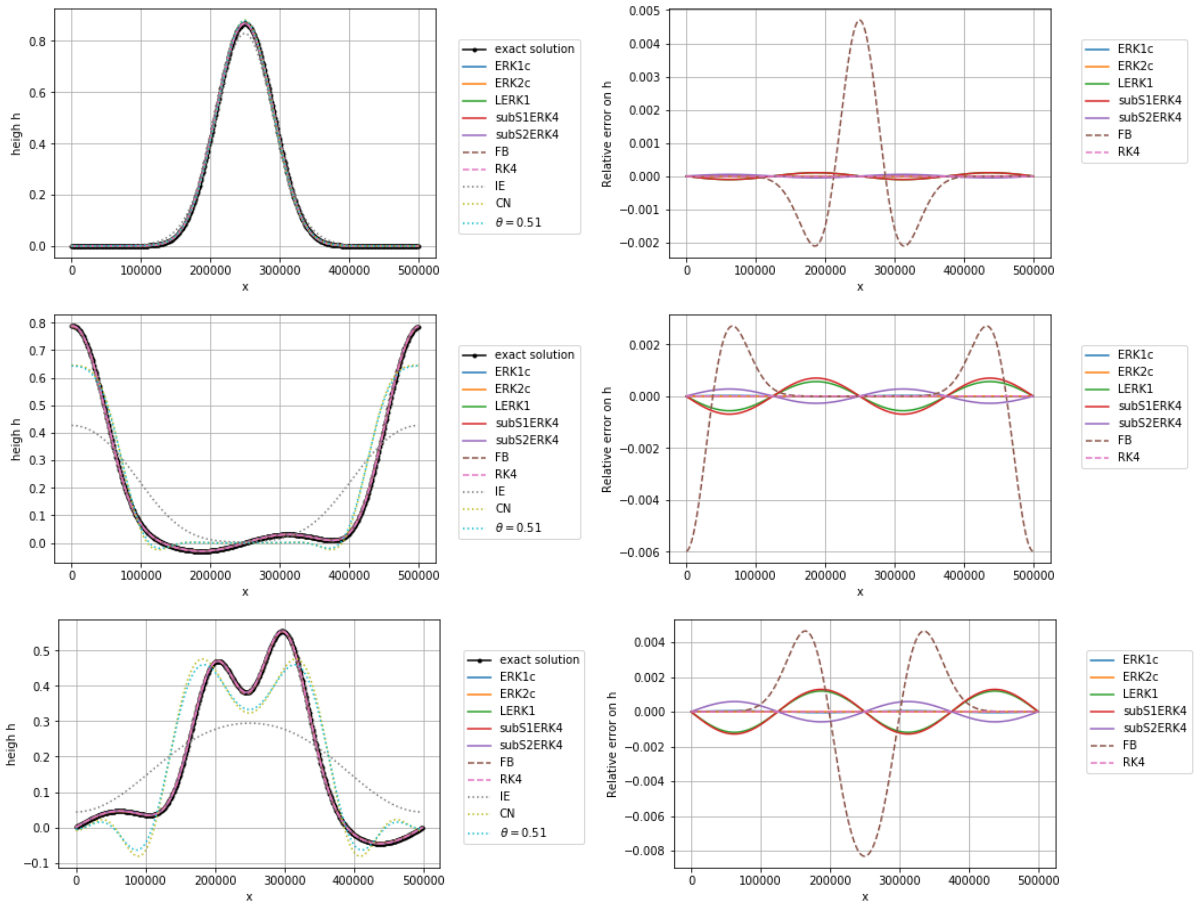

FIGURE 13. Space time source term test case. Numerical solution $\mathfrak{h}$ computed using fourth order C4 spatial discretization with $N = 500$ grid points and $\bar{h} = 100$m (shallow ocean). First line panels: solution and relative error on h at $t = 10$ minutes. Second line panels: solution and relative error on h at $t = 2$ hours. Third line panels: solution and relative error on h at $t = 4$ hours.

Evolution of the computational cost with the time steps. In Table 10, we give the number of right hand side evaluations to reach $t = 2$ hours using C4 and different time schemes. The values obtained for explicit time schemes, implicit time schemes and exponential integrators are compared.
Exponential integrators computational costs are linked to the number of exponential involved. ERK1c, LERK1 and subS1ERK4 require the computation of one exponential function, ERK2c and subS2ERK4 require two exponentials and LERK3 and LERK4 require three exponentials per time step.
Low order exponential methods ERK1c, LERK1 and subS1ERK4 computational cost does not reduce significantly using $\Delta t = 3600$s instead of $\Delta t = 600$s while LERK3 and LERK4 computational cost are cheaper. This is because first exponential functions to compute (when we compute K_i, see Appendix A.4) become cheaper than the last exponential with large time step.
From $\Delta t = 600$s, ERKc, LERK1 and subSERK4 are cheaper than explicit time schemes it was already the case for ERK1c with $\Delta t = 300$s.

	$\mathbf{\Delta t = 300}$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 ($\Delta t = 38.7$s)		744	
FB ($\Delta t = 27.3$s)		263	

TABLE 10. Space time dependent source term test case. Number of right hand side evalutions to reach $t = 2$ hours using C4 space operators with $N = 500$ grid points and $\bar{h} = 100$ meters.

Accuracy/Computational cost. As in the previous test cases, we consider the error evolution according to the number of right hand side evaluations. 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 used time steps are the same as before. They are given in Table 6 for explicit integrators. The time steps for A-stable schemes are $\Delta t \in \{40, 150, 600, 3600, 5400\}$ (shallow ocean) and $\Delta t \in \{10, 50, 150, 600, 900\}$ (deep ocean).
Results give us the following remarks.

- Since splitting errors do not cancel, we remark the lower accuracy of splitting exponential integrators (S1ERK4 and S2ERK4). This is the case even when smaller sub-time steps are used for the non linear part (subS1ERK4 and subS2ERK4 schemes).
- Linearly exact integrators are more expansive. This is because the right hand side is not anymore in the kernel of the linear part. Then, the number of calls of the right hand side is impacted by the number of exponential of matrices occurring in LERK3 and LERK4. LERK1 is poorly accurate.
- Since only two exponential of matrices are required for ERK2c and one for ERK1c, these integrators are among the cheaper. ERKc methods represent a good compromise between computational cost and accuracy (corner bottom left on plots).

Exponential integrators ERKc are more accurate with deep ocean than with shallow ocean. Among the largest time steps, many of errors are smaller explicit time schemes errors. ERK1c, with $\Delta t = 600$seconds, is more accurate and less expansive than RK4.
Splitting time schemes errors are significantly larger than ERKc errors even if order of accuracy are the same (ERK1c/S2ERK4). However, computational cost are the same.

In the experiments we have done, ERK1c seems to be a good compromise between accuracy and computational cost. It is the cheaper A-stable time scheme and the more accurate.
Exponential integrators are more adapted to deep ocean. A more accurate solution is obtained with the same time step Δt (see Figures 14 and 15).

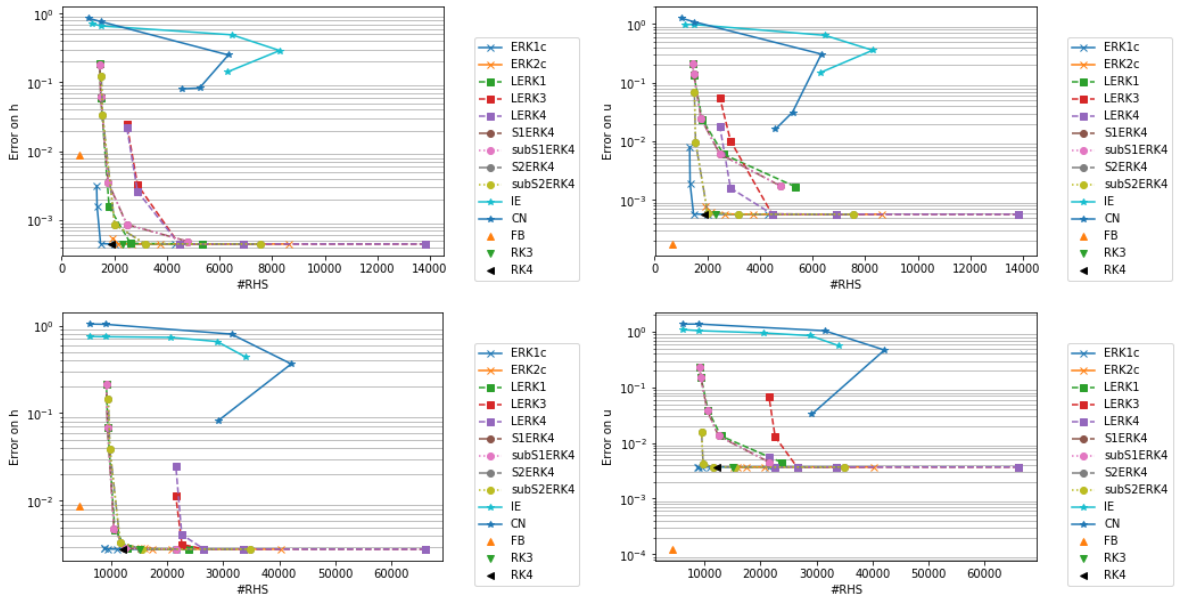

FIGURE 14. Space time source term test case. Relative error on h and u related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$) 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.

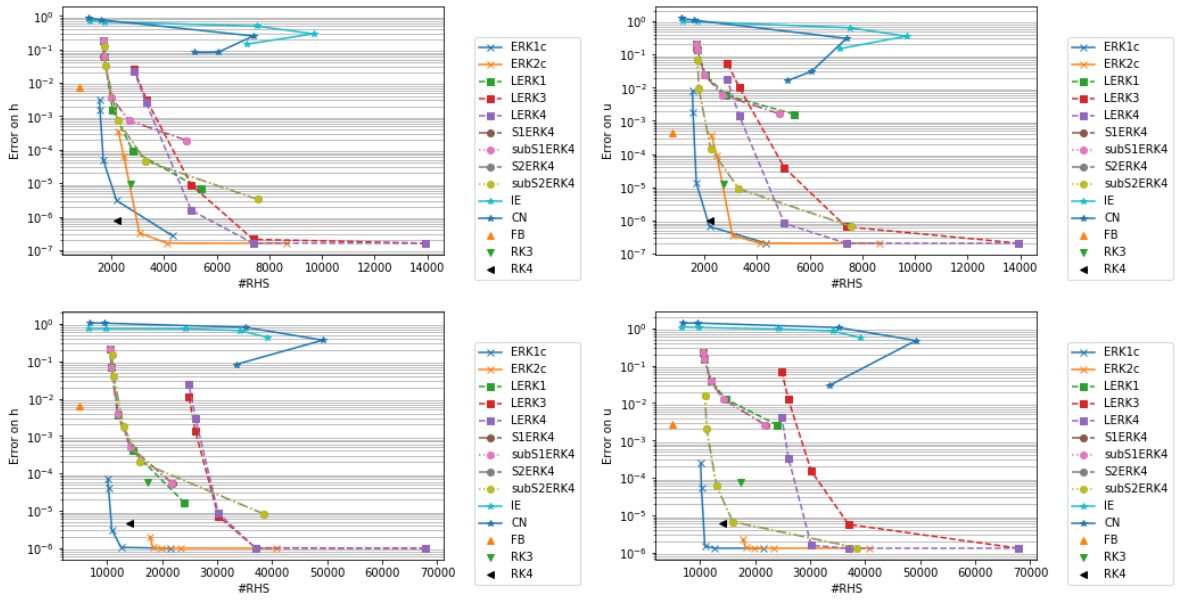

FIGURE 15. Space time source term test case. Relative error on h and u related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$) 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.

6. Conclusion

In this article, we analyze a large set of time schemes to solve LSWE (1.9).

- *Explicit time schemes:* Runge-Kutta methods third and fourth order accurate and Forward-Backward scheme. This integrators are cheaper by time step but a large number of time iterations is required 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 desired. Unfortunately, implicit time schemes suffer dramatically 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. 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 CFL conditions using C4 than C2. Furthermore, Krylov methods and conjugate gradient converge more slowly with high order accurate space discretization. These is due to the larger spectral radius of matrices.

As mentioned, due to their poor dissipation and dispersion properties implicit integrators suffer from poor accuracy. Numerical solution become flat or distort, in particular with large Courant number. Exponential integrators have better dissipation and dispersion properties. 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.

We analyze various exponential integrators. High order accurate linearly exact Runge-Kutta integrators are the more accurate time scheme considered. However they are expansive because there are several exponential of matrices. Exponential Runge-Kutta integrators and splitting exponential integrators are cheaper. Splitting exponential integrators suffer to the splitting error when source term is space dependent. Exponential Runge-Kutta integrators are a good compromise between accuracy and computational cost.

Exponential integrators create an opportunity to perform an accurate solver for shallow water based equation using larger time steps than explicit integrators. Interesting applications are the bedload transport on which numerical diffusion is large on the bottom. It could also be competitive to increase accuracy in ocean model.

The main disadvantage (but not least) is the computational cost. To be competitive, large time steps have to be considered in exponential integrators. There exist a large panel of methods allowing to reduce the computational cost of implicit time integrators. We hope to use the same kind of method to reduce the computational cost of exponential integrators. Among others we mention: parallelism [11, 30] and domain decomposition [4, 21].

This possible developments are address to future work.

Acknowledgements

ANR ADOM. H2020 IMMERSE.

Appendix A. Time integrators

Consider the ordinary differential equation

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

We call autonomous the case such that $\mathcal{F}$ 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 (1.9), the function $\mathcal{N}(X,t)$ does not depend on X, more precisely:

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

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 third and fourth order accurate and a Forward Backward scheme.

A.1.1. *The Ralston's third order integrator.* The Ralston's third order integrator [27] 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 explicit Runge-Kutta integrator (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 (1.9) (see [31] and references therein). It has a small computational cost per iteration: the right hand side is evaluated only once per time iteration.
The 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 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

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

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, IE). The

Crank-Nicholson scheme (CN) corresponds to $\theta = 1/2$ and for this value of θ, the scheme is neutral (no dissipation). In the numerical experiments we done, to add a small dissipation and stay close to CN, we also consider the case $\theta = 0.51$.
There are no stability criterion while $\theta \geq 1/2$.

A.3. **Exponential Runge-Kutta Integrators.** Exponential Runge-Kutta Integrators (ERK) 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 [15] to review. In this section, we remind ideas of exponential Runge-Kutta integrators.

We start by consider an autonomous system:

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

and 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 ERK integrators. These depend of matrix functions of the set $(\varphi_k)_k$ define by

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

ERK integrators are A-stable and exact for linear equations. In the following, we present different kind of exponential Runge-Kutta 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):

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

The function φ_1 is $\varphi_1(z) = \frac{e^z - 1}{z}$ and extends by $\varphi_1(0) = 1$.
The ERK1 exponential operator is first order accurate. If L contains the full linear part (see Rosenbrock exponential integrators [16]):

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

it is second order accurate. Here, $\mathrm{Jac}_{X^n}\mathcal{F}$ is the Jacobian matrix. L_n may be required to be updated at each time step. Considering (A.11), ERK1 is 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 exponential Runge-Kutta integrator.* A more accurate scheme is ERK2:

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

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 set 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 [16], 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

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

We denote this scheme ERK1c and ERK2c.
The schemes (A.17) and (A.18) are respectively accurate order 1 and 2. They are $A-stable$ 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.

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 [19]. A review of these methods is given in [5, 22] and reference therein.
In the following, we call these methods LERK for linearly exact Runge-Kutta methods.

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

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

where $\mathcal{N}(X,t) = \mathcal{F}(X,t) - LX$, in our case $\mathcal{N}(X,t) = \mathcal{N}(t)$.
At this step, it is sufficient to apply an explicit time scheme on (A.20). If $L = 0$, LERK coincide with an explicit time scheme. If $\mathcal{N}(t)$ satisfy $\mathcal{N}(t)_i = \mathcal{N}(t)_j$ for all $i \neq j$, then $\mathcal{N}(t)$ is in the kernel of L and $\exp(\alpha L)\mathcal{N}(t)$ is compute into a single iteration thanks Krylov methods:

$$\exp(\alpha L)\mathcal{N}(t) = \mathcal{N}(t) \in \mathcal{K}_1\left(L,\mathcal{N}(t)\right). \tag{A.21}$$

These reduce significantly the computational cost of this integrators.
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 (1.9) but the exponential part stabilize it. It is given by the formula

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

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

$$X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \Delta t b(t^n)\right]. \tag{A.23}$$

One exponential is required by time step.

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

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

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

$$\begin{cases} K_1 = \mathcal{N}(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{3\Delta t}{4}L\right)\mathcal{N}\left(t^n + \frac{3\Delta t}{4}\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{9}(2K_1 + 3K_2 + 4K_3)\right]. \end{cases} \tag{A.25}$$

LERK3 contains five exponential of matrices while ERK2c contains 2 matrix functions φ_1 and φ_3. As consequence, LERK3 should be more expansive than ERK2c. If $\mathcal{N}(X,t) = \mathcal{N}(t)$, then LERK3 contains only three exponential of matrices.

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

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_1\right), t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_2\right), t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\mathcal{N}\left(\exp\left(\Delta t L\right)\left(X^n + \Delta t K_3\right), t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{A.26}$$

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 $\mathcal{N}(X,t) = \mathcal{N}(t)$, LERK4 is less expansive and is simplified into:

$$\begin{cases} K_1 = \mathcal{N}(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\mathcal{N}\left(t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 4K_2 + K_4)\right] \end{cases} \tag{A.27}$$

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 [17] and reference therein).
Two splitting are considered here : Lie and Strang. We use a 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 $\mathcal{S}_{l,\Delta t} \circ \mathcal{S}_{nl,\Delta t}$ The scheme is then :

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^n + \Delta t K_3, t^n + \Delta t\right) \\ X^* = X^n + \dfrac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ X^{n+1} = \exp\left(\Delta t L\right) X^*. \end{cases} \tag{A.28}$$

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 non linear term in (A.2) depends only on time, the splitting scheme is 4-th order accurate because splitting error is null. We will note this time scheme S1ERK4.

Sub-time steps can be considered in the RK4 part to solve the non-linear part. Denote subS1ERK4 this variant of the method:

$$\mathcal{S}_{l,\Delta t} \circ \left(\mathcal{S}_{nl,\Delta t/N_s}\right)^{N_s}. \tag{A.29}$$

The number of substeps N_s is chosen to be egual to maximum time step allowed for RK4 :

$$\frac{c\Delta t/N_s}{\Delta x} \leq K_{\mathrm{RK4}} \tag{A.30}$$

where K_{RK4} is the RK4 stability constraint (see Table 4).
The time integrator subS1ERK4 suffers from splitting error as S1ERK4. It is first order accurate. However it is fourth order accurate if $\mathcal{N}(X,t) \in \mathrm{Ker}(L)$.
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 equations.
The method is $\mathcal{S}_{l,\Delta t/2} \circ \mathcal{S}_{nl,\Delta t} \circ \mathcal{S}_{l,\Delta t/2}$. The corresponding algorithm is

$$\begin{cases} X^* = \exp\left(\dfrac{\Delta t}{2} L\right) X^n \\ K_1 = \mathcal{N}(X^*, t^n) \\ K_2 = \mathcal{N}\left(X^* + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^* + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^* + \Delta t K_3, t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\dfrac{\Delta t}{2} L\right)\left[X^* + \dfrac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{A.31}$$

We call this scheme S2ERK4. It is second order accurate and two exponential of matrices are occurring at each time step.

As subS1ERK4, it is possible to increase accuracy on the non linear equation by considering N_s sub-time steps. The integrator is subS2ERK4 and is given by

$$\mathcal{S}_{l,\Delta t/2} \circ \left(\mathcal{S}_{nl,\Delta t/N_s}\right)^{N_s} \circ \mathcal{S}_{l,\Delta t/2} \tag{A.32}$$

Two exponential of matrices are used at each time step. As in the Lie splitting integrators S1ERK4 and subS1ERK4, the time scheme is 4-th order accurate when non linear part satisfies $\mathcal{N}(X,t) \in \mathrm{Ker}(L)$. It is second order accurate in general (i.e. if $\mathcal{N}(X,t) \notin \mathrm{Ker}(L)$). The number N_s of substeps is chosen thanks (A.30).

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

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

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

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

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

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

[24] 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.

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

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

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

[28] Y. Saad. Analysis of some Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 29(1):209–228, 1992.
[29] Y. Saad. *Iterative methods for sparse linear systems*, volume 82. SIAM, 2003.
[30] 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.
[31] 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.
[32] 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.
[33] 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.
[34] 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.

†Univ. Grenoble Alpes, Inria, CNRS (UMR 5224), Grenoble INP∗, LJK, 38000 Grenoble, France
E-mail address: [1]matthieu.brachet@inria.fr, [2]laurent.debreu@inria.fr, [3]christopher.eldred@inria.fr,