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

Matthieu Brachet, Laurent Debreu, Christopher Eldred

COMPARISON OF EXPONENTIAL INTEGRATORS AND TRADITIONAL TIME INTEGRATION SCHEMES FOR THE SHALLOW WATER EQUATIONS

MATTHIEU BRACHET, LAURENT DEBREU, AND CHRISTOPHER ELDRED

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. This 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 constrained by the Courant-Friedrichs-Lewy stability condition. Implicit schemes can be unconditionally stable but, in practice lack accuracy when used with large time steps. In this paper we propose a detailed comparison of such classical schemes with exponential integrators. The accuracy and the computational costs are analyzed in different configurations.

Keywords: Shallow water equations, Time stepping, Exponential integrators, Finite differences, Krylov methods

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 fluid thickness. In a one-dimensional framework, the unknowns are h, the fluid thickness, and u, the horizontal velocity. The system is expressed as:

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

for all $x \in [0,d]$, $t \geq 0$ and the initial data are $(u_0(x), h_0(x))$. Equations (1) are closed with appropriate boundary conditions, which we consider here to be periodic. g is the gravity. 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). 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 separately from the 3D equations using a shallow water system forced by the depth average of the 3D right hand side (see [34] and reference therein).
- *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 [10].
- *Atmosphere model*: shallow water system on a rotating sphere is the simplest atmospheric model of interest. It is considered as a first step in the study of a numerical solver for geophysical fluids [37].

LD acknowledges support by the ANR through contract ANR-18-CE46-0008 (ADOM). MB and LD have received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 821926 (IMMERSE)..

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under Contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

The desired degree of accuracy of the numerical solution of (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 moderate. This is the case for example for large scale (e.g. climate) oceanic applications. Operational (and thus time to solution) constraints can also favor the use of such schemes. For other applications, like bedload transport or tidal model, 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, the Kinnmark-Gray 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 are limited by a Courant-Friedrichs-Lewy (CFL) stability condition (see [22] for a review on the different stability conditions of an ocean model):

$$\frac{c\Delta t}{\Delta x} \leq \mathrm{C}^{\mathrm{ste}}, \tag{2}$$

where Δt and Δx are the temporal and spatial discretization steps. $\dfrac{c\Delta t}{\Delta x}$ is the Courant number where c is a characteristic velocity which corresponds in the shallow water model to $c = \sqrt{gH}$ where H is the mean total depth [20]. In the context of ocean models, Forward-Backward schemes are considered in [34] and Runge-Kutta integrators in [11, 36] among others.

An alternative is to use implicit schemes, the simplest one being the Euler Backward scheme. This family of time integrators are potentially A-stable and allow much greater time steps than explicit schemes. The price to pay is 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, interest has grown in the use of exponential integrators to solve partial differential equations of the form

$$\frac{dX}{dt} = \mathcal{F}(X,t). \tag{3}$$

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

A review of these schemes is available in [17] in a general framework. The main property of this class of integrators is exact solution of linear equations, i.e. when $\mathcal{N} = 0$. A consequence is A-stability: these 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 exponentials. This can be done using factorization methods or with polynomial approximation but it is generally expensive and ill-conditioned. A classical review is [26]. Advances in the application of Krylov methods [8, 15] for performing this computation have brought more interest in these methods.

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

$$X(t) = \exp(tL)\,X(0) + \int_0^t \exp\left((t-\tau)L\right)\mathcal{N}\left(X(\tau)\right)d\tau, \tag{4}$$

  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 (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, 9] where it is shown that these methods are as accurate as explicit Runge-Kutta methods but allow the use of much larger time steps. In all of these articles, the accuracy is analyzed on the classical set of test cases [37] for which no time dependent analytical solution is available. Although the results are encouraging, they do not allow a precise estimate of accuracy and analysis of errors. A multi-step exponential integrator is considered in [14] in a similar context. This integrator requires an efficient algorithm for computing a linear combination of exponential functions, in particular the **phipm** algorithm developed by [27]).

In [13], ERK integrators are compared to implicit time schemes for a three-dimensional Boussinesq thermal convection system. These methods are found to be more accurate than implicit schemes but also more expensive.
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 scaling and squaring with a Padé approximation. It is the faster method but matrix exponentials have to be stored in memory. The second way relies on computing matrix exponentials at each time step using Krylov methods. In that case, the matrix 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 [29], exponential integrators are considered for the solution of multi-layer shallow water equations. The spatial discretization is performed with mimetic elements leading to skew-symmetric matrices and exponential functions are computed thanks to 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 provide a way to solve autonomous equations. We denote by ERKc (see A.3 and [18]) the ERK integrators corrected for the solution of non-autonomous systems. The order of accuracy of ERKc is the same as the equivalent ERK integrators. The computational cost is the same but they require an approximation of the time derivative of the right hand side.

- To avoid quadrature in the integral part of (4), we can proceed to the following change of variable:

$$V(t) = \exp((t^n - t)L)X(t) \tag{5}$$

in (3) and use an explicit scheme for the equation satisfied by V:

$$V'(t) = \exp((t^n - t)L)\mathcal{N}\left(\exp((t - t^n)L)V(t), t\right). \tag{6}$$

Equation (6) can be solved with any time integration scheme. When Runge-Kutta integrators are considered, this corresponds to Lawson integrators.
These exponential integrators are called Linearly Exact Runge-Kutta integrators (LERK) in the following. They are used in [28] 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 as the Korteweg-de Vries equation. The error is numerically investigated.
It is easy to build a high order accurate LERK integrator by considering a high order Runge-Kutta method on (6). Unfortunately, the computation of many matrix exponentials is 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, Lie 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:

$$\begin{cases} \dfrac{dX}{dt} = LX \\ X(0) = X^n, \end{cases} \tag{7}$$

using the matrix exponential. This step 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^*:

$$\begin{cases} \dfrac{dX}{dt} = \mathcal{N}(X, t) \\ X(0) = X^*, \end{cases} \tag{8}$$

using an explicit scheme such as the fourth order Runge-Kutta (RK4) scheme. The solver for this 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 matrix exponential is required for the computation of $\mathcal{S}_{\mathrm{l},\Delta t}$. Each equation is solved with a highly accurate scheme. Lie splitting will be denoted S1ERK4. To increase the accuracy of the non-linear integration, especially when the non-linear term has a high temporal frequency, we can consider substepping in time (N_s

sub-steps) for the computation of $\mathcal{S}_{\mathrm{nl},\Delta t}$. Lie splitting becomes $\left(\mathcal{S}_{\mathrm{nl},\Delta t/N_s}\right)^{N_s} \circ \mathcal{S}_{\mathrm{l},\Delta t}$ and will be denoted subS1ERK4. Due to splitting errors, the resulting scheme is only first order accurate.
More accurate methods are based on Strang 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 substepping version). These schemes are second order accurate and require the computation of two matrix exponentials but the number of matrix exponentials required can be reduced to one by combining two time iterations.
Integrators relying on splitting methods have been used in [19] 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 notation P/Q for a given scheme means that the order accuracy is P and the number of required matrix exponentials is Q. Two cases are distinguished:

- The first case (left column) corresponds to a non linear part $\mathcal{N}(X,t)$ depending on X and not just on time t. In their Rosenbrock formulation, ERKc methods benefit from an updated linear part thanks to the Jacobian and gain one order of accuracy without increasing the computational cost. LERK integrators are higher order accurate but suffer from the number of matrix exponentials occurring (5 exponentials for LERK3, 7 for LERK4). LERK3 is more expensive than ERK2c while the order of accuracy is the same as in the Rosenbrock case. Splitting methods have a computational cost corresponding to their accuracy: n-th order methods have n matrix exponentials to compute per time step. Splitting errors are the limiting factor on accuracy.
- The second case (right column) is devoted to a specific type of equations in which the non linear term depends only on time $\mathcal{N}(X,t) = \mathcal{N}(t)$. In this configuration, ERKc automatically benefit from the advantages of the Rosenbrock case and are more accurate with the same number of matrix exponentials to compute.
  LERK methods are less expensive than in the first case due to simplifications occurring in the algorithm (see A.4): they require less matrix exponentials. Furthermore, if $\mathcal{N}(t)$ is in the kernel of L, some exponentials are easily computed (*i.e.* $\exp(\Delta t L)\mathcal{N}(t) = \mathcal{N}(t)$) and this significantly reduces their computational cost. It does not impact the accuracy. Splitting based integrators gain in accuracy when $\mathcal{N}(t) \in \mathrm{Ker}(L)$ because splitting errors cancel but the number of exponentials is the same as 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 matrix exponentials for one time step. ERK are a specific case of ERKc for autonomous PDEs.

In the following, we will consider linearized equations around a steady state $(\bar{h}, 0)$ where $\bar{h}$ is a positive constant fluid height. The system of linearized shallow water equations (LSWE) is given by

$$\begin{cases} \dfrac{\partial h'}{\partial t} + \bar{h}\dfrac{\partial u'}{\partial x} = 0 \\ \dfrac{\partial u'}{\partial t} + g\dfrac{\partial h'}{\partial x} = f(t,x) \end{cases} \tag{9}$$

where (h', u') are small perturbations around the resting steady state $(\bar{h}, 0)$. This linear system is a good model for the nonlinear equations. Indeed, in atmospheric or oceanic cases, the models are usually weakly nonlinear therefore most of the relevant behavior is captured by (9) through an appropriate choice of $f(t,x)$. The characteristic velocity of (9) is $c = \sqrt{g\bar{h}}$. In the following, we will drop the $'$ to simplify notations.

Some exponential integrators are specially adapted for linear problems such as (9) (see quadrature rules in [17]). These time integrators are not considered in this article because they are not suitable for nonlinear problems, which are the ultimate goal.

The purpose of this paper is to compare various time schemes for linearized shallow water equations. This includes explicit Runge-Kutta, Kinnmark-Gray and Forward-Backward schemes, implicit θ-schemes and exponential integrators. The stability constraints associated to each of these scheme are first recalled. Then the accuracy is studied and we are particularly interested in how the frequency of the forcing term f in (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. Section 3 is devoted to the description of 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 different specifications of the forcing term $f(t,x)$ in section 5.

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

Equations (9) are solved using the method of lines. The first step is the discretization in space. We consider two centered finite difference order operators, second (C2) and fourth (C4) order accurate, on a staggered grid, a one-dimensional version of the C-grid in the Arakawa classification (see [1]).
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 constraints and thus on the computational cost of the spatio-temporal schemes.

Two staggered grids are considered for the space discretization of (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

$$\begin{cases} x_i := i\Delta x \\ x_{i+1/2} := x_i + \dfrac{\Delta x}{2} = \left(i + \dfrac{1}{2}\right)\Delta x. \end{cases} \tag{10}$$

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 $(x_{i+1/2})_{0\leq i\leq N-1}$:

$$\begin{cases} h(\cdot, x_i) \approx \mathfrak{h}_i \text{ with } 0 \leq i \leq N-1, \\ u(\cdot, x_{i+1/2}) \approx \mathfrak{u}_{i+1/2} \text{ with } 0 \leq i \leq N-1. \end{cases} \tag{11}$$

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

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

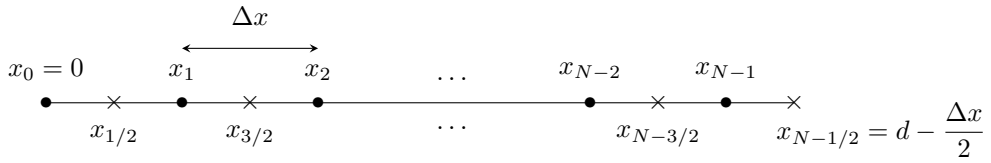

FIGURE 1. Staggered grid representing $(x_i)_{0\leq i\leq N-1} \in \mathbb{R}^N$ and $(x_{i+1/2})_{0\leq i\leq N-1} \in \mathbb{R}^N$.

- *Order 2 FD Operators (C2):*

$$\begin{cases} \dfrac{\partial u}{\partial x}_i \approx \delta_x \mathfrak{u}_i := \dfrac{\mathfrak{u}_{i+1/2} - \mathfrak{u}_{i-1/2}}{\Delta x} \\ \dfrac{\partial h}{\partial x}_{i+1/2} \approx \delta_x \mathfrak{h}_{i+1/2} := \dfrac{\mathfrak{h}_{i+1} - \mathfrak{h}_i}{\Delta x} \end{cases} \tag{12}$$

- *Order 4 FD Operators (C4):*

$$\begin{cases} \dfrac{\partial u}{\partial x}_i \approx \delta_x \mathfrak{u}_i := \dfrac{9}{8}\dfrac{\mathfrak{u}_{i+1/2} - \mathfrak{u}_{i-1/2}}{\Delta x} - \dfrac{1}{8}\dfrac{\mathfrak{u}_{i+3/2} - \mathfrak{u}_{i-3/2}}{3\Delta x} \\ \dfrac{\partial h}{\partial x}_{i+1/2} \approx \delta_x \mathfrak{h}_{i+1/2} := \dfrac{9}{8}\dfrac{\mathfrak{h}_{i+1} - \mathfrak{h}_i}{\Delta x} - \dfrac{1}{8}\dfrac{\mathfrak{h}_{i+2} - \mathfrak{h}_{i-1}}{3\Delta x} \end{cases} \tag{13}$$

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 gives us the phase error:

$$e_\Phi := \frac{\arg \exp\left(\lambda_x(\theta)\right)}{\theta} \tag{14}$$

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 the C2 operator.

Space operator	**Phase error**
C2	$\dfrac{\sin(\theta/2)}{\theta/2}$
C4	$\dfrac{(13 - \cos\theta)\sin(\theta/2)}{6\theta}$

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

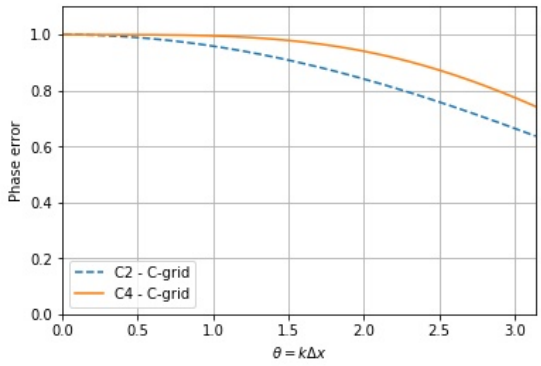

FIGURE 2. Numerical phase error for second and fourth order centered approximations on a staggered grid. The normalized wave number is $\theta = k\Delta x$.

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

The computation of exponential integrators involve several matrix vector product functions $\varphi_k(A)b$ (see Appendix A.3). The functions φ_k used in our numerical schemes are:

$$\begin{cases} \varphi_0(z) = \exp(z) \\ \varphi_1(z) = \dfrac{\exp(z)-1}{z} \\ \varphi_2(z) = \dfrac{\exp(z)-1-z}{z^2}, \\ \varphi_3(z) = \dfrac{\exp(z)-1-z-z^2/2}{z^3}, \end{cases} \tag{15}$$

where the $\varphi_k(z)$ functions are extended to 0 by $1/k!$.
The problem is solved thanks to Krylov methods [16, 27, 32] which are based on the projection of the matrix A onto a Krylov subspace using Arnoldi's algorithm. The $\varphi_k(A)b$ products are then approximated in this subspace.

3.1. Arnoldi's method for Krylov subspaces.

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

$$\mathcal{K}_m(A,b) = \mathrm{Span}(b, Ab, A^2b, \cdots, A^{m-1}b). \tag{16}$$

An upper Hessenberg matrix $H_m \in \mathbb{M}_m(\mathbb{R})$ and a matrix V_m are built using Algorithm 1. The coefficients of H_m are $(h_{i,j})_{1\le i,j\le m}$ and the columns of V_m are the vectors $(v_i)_{1\le i\le m}$.

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 equality [32]:

$$V_m^T A V_m = H_m. \tag{17}$$

The computational cost of the Arnoldi Gram-Schmidt algorithm is $\mathcal{O}(m^2 n)$ where n is the size of A (see [32]).

3.2. **Computing matrix function product.** As mentioned in [29], the use of exponential integrators requires efficient computation of the product $\varphi_k(A)b$ with A a matrix and b a vector. Indeed, for large matrices A, $\varphi_k(A)$ cannot be saved once and for all because it is dense *a priori* and therefore storage costs would be prohibitive.
The first step is to build 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 approximated by

$$\varphi_k(A)b \approx x_m = \|b\|_2 V_m \varphi_k(H_m) e_1. \tag{18}$$

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 high order Padé approximant. The error between the Padé approximant and the exact value of $\varphi_k(H_m)$ is linked to the norm of H_m, and thus scaling and squaring methods are used (see [26, 35]).

3.2.1. *Stopping criterion.* A stopping criterion is given in the exponential case (i.e. $k = 0$) in [31]. It is adapted and used for a general φ_k in [27]. Following ideas given in these papers, we use the error formula

$$\varphi_k(A)b - x_m = \|b\|_2 h_{m+1,m} \left(\sum_{i=k+1}^{\infty} A^{i-k+1} v_{m+1} e_m^T \varphi_i(H_m) \right) e_1. \tag{19}$$

The norm of the first term of this series

$$\|b\|_2 \cdot |h_{m+1,m}| \| v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 \|_2 \tag{20}$$

provides a good error estimate which we will use as a stopping criterion in our experiments. From 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 (56). In [31], author suggests to consider more terms of the series for the stopping criterion but this has not been 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

$$\|b\|_2 \cdot \| h_{m+1,m} v_{m+1} e_m^T \varphi_{k+1}(H_m) e_1 \|_2 \tag{21}$$

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

$$L = \begin{bmatrix} 0 & -\bar{h}\delta_x \\ -g\delta_x & 0 \end{bmatrix}, \tag{22}$$

$X = [\mathfrak{h}, \mathfrak{u}]^T \in \mathbb{R}^{2N}$. The vector $b \in \mathbb{R}^{2N}$ is randomized to contain all frequencies. Results are given in Figures 3 and 4 for respectively the second and fourth order discretizations. 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 in Figures 3 and 4, the number of iterations increases with the Courant number. This was expected due to the theorem 4.3 in [31]: a larger Courant number

increases the spectral radius and the *a priori* error bound. C4 operator is slightly 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, as soon as it is sufficiently small, does not impact significantly the computational cost.

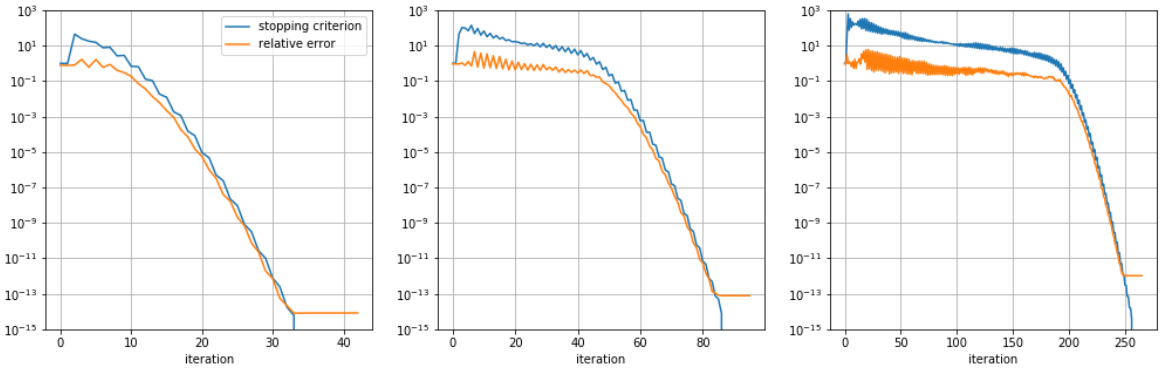

FIGURE 3. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There are $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel),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

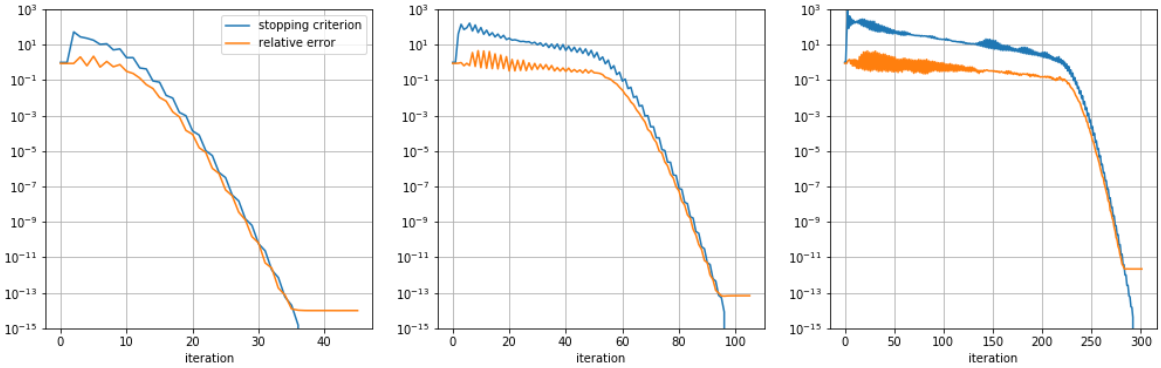

FIGURE 4. Relative error and stopping criterion for the Krylov estimation of $\varphi_1(\Delta t L)b$ with $b \in \mathbb{R}^{2N}$ a random vector. There are $N = 500$ grid points. The fluid thickness is $\bar{h} = 100$ meters. The time step Δt is such that the Courant number is equal to 5 (left panel),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

3.2.2. *Influence of Courant number and discretization parameter on Krylov method.* In [31] (theorem 4.2), it is proved the error for a Krylov approximation of the matrix exponential is

$$\|x - x_m\|_2 \leq 2\|b\|_2 \frac{\rho(A)^m e^{\rho(A)}}{m!} \tag{23}$$

where x_m is the Krylov approximation of $x = e^A b$ and $\rho(A)$ is the spectral radius of A. This result is true whatever the matrix A and the vector b are.
Using $A = \Delta t L$, where L is given by (22), $\rho(A)$ is proportional to $\dfrac{c\Delta t}{\Delta x}$. Thus, fixing tolerance, m is expected to depend mainly on the Courant number $\dfrac{c\Delta t}{\Delta x}$.

To check this result, we determine the number of Krylov iterations m required to to compute $\varphi_0(\Delta t L)b$ with a stopping criterion smaller than 10^{-10}. As in the previous experiment, b is a randomized vector. We consider different grid parameters N and Courant numbers $c\Delta t/\Delta x$. The results are summarized in Table 3.

Grid parameter N	$\frac{c\Delta t}{\Delta x} = 5$	$\frac{c\Delta t}{\Delta x} = 25$	$\frac{c\Delta t}{\Delta x} = 100$	$\frac{c\Delta t}{\Delta x} = 200$	$\frac{c\Delta t}{\Delta x} = 500$	$\frac{c\Delta t}{\Delta x} = 1000$
200	32	92	217	217	219	221
500	32	92	286	509	563	565
1000	32	92	286	532	1143	1176
2000	33	92	286	534	1256	2307
5000	33	94	288	534	1258	2448
10000	34	94	288	536	1260	2450

TABLE 3. Dimension of $\mathcal{K}_m(\Delta t L, b)$ to compute $\varphi_0(\Delta t L)b$ such that stopping criterion's (21) is smaller than 10^{-10}. The vector $b \in \mathbb{R}^{2N}$ is a random vector. The space discretization is C4.

For a fixed Courant number $c\Delta t/\Delta x$, it is remarkable that m is constant while N increases. For this reason, the value m should be small compared to N for thin grids (i.e. large value N). For coarse grids, the value m seems to be at worst $m \approx N$ where as the Krylov approximation is applied on a $2N \times 2N$ matrix. Furthermore, as expected, the Krylov dimension m required increases with the Courant number. Similar results are obtained for all φ_k function considered in our experiments.

Due to these results, we will focus on the number of matrix products Lb (and of right hand side evaluations) occurring as a significant part of the computational cost.
In our experiments, we will consider $N = 500$ because the number of linear parts occurring does not depend significantly on N and this value is representative of the behaviors for the considered methods.

4. Properties of time integration schemes

We here consider various time integration schemes, combined with the previous spatial discretization operators to solve LSWE (9). The detailed formulations of the time schemes are given in Appendix A. Notations used are given in Table 4.

Time integrators:	
FB	Forward-Backward.
RK(n)	Runge-Kutta integrator (n)-th order accurate.
RK-KG(2,6)	RK - Kinnmark-Grey 2-nd order accurate with 6 steps.
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 4. Time integration schemes used in this paper.

The stability condition is one of the main property of a numerical scheme. Consider a one step time integration scheme to solve (9) (with $f(t,x) = 0$). There exists a matrix $G \in \mathbb{M}_{2N}(\mathbb{R})$ such that:

$$X^{n+1} = GX^n, \tag{24}$$

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

- Forward-Backward:

$$G = Id + \Delta t L_{\text{FB}} \text{ where } L_{\text{FB}} = \begin{bmatrix} 0 & -\bar{h}\delta_x \\ -g\delta_x & \Delta t g \bar{h} (\delta_x)^2 \end{bmatrix}.$$

- RK3:

$$G = \text{Id} + \Delta t L + \frac{1}{2}(\Delta t L)^2 + \frac{1}{6}(\Delta t L)^3, \tag{25}$$

- RK4:

$$G = \text{Id} + \Delta t L + \frac{1}{2}(\Delta t L)^2 + \frac{1}{6}(\Delta t L)^3 + \frac{1}{24}(\Delta t L)^4, \tag{26}$$

- RK-KG(2,6):

$$G = \text{Id} + \Delta t L + \frac{1}{2}(\Delta t L)^2 + \frac{1}{6}(\Delta t L)^3 + \frac{1}{24}(\Delta t L)^4 + \frac{1}{180}(\Delta t L)^5 + \frac{1}{1080}(\Delta t L)^6, \tag{27}$$

- θ-scheme:

$$G = (\text{Id} - \theta \Delta t L)^{-1} \cdot (\text{Id} + (1-\theta)\Delta t L), \tag{28}$$

- Exponential Integrator:

$$G = \exp(\Delta t L). \tag{29}$$

where the matrix L has been defined by (22). Note that L_{FB} is a small perturbation of L.
A time scheme is stable if and only if $\|X^{n+1}\|_2 \leq \|X^n\|_2$, or equivalently $\text{Sp}(G) \subset \mathcal{D}(0,1)$ in $\mathbb{C}$. In Table 5, we summarize the maximum Courant numbers $c\Delta t/\Delta x$ required to maintain stability using different explicit schemes. The RK-KG(2,6) scheme has the largest stability limit while the RK3 scheme has a stronger stability constraint.
Whatever the time scheme considered, the second order discretization allows for a time step larger than the one for the fourth order discretization, the ratio between the two is about 1/0.86.

Scheme	RK-KG(2,6)	RK4	RK3	FB
C2	$2\sqrt{6} \approx 2.45$	$\sqrt{2} \approx 1.41$	$\frac{\sqrt{3}}{2} \approx 0.86$	1
C4	$\frac{48\sqrt{376742901}}{443749} \approx 2.10$	$48 \cdot \sqrt{\frac{283}{443749}} \approx 1.21$	$12 \cdot \sqrt{\frac{1698}{443749}} \approx 0.74$	$24 \cdot \sqrt{\frac{566}{443749}} \approx 0.86$

TABLE 5. 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. Using explicit methods, it is mainly given by the number of right hand side evaluations of (9). In Table 6, we compute the maximum Courant number divided by the number of right hand side evaluations. The greater the value is, cheaper the method is. The Forward-Backward scheme is the cheapest among the explicit time schemes but also the least accurate.The Runge Kutta integrator RK4 is more accurate and less expensive than RK3. RK-KG(2,6) has the largest computational cost per time iteration but has a good stability. Among the Runge-Kutta integrators considered, it is the least expensive to reach final time. Unfortunately, this integrator is only second order accurate.

Scheme	RK-KG(2,6)	RK4	RK3	FB
C2	$\frac{\sqrt{6}}{6} \approx 0.41$	$\frac{\sqrt{2}}{4} \approx 0.35$	$\frac{\sqrt{3}}{6} \approx 0.29$	1
C4	$\frac{8\sqrt{376742901}}{443749} \approx 0.35$	$12 \cdot \sqrt{\frac{283}{443749}} \approx 0.30$	$4 \cdot \sqrt{\frac{1698}{443749}} \approx 0.25$	$24 \cdot \sqrt{\frac{566}{443749}} \approx 0.86$

TABLE 6. 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.

Dissipation and dispersion are the ability of a numerical scheme to keep unchanged the shape and velocity of a simple wave. It is measured by the dissipation map:

$$(\lambda, \theta) \mapsto |\zeta(\lambda, \theta)|$$

and the dispersion map:

$$(\lambda, \theta) \mapsto \frac{\arg \zeta(\lambda, \theta)}{\lambda\theta}$$

where $\lambda = c\Delta t/\Delta x$ is the Courant number, $\theta = k\Delta x$ is the normalized wave number and $\zeta(\lambda, \theta)$ is the eigenvalue of G associated to frequency k.
A perfect scheme would have each of these functions equal to one, and a good scheme stays close to one. When the dissipation map is larger than 1, the numerical scheme is unstable. A dispersion map far from 1 (at a given wave number) indicates a phase delay or advance.

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 allow 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 disperses (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 the 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 dispersion properties do not depend on the Courant number since the only error comes from the spatial discretization.

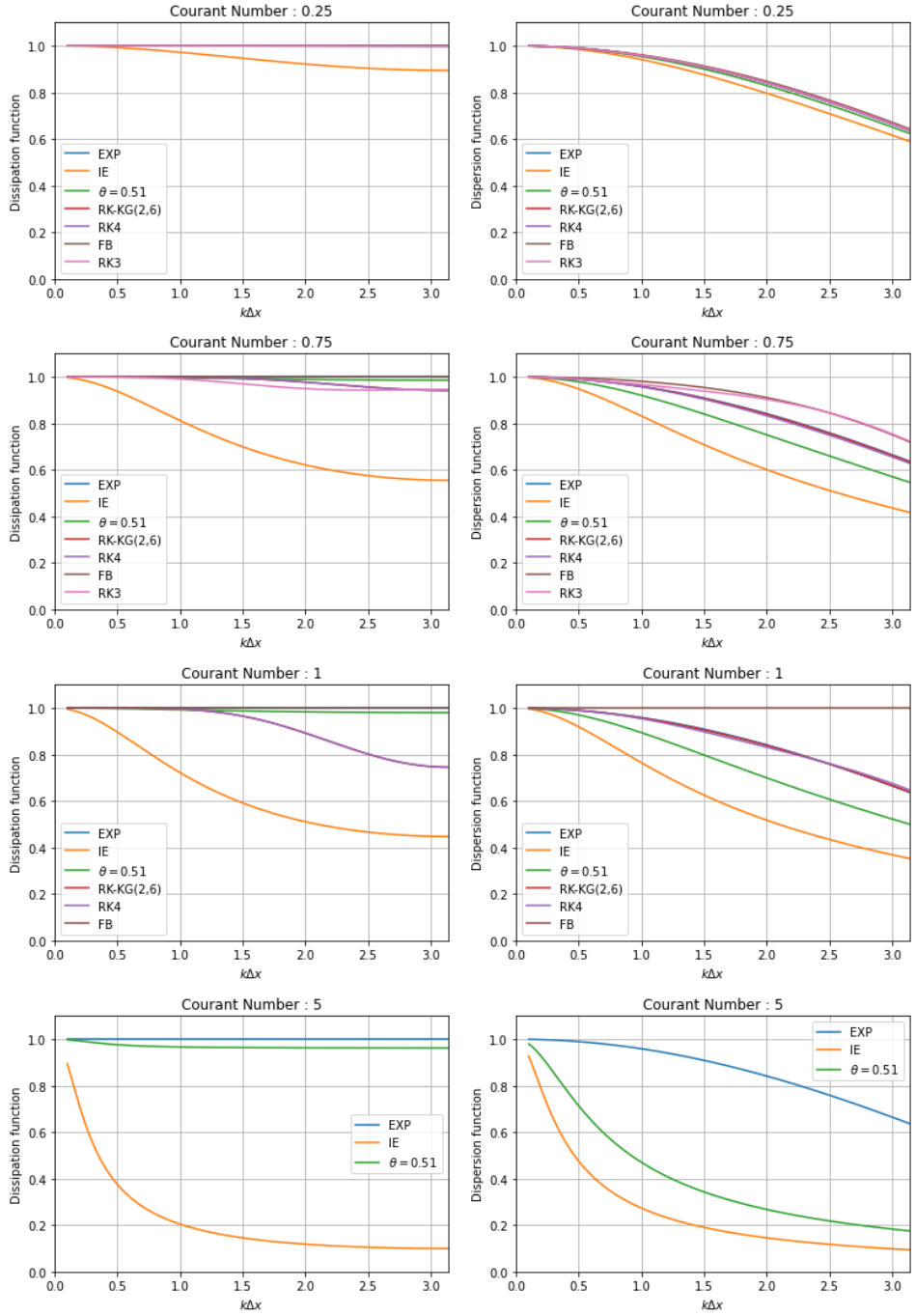

FIGURE 5. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C2 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered: 0.25, 0.75, 1 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

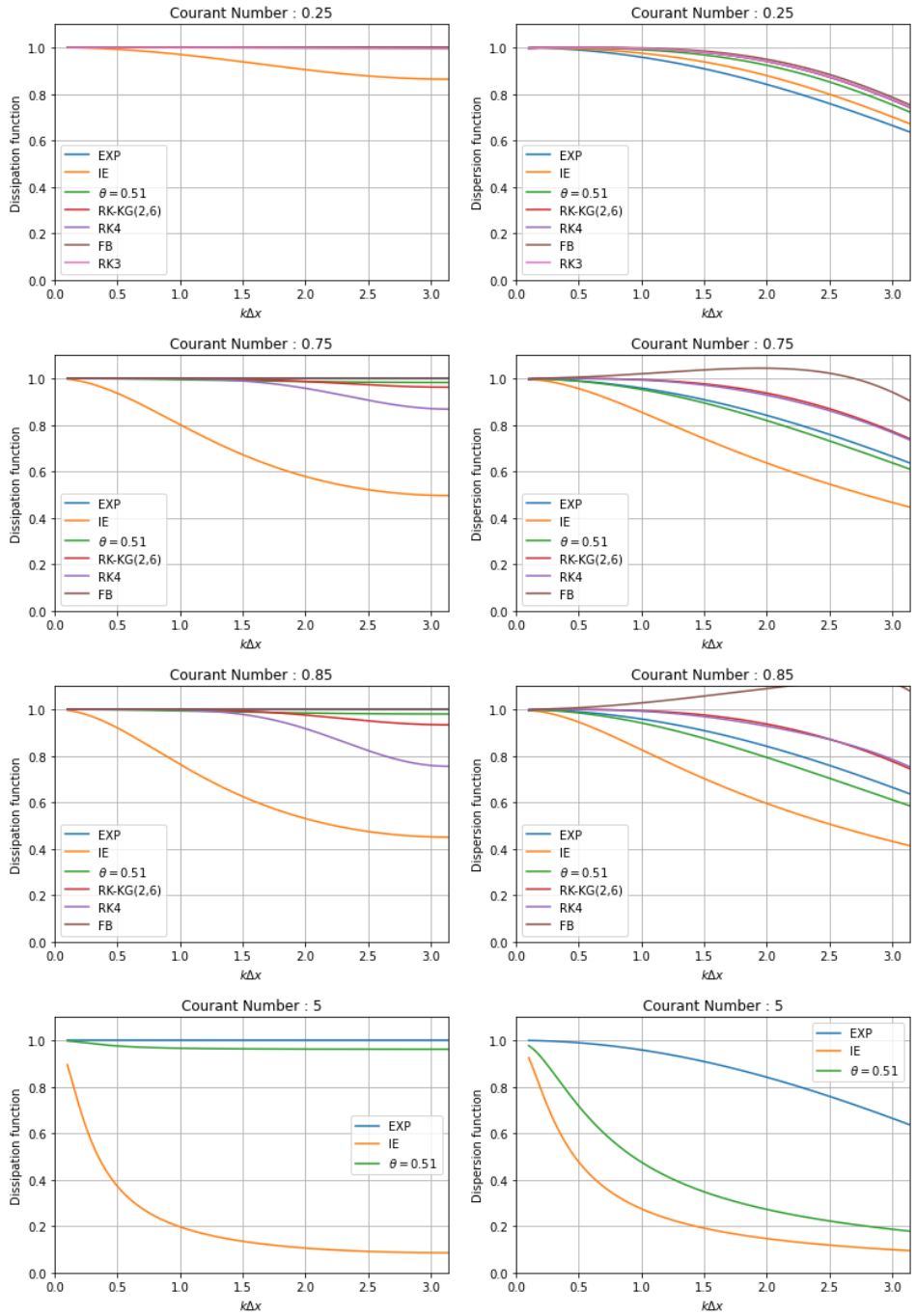

FIGURE 6. Left panel: dissipation map, Right panel: dispersion map using different time integrators and C4 space scheme. Three Courant numbers $c\Delta t/\Delta x$ are considered: 0.25, 0.75, 0.85 and 5. When $c\Delta t/\Delta x = 5$, explicit integrators are not stable.

5. NUMERICAL RESULTS

To assess the performance of each scheme, we consider several test cases for the linearized shallow water equations (9). Both the accuracy and stability are evaluated. As mentioned before, exponential integrators

and implicit time schemes are A-stable allowing the use of large time steps. Implicit methods require the solution of a linear system which is done using a conjugate gradient method (the linear system to solve is reduced to a smaller symmetric system as detailed in A.2).

Consider (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

$$\begin{aligned} h(t,x) =& \frac{1}{2}\sqrt{\frac{\bar{h}}{g}}\left[u_0(x-ct) - u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) + h_0(x+ct))\right] + \dots \\ & \cdots + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) - f(\tau, x+c(t-\tau))\right) d\tau \end{aligned} \tag{30}$$

and the velocity u is

$$\begin{aligned} u(t,x) =& \frac{1}{2}\left[u_0(x-ct) + u_0(x+ct) + \sqrt{\frac{g}{\bar{h}}}(h_0(x-ct) - h_0(x+ct))\right] + \dots \\ & \cdots + \frac{1}{2}\int_0^t \left(f(\tau, x-c(t-\tau)) + f(\tau, x+c(t-\tau))\right) d\tau \end{aligned} \tag{31}$$

with the characteristic velocity $c = \sqrt{g\bar{h}}$.
Three source functions f will be considered:

- The first is $f \equiv 0$, 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 order. 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. LERK and splitting integrators benefit from the fact the source function $\mathcal{N}(t)$ is in the kernel of L. LERK integrators have a reduced computational cost because only one exponential has to be compute at each time step. Splitting methods benefit from error cancellation and become fourth order accurate when used in conjunction with the RK4 scheme.
- In the third test case, more general, f depends both on time and space. Most of the advantages of the preceding simplified cases disappear. Splitting based methods have limited order of accuracy. Additionally, LERK integrators are more expensive 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 gravity $g = 9.81\mathrm{m}\cdot\mathrm{s}^{-2}$. Initial states (h_0, u_0) are given by:

$$\begin{cases} h_0(x) = h_m \cdot \exp\left(-\left(\dfrac{x - 0.5\cdot d}{\sigma}\right)^2\right) \\ u_0(x) = 0 \end{cases} \tag{32}$$

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 in a single iteration.
Two oceanic configurations will be considered: a shallow ocean ($\bar{h} = 100\mathrm{m}$) and a deep ocean ($\bar{h} = 4000\mathrm{m}$). The change of depth impacts the value of the propagation speed c and thus of the Courant number. The ratio between the two thicknesses is $4000/100 = 40$ and results in a ratio of the Courant numbers $\sqrt{4000}/\sqrt{100} \approx 6.32$.

Relative errors are computed at time t^n for h using

$$e_h = \sqrt{\frac{d}{N} \cdot \frac{\frac{1}{N_t}\sum_{n=1}^{N_t} \|\mathfrak{h}^n - h(t^n)\|^2_{L^2([0,d])}}{\|h\|^2_{L^2([0,T_f]\times[0,d])}}} \tag{33}$$

and for u by

$$e_u = \sqrt{\frac{d}{N} \cdot \frac{\frac{1}{N_t}\sum_{n=1}^{N_t} \|\mathfrak{u}^n - u(t^n)\|^2_{L^2([0,d])}}{\|u\|^2_{L^2([0,T_f]\times[0,d])}}} \tag{34}$$

where $\mathfrak{h}$ (resp. $\mathfrak{u}$) is the numerical approximate of h (resp. u). N is the number of spatial grid points chosen to be equal to $N = 500$ in our experiments. It leads to $\Delta x = 1000$ meters. N_t is the number of time iterations to reach the final time $T = N_t \Delta t$.

5.1. **No forcing Case.** Consider (9) without source term. More precisely, f is

$$f(t,x) = 0 \text{ for all } x \in [0,d],\ t \geq 0. \tag{35}$$

Then, (9) is a linear autonomous equation. In this particular case, there are only space discretization errors when exponential integrators are used. That is why it is sufficient to consider ERK1 exponential integrators, indeed, all exponential differencing schemes are equivalent.
At each time $t \geq 0$, the solution is

$$\begin{cases} h(t,x) = \dfrac{1}{2}\left(h_0(x-ct) + h_0(x+ct)\right) \\ u(t,x) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left(u_0(x-ct) - u_0(x+ct)\right) \end{cases} \tag{36}$$

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

Visualisation of dissipation/dispersion errors of the unconditionally stable schemes. In 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). The spatial operator is fourth order accurate. The time step is $\Delta t = 300$s and corresponds to a Courant number $c\Delta t/\Delta x = 9.4$. In this high resolution case, the solution is spatially well resolved and most of the errors come from the time stepping algorithm.

We observe the dissipation of the solution computed with Backward Euler and the dispersion using Crank-Nicolson. The curves associated to θ-scheme ($\theta = 0.51$) show less dispersion error than the Crank-Nicolson 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. Results shown in Figure 7 were obtained with the C4 scheme but similar conclusions are obtained using a second order accurate scheme (C2).

Accuracy / Computational cost. We here compare the error in h and u for the different time schemes, and the associated number of right hand side evaluations. According to experiment done in section 3.2.2, the computational costs are assumed to be directly linked to the number of right hand side evaluations. For exponential integrators and implicit schemes, these calls arise from the matrix-vector products of the Krylov and conjugate gradient methods.
For each explicit scheme, 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 given in Table 7.
For the implicit and exponential integrators, which are unconditionally stable, we consider values given in Table 8, along with the corresponding Courant numbers.
Results are plotted in Figure 8 using second order spatial discretization C2, and in Figure 9 for the fourth

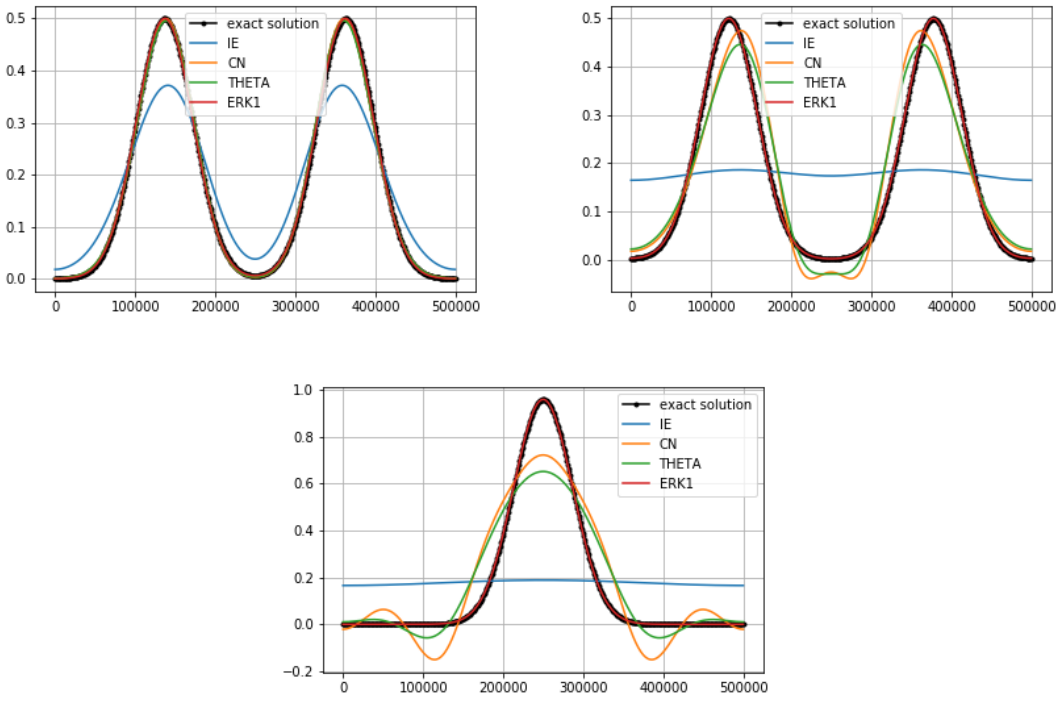

FIGURE 7. Linear test case. Numerical solutions $\mathfrak{h}$ computed using C4 spatial operators and $\bar{h} = 100$ meters. $\Delta t = 300$s ($c\Delta t/\Delta x = 9.4$). Top left panel: solution at $t = 1$h. Top right panel: solution at $t = 10$h. Bottom panel: solution at $t = 40$h.

	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
RK-KG(2,6)	78.2	67.0	12.3	10.6

TABLE 7. Time steps Δt considered for explicit time integrators FB, RK3, RK4 and RK-KG(2,6). These time steps are close to the maximum value allowed to ensure stability.

Δt	100	600	1200	3600	5400
Shallow ocean $c\Delta t/\Delta x$	3.13	18.79	37.58	112.76	169.13
Deep ocean $c\Delta t/\Delta x$	19.81	118.85	273.71	713.12	1069.69

TABLE 8. Time steps Δt used for implicit and exponential integrators and associated Courant numbers $c\Delta t/\Delta x$ in case of a shallow ocean ($\bar{h} = 100$ meters) and a deep ocean ($\bar{h} = 4000$ meters).

order accurate operator C4. What is plotted are the relative errors as a function of the number of right hand side evaluations. Each symbol corresponds to a value obtained with the times steps of Table 8.

As mentioned before, the explicit Forward Backward scheme along with the C2 discretization takes advantage of very good dissipation and dispersion properties when Courant number is close to 1 (see Figure 5). This is not true using C4 operators for which only limited accuracy is achieved. But its cost, in comparison with RK3, RK4 and RK-KG(2,6) schemes, is much lower. The Forward Backward scheme is thus a good

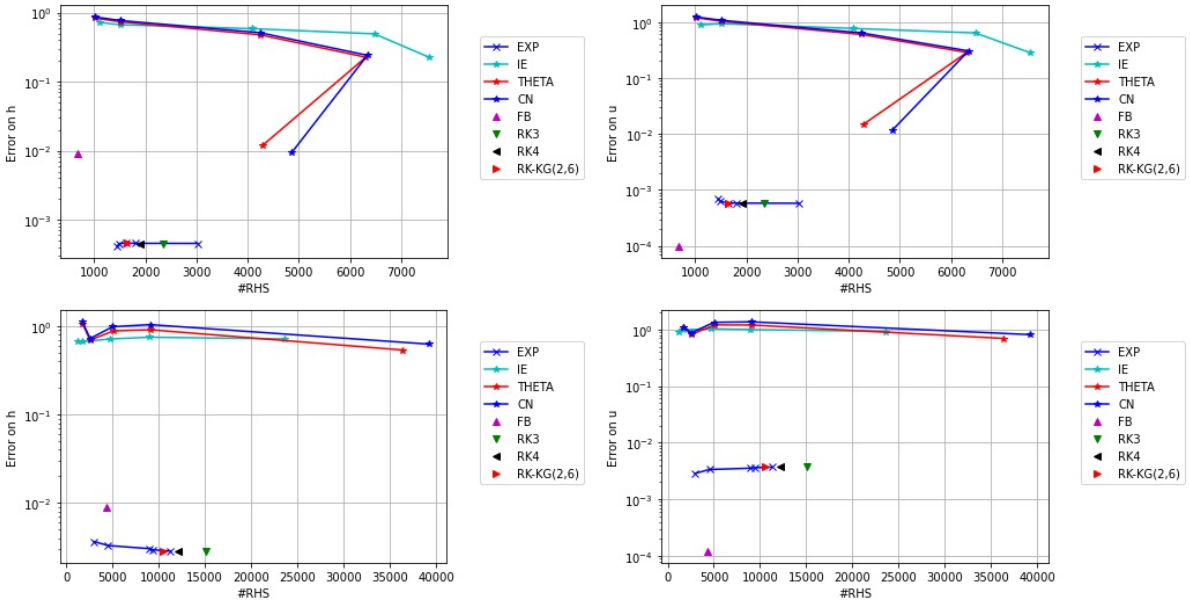

FIGURE 8. Linear test case. Relative error as a function of the number of right hand side (RHS) evaluations (which is itself given by the choice of the time step). The period of integration is $T = 6$ hours. On the top panels, we consider the shallow ocean $\bar{h} = 100$ meters; on the bottom panels, the deep ocean with $\bar{h} = 4000$ meters. The spatial discretization scheme is C2.

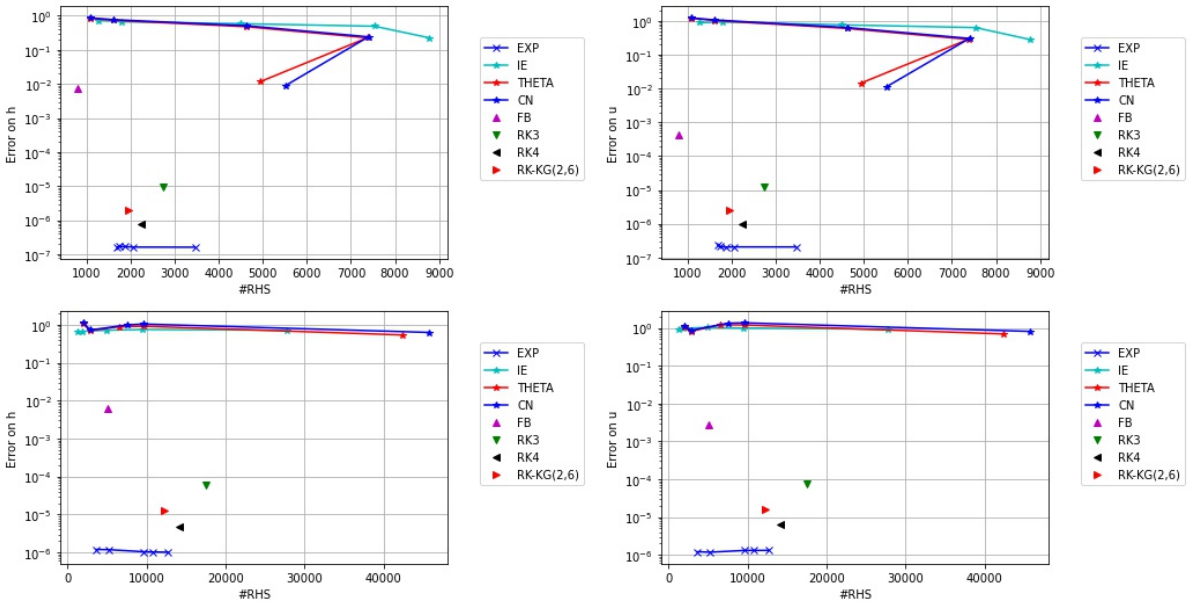

FIGURE 9. Linear test case. Same plots as Figure 8 with C4 spatial operators.

alternative, when compared to any other schemes (either explicit or implicit) when accuracy is not the factor limiting the choice of the time step. RK-KG(2,6) has a computational cost slightly smaller than RK4 while accuracy is similar.

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 EXP 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.

Computational cost. To explore the sensitivity of the ERK1 scheme to the discretization order and to the Courant number, the numbers of RHS evaluations with different ocean thicknesses and spatial operators are given in Table 9. As previously mentioned, we focus on the number of RHS evaluations shown on Table 9 as

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

TABLE 9. Linear test case. Number of RHS evaluations to reach $T = 6$ hours with ERK1 and $\Delta t = 100$s.

an important part of the computational cost.

ERK1 is 1.06 times more expensive using C4 instead of C2 in the shallow ocean case and 1.14 times in the deep ocean case. With the same change (C4 instead of C2), explicit time integration schemes are 1.16 more costly due to change in CFL conditions (see Table 7). Thus, ERK1 is as impacted as explicit time schemes by an increase of the spatial discretization order.
As already mentioned, the deep ocean configuration instead of shallow ocean leads a Courant number approximately ≈ 6.32 times greater than the shallow ocean configuration. Explicit schemes have thus to reduce their time step by a corresponding factor and are 6.32 more expensive than for the shallow ocean case. 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 Courant number, has less impact on the exponential integrators than on the explicit time integration schemes.

For 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 accurate due to their dissipation and dissipation properties.
In the next subsections, we evaluate the impact of having a non zero forcing term.

5.2. **Time dependent forcing test case.** We consider here the time dependent forcing case: f is given by $f(t) = K\sin(\omega t)$, $K = 1 \cdot 10^{-5}\mathrm{m}\cdot\mathrm{s}^{-2}$. It does not depend on x. According to (30) and (31), the exact solution is

$$\begin{cases} h(t,x) = \dfrac{1}{2}\left[h_0(x-ct) + h_0(x+ct)\right] \\ u(t,x) = \dfrac{1}{2}\sqrt{\dfrac{g}{\bar{h}}}\left[h_0(x-ct) - h_0(x+ct)\right] + \dfrac{K}{\omega}\left(1-\cos\omega t\right). \end{cases} \tag{37}$$

Error as a function of the frequency ω. The source term f is T_p-periodic with $T_p = 2\pi/\omega$. In 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 a shallow ocean for the second and fourth order spatial discretizations. $e_h(\omega)$ (resp. $e_u(\omega)$) is given by (33) (resp. (34)). 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 number of RHS of the exponential integrators are smaller or equal to those using RK3 or RK4. Time steps for the explicit time schemes are as in the previous subsection (see Table 7). We specify that subS1ERK4 and subS2ERK4 have $N_s(\Delta t)$ sub-time steps where

$$N_s(\Delta t) = \max\left\{ N \in \mathbb{N}^* \text{ s.t. } \frac{\sqrt{g\bar{h}}\Delta t/N}{\Delta x} \leq K_{\mathrm{RK4}} \right\}, \tag{38}$$

where K_{RK4} is the stability constraint of the RK4 schemes (see Table 5). Then, in this simulation with C2 (resp. C4) and shallow ocean, we have $\Delta t/N_s = 42.86$s, $N_s = 14$ (resp. 37.5s, $N_s = 16$) which is slightly lower than $\Delta t_{\mathrm{RK4}} = 45.1$s (resp. 38.7s).

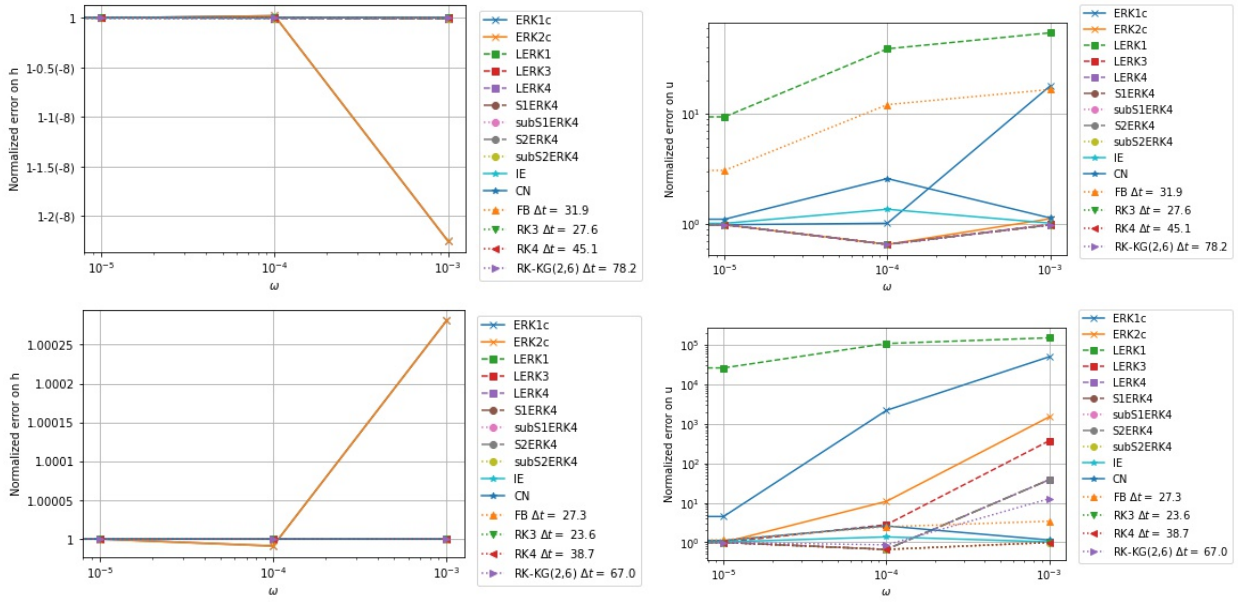

FIGURE 10. Time dependent source term test case. We plot the relative error on h and u using different time schemes and values of ω. Top panel: C2 spatial scheme, Bottom panel: C4 spatial scheme. Parameters are $\bar{h} = 100$ meters, $t_{\max} = 6$ hours. A-stable time schemes are using $\Delta t = 600$ seconds. Explicit time schemes are using time steps close to the maximum allowed (see Table 7).

The error for h does not depend significantly on ω because the source function does not impact the exact $h(x,t)$ solution. Conversely, 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 $\Delta t = 600$s and for different frequencies, are given in Table 10.

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

TABLE 10. 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.

For a frequency less or equal to $\omega = 10^{-4}\mathrm{s}^{-1}$, the number of time steps per period is large (≥ 100). 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.

In Figure 10, S1ERK4, S2ERK4, LERK4 curves are not distinguishable, just as subS1ERK4, suS2ERK4 and RK4 curves. 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 for 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.

As mentioned before, in this specific case where the source term does not depend on space, there is no splitting error. Then, splitting time schemes subS1ERK4 and subS2ERK4 are all fourth order accurate and values $e_h(\omega)/e_h(0)$ and $e_u(\omega)/e_u(0)$ remain small whatever the value ω. They are impacted at the same level as RK4 thanks to good representation of source term f, fourth order accuracy and small time steps.

Implicit time schemes (IE and CN) are not sensitive to the frequencies ω, but this just reinforces the fact that even when ω is zero, the error is already large (see previous paragraph).
Explicit time schemes (FB, RK3, RK4 and RK-KG(2,6)) 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, Figures 11 and 12 show plots of relative errors according 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 case, the time steps are $\Delta t \in \{40, 150, 600, 3600, 5400\}$ while they are $\Delta t \in \{10, 50, 150, 600, 900\}$ for the deep ocean case to leading similar Courant numbers for the two cases.
Results, with the second order C2 scheme, are plotted in Figure 11. Spatial errors are larger than time errors as in the no forcing test case seen in section 5.1. Methods with a low order of accuracy (ERKc and LERK1) are the only ones whose time error is visible. Other exponential integrators (high order LERK and splitting methods), when used with larger time steps Δt, are as accurate and cheaper than RK3 and RK4.
As illustration, in case of deep ocean, 9000 RHS evaluations are necessary for the exponential integrators to produce an error less than 10^{-3} while 17000 RHS evaluations are at necessary for RK3, 14000 for RK4 and 12000 for RK-KG(2,6).
FB is accurate and cheap, but this is again a special case due to the use of C2 space operator.

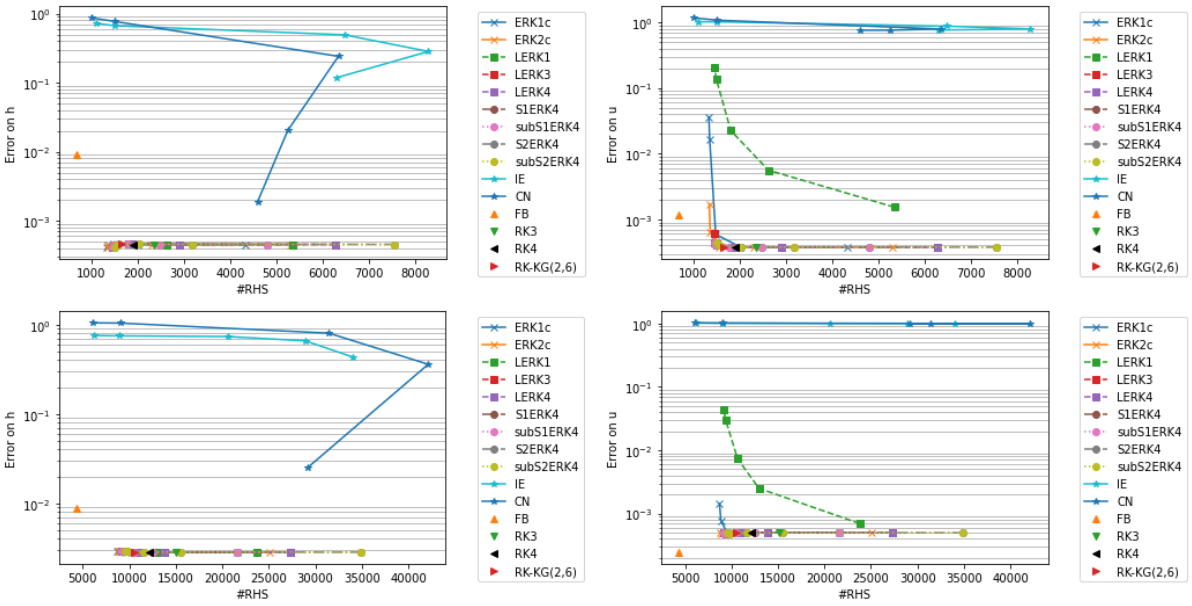

FIGURE 11. Time source term test case. Relative errors related to the number of right hand side called to reach $t = 6$hours. Top panels: shallow ocean ($\bar{h} = 100$ meters) with $\Delta t \in \{40, 150, 600, 3600, 5400\}$ for A-stable schemes. Bottom panels: deep ocean ($\bar{h} = 4000$ meters) with $\Delta t \in \{10, 50, 150, 600, 900\}$ for A-stable schemes. Explicit time steps are given in Table 7. For both, C2 spatial discretization is used.

Figure 12 shows the results obtained with the fourth order accurate C4 scheme.
On the computational cost side, ERK1c, LERK and SERK4, when used with large time steps, are less costly

than Runge-Kutta schemes.
Here, the spatial error is reduced and time errors become visible. In shallow ocean, errors are large except for splitting exponential integrators. When ocean is deep, high order LERK and splitting integrators benefit from their high accuracy and are more accurate than RK4. More precisely, 10000 RHS evaluations are necessary to reach an error smaller than 10^{-5} with high order exponential integrators LERK3, LERK4, SERK4 and subSERK4, 12000 RHS evaluations with ERK1c and 17000 RHS evaluations with ERK2c. To compare, at least 17000 RHS evaluations are needed with RK3 and 14000 with RK4. LERK1 have an error larger than 10^{-5} whatever the time step considered. For both cases (C2 and C4), LERK3 and LERK4 are

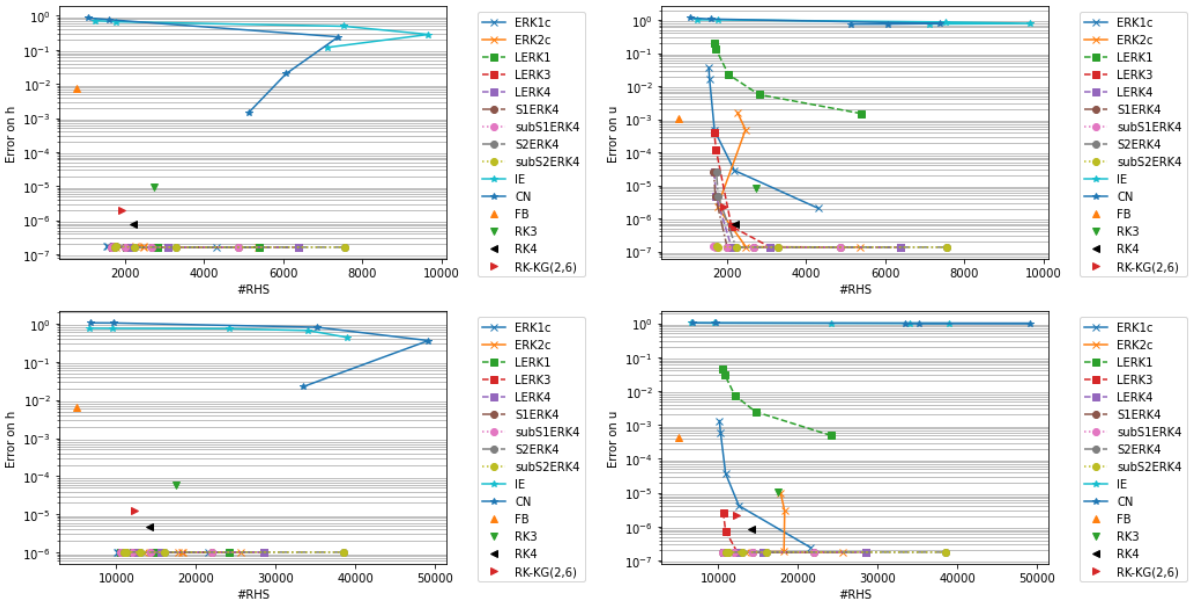

FIGURE 12. Time source term test case. Same than Figure 11 with C4 space operators. Explicit time steps are given in Table 7.

cheap considering the number of matrix exponentials occurring. In addition, splitting errors cancel out. As mentioned in the introduction of this section, this was expected due to the temporal only dependency of the source term.
The next section considers the case of a spatio-temporal varying source term.

5.3. **Time and space dependent forcing test case.** The preceding time dependent only forcing case favored the use of splitting exponential integrators schemes (SERK4 ans subSERK4) and of high order accurate LERK schemes. As we pointed it out, this was a consequence of the particularity 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:

$$f(t,x) = K \sin\omega t \cos kx. \tag{39}$$

At each time $t \geq 0$ and for all $x \in \mathbb{R}$, the exact water height h and velocity u are given by:

$$h(t,x) = \frac{1}{2}\left(h_0(x-ct) + h_0(x+ct)\right) + \sqrt{\frac{\bar{h}}{g}} K \frac{\omega \sin(kct) - ck\sin(\omega t)}{\omega^2 - c^2k^2} \sin(kx), \tag{40}$$

and

$$u(t,x) = \frac{1}{2}\sqrt{\frac{g}{\bar{h}}}\left(h_0(x-ct) - h_0(x+ct)\right) + K\omega \frac{\cos(kct) - \cos(\omega t)}{\omega^2 - c^2k^2} \cos(kx). \tag{41}$$

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

Visualization of h and error. The numerical solution h and the relative error are plotted on Figure 13 using different time schemes and C4 spatial 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 for explicit schemes are $\Delta t_{\text{RK-KG(2,6)}} = 60$s ($c\Delta t/\Delta x \approx 1.88$), $\Delta t_{\text{RK4}} = 30$s ($c\Delta t/\Delta x \approx 0.94$), $\Delta t_{\text{RK3}} = 20$s and $\Delta t_{\text{FB}} = 20$s ($c\Delta t/\Delta x \approx 0.63$). The time steps of explicit time schemes are smaller than in Table 7 but this ensures the final time of integration is exactly the same than that of unconditionnally stable schemes.
The relative errors in Figure 13 are computed using the following formula:

$$\frac{\mathfrak{h}_j^n - h(t^n, x_j)}{\max_j |h(t^n, x_j)|}. \tag{42}$$

As in the previous cases, the accuracy of implicit time schemes is low, either due to dissipation or dispersion

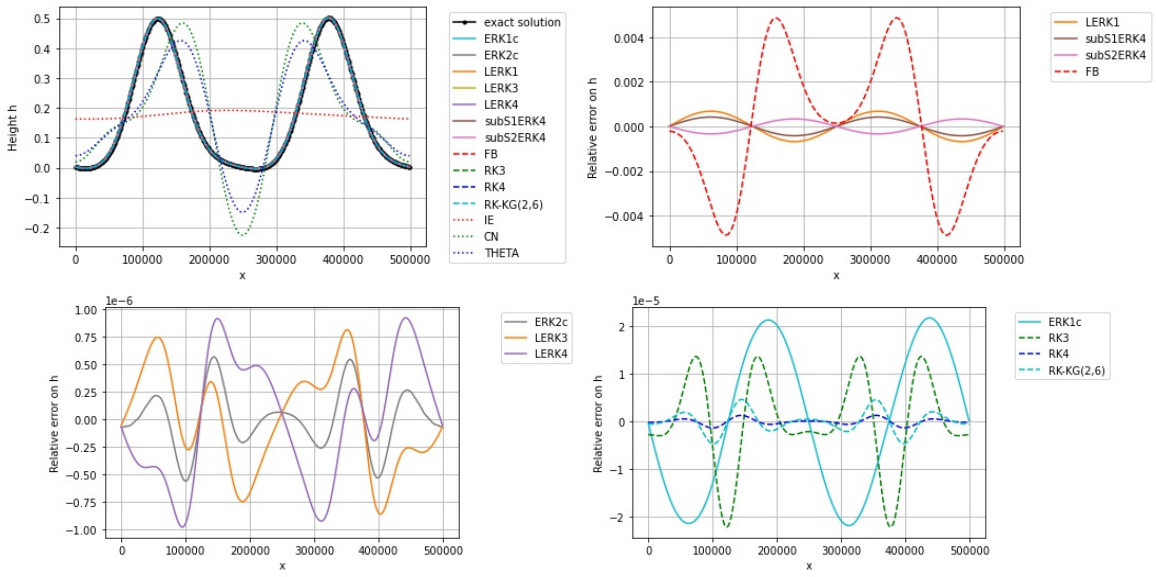

FIGURE 13. Spatio-temporal source term test case. Numerical solution $\mathfrak{h}$ computed using fourth order C4 spatial discretization with $\bar{h} = 100$m (shallow ocean). First plot (top left): solution h at $t = 10$ hours. Three next plots: relative errors on h. A-stable schemes have $\Delta t = 600$s, Explicit schemes time steps $\Delta t_{\text{RK4}} = 30$s, $\Delta t_{\text{FB}} = \Delta t_{\text{RK3}} = 20$s and $\Delta t_{\text{RK-KG(2,6)}} = 60$s.

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 least accurate among the explicit schemes. RK3, RK4 and RK-KG(2,6) have 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 performs better but is the least accurate exponential integrators among LERK3, LERK4 and ERKc.

Evolution of the computational cost with the time steps. In Table 11, we give the number of right hand side evaluations to reach $t = 2$ hours using the C4 spatial discretization scheme and different time schemes. For unconditionally stable schemes, the three columns of Table 11 corresponds to increasing values of the time

	$\mathbf{\Delta t = 300}$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
RK-KG(2,6) ($\Delta t = 67.0$s)		644	
RK4 ($\Delta t = 38.7$s)		744	
RK3 ($\Delta t = 23.6$s)		915	
FB ($\Delta t = 27.3$s)		263	

TABLE 11. Spatio-temporal source term test case. Number of right hand side evaluations to reach $t = 2$ hours using C4 space operators in a shallow ocean. In the first row, the values of the time step used for all the unconditionally stable schemes is indicated.

step $\Delta t = 300$s, $\Delta t = 600$s and $\Delta t = 3600$s. 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 matrix exponentials involved. ERK1c, LERK1 and subS1ERK4 require the computation of one matrix exponential, ERK2c and subS2ERK4 require two matrix exponentials and LERK3 and LERK4 require three matrix exponentials per time step. In Table 11, we see that 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 (and thus taking a smaller number of time steps to reach $t = 2$ hours).

Conversely ERK2c, LERK3 and LERK4 computational costs are reduced by increasing the time step. This decrease in cost seems related to the low computational cost of the first exponential matrices (when we compute K_i, see (70) and (72)) compared to the last exponential. As an illustration, we give in Table 12, the computational cost for each step of LERK3 and LERK4 with C4. In particular, the computations of K_i have approximately the same cost whatever the time step. An explanation for this is the fact that

	LERK3		**LERK4**	
Δt	600	3600	600	3600
K_1	0	0	0	0
K_2	35	48	35	48
K_3	35	48	-	-
K_4	-	-	35	48
X^{n+1}	56	286	56	286

TABLE 12. Spatio-temporal source term test case. Mean value of the number of right hand side evaluations to compute each step to reach $t = 2$ hours using C4 space operators in a shallow ocean.

computing $\exp(-\Delta t L)\mathcal{N}(t)$ is easier than computing $\exp(-\Delta t L)(X^n + \cdots)$ due to the frequencies included in $\mathcal{N}$. Indeed, $\mathcal{N}(t)$ is simply given by a trigonometric function while $X^n + \cdots$ contain frequencies due to the initial functions (h_0, u_0). Then, computing $\exp(-\Delta t L)\mathcal{N}(t)$ is less impacted by changes of Δt than computing the final step. Furthermore, LERK3 and LERK4 have exactly the same computational cost because they

differ only in one exponential matrix when $\mathcal{N}(X,t) = \mathcal{N}(t)$ (see Appendix A.4) and this difference does not impact significantly the convergence of Krylov methods.
When Δt is larger than 600s, ERKc, LERK1, subS1ERK4 and subS2ERK4 are cheaper than explicit Runge Kutta time schemes.

Accuracy/Computational cost. As in the previous test cases, we consider the error evolution according to the number of right hand side evaluations. Errors in h and u are computed with equations (33) and (34). 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 7 for explicit integrators.
The results lead us to the following remarks:

- Since splitting errors do not cancel, we see 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). The S1ERK4 (resp. S2ERK4) scheme is not distinguishable from the subS1ERK4 (resp. subS2ERK4) scheme.
  However, computational costs are the same as those of ERKc time schemes which have the same number of exponential matrices per time step.
- High order linearly exact integrators are more expensive. This is because the right hand side is no longer 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 matrices are required for ERK2c and one for ERK1c, these integrators are among the cheapest. ERKc methods represent a good compromise between computational cost and accuracy (corner bottom left on plots).

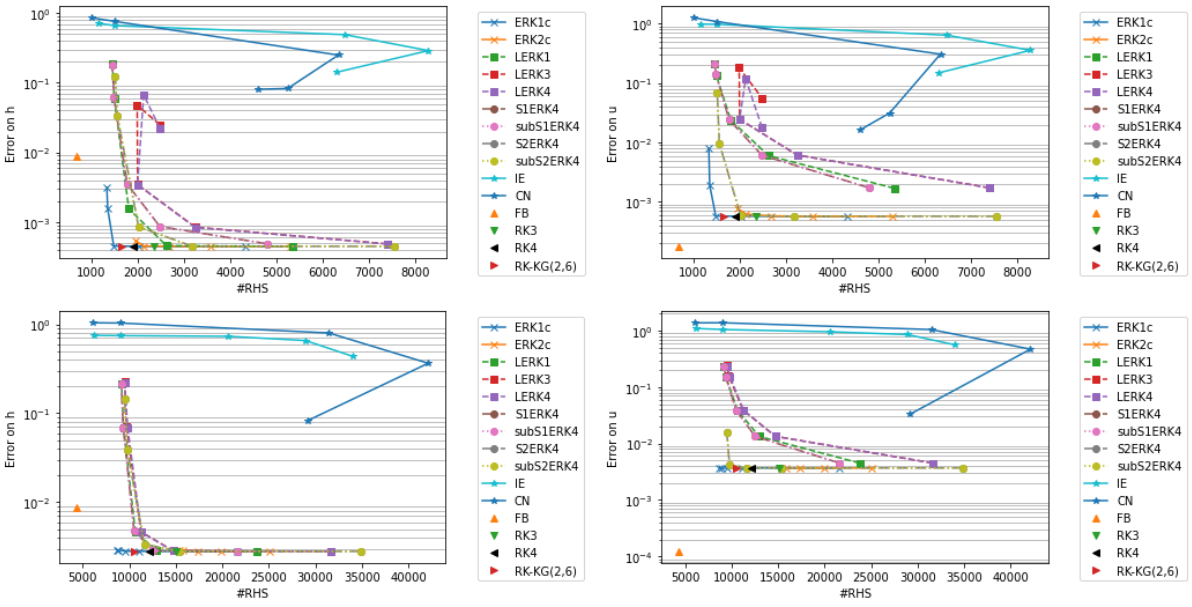

FIGURE 14. Space time source term test case. Same as Figure 11. C2 spatial discretization.

ERK1c, with $\Delta t = 600$seconds, is more accurate and less expensive than RK4. It was the opposite in the time dependant forcing test case. However, the reduction in error is not significant and considering all the test cases RK4 and ERK1c have an error of the same magnitude.
As shown above in Table 11, for larger time steps, ERK1c become even cheaper than RK4, at the price of a reduced accuracy.
In these experiments, ERK1c seems to be a good compromise between accuracy and computational cost.

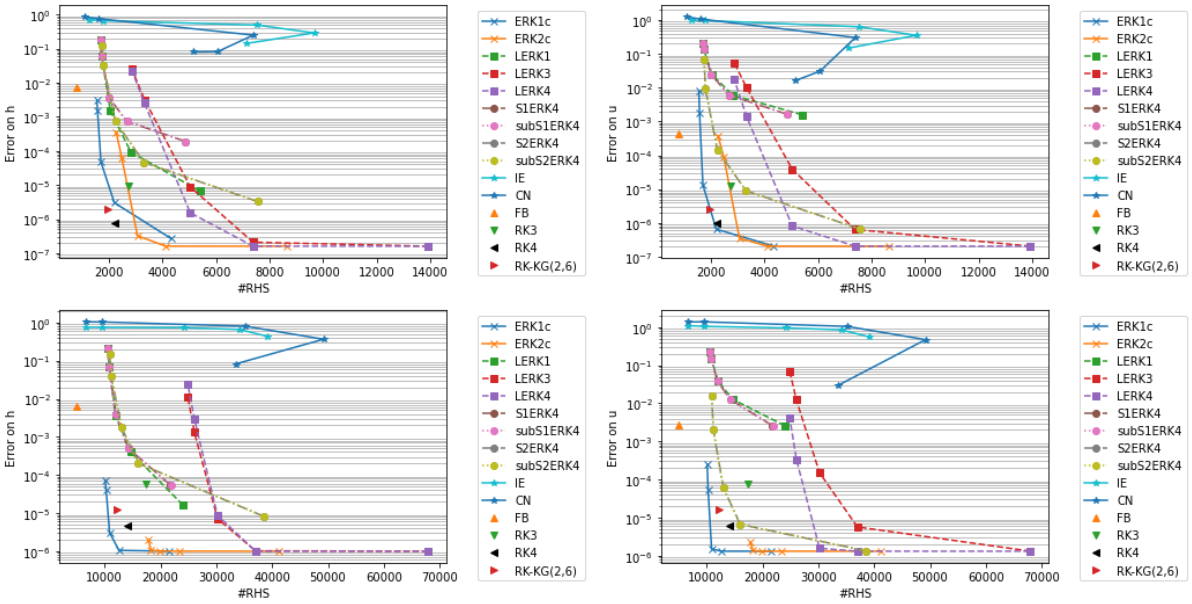

FIGURE 15. Space time source term test case. Same as Figure 12. C4 spatial discretization.

Exponential integrators are more adapted to deep ocean where stability constraints are important. For example, RK4 is a better compromise of accuracy/computational cost than ERK1c with shallow water while it is the contrary for deep ocean.

6. Conclusion

In this paper, we analyzed a number of time integration schemes for the solution of the linearized shallow water equations (9).

- *Explicit time schemes:* Runge-Kutta methods and Forward-Backward scheme. These integrators are cheaper per time step but a large number of time iterations are required to reach final time. A Kinnmark-Gray scheme is considered and benefit from a large stability for an explicit integrator.
- *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 large as desired. Unfortunately, implicit time schemes suffer dramatically from poor dissipation and dispersion properties when used with large Courant numbers.
- *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 spatial discretization was done on a staggered grid either with second order accurate (C2) scheme or with a fourth order accurate (C4) scheme. As it is known, explicit time schemes have more restrictive CFL conditions using C4 than C2. Furthermore, Krylov methods and conjugate gradient converge more slowly when a high order accurate spatial discretization is used. This is due to the larger spectral radius of matrices.

Exponential integrators have better dissipation and dispersion properties related solely to the spatial discretization. These 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.

Various exponential integrators have been analyzed in particular with respect to the spatio-temporal variability of the forcing term on the right hand side of the equations. Among them, high order accurate linearly exact Runge-Kutta integrators are the most accurate. However they are expensive since they involve a large number of matrix exponentials. To this respect, Exponential Runge-Kutta integrators and splitting exponential integrators are cheaper. Splitting exponential integrators suffer splitting error when the source term is space dependent. Splitting exponential integrators have been introduced in order to capture the high temporal variability of the forcing. However when the forcing is also space dependent, splitting errors limit the accuracy of these schemes. Exponential Runge-Kutta integrators are a good compromise between accuracy and computational cost. We have shown that they are able to be as accurate as the explicit RK4 scheme and at a potentially lower cost.

The main difficulty using exponential integrators is to compute efficiently matrix exponentials. That is the equivalent to solving a system for implicit integrators. For them, there exist a large body of methods allowing reduction of the computational cost. Some of them have recently been adapted to exponential integrators: as an example we mention domain decomposition [4, 23], preconditioning [24, 12] and parallel approximation [33]. We hope to use the same kind of methods to reduce the computational cost of exponential integrators for hyperbolic PDEs in future work.

Another possible issue is due to memory storage. The Arnoldi Gram Schmidt algorithm requires storage of $\mathcal{O}((m+1)N)$ values. This is prohibitive when the grid is thin (thus N is large) but it is possible to reduce this value using the incomplete orthogonalization process [32]. The present analysis should be continued with this in mind.

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. 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. A center compact scheme for the shallow water equations on the sphere. *Computers & Fluids*, page 105286, 2022.

[7] S. Calandrini, K. Pieper, and M. D. Gunzburger. Exponential time differencing for the tracer equations appearing in primitive equation ocean models. *Comput. Meth. in App. Mech. and Eng.*, 365:113002, 2020.

[8] M. Caliari, F. Cassini, and F. Zivcovich. Approximation of the matrix exponential for matrices with a skinny field of values. *BIT Numerical Mathematics*, 60(4):1113–1131, 2020.

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

[14] 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.

[15] Stéphane Gaudreault, Greg Rainwater, and Mayya Tokman. Kiops: A fast adaptive krylov subspace solver for exponential integrators. *Journal of Computational Physics*, 372:236–255, 2018.

[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 J. Num. Anal.*, 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. *J. Comput. Phys.*, 228(8):2874–2905, 2009.

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

[22] 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.

[23] 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. *Comput. & Fluids*, 194:104307, 2019.

[24] V. T. Luan, M. Tokman, and G. Rainwater. Preconditioned implicit-exponential integrators (IMEXP) for stiff PDEs. *Jour. Comput. Phys.*, 335:846–864, 2017.

[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. *IMA J. Num. Anal.*, 40(4):2399–2414, 2020.

[29] K. Pieper, K. C. Sockwell, and M. Gunzburger. Exponential time differencing for mimetic multilayer ocean models. *J. Comput. Phys.*, 398:108900, 2019.

[30] A. Ralston. Runge-Kutta methods with minimum error bounds. *Math. of Comput.*, 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] 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.

[37] 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.

Appendix A. Time integrators

Consider the ordinary differential equation

$$\frac{dX}{dt} = \mathcal{F}(X,t). \tag{43}$$

We call autonomous the case such that $\mathcal{F}$ does not depend on time, $\mathcal{F}(X,t) = \mathcal{F}(X)$.
Equation (43) can be split into a linear and a non linear part:

$$\mathcal{F}(X,t) = LX + \mathcal{N}(X,t) \tag{44}$$

where L is a fixed matrix or is adapted at each time step to be $\mathrm{Jac}_{X^n}\mathcal{F}$.
In the context of (9), the function $\mathcal{N}(X,t)$ does not depend on X, more precisely:

$$\frac{dX}{dt} = LX + \mathcal{N}(t). \tag{45}$$

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 defined 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 equations. In this section, we consider two Runge-Kutta methods respectively third and fourth order accurate and also a simple Forward Backward scheme.

A.1.1. *Kinnmark-Gray's second order integrator.* Kinnmark-Gray's integrators are Runge-Kutta schemes for which stability is preferred over accuracy. The following KG integrator is denoted RK-KG(2,6). This integrator was suggested by Andrew Steyer, from Sandia National Laboratories, in a personal communication. It is second order accurate.

$$\begin{cases} K_1 = \mathcal{F}\left(X^n, t^n\right) \\ K_2 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{6}K_1, t^n + \dfrac{\Delta t}{6}\right) \\ K_3 = \mathcal{F}\left(X^n + \dfrac{2\Delta t}{15}K_2, t^n + \dfrac{2\Delta t}{15}\right) \\ K_4 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{4}K_3, t^n + \dfrac{\Delta t}{4}\right) \\ K_5 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{3}K_4, t^n + \dfrac{\Delta t}{3}\right) \\ K_6 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2}K_5, t^n + \dfrac{\Delta t}{2}\right) \\ X^{n+1} = X^n + \Delta t K_6. \end{cases} \tag{46}$$

At each time step, $\mathcal{F}$ is evaluated six times.

A.1.2. *Ralston's third order integrator.* Ralston's third order integrator (see [30]) is a third order accurate explicit Runge-Kutta integrator. We called it RK3 in this article. It is given by

$$\begin{cases} K_1 = \mathcal{F}\left(X^n, t^n\right) \\ K_2 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2}K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{F}\left(X^n + \dfrac{3\Delta t}{4}K_2, t^n + \dfrac{3\Delta t}{4}\right) \\ X^{n+1} = X^n + \dfrac{\Delta t}{9}\left(2K_1 + 3K_2 + 4K_3\right). \end{cases} \tag{47}$$

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

A.1.3. *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:

$$\begin{cases} K_1 = \mathcal{F}\left(X^n, t^n\right) \\ K_2 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{F}\left(X^n + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{F}\left(X^n + \Delta t K_3, t^n + \Delta t\right) \\ X^{n+1} = X^n + \dfrac{\Delta t}{6}\left(K_1 + 2K_2 + 2K_3 + K_4\right). \end{cases} \tag{48}$$

This time scheme is more accurate than RK3 but the right hand side part is evaluated one time more (4 instead of 3).

A.1.4. *Forward-Backward time scheme.* The Forward-Backward (FB) time scheme is a well known time integrator for linearized shallow water equation (9) (see [34] 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

$$\begin{cases} \mathfrak{h}^{n+1}_{i+1/2} = \mathfrak{h}^n_{i+1/2} - \Delta t \bar{h} \delta_x \mathfrak{u}^n_{i+1/2} \\ \mathfrak{u}^{n+1}_i = \mathfrak{u}^n_i - \Delta t g \delta_x \mathfrak{h}^{n+1}_{i+1/2} \end{cases} \tag{49}$$

where δ_x is a finite difference operators for the first order spatial derivative.

A.2. **Implicit Time Scheme.** Implicit time schemes allow us to use greater time steps (without CFL restriction).
An example of one step implicit integrator is the θ-scheme given by

$$\frac{X^{n+1} - X^n}{\Delta t} = \theta \mathcal{F}(X^{n+1}, t^{n+1}) + (1-\theta)\mathcal{F}(X^n, t^n) \tag{50}$$

where $\theta \in [0,1]$. As soon as $\theta \neq 0$, a linear system has to be solved at each time step, this can be done using Newton's algorithm or a linear solver in the case of linear equation.
In the context of (45), the scheme is

$$(\mathrm{Id} - \theta \Delta t L) X^{n+1} = X^n - \Delta t \left(\theta \mathcal{N}(t^{n+1}) + (1-\theta)\mathcal{N}(t^n) + (1-\theta) L X^n\right). \tag{51}$$

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 performed, to add a small dissipation and stay close to CN, we also consider the case $\theta = 0.51$.
The θ-scheme is unconditionally stable as soon as $\theta \geq 1/2$.

If applied to (9) with the finite difference approximation C2 or C4, the system to solve can be rewritten as

$$(\mathrm{Id} - \Delta t^2 g \bar{h} \delta_x^2)(\mathfrak{h}^{n+1} - \mathfrak{h}^n) = -\Delta t \bar{h} \delta_x \mathfrak{u}^n + \theta \Delta t^2 \bar{h} g \delta_x^2 \bar{h}^n - \Delta t^2 \bar{h} \theta \delta_x \bar{F} \tag{52}$$

where $\bar{F}$ is the vector with $\bar{F}_j = \theta f(t^{n+1}, x_j) + (1-\theta) f(t^n, x_j)$ for all $0 \leq j \leq N-1$.
Computing $\mathfrak{h}^{n+1}$, $\mathfrak{u}^{n+1}$ is updated

$$\mathfrak{u}^{n+1} = \mathfrak{u}^n - \Delta t \left(g\theta \delta_x (\mathfrak{h}^{n+1} - \mathfrak{h}^n) + \bar{F} - g \delta_x \mathfrak{h}^n\right). \tag{53}$$

In practice, the system (52) is solved using the conjugate gradient method.

A.3. **Exponential Runge-Kutta Integrators.** Exponential Runge-Kutta Integrators (ERK) are an alternative to implicit schemes to solve stiff problems and keep stability. These allow to overcome the CFL condition as for implicit schemes but with a higher accuracy. We refer to [17] for a review.

We start by considering an autonomous system:

$$\frac{dX}{dt} = \mathcal{F}(X). \tag{54}$$

The function $X : t \mapsto X(t)$ satisfies

$$X(t^n + \Delta t) = \exp(\Delta t L)\, X(t^n) + \int_0^{t^n} \exp((\Delta t - \tau)L)\, \mathcal{N}(X(t^n + \tau))\, d\tau \tag{55}$$

Various quadrature rules for the integral lead to different ERK integrators. They depend on matrix functions of the set $(\varphi_k)_k$ defined by

$$\begin{cases} \varphi_{k+1}(z) = \dfrac{\varphi_k(z) - \varphi_k(0)}{z} \\ \varphi_k(0) = 1/k! \\ \varphi_0(z) = \exp(z). \end{cases} \tag{56}$$

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 (55). It leads to the exponential Euler integrator (ERK1):

$$X^{n+1} = \exp(\Delta t L)\, X^n + \Delta t \varphi_1(\Delta t L)\, \mathcal{N}(X^n) \tag{57}$$

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 when L is fixed once for all. If L contains the full linear part (see Rosenbrock exponential integrators [18]):

$$L = L_n = \mathrm{Jac}_{X^n} \mathcal{F}, \tag{58}$$

it is second order accurate. Here, $\mathrm{Jac}_{X^n}\mathcal{F}$ is the Jacobian matrix and may be required to be updated at each time step. Considering (56), ERK1 is written

$$X^{n+1} = X^n + \Delta t \varphi_1(\Delta t L) \mathcal{F}(X^n). \tag{59}$$

These allow to compute only one matrix function instead of two in (57).

A.3.2. *Second order exponential Runge-Kutta integrator.* A more accurate scheme is ERK2:

$$\begin{cases} a^n = X^n + \Delta t \varphi_1(\Delta t L) \mathcal{F}(X^n) \\ X^{n+1} = X^n + 2\Delta t \varphi_3(\Delta t L)\left(\mathcal{N}(a^n) - \mathcal{N}(X^n)\right). \end{cases} \tag{60}$$

The function φ_3 is

$$\begin{cases} \varphi_3(z) = \dfrac{e^z - 1 - z - z^2/2}{z^3} \text{ if } z \neq 0 \\ \varphi_3(0) = 1/6. \end{cases} \tag{61}$$

Two matrix functions are required in (60). ERK2 is second order accurate in general but if L is given by (58), 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 (59) 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 occur and accuracy is poor.

In [18], a modification of (59) and (60) is described to solve autonomous problems.
ERK1 is modified into

$$X^{n+1}=X^n+\Delta t\mathcal{F}\left(X^n,t^n\right)+\Delta t^2\varphi_2\left(\Delta tL\right)\left[L\mathcal{F}(X^n,t^n)+\frac{\partial\mathcal{F}}{\partial t}(X^n,t^n)\right] \tag{62}$$

with $\varphi_2(z)=(e^z-1-z)/z^2$ and $\varphi_2(0)=1/2$.
The higher order version (60) becomes

$$\begin{cases} a^n=X^n+\Delta t\mathcal{F}\left(X^n,t^n\right)+\Delta t^2\varphi_2\left(\Delta tL\right)\left[L\mathcal{F}(X^n,t^n)+\frac{\partial\mathcal{F}}{\partial t}(X^n,t^n)\right] \\ X^{n+1}=a^n+2\Delta t\varphi_3(\Delta tL)\left[\mathcal{F}(a^n,t^n+\Delta t)-\mathcal{F}(X^n,t^n)-L(a^n-X^n)-\Delta t\frac{\partial\mathcal{F}}{\partial t}(X^n,t^n)\right]. \end{cases} \tag{63}$$

We denote these schemes ERK1c and ERK2c.
Time integrators (62) and (63) 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 (58)), 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 described first in [21]. 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.

Consider the change of variable

$$V(t)=\exp((t^n-t)L)X(t) \tag{64}$$

in equation (45). Then, V is solution of

$$V'(t)=\exp\left((t^n-t)L\right)\mathcal{N}\left(\exp\left((t-t^n)L\right)V(t),t\right) \tag{65}$$

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 (65). If $L=0$, LERK coincide with an explicit time scheme. If $\mathcal{N}(t)$ is in the kernel of L then $\exp(\alpha L)\mathcal{N}(t)$ can be computed thanks to a single iteration of a Krylov method:

$$\exp(\alpha L)\mathcal{N}(t)=\mathcal{N}(t)\in\mathcal{K}_1\left(L,\mathcal{N}(t)\right). \tag{66}$$

where $\mathcal{K}_1$ is the Krylov subspace defined in 3. This significantly reduces the computational cost of these 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 the Forward Euler scheme. The Forward Euler scheme is not stable when applied directly on (9) but the exponential part stabilizes it. It is given by the formula

$$X^{n+1}=\exp\left(\Delta tL\right)\left[X^n+\Delta t\mathcal{N}(X^n,t^n)\right]. \tag{67}$$

This scheme is called LERK1. In the case of (45), it is written:

$$X^{n+1}=\exp\left(\Delta tL\right)\left[X^n+\Delta t\mathcal{N}(t^n)\right]. \tag{68}$$

One exponential is required per time step.

A.4.2. *Linearly Exact RK3.* Applying the RK3 time scheme (47) on (65), we obtain a third order LERK integrator:

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_1\right), t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{3\Delta t}{4}L\right)\mathcal{N}\left(\exp\left(\frac{3\Delta t}{4}L\right)\left(X^n + \frac{\Delta t}{2}K_2\right), t^n + \frac{3\Delta t}{4}\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{9}(2K_1 + 3K_2 + 4K_3)\right]. \end{cases} \tag{69}$$

This scheme will be called LERK3 in the following. It contains five matrix exponentials.
In the case of (45) where $\mathcal{N}(X,t) = \mathcal{N}(t)$, LERK3 takes the following form:

$$\begin{cases} K_1 = \mathcal{N}(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{3\Delta t}{4}L\right)\mathcal{N}\left(t^n + \frac{3\Delta t}{4}\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{9}(2K_1 + 3K_2 + 4K_3)\right]. \end{cases} \tag{70}$$

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

A.4.3. *Linearly Exact RK4.* Considering fourth order Runge-Kutta (48) applied to (65), we obtain a fourth order accurate LERK time stepping. This scheme will be called LERK4 and is given by:

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_1\right), t^n + \frac{\Delta t}{2}\right) \\ K_3 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(\exp\left(\frac{\Delta t}{2}L\right)\left(X^n + \frac{\Delta t}{2}K_2\right), t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\mathcal{N}\left(\exp\left(\Delta t L\right)\left(X^n + \Delta t K_3\right), t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{71}$$

In this time integrator, seven matrix exponentials are called at each time step. It is the most expensive time integrator considered here. In the case of (45) where $\mathcal{N}(X,t) = \mathcal{N}(t)$, LERK4 is less expensive and is simplified into:

$$\begin{cases} K_1 = \mathcal{N}(t^n) \\ K_2 = \exp\left(-\frac{\Delta t}{2}L\right)\mathcal{N}\left(t^n + \frac{\Delta t}{2}\right) \\ K_4 = \exp\left(-\Delta t L\right)\mathcal{N}\left(t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\Delta t L\right)\left[X^n + \frac{\Delta t}{6}(K_1 + 4K_2 + K_4)\right] \end{cases} \tag{72}$$

because $K_2 = K_3$. LERK4 is a fourth order scheme.

A.5. **Splitting Exponential Integrators.** A possibility to reduce the computational cost is to consider a time splitting scheme. By separating the linear part and non linear part, we consider two equations to be solved: a linear equation (we denote by $\mathcal{S}_{l,\Delta t}$ the solver) which can be solved 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 and potentially with a smaller time step in order to increase the accuracy.
Stability is then ensured by the exponential part (exact in time) and accuracy is ensured by the explicit scheme. By using this method a splitting error occurs. The choice of the splitting is thus important (see [19] and reference therein).
Two splittings are considered here: Lie and Strang. We use a RK4 scheme for the non linear part and exponential integrator for the linear part.

A.5.1. *Lie splitting.* Lie splitting is based on first solving the non linear equation and using its solution as an initial condition for the linear equation. The method can be represented by $\mathcal{S}_{l,\Delta t} \circ \mathcal{S}_{nl,\Delta t}$. The scheme is then :

$$\begin{cases} K_1 = \mathcal{N}(X^n, t^n) \\ K_2 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_1, t^n + \dfrac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^n + \dfrac{\Delta t}{2} K_2, t^n + \dfrac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^n + \Delta t K_3, t^n + \Delta t\right) \\ X^* = X^n + \dfrac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ X^{n+1} = \exp\left(\Delta t L\right) X^*. \end{cases} \tag{73}$$

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. We will denote this time scheme S1ERK4.
In the particular case where the non linear term in (44) depends only on time, the splitting error cancels and the scheme is 4-th order accurate.

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

$$\mathcal{S}_{l,\Delta t} \circ \left(\mathcal{S}_{nl,\Delta t/N_s}\right)^{N_s}. \tag{74}$$

The number of substeps N_s is chosen to lead to a time step satisfying the RK4 stability constraint:

$$N_s = \max\left\{ N^\star \in \mathbb{N} \text{ s.t. } \frac{c\Delta t/N}{\Delta x} \leq K_{\mathrm{RK4}} \right\} \tag{75}$$

where K_{RK4} is the RK4 stability constraint (see Table 5).
The time integrator subS1ERK4 suffers from splitting error as S1ERK4. It is first order accurate. Again, it is fourth order accurate if $\mathcal{N}(X, t)$ depends only on time. A more accurate scheme is based on Strang Splitting.

A.5.2. *Strang splitting.* Strang splitting writes $\mathcal{S}_{l,\Delta t/2} \circ \mathcal{S}_{nl,\Delta t} \circ \mathcal{S}_{l,\Delta t/2}$. The corresponding algorithm is

$$\begin{cases} X^* = \exp\left(\frac{\Delta t}{2} L\right) X^n \\ K_1 = \mathcal{N}(X^*, t^n) \\ K_2 = \mathcal{N}\left(X^* + \frac{\Delta t}{2} K_1, t^n + \frac{\Delta t}{2}\right) \\ K_3 = \mathcal{N}\left(X^* + \frac{\Delta t}{2} K_2, t^n + \frac{\Delta t}{2}\right) \\ K_4 = \mathcal{N}\left(X^* + \Delta t K_3, t^n + \Delta t\right) \\ X^{n+1} = \exp\left(\frac{\Delta t}{2} L\right) \left[X^* + \frac{\Delta t}{6}(K_1 + 2K_2 + 2K_3 + K_4)\right]. \end{cases} \tag{76}$$

We call this scheme S2ERK4. It is second order accurate and two matrix exponentials are required at each time step.

As with 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

$$\mathcal{S}_{l,\Delta t/2} \circ \left(\mathcal{S}_{nl,\Delta t/N_s}\right)^{N_s} \circ \mathcal{S}_{l,\Delta t/2} \tag{77}$$

As in Lie splitting, the time scheme are 4-th order accurate when non linear part satisfies $\mathcal{N}(X,t) \in \mathrm{Ker}(L)$ (for example when $\mathcal{N}(X,t)$ depends only on time). It is second order accurate in general. The number N_s of substeps is chosen using (75).

Lab. de Mathématiques et Applications, Univ. Poitiers, CNRS, UMR 7348, F86073 Poitiers, France.
Email address: matthieu.brachet@math.univ-poitiers.fr

Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.
Email address: laurent.debreu@inria.fr

Org 01446 (Computational Science), Sandia National Laboratories
Email address: celdred@sandia.gov