High-order locally A-stable implicit schemes for linear ODEs

Hélène Barucq, Marc Duruflé, Mamadou N'Diaye

Noname manuscript No.
(will be inserted by the editor)

High-order locally A-stable implicit schemes for linear ODEs

Hélène Barucq · Marc Duruflé · Mamadou N'Diaye

Received: date / Accepted: date

Abstract Accurate simulations of wave propagation in complex media like Earth subsurface can be performed with a reasonable computational burden by using hybrid meshes stuffing fine and coarse cells. Locally implicit time discretizations are then of great interest. They indeed allow using unconditionally stable schemes in the regions of computational domain covered by small cells. The receivable values of the time step are then increased which reduces the computational costs while limiting the dispersion effects. In this work we construct a method that combines optimized explicit schemes and implicit schemes to form locally implicit schemes for linear ODEs, including in particular semi-discretized wave problems that are considered herein for numerical experiments. Both the explicit and implicit schemes used are one-step methods constructed using their stability function. The stability function of the explicit schemes are computed by maximizing the time step that can be chosen. The implicit schemes used are unconditionally stable and do not necessary require the same number of stages as the explicit schemes. The performance assessment we provide shows a very good level of accuracy for locally implicit schemes. It also shows that a locally implicit scheme is a good compromise between purely explicit and purely implicit schemes in terms of computational time and memory usage.

Keywords Time integration · hybrid discontinuous Galerkin method · hyperbolic problems · wave equations

Hélène Barucq
INRIA Magique-3D, E2S UPPA, CNRS, Avenue de l'Université, 64012 Pau, France.
E-mail: helene.barucq@inria.fr

Marc Duruflé
Bordeaux INP, Institut de Mathématiques de Bordeaux UMR 5251, Bordeaux, France.
INRIA Magique-3D, E2S UPPA, CNRS, 200 Avenue de la Vieille Tour, 33405 Talence, France.
E-mail: marc.durufle@inria.fr

Mamadou N'diaye
Stanford University, School of Earth, Energy & Environmental Sciences, Department of Energy Resources Engineering, 367 Panama Mall, 94305 Stanford, CA-USA.
Previously at INRIA Magique-3D, E2S UPPA, CNRS, Pau, France.
E-mail: mndiaye@stanford.edu / ndiaye.mamadou24@gmail.com

1 Introduction

The interest of using high order finite elements to simulate wave problems has been demonstrated in many works as for example in [1,2,3,4,5]. When the propagation medium is complex, it may be necessary to refine the mesh locally to accurately track the discontinuities in the medium, whether geometric or constitutive. In this case, it might be necessary to locally couple high order elements with low order elements. The question then arises of choosing the most suitable time scheme so as not to destroy the quality of the approximation in space. Explicit time schemes [6,7] are very popular as a low-cost in memory and highly scalable integration process. However, their stability is only guaranteed if the so-called Courant-Friedrichs-Lewy (CFL) condition is met. This condition sets the time step as a function of the smallest mesh step through a constant called CFL number (see e.g. [6,8]). The value of the CFL number is determined by the schemes used for space and time discretization. This unnecessarily increases the computational costs if the CFL constant tends to be small. There is some work that seeks to construct time integration schemes with maximal CFL number as in [9,10,11,12,13,14]. In those works, the obtained schemes are called optimal in the sense that they allow the use of a large time step compared to the classical explicit schemes. However, even if there is an interesting gain in computational time, it is not completely satisfactory, especially when the ratio between small and large cells of the mesh is low. For this reason, the idea of developing schemes with an adaptive time step based on a local CFL number has been explored by several authors, like in [15,16,17,18]. Locally explicit time stepping methods allow using different time steps which are employed after the computational domain has been divided into different zones distinguishing themselves from the mean size of the mesh elements. This process allows to take small time steps precisely where the smallest elements are located and reduces by this way the overall computational time. However, it is difficult to implement with high order schemes, which certainly explains why most developments provide only second order schemes. In addition, implementation difficulties have been reported when the mesh is irregular with cells having sizes that differ by several orders of magnitude.

Another option is to prefer an implicit integration scheme that tends to have better stability property compared to explicit scheme (see [8,7,19]). Some are actually unconditionally stable like those presented in [19,20] in the sense that their stability is guaranteed without any constraint on the time step. In those cases, the time step has an upper bound only to ensure good accuracy. However, to solve large 3D problems, such as seismic wave problems, the use of implicit schemes can be problematic. Indeed, the implementation of an implicit scheme requires solving a linear system at each iteration, and in some cases the limits available in memory can be reached quickly (see e.g. [21]).

A compromise can be found when we can take advantage of a good stability property of implicit schemes and the less memory consuming process of explicit schemes. For instance, when a computational mesh contains few small cells, an interesting approach is to couple an unconditionally stable implicit scheme with an explicit scheme, the latter being reserved for areas paved with large cells. In this way, we would avoid applying a small time step on the whole area as it would be the case with a global explicit scheme while making better use

of memory compared to global implicit integration. There are several works in the literature devoted to the development of coupled implicit and explicit time integration schemes. They can be grouped into two classes of schemes: Implicit Explicit Schemes (IMEX) and locally implicit schemes [17,22] which are most often limited to order two for implementation difficulties, as already mentioned above for locally explicit schemes. IMEX schemes, originally based on operator splitting, have been widely used for the time integration of spatially discretized partial differential equations (PDEs) of diffusion-convection type where an implicit scheme is used for the diffusion term while the convection term is handled with an explicit scheme (see [23,24,25] which are among the first works devoted to this type of scheme). Regarding wave problems, there are few references on the construction of IMEX schemes. This can be explained by the fact that these time integration methods were originally developed to integrate operator-governed systems consisting of a stiff and a non-stiff term. On solving wave problems, the motivation for having such schemes is expressed when the propagation domain has geometrical characteristics including large scale variations. We then speak of geometry induced-stiffness or scale-separation stiffness (see [26]). In this case, Discontinuous Galerkin (DG) discretization methods are of great interest because they are well suitable for hp-adaptivity. A DG method coupled with IMEX time integration scheme is well described in [26], which is followed by many others works until a recent one presented in [27].
If we get back to the case of the wave problems, a second-order locally implicit schemes have been recently developed for Maxwells equation in [28]. It is shown in [29] that this procedure does not change the second-order convergence of the obtained scheme. Based on the work in [22], an error analysis of the locally implicit time integration scheme is proposed in [30,31] for the linear Maxwell's equations. In those works, the second order locally implicit methods are built using the Crank-Nicolson scheme (equivalent to the second order Padé scheme for linear ODE [32]) and the Runge-Kutta (RK) implicit mid-point rule (see [8]). With a DG method and the developed locally implicit schemes, they prove the stability of the fully discrete Maxwell's system under a CFL condition and get convergence results of order two as well.

The main objective of this paper is to provide higher order locally implicit schemes for linear ODEs, including linear wave problems. To this end, following [16] and [28], we propose a method to construct locally implicit time integration schemes with an arbitrary order of accuracy for linear ODEs. The general rules for their development share the same ideas as in [16] except that we use implicit schemes in the region covered with a refined mesh. The locally implicit methods we propose in this work are more flexible than IMEX schemes in the sense that they allow to combine implicit and explicit schemes with different internal stages. However, their extension to non-linear ODEs $y'(t) = f(t, y)$ seem difficult since the linearity of the ODE is used in many places while constructing the schemes. We validate the developed scheme by solving the acoustic wave equation with a hybridizable discontinuous Galerkin (HDG) formulation [21,33, 34].

The remaining of this paper is structured as follows. First we give a general setting of the construction of time integration schemes using a stability function. Then we introduce unconditionally implicit and optimized explicit schemes which form the basis of the locally implicit schemes we propose. In section 3, we

present the methodology we use to develop high-order locally implicit schemes and briefly introduce the additive RK schemes [35] for comparison. In the follow up section, we discuss the splitting technique used to divide the computational domain into coarse and fine regions. In section 6, numerical results for both 2-D and 3-D cases are displayed. We also study the time and space-time convergence of the locally implicit schemes and illustrate the order reduction phenomenon [38] when using the implicit LSDIRK (Linear Singly Diagonally Implicit Runge Kutta) schemes [32].

2 Construction of numerical time integration schemes from a stability function

2.1 Stability function of a time scheme

In this section, we settle preliminary statements of time integration schemes for linear ODEs that we consider in this paper.

Let $\Omega \subset \mathbb{R}^n$ be an open set representing the computational domain and $y(t) \in \Omega$, $t \in [0,T]$, $T \in \mathbb{R}^+$ be the solution of the following ODE

$$\begin{cases} M_h \dfrac{dy(t)}{dt} + K_h y(t) = F(t), \\ y(0) = y_0. \end{cases} \tag{1}$$

The initial problem (1) is a standard one obtained after spatial discretization of a partial differential equation (PDE), where M_h is the mass matrix and K_h is the stiffness matrix. h denotes the mesh size, $F(t)$ is a source term obtained after discretizing the continuous source term in space and y_0 is the initial condition. In Chapter 3 of [21], it is detailed how this ODE is obtained in the case of wave equations. Let $t_0 < t_1 < \cdots < t_{N-1} < t_N$, $N \in \mathbb{N}$ be a uniform grid of the time interval $[0,T]$:

$$t_n = n\Delta t,$$

where Δt is the time step. The analytical solution to (1) is given by

$$y(t_{n+1}) = \mathrm{e}^{\Delta t A}\left(y(t_n) + \int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u\right), \tag{2}$$

where $A = -M_h^{-1}K_h$.

The numerical solution can then be constructed by approximating the exponential, i.e. find R such that

$$\mathrm{e}^{\Delta t A} \approx R(\Delta t A). \tag{3}$$

Herein R is a rational function where both the numerator and denominator are polynomials of $\Delta t A$. The numerical process considered here aims at computing a sequence X_n, which is an accurate approximation of the analytical solution $X(t_n)$. The corresponding numerical scheme reads as:

$$y_{n+1} = R(\Delta t A) y_n + \widetilde{\phi}_n, \tag{4}$$

where $\widetilde{\phi}_n$ is given as an approximation of the following quantity:

$$\widetilde{\phi}_n \approx R(\Delta t A) \int_0^{\Delta t} \mathrm{e}^{-uA} M_h^{-1} F(n\Delta t + u)\,\mathrm{d}u.$$

The function R in (3) is the stability function of the corresponding numerical scheme.

Implicit schemes are known to have better stability properties than explicit formulations [7,8,20]. In a previous work [32], we have developed high-order A-stable implicit one-step time integration schemes. The construction of those schemes is based on the definition of their stability function which has the following form:

$$R(z) = \frac{N(z)}{D(z)},\ \forall z \in \mathbb{C}^-,$$

where $N(z)$ and $D(z)$ are polynomials of z. The corresponding numerical scheme is implicit when the degree of the polynomial function D is greater than one otherwise the numerical scheme is explicit. Herein, we propose to construct locally implicit schemes involving those implicit unconditionally stable schemes, namely Padé and LSDIRK schemes. In the next section, we briefly introduce the construction of optimized CFL number explicit schemes used in this paper.

2.2 Optimized explicit time schemes for linear ODEs

Many efforts have been made on the construction of higher-order optimal CFL number explicit schemes. In [10,11], the authors propose the modified equation technique to obtain high-order time schemes with an optimal CFL number for linear second order ODEs (i.e. ODEs of the type $y'' = f(y)$ where f is linear). In the proceedings [14], we propose optimized high-order explicit Runge-Kutta (ERK) Nyström methods for non-linear second-order ODEs. Other works addressed the optimization of the stability domain of ERK schemes [8,13,9,12]. All those schemes are constructed for both linear and non-linear ODEs. The number of non-linear conditions to be satisfied to obtain the ERK coefficients [7] becomes very large, especially for higher orders. For this reason, most optimized ERK schemes available in the literature are at most of order 4. In this sub-section, we present a technique to build high-order optimized ERK schemes for linear ODEs. The construction is based on the computation of a stability function that leads to a maximal CFL number for the corresponding scheme.

Let us denote by $R_s^\ell(z)$, $s, \ell \in \mathbb{N}$, the stability function of a $m = s + \ell$ stages Linear-ERK schemes. It is defined by:

$$R_s^\ell(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^s}{s!} + \alpha_{s+1} z^{s+1} + \cdots + \alpha_{s+\ell} z^{s+\ell}, \tag{5}$$

where $\alpha_i \in \mathbb{R}$, $i = s+1, \cdots, s+\ell$ are free parameters. The optimized Linear-ERK scheme with $m = s + \ell$ stages (denoted as ERK $s - \ell$) is given as

$$X_{n+1} = R_s^\ell(\Delta t A) X_n. \tag{6}$$

We note that this explicit scheme has an order s by definition (for a linear ODE with $F(t) = 0$).

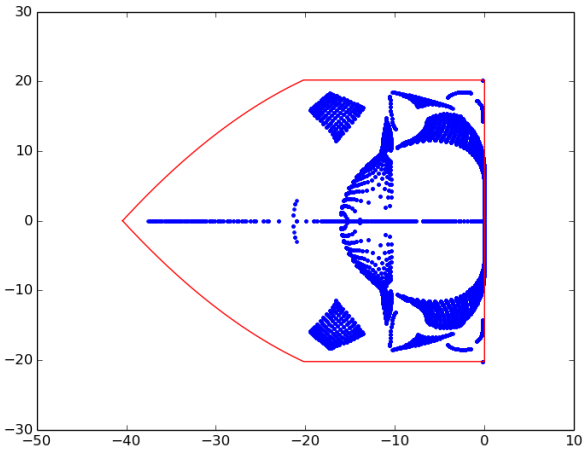


Fig. 1: Spectrum of $A = -M_h^{-1}K_h$ (blue points are eigenvalues of A) in the complex plane included in a set $\max_{\lambda\in sp(A)|Imag(\lambda)|} \times Cabane$ (inside the red polygon) obtained when solving the acoustic wave equation with HDG formulation and $\mathbb{Q}_3$ polynomials. The computational domain is a square $[-4,4]^2$ of 15×15 elements. Neumann condition is set on the boundary of the square.

The parameters $\alpha_i \in \mathbb{R}$, $i = s+1, \cdots, s+l$ can be chosen such that they maximize the CFL number with the methodology described in [13]. The first step of this methodology consists in providing the spectrum $sp(A)$ of the linear operator $A = -M_h^{-1}K_h$ obtained from (1). The linear operator depends on the equation, the physical environment, the mesh, the chosen FEM formulation (which can be continuous or discontinuous), etc. Herein, we have chosen to conduct the optimization for a HDG formulation for the acoustic wave equation, see [33] and the second chapter in [21]). In Figure 1, we display the spectrum obtained for a 2-D regular mesh made of quadrilateral elements covering $[-4,4]^2$ and $\mathbb{Q}_3$ HDG approximation (with external Neumann boundary conditions). This is the typical spectrum obtained for a mesh containing elements of the same size. From this spectrum, we have defined a set denoted by $Cabane$, that includes this typical spectrum (see Figure 1, the normalized $Cabane$ has to be multiplied by $\max_{\lambda\in sp(A)} |Imag(\lambda)|$ to include the considered spectrum where $Imag$ denotes the imaginary part of a complex number).

The upper part of the normalized envelope $Cabane$ is defined as the junction of three curves in the complex plane

$$Cabane^+ := \{it,\ t \in [0,1]\} \oplus \{i - t,\ t \in [0,1]\} \oplus \left\{ t - 2 + i\frac{t}{10}(14 - 4t),\ t \in [0,1] \right\}. \tag{7}$$

The lower part of $Cabane$ is obtained by symmetry with respect to the real axis. The optimization problem then reads

$$\underset{\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell}}{\arg\max} \quad \mathbf{cfl}(\alpha_{s+1}, \alpha_{s+2}, \cdots, \alpha_{s+\ell}),$$

where cfl is the CFL number defined as

$$\text{cfl}(\alpha_{s+1},\cdots,\alpha_{s+\ell}) = \max\{\Delta t \text{ such that } \Delta t \times Cabane \subset \mathcal{S}(\alpha_{s+1},\cdots,\alpha_{s+\ell})\},$$

with $\mathcal{S}(\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell})$ defining the stability region of the corresponding Linear-ERK scheme which is given by:

$$\mathcal{S}(\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell}) = \{z\in\mathbb{C},\ |R_s^\ell(z)|\leq 1\}.$$

If the spectrum $sp(A)$ has exactly the same shape as the one defined by $Cabane$, the final time step should satisfy

$$\Delta t \leq \frac{\text{cfl}(\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell})}{\max_{\lambda\in sp(A)}|Imag(\lambda)|}, \tag{8}$$

to get a stable solution. Unfortunately, this is not usually the case and the maximal time step is given by

$$\Delta t = \max\left\{\Delta t \text{ such that } \lambda\Delta t\in\mathcal{S}(\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell})\ \forall\lambda\in sp(A)\right\}. \tag{9}$$

Note that in [13] the authors use the true spectrum of the operator and look for the maximal time step Δt that satisfies the stability constraint. After optimization the obtained value Δt represents the maximal time step such that $z=\lambda\Delta t\in\mathcal{S}$ for all $\lambda\in sp(A)$. Here, we consider a normalized envelope $Cabane$, and for the operator A the maximal time step is obtained with (9). The stability domains of the different schemes of order 4 and 8 are shown in Figure 2. We show in Figure 2(b) and 2(d) the stability region taking into account the number of stages for each scheme. For this reason we have chosen $Real(z)/(s+l)$ and $Imag(z)/(s+l)$ in the x and y coordinate respectively. We observe that by increasing the number of additional stages l, the stability region tends to the desired shape $Cabane$. The optimal coefficients and CFL numbers obtained for Linear ERK schemes of order 2, 4, 6 and 8 are given in Chapter 7 of [21]. We define the efficiency of the scheme as follows:

$$\texttt{Efficiency} = \frac{\text{cfl}(\alpha_{s+1},\alpha_{s+2},\cdots,\alpha_{s+\ell})}{s+\ell}. \tag{10}$$

When the number of additional stages ℓ is increased, the CFL increases as well, but the scheme is more costly since it involves more matrix-vector products. That's why the ratio `Efficiency` measures correctly the actual efficiency of the scheme. In table 1, we have reported the efficiency obtained for optimal coefficients and different values of ℓ. The efficiency is equal to 0 for ERK 2-0 and ERK

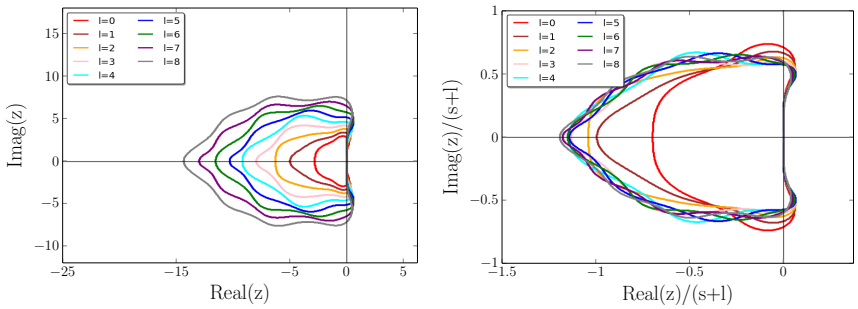
ℓ $(s=2)$	0	2	4
`Efficiency`	0 %	56.2 %	59.6 %

ℓ $(s=4)$	0	2	4
`Efficiency`	34.8 %	52.1 %	57.2 %

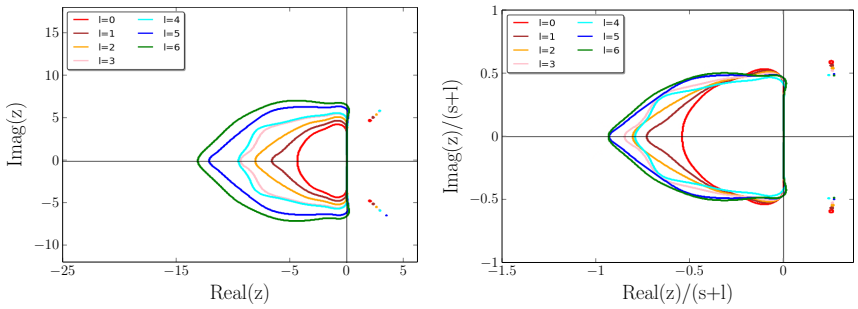
ℓ $(s=6)$	0	2	4
`Efficiency`	0 %	36.1 %	35.6 %

ℓ $(s=8)$	0	2	4
`Efficiency`	26.9 %	39.7 %	45.1 %

Table 1: Efficiency obtained for ERK schemes of order 2, 4, 6, and 8.



(a) Stability region (b) Stability region with the complexity



(c) Stability region (d) Stability region with the complexity

Fig. 2: Stability region of the Linear-ERK schemes of order 4 (top) and 8 (bottom) for different values of ℓ.

6-0 because the stability domain does not include any part of the imaginary axis (except zero). We can see that the efficiency can be significantly increased with only two additional stages. We have observed that low-order schemes (order 2 and 4) have a better efficiency but they are limited by the accuracy. Higher-order schemes (order 6 and 8) provide very accurate results but are limited by the CFL. The evaluation of optimized schemes (6) can be made with Hörner's algorithm in a way that they only require a storage of three vectors of size N. As a result, these schemes are both interesting in terms of computation time and memory usage.

3 Construction of locally implicit time schemes

For the construction of locally implicit time schemes, we have adapted the procedure described in [16] in order to use the optimized explicit schemes introduced in the previous section. For notation convenience we write the first equation of (1) as

$$y'(t) = Ay(t) + F(t), \tag{11}$$

where $A = -M_h^{-1} K_h$ is a linear operator and $y'(t)$ is the time derivative of the unknown y. We split the unknown as follows:

$$y(t) = y^{(c)}(t) + y^{(f)}(t),$$

with

$$\begin{aligned} y^{(c)} &= (I - P)y(t), \\ y^{(f)} &= Py(t), \end{aligned}$$

where $y^{(c)}(t)$ corresponds to the solution in the coarse part of the mesh and $y^{(f)}(t)$ corresponds to the solution in the fine part. P is a diagonal matrix with entries $P_{i,i} = 1$ if i corresponds to a degree of freedom in the fine area of the mesh and 0 elsewhere. After integration of (11) over the interval $[t_n, t_n + \xi\Delta t]$ where ξ is real between 0 and 1, we obtain the following expression of $y(t_n + \xi\Delta t)$:

$$\begin{aligned} y(t_n + \xi\Delta t) = y(t_n) + \underbrace{\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt + \overbrace{\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt}^{\text{source term}}}_{\text{Coarse part}} \\ + \underbrace{\int_{t_n}^{t_n+\xi\Delta t} APy(t)dt + \overbrace{\int_{t_n}^{t_n+\xi\Delta t} PF(t)dt}^{\text{source term}}}_{\text{Fine part}}. \end{aligned} \tag{13}$$

In the coarse part we apply the optimized explicit schemes of 2.2 to integrate the ODE while A-stable implicit schemes developed in [32] are used in the fine part of the mesh.
In practice, we also use the implicit scheme to integrate the ODE in all elements that are adjacent to the fine region. In fact, after splitting the computational domain into a fine and coarse region following a procedure described in the section 4, we first define close degrees of freedom (dofs) as all the dofs in the fine region and its adjacent elements. Then, we define far dofs as the dofs belonging to the remaining elements of the coarse region. The reason of this extra level of splitting is due to the fact that the locally implicit schemes require solving a linear system in part of the domain which is not the case for local time stepping. However, with the HDG formulation for the acoustic wave equation we are using, we just need to solve a linear system for dofs that are in the fine region. These details are explained in the sub-section 3.2 and the appendix A.

3.1 Treatment of the coarse part

To treat the coarse part we first approximate the integral term containing $y(t)$ using a s-point quadrature method. Let c_i be the quadrature points and b_i the corresponding weights. We obtain

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{i=1}^{s} b_i\, y(t_n + c_i\xi\Delta t).$$

We replace $y(t_n + c_i\xi\Delta t)$ by its Taylor expansion of order $r-1$ to get

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{i=1}^{s} b_i \sum_{j=0}^{r-1} \frac{(c_i\xi\Delta t)^j}{j!} y^{(j)}(t_n).$$

Then, we require that the quadrature method is of order $r-1$ at least. So we can use

$$\sum_{i=1}^{s} b_i c_i^j = \frac{1}{j+1}, \quad j = 0, \cdots, r-1$$

we then obtain

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} y^{(j)}(t_n).$$

To handle the derivative of y in the previous sum, we differentiate the ODE (11) $r-1$ times and replace it into the equation to obtain

$$\begin{aligned}&\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \\ &\approx \xi\Delta t A(I-P)\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!}\left(A^j y(t_n) + \sum_{\ell=1}^{j} A^{j-\ell}F^{(\ell-1)}(t_n)\right).\end{aligned}$$

From here, to optimize the CFL number of the explicit scheme, we can add extra terms beyond the order $r-1$ term of the Taylor expansion without changing the order of accuracy. In this paper, we use the following expression for the stability function $R(z)$ of the chosen explicit scheme ($m+1 = s+\ell$):

$$R(z) = \alpha_0 + \alpha_1 z + \alpha_r z^r + \cdots + \alpha_{m+1} z^{m+1},\ m \geq r.$$

In order to coincide with explicit optimized schemes, we add extra-terms to obtain:

$$\begin{aligned}&\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \approx \xi\Delta t A(I-P)\left(\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!} A^j y(t_n) + \sum_{j=r}^{m} \alpha_{j+1}(\xi\Delta t)^j A^j y(t_n)\right) \\ &+ \xi\Delta t A(I-P)\left(\sum_{j=0}^{r-1} \frac{(\xi\Delta t)^j}{(j+1)!}\sum_{\ell=1}^{j} A^{j-\ell}F^{(\ell-1)}(t_n) + \sum_{j=r}^{m} \alpha_{j+1}(\xi\Delta t)^j \sum_{\ell=1}^{j} A^{j-\ell}F^{(\ell-1)}(t_n)\right);\end{aligned}$$

that can be written as

$$\begin{aligned}\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \ \approx\ & \xi\Delta t A(I-P)\left(\sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j A^j y(t_n)\right) \\ &+ \xi\Delta t A(I-P)\left(\sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j \sum_{\ell=1}^{j} A^{j-\ell}F^{(\ell-1)}(t_n)\right),\end{aligned}$$

since

$$\alpha_j = \frac{1}{j!},\ j \leq r.$$

This is due to the fact that the chosen explicit scheme is of order r. We then replace F by its interpolation with quadrature points c_i (denoted by Q):

$$F(t_n + \xi\Delta t) \ \approx \ Q(t_n + \xi\Delta t).$$

Then the interpolation polynomial Q reads

$$Q(t_n + \xi\Delta t) = \widetilde{Q}(\xi) = \sum_{i=1}^{s} F_i\, \widetilde{\varphi}_i(\xi), \tag{14}$$

with

$$F_i = F(t_n + c_i\Delta t),$$

and the basis functions $\widetilde{\varphi}_i$ valued between 0 and 1 are given by

$$\widetilde{\varphi}_i(\xi) = \frac{\prod_{j\neq i} \xi - c_j}{\prod_{j\neq i} c_i - c_j}.$$

We introduce the following notation

$$\widetilde{w}_j \ = \ A^j y(t_n) + \sum_{\ell=1}^{j} A^{j-\ell} Q^{(\ell-1)}(t_n),$$

to end up with a simplified expression

$$\int_{t_n}^{t_n+\xi\Delta t} A(I-P)y(t)dt \ \approx \ \xi\Delta t A(I-P)\left(\sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^j \widetilde{w}_j\right). \tag{15}$$

We then use the following relation

$$Q^{(\ell)}(t_n) = \frac{1}{\Delta t^\ell}\, \widetilde{Q}^{(\ell)}(0).$$

to compute $Q^{(\ell-1)}(t_n)$ as a linear combination of F_i:

$$Q^{(\ell-1)}(t_n) = \sum_{i=1}^{s} \frac{\widetilde{\varphi}_i^{(\ell-1)}(0)}{(\Delta t)^{\ell-1}} F_i.$$

The coefficients

$$D_{i,\ell} = \frac{\widetilde{\varphi}_i^{(\ell)}(0)}{(\Delta t)^\ell},$$

can be pre-computed prior to time iterations. In practice, we will compute and store the vectors

$$\zeta_j = \alpha_{j+1} A(I-P)\widetilde{w}_j. \tag{16}$$

An optimal way to do that is provided by Algorithm 1.

Algorithm 1 Computation of ζ_j (16)

Coefficients $D_{i,\ell}$ are pre-computed as
$D_{i,\ell} = \dfrac{\widetilde{\varphi}_i^{(\ell)}(0)}{(\Delta t)^\ell}$
for $i = 1 \dots s$ **do**
  compute $F_i = F(t_n + c_i \Delta t)$
end for
$w = y_n$
for $j = 0 \dots m$ **do**
  compute $z = A\,(I - P)\,w$ and $z_p = A\,P\,w$
  $\zeta_j = \alpha_{j+1}\, z$
  compute $Q^{(j)} = \sum_{i=1}^{s} D_{i,j} F_i$
  $w = z + z_p + Q^{(j)}$
end for

Now we treat the second-integral in (13) corresponding to the source term in the coarse part. It is approximated as follows:

$$\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt \approx (I-P)\xi\Delta t \sum_{i=1}^{s} b_i Q(t_n + c_i \xi \Delta t).$$

For degrees of freedom (dofs) that are far from the fine region (i.e. dofs belonging to elements that are not adjacent to or inside the fine part of the mesh), the corresponding row of AP is zero, and $P_{i,i} = 0$. As a result, the contribution of fine part in (13) can be dropped. The vector y_{n+1} (with $\xi = 1$) is computed as follows for these degrees of freedom:

$$y_{n+1} = y_n + \Delta t A(I-P) \sum_{j=0}^{m} \alpha_{j+1} \Delta t^j \widetilde{w}_j + \Delta t \sum_{i=1}^{s} b_i F_i$$

or equivalently

$$y_{n+1} = y_n + \Delta t \sum_{j=0}^{m} \Delta t^j \zeta_j + \Delta t \sum_{i=1}^{s} b_i F_i. \tag{17}$$

We see that this expression is a linear combination of vectors F_i and ζ_j computed in Algorithm 1.

Remark 1 If the fine part is void (i.e. $P = 0$), the obtained explicit scheme can be written in the usual form given in [21]:

$$y_{n+1} = \sum_{j=0}^{m+1} \alpha_j \Delta t^j A^j y_n + \Delta t \sum_{k=0}^{m} \Delta t^k A^k \sum_{i=1}^{s} \omega_i^k F_i,$$

where

$$\omega_i^k = \begin{cases} \displaystyle\sum_{\ell=1}^{m+1-k} \alpha_{k+\ell}\, \widetilde{\varphi_i}^{(\ell-1)}(0), \text{ if } k > 0 \\ b_i, \text{ if } k = 0 \end{cases}.$$

The result is obtained by introducing $k = j + 1 - \ell$.

Remark 2 In formula (17), we can observe that the coarse part is advanced by using Hörner's algorithm (instead of the stable algorithm given in [21]). This algorithm is necessary to advance the fine part efficiently. That is the main reason why we preferred the optimized explicit schemes with a small number of additional stages, such that the Hörner's algorithm should not deteriorate the accuracy.

3.2 Treatment of the coarse part combined with the fine part

We recall that Q (14) denotes the polynomial that interpolates the source function F. We introduce $\hat{Q}$ as the anti-derivative of Q. Then the integral of the source term in the coarse region (that will be in contact with the fine region) can be approximated as follows:

$$\int_{t_n}^{t_n+\xi\Delta t} (I-P)F(t)dt \approx (I-P)\left(\hat{Q}(t_n+\xi\Delta t) - \hat{Q}(t_n)\right). \tag{18}$$

Using the equation (18) and the approximation (15) in the coarse region, the equation (13) becomes:

$$\begin{aligned} y(t_n+\xi\Delta t) \approx y_n &+ A(I-P)\sum_{j=0}^{m} \alpha_{j+1}(\xi\Delta t)^{j+1}\widetilde{w}_j \\ &+ (I-P)\left(\hat{Q}(t_n+\xi\Delta t) - \hat{Q}(t_n)\right) \\ &+ \int_{t_n}^{t_n+\xi\Delta t} APy(t) + PF(t)dt \end{aligned}$$

We introduce the variable $\tau = \xi\Delta t$ to obtain

$$\begin{aligned} y(t_n+\tau) = y_n + A(I-P)\sum_{j=0}^{m} \alpha_{j+1}\,\tau^{j+1}\widetilde{w}_j + (I-P)\left(\hat{Q}(t_n+\tau) - \hat{Q}(t_n)\right) \\ + \int_{t_n}^{t_n+\tau} APy(t) + PF(t)dt. \end{aligned} \tag{19}$$

Let $\widetilde{y}$ be defined as

$$\widetilde{y}(\tau) = y(t_n+\tau).$$

Like in [16], we differentiate (19) with respect to τ to obtain the following ODE:

$$\frac{d\widetilde{y}(\tau)}{d\tau} = \overbrace{A(I-P)\sum_{j=0}^{m}(j+1)\alpha_{j+1}\,\tau^{j}\widetilde{w}_j + (I-P)Q(t_n+\tau) + PF(t_n+\tau)}^{\text{updated source term}} + AP\widetilde{y}(\tau). \tag{20}$$

The equation (20) represents the ODE that has to be solved in the fine region with an updated source term. This updated source term is known since its comes from the explicit scheme used in the coarse region and the source term set in the

fine region. We have made the choice of approximating F with Q in this term as well. With this approach, we obtain the following ODE:

$$\frac{d\widetilde{y}(\tau)}{d\tau} = AP\widetilde{y}(\tau) + \widetilde{F}(\tau), \tag{21}$$

where

$$\widetilde{F}(\tau) = \sum_{j=0}^{m}(j+1)\,\tau^j\,\zeta_j + Q(t_n+\tau) = \sum_{j=0}^{m}(j+1)\,\tau^j\,\zeta_j + \sum_{i=1}^{s}\widetilde{\varphi}_i\left(\frac{\tau}{\Delta t}\right)F_i. \tag{22}$$

The ODE (21) can be solved using either explicit schemes with small time step as in [16] or implicit schemes as we propose here. We use A-stable implicit schemes developed in [32] to solve (21), to obtain locally implicit schemes. By this way, the most stiff part of the computational domain becomes a CFL-free region since we use A-stable time integration schemes. This allows increasing the admissible values of time step which contributes to reduce the computational costs.
As mentioned above, in practice we define a close region that comprises the fine region and its adjacent elements. The adjacent elements to the fine part are elements that share an edge (face in 3-D) with the fine region. The dofs associated with elements in the close region are called close degrees of freedom (close dofs) whereas the dofs of other elements are named far degrees of freedom (far dofs). The close dofs correspond to non-null rows of operator AP, meaning the ODE (21) is solved implicitly for all the close dofs not only the dofs of the fine region. However, since we use a HDG formulation, the linear system to be solved involves only the dofs associated with a unknown multiplier λ for edges (faces in 3-D) of the fine region only. Then the unknowns u and v (for the acoustic wave equation (25)) are reconstructed in the close region (fine region and adjacent elements), as explained in appendix A.
In Algorithm 2 we present a sequence to compute y_{n+1} from y_n using the locally implicit method. It is worth noting that the tasks 1 and 2 in this algorithm can be performed in parallel although the cost of these two tasks is very different. In fact, the task 1 consists of a linear combination as shown in sub-section 3.1 while task 2 involves the solution of several linear systems.

Algorithm 2 Computation of y_{n+1}

compute vectors F_i and ζ_j with algorithm 1
Task 1 : compute y_{n+1} for far degrees of freedom with formula (17)
Task 2 : compute y_{n+1} for close degrees of freedom by solving the ODE (21) with an implicit scheme

Remark 3 The ODE (21) holds true for far degrees of freedom. For these degrees of freedom, it reduces to $d\widetilde{y}(\tau)/d\tau = \widetilde{F}(\tau)$ since the corresponding rows of AP are null. This ODE can be integrated exactly, and we obtain the expression (17) for y_{n+1}.

3.3 Comparison with ARK schemes

IMEX Runge-Kutta schemes can be applied to the following splitting of the linear ODE (11) :

$$Ay(t) + F(t) = \underbrace{A(I-P)y(t) + (I-P)F(t)}_{\text{Coarse part}} + \underbrace{APy(t) + PF(t)}_{\text{Fine part}}.$$

Let us denote

$$f_E(t, y(t)) = A(I-P)y(t) + (I-P)F(t), \quad f_I(t, y(t)) = APy(t) + PF(t).$$

Additive Runge-Kutta (ARK) methods can be written:

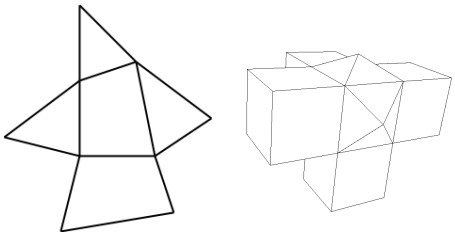
$$\begin{cases} u_i = y_n + \Delta t \sum_{j=1}^{i-1} a^E_{i,j} f_E(t_n + c_i\Delta t, u_j) + \Delta t \sum_{j=1}^{i} a^I_{i,j} f_I(t_n + c_i\Delta t, u_j), \\ \\ y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i \left(Au_i + F(t_n + c_i\Delta t)\right). \end{cases} \tag{23}$$

For coefficients $a^I_{i,j}, a^E_{i,j}, b_i, c_i$, we use the values given in [35]. The scheme (23) is implemented in practice with algorithm 3. In this algorithm, $\gamma = a^I_{2,2}$ is

Algorithm 3 Computation of y_{n+1} with ARK schemes

for $i = 1 \dots s$ **do**
  Compute $F_i = F(t_n + c_i\Delta t)$
  if $i = 1$ **then**
    Set $u_i = y_n$
  else
    Compute $b = y_n + \Delta t \left(\sum_{j=1}^{i-1} a^I_{i,j} w^I_j + a^E_{i,j} w^E_j\right) + \gamma \Delta t P F_i$
    Solve the implicit part such that $(I - \gamma\Delta t AP)u_i = b$
  end if
  Store $w^I_i = APu_i + PF_i$ and $w^E_i = A(I-P)u_i + (I-P)F_i$
end for
Compute $y_{n+1} = y_n + \Delta t \left(\sum_{i=1}^{s} b_i (w^I_i + w^E_i)\right)$

the coefficient on the diagonal. We can observe that the storage of vectors w^I_i and w^E_i is required, which can be compared to the storage of vectors ζ_j for our schemes (algorithm 2). The vectors F_i are usually stored as sparse vectors since the source is usually located on the boundary. As a result, we can conclude that the storage requirements for both schemes are similar. ARK schemes have a drawback since they require that the points c_i coincide for the explicit and implicit schemes used, meaning we have to choose the same number of stages for the explicit part and the implicit part. Regarding this issue, the locally implicit schemes we propose are more flexible since they do not necessary require to have the same number of stages for the explicit and implicit schemes used. This is interesting because we can use less stages for the implicit part



(a) 2D (b) 3D

Fig. 3: Example of 2D and 3D small meshes used to evaluate the local time step of each element.

and reduce the computational time. Finally, the efficiency of the third fourth and fifth-order ARK schemes as defined by equation (10) is given in table 2. These schemes are referred as ARK3, ARK4 and ARK5 (respectively denoted

Scheme	ARK3	ARK4	ARK5
Efficiency	45.8 %	35.3 %	9.9 %

Table 2: Efficiency obtained for the explicit part of ARK schemes.

ARK3(2)4L[2]SA, ARK4(3)6L[2]SA and ARK5(4)8L[2]SA in appendix C of [35]). We can see that the efficiency of these schemes is lower than the one of optimized explicit schemes we presented in sub-section 2.2. This imply that for a given splitting of the mesh (operator P), ARK schemes should require a smaller time step compared to the proposed locally implicit schemes. Numerical experiments conducted in the section 6 confirm these preliminary comparisons.

4 Splitting based on the local time step

Let $\Omega \subset \mathbb{R}^n$ be an open set that represents the computational domain. The domain Ω is assumed to be meshed with elements K_i such that

$$\Omega = \bigcup K_i.$$

We consider the following semi-discrete ODE for a sub-mesh Ω_i of the computational domain Ω:

$$\begin{cases} M_h \dfrac{dy(t)}{dt} + K_h y(t) = F(t) \quad t \in (0,T] \\ y(0) = y_0 \end{cases}. \tag{24}$$

The sub-mesh Ω_i comprises the element K_i and its adjacent elements (elements that share an edge in 2D (a face in 3D) with the element K_i). In Figure 3, we have represented examples of a sub-mesh Ω_i in 2D and 3D in the case of an hybrid mesh. In (24), M_h is the local mass matrix and K_h is the local stiffness

matrix. We denote by $A_i = -M_h^{-1} K_h|_{\Omega_i}$ the product of the inverse of the local mass matrix with the local stiffness matrix of the sub-mesh Ω_i. The local time step Δt_i is then given as

$$\Delta t_i = \max \left\{ \Delta t \text{ such that } |R_s^\ell(\lambda \Delta t)| \leq 1, \ \forall \lambda \in sp(A_i) \right\}.$$

This local time step is expensive to compute, since we need to compute all the eigenvalues of A_i and then compute the maximum. We preferred to apply the algorithm 4 to evaluate an approximation of Δt_i.

Algorithm 4 Computation of Δt_i

Choose a nominal time step Δt_{nom} large enough.
Find the largest eigenvalue λ_{max} of A_i. Eigenvalues are sorted by increasing modulus $|R_s^\ell(\lambda \Delta t_{\text{nom}})|$.
Apply the bisection method to find $\Delta t_i \in [0, \Delta t_{\text{nom}}]$ such that $|R(\lambda_{\max} \Delta t_i)| = 1$.

In this algorithm, we find the eigenvalues λ with the largest modulus $|R_s^\ell(\lambda \Delta t_{\text{nom}})|$, by using Slepc software [36]. This software allows the user to provide its own function that compares two eigenvalues (through function EPSSetEigenvalue-Comparison).

Once the local time steps Δt_i are approximated, the splitting between fine and coarse region is done by comparing these local time steps to a reference time step Δt_{ref}:

$$\begin{cases} \text{if } \Delta t_i \leq \Delta t_{ref}, \ K_i \in \Omega^{fine} = \text{ fine region} \\ \text{if } \Delta t_i > \Delta t_{ref}, \ K_i \in \Omega^{coarse} = \text{ coarse region} \end{cases}.$$

5 Accuracy of the locally implicit methods

In this section, we aim at showing that the locally implicit methods obtained from the combination of the Linear-ERK schemes of order r with implicit schemes of the same order (Padé or Linear-SDIRK schemes), are of order r. This result follows from the result obtained for the local time-stepping method presented in [16].

By construction, the algorithm used to advance far degrees of freedom with formula (17) is of order r, i.e. the consistency error of the formula is in $O(\Delta t^{r+1})$. The algorithm used to advance close degrees of freedom (an implicit scheme of order r) will also provide a local error in $O(\Delta t^{r+1})$. So it seems clear that the global scheme has a local error in $O(\Delta t^{r+1})$. As a result, if the global scheme is stable, the global error should be in $O(\Delta t^r)$. We do not have any proof for the stability, but since the ODE in the close region is advanced with an A-stable scheme, the CFL of the global scheme should be controlled by the CFL of the coarse part.

6 Numerical results

This section provides numerical results in order to validate the locally implicit schemes implemented and to evaluate the efficiency of the schemes. The schemes have been implemented in the finite element C++ code Montjoie [5] (http://montjoie.gforge.inria.fr/) with a HDG formulation as described in [21]. The numerical simulations have been performed on 16 cores, the computation time is the real time spent to complete the simulation until the final time. The memory usage is measured for a sequential execution (on one core). The parallelization has been performed by parallelizing directly each step of the algorithm 2.

We first consider the acoustic wave equation. The scalar field u and vectorial field v depend on the space $\mathbf{x}$ and the time t and are solutions to the following boundary value problem:

$$\begin{cases} \rho\,\partial_t u - \operatorname{div} v = f, \quad \forall(\mathbf{x},t) \in \Omega \times \mathbb{R}^+ \\ \mu^{-1}\partial_t v - \nabla u = 0, \quad \forall(\mathbf{x},t) \in \Omega \times \mathbb{R}^+ \\ u(\mathbf{x},0) = \partial_t u(\mathbf{x},0) = 0, \quad \forall \mathbf{x} \in \Omega \quad \text{(null initial conditions)} \\ u = f_D, \quad \mathbf{x} \in \Gamma_D \quad \text{(Dirichlet condition)} \\ \mu\partial_n u = f_N, \quad \mathbf{x} \in \Gamma_N \quad \text{(Neumann condition)} \\ \mu\partial_n u + \sqrt{\rho\mu}\,\partial_t u = f_A, \quad \mathbf{x} \in \Gamma_A \quad \text{(Absorbing condition)} \end{cases} \tag{25}$$

where Ω is the computational domain. Γ_D, Γ_N and Γ_A are the boundaries associated respectively with Dirichlet, Neumann and absorbing boundary condition. n is the normal vector outgoing to the considered boundary, ρ and μ are physical indexes, which are piecewise constant. f_D, f_N and f_A are given source functions. Hybrid Discontinuous Galerkin formulation is used for the space discretization of these equations (see [33]). Schur complement technique is performed such that the linear system to be solved has a reduced size (see details in [21]). This linear system is factorized and solved by calling Mumps routines [37]. For implicit schemes, we can use Padé schemes, denoted as Padé r where r is the order. We can also use Linear-Sdirk schemes, denoted as LSDIRK $r-\ell$ where r is the order and ℓ the number of additional stages. For these schemes, the total number of stages is equal to $s+\ell$ where $s = r-1$.

6.1 Time convergence

In this sub-section, we consider a square $[-5,5]^2$ with homogeneous indexes $\rho = \mu = 1$. The inner square $[-1,1]^2$ is refined (see figure 4) in order to have a fine resolution where the source is not null. This inner square will correspond to the fine region (i.e. implicit part) where an implicit scheme will be used. We choose the following source term (Gaussian in space)

$$f(x,y,t) = \beta \exp(-\alpha(x^2+y^2))h(t),$$

where

$$r_0 = 1, \quad \alpha = \frac{\log(10^6)}{r_0^2}, \quad \beta = \sqrt{\frac{\alpha}{\pi}}.$$

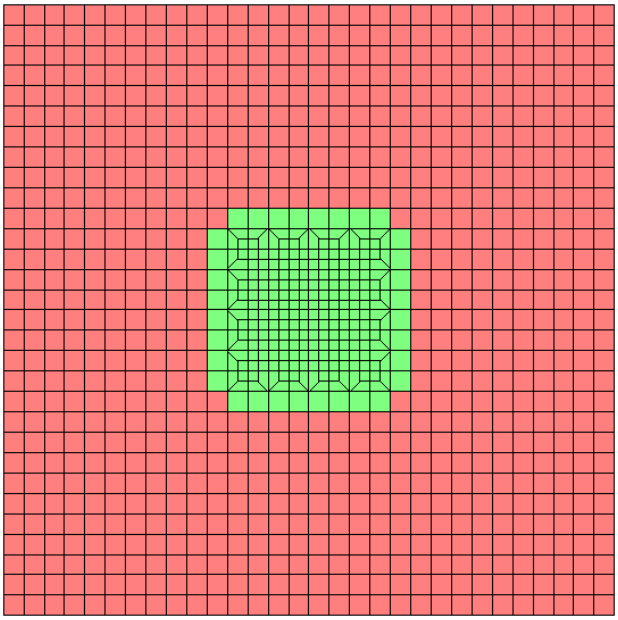


Fig. 4: Mesh used for time convergence (Green zone: fine region, Red zone: coarse region). The close region comprises the fine region (green zone) and adjacent elements (elements that share an edge with an element in the green zone).

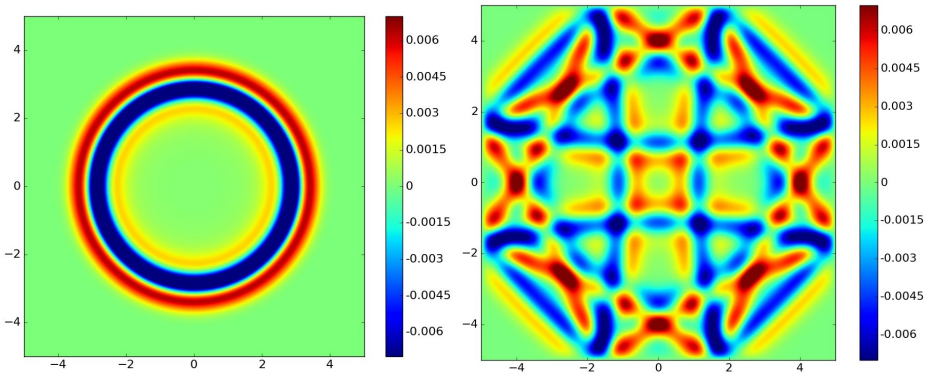


Fig. 5: Solution for the square at time $t = 4$ and $t = 20$.

The time pulse $h(t)$ is chosen as a derivative of a Gaussian

$$h(t) = -(f_0 t - 1) \exp\left(-(f_0 t - 1)^2 \pi^2\right),$$

where f_0 is the central frequency (we choose $f_0 = 1$). Homogeneous Dirichlet conditions are set on the boundary of the square. The solution obtained for $t = 4$ and $t = 20$ is plotted in figure 5. In this sub-section, we study the convergence of

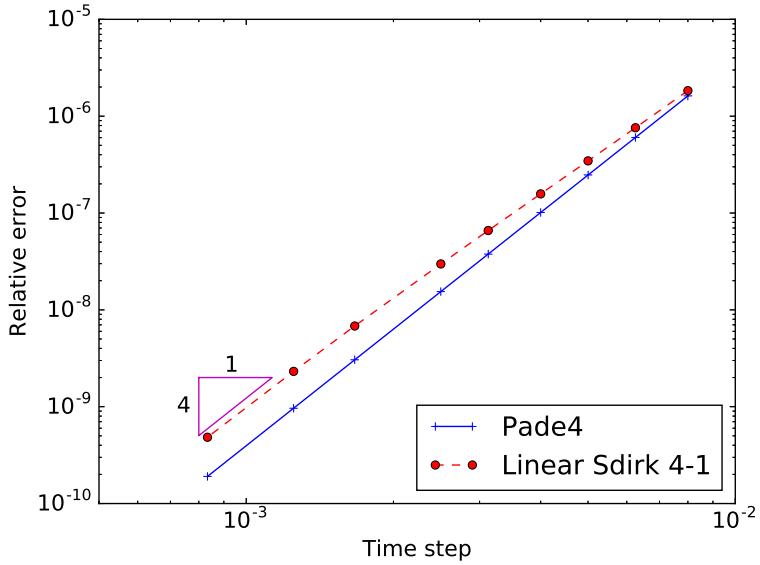


Fig. 6: Relative L^2 error versus Δt for ERK 4-2 and Padé 4 (or Linear SDIRK 4-1).

the local implicit scheme. As a result, the space discretization is fixed with the mesh displayed in figure 4 and $\mathbb{Q}_{12}$ space approximation such that the space error is close to 10^{-12} (due to round-off errors in double precision). The reference solution is computed with an explicit eighth-order Runge-Kutta scheme ($\Delta t = 0.001$), the relative error is computed with final time $t = 20$. In figure 6, 7 and 8, we can observe the convergence in time of local implicit schemes. Linear ERK schemes are used for the explicit part (respectively ERK 4-2, ERK 6-2 and ERK 8-2). Padé schemes or Linear SDIRK schemes with one additional stage are used for the implicit part. For fourth-order schemes (see figure 6), we observe a nice convergence in $O(\Delta t^4)$ with either Padé scheme or Linear SDIRK for the implicit part. For sixth-order schemes (see figure 7), we obtain a convergence in $O(\Delta t^6)$ when Padé scheme is used. The stagnation below 10^{-12} is due to round off errors (double precision is used). When Linear SDIRK scheme is used, the convergence seems slower, the relative error is much larger than with Padé schemes. For eighth-order, Padé scheme would probably provide a better convergence in $O(\Delta t^8)$ but it cannot be observed since the error is already in 10^{-12} for the maximal time step. Linear SDIRK scheme provides a reduced convergence (here we measure a sixth-order convergence in figure 8). For the figures 6, 7 and 8, the time step is chosen lower than the maximal time step. Because of the restrictive CFL, this maximal time step is quite small, and the error obtained with this time step can be very small.

6.2 Space-time convergence

In this sub-section, the convergence both in space and time is studied We con-

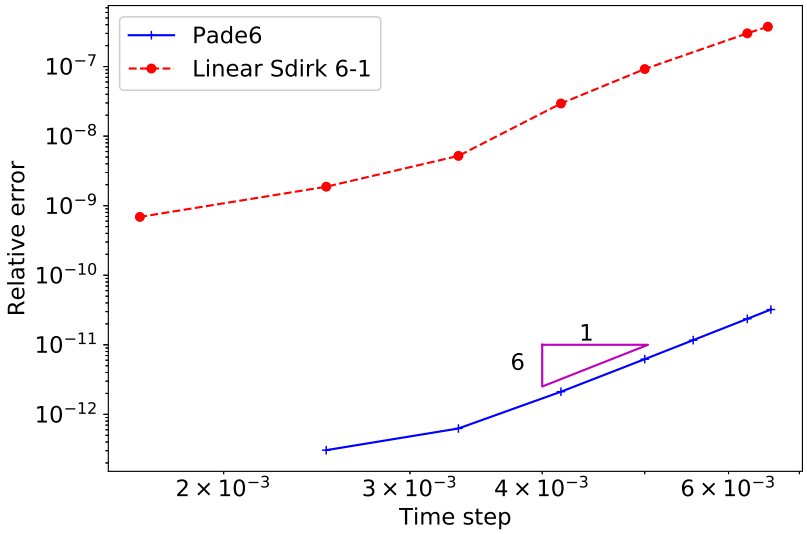


Fig. 7: Relative L^2 error versus Δt for ERK 6-2 and Padé 6 (or Linear SDIRK 6-1).

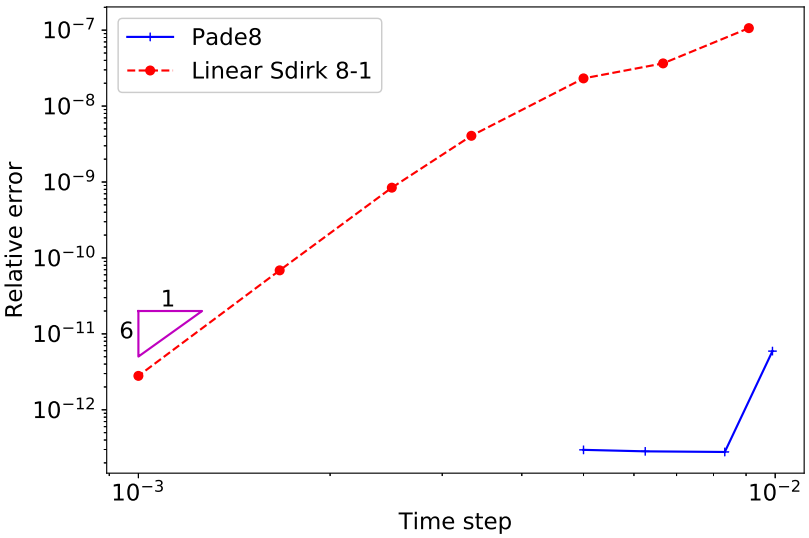


Fig. 8: Relative L^2 error versus Δt for ERK 8-2 and Padé 8 (or Linear SDIRK 8-1).

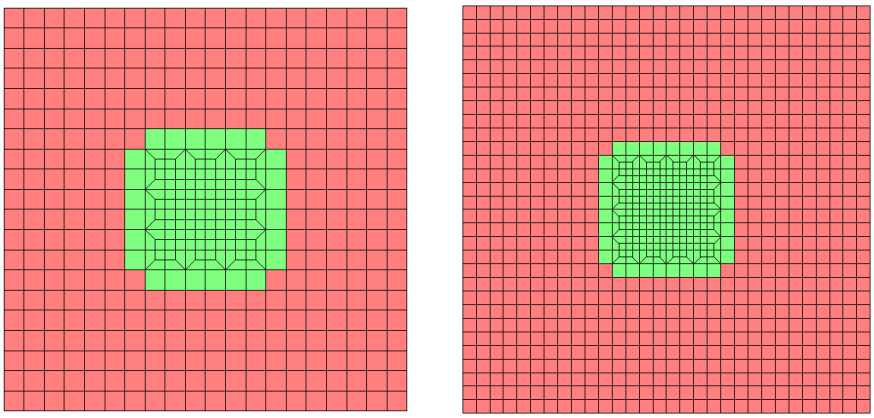


Fig. 9: Example of two meshes used for space-time convergence.

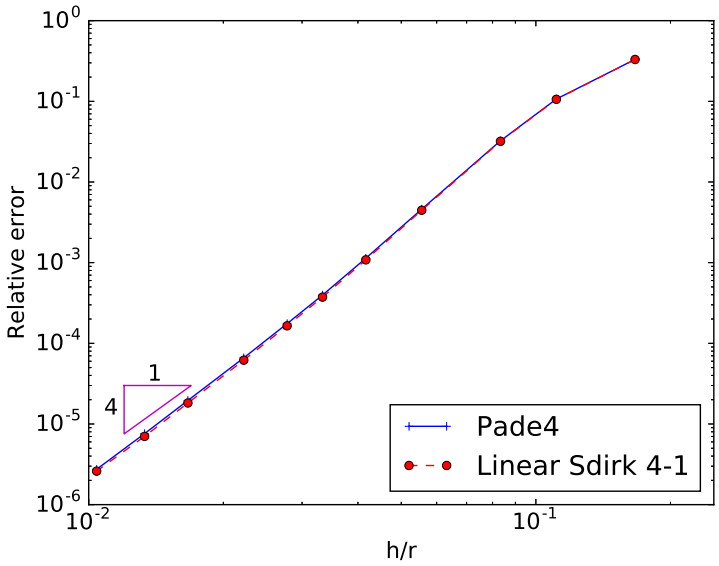


Fig. 10: Space-time convergence with $\mathbb{Q}_3$, ERK4-2 and Pade 4 (or Linear SDIRK 4-1).

sider the same case than in the previous section. Only the mesh size will vary, an example of two meshes is given in figure 9. The reference solution is computed with $\mathbb{Q}_{12}$ approximation and 40 elements in x-direction with an explicit eighth-order Runge-Kutta scheme ($\Delta t = 0.001$).

We consider $\mathbb{Q}_3, \mathbb{Q}_5$ and $\mathbb{Q}_7$ approximation in order to obtain a convergence in $O(h^4), O(h^6)$ and $O(h^8)$ where h is the mesh size. The time step is chosen

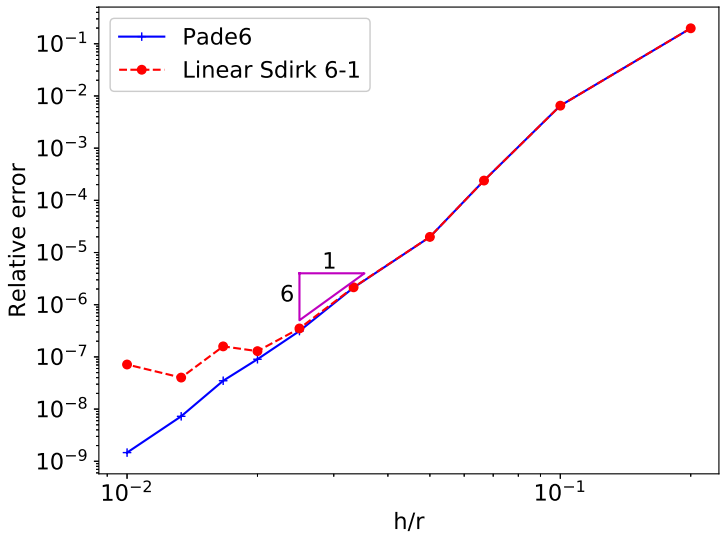


Fig. 11: Space-time convergence with $\mathbb{Q}_5$, ERK6-2 and Pade 6 (or Linear SDIRK 6-1).

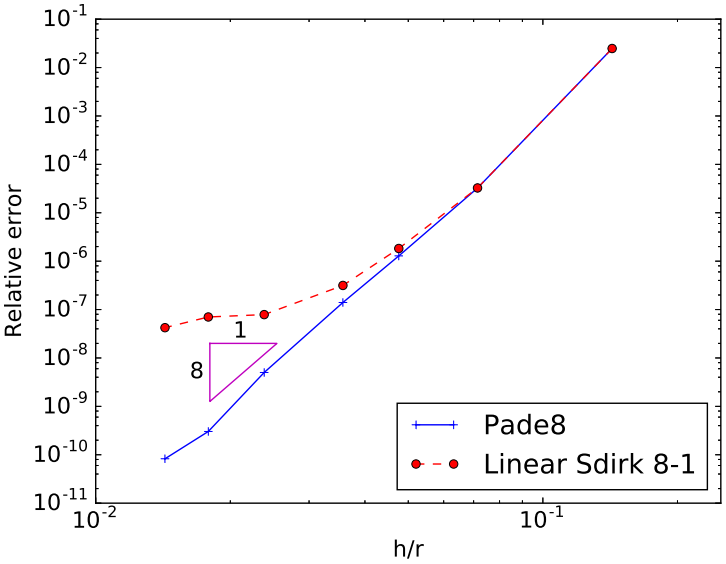


Fig. 12: Space-time convergence with $\mathbb{Q}_7$, ERK8-2 and Pade 8 (or Linear SDIRK 8-1).

Order	h	Local LSDIRK	Local Padé	Explicit	Slepc
4 ($\mathbb{Q}_3$)	0.25	0.0727	0.0727	0.0727	0.0727
6 ($\mathbb{Q}_3$)	0.5	0.0512	0.0516	0.0543	0.0567
8 ($\mathbb{Q}_7$)	0.5	0.0410	0.0418	0.0428	0.0428

Table 3: Maximal time step for the square will local refinement for local implicit schemes, without refinement for explicit scheme. Last column gives the estimate computed with Slepc.

close to the maximal time step, namely we choose

$$\Delta t = \frac{h}{3.5}, \quad \Delta t = \frac{h}{10}, \quad \Delta t = \frac{h}{12.5},$$

for respectively $\mathbb{Q}_3, \mathbb{Q}_5$ and $\mathbb{Q}_7$ space approximation. In the table 3, the measured maximal time step for this case is given for the locally implicit schemes. This maximal time step is found manually by checking that the solution is stable until $T = 10000$. It is compared with the time step measured for a regular square (with only ERK scheme), in order to check if the local implicit scheme does not deteriorate the stability condition due to the coarse part of the mesh. It is observed that the maximal time step is slightly smaller with locally implicit schemes from order 6. In this table, the maximal time step computed by Slepc (see section 4) is also given to show that this approximation is fairly good, sometimes this procedure gives exactly the maximal time step (here for order 4 and 8). In figures 10, 11 and 12, we plot the relative L^2 error versus the mesh size h for fourth-order, sixth-order and eighth-order schemes. When a local implicit scheme is used with Padé scheme for the implicit part, we observe a nice convergence in $O(h^{r+1})$ as expected (where r is the order of space approximation). When a LSDIRK scheme is used, we observe a good convergence for fourth-order scheme but a deteriorated convergence for higher order schemes. This deterioration could be due to spurious oscillations that have been previously observed in [32]. Nevertheless this issue appears for small values of h and the relative L^2 error is satisfactory. In order to illustrate this issue, we have computed the L^2 error when the following scalar ODE is solved

$$y'(t) = iLy(t) + f(t)$$

for large values of L with a simple source function

$$f(t) = e^{-3(t-5)^2} cos(2\pi t).$$

Null initial conditions are imposed such that the solution is not highly oscillatory even for large values of L. This kind of scalar ODE can be obtained when the general ODE (1) is projected on an eigenmode associated with the eigenvalue iL. Spurious oscillations are associated with highly oscillating eigenmodes (obtained when L is large). The L^2 error is computed on the time interval $[0, 10]$. The L^2 error obtained for Padé and LSDIRK schemes is plotted in figure 13. We can observe that fourth-order schemes (both Padé 4 and LSDIRK 4-1) are robust since the L^2 error remains low for large values of L. When the time step is divided by two, the error decreases for all values of L. For eighth-order schemes, only Padé 8 is robust. When LSDIRK 8-l, $l = 1, 2, 3$, is used, the L^2 error goes

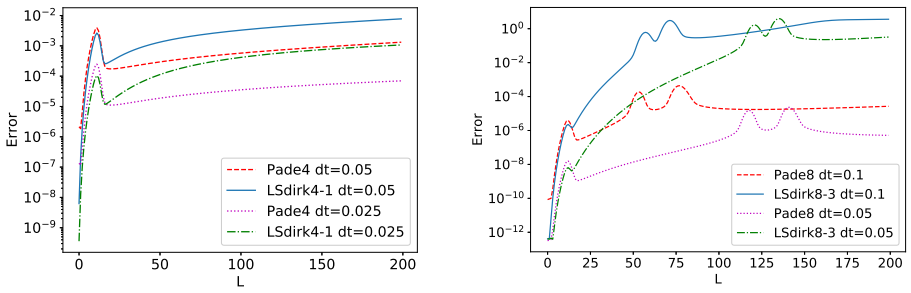


Fig. 13: On the left, the relative L^2 error obtained for the scalar ODE solved with the fourth-order implicit schemes. On the right, the relative L^2 error with the eighth-order implicit schemes.

beyond 100% for large values of L. This phenomenon is known as a phenomenon of order reduction [38].

6.3 2-D case

In this section, we consider the scattering by a magnetron (see figure 14). The medium is homogeneous ($\rho = \mu = 1$), there are 16 circular cavities of radius 4.0. The domain has a diameter of 140 wavelengths, since the radius of the outer circle is equal to 70, and the central frequency of the source is equal to 1.0. A homogeneous Neumann boundary condition is set on all the boundaries except at the external circle. On this outer circle, an absorbing boundary condition (ABC) is set to simulate the propagation of an incident plane wave with $f_A = \mu\partial_n u^{inc} + \sqrt{\rho\mu}\partial_t u^{inc}$, which corresponds to an ABC applied on the reflected wave. The incident field is given by $u^{inc} = h(t - 1.5 - \mathbf{x})$. The time source is a sine function modulated by a Gaussian

$$h(t) = \sin(2\pi t)\exp(-0.1(t - t_0)^2),$$

t_0 is chosen such that the Gaussian is equal to 10^{-6} at $t = 0$:

$$t_0 = \sqrt{10\,\log(10^6)}.$$

The source term f_A is given as

$$f_A(x, t) = (1 - u \cdot n)h'(t - u \cdot x - 70),$$

where $u = (-1, 0)$ is the orientation of the plane wave and n the outgoing normal. The solution obtained is plotted in figure 15 for different times. The reference solution is computed with eighth-order Padé scheme and $\Delta t = 0.01$. Among fourth-order time schemes, we compare the following schemes:

- Classical RK4 scheme (ERK 4-0)
- ERK 4-2 with local implicit LSDIRK 4-1

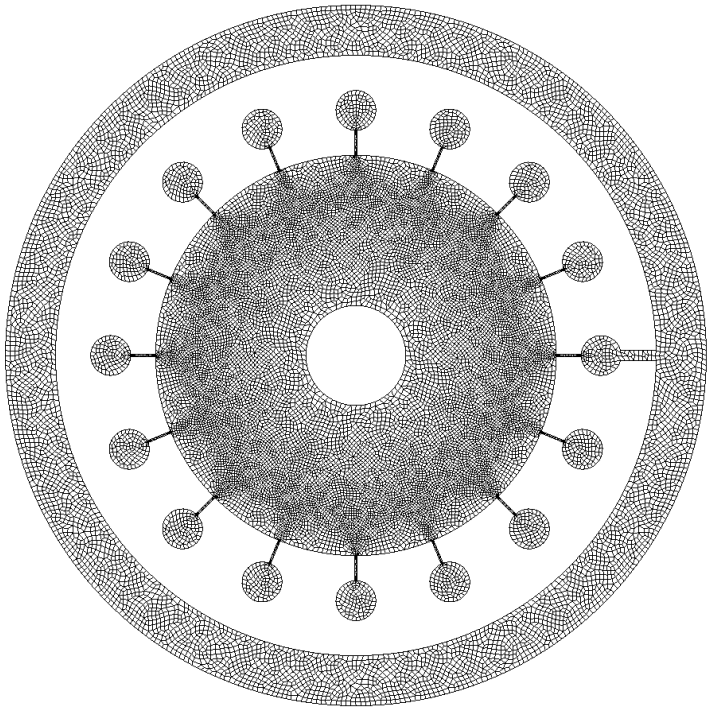


Fig. 14: Mesh used for the simulations for the scattering of a magnetron ($\mathbb{Q}_8$ approximation).

- ERK 4-2 with local implicit Padé 4
- ERK 4-4 with local implicit Padé 4
- ARK4
- LSDIRK 4-1
- Padé 4

Local implicit schemes use the splitting computed as described in section 4 with $\Delta t_{\text{ref}} = 0.032$. The obtained coarse and fine regions are displayed in figure 16. The time step for local implicit schemes is then chosen close to the maximal time step. The results obtained for fourth-order schemes are summarized in table 4. For this case, we observe that the explicit scheme (ERK 4-0) is cheap in memory (745 Mo) but is expensive in computational time (more than 7 hours). On the other hand, the implicit fourth-order Padé scheme is the fastest (21 minutes) but requires much more memory (4.3 Go). Locally implicit schemes achieve a compromise between these two schemes, since the storage requirement is lower than for implicit schemes and the computational time is smaller than for explicit schemes. ARK4 scheme is less efficient because the maximal time step is smaller (see table 2 that gives the efficiency of this scheme) than by using local implicit schemes based on ERK4-2 scheme. Among eighth-order time schemes, we compare the following schemes:

- ERK 8-2 with local implicit LSDIRK 8-1

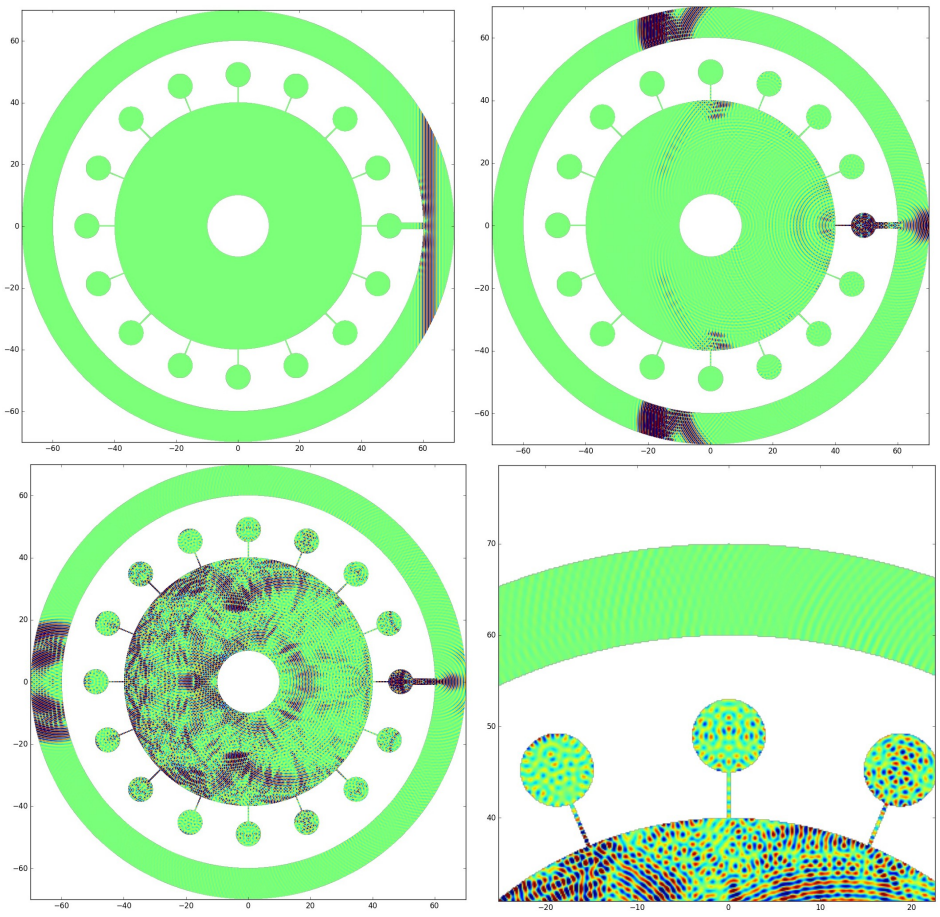


Fig. 15: Solution for the magnetron for $t = 20, 100, 200$. On right bottom, zoom of the solution near circular cavities.

Time Scheme	Time step	L^2 error	Computation time	Memory
Explicit RK4	$4.57 \cdot 10^{-4}$	$8.12 \cdot 10^{-10}$	7h10min	**748 Mo**
Local LSDIRK 4-1 (ERK 4-2)	0.025	$1.17 \cdot 10^{-3}$	42min9s	1.62 Go
Local Padé 4 (ERK 4-2)	0.025	$1.36 \cdot 10^{-3}$	28min10s	2.07 Go
Local Padé 4 (ERK 4-4)	0.0385	$3.71 \cdot 10^{-3}$	22min8s	2.17 Go
ARK4	0.0179	$1.94 \cdot 10^{-4}$	60min9s	1.60 Go
LSDIRK 4-1	0.04	$2.645 \cdot 10^{-3}$	44min32s	2.7 Go
Padé 4	0.0333	$3.454 \cdot 10^{-3}$	**21min23s**	4.3 Go

Table 4: Comparison of different fourth-order time schemes for the magnetron.

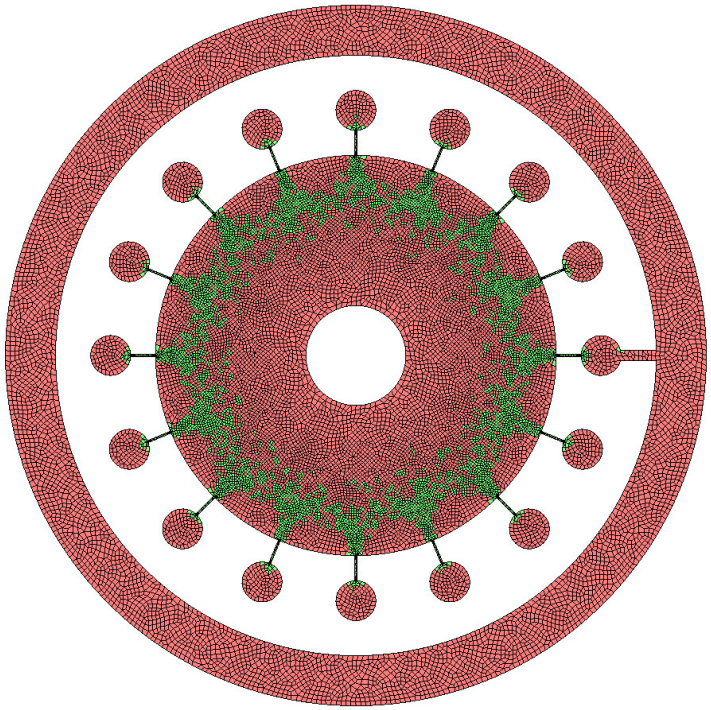


Fig. 16: Splitting of the magnetron mesh into the fine part (green) and coarse part (red).

Time Scheme	Time step	L^2 error	Computation time	Memory
Local LSDIRK 8-1 (ERK 8-2)	$1/30$	$1.92 \cdot 10^{-6}$	60min22s	**1.78 Go**
Local LSDIRK 8-1 (ERK 8-6)	$1/22$	$9.93 \cdot 10^{-6}$	49min28s	1.91 Go
Local Padé 8 (ERK 8-2)	$1/30$	$2.25 \cdot 10^{-9}$	39min3s	2.93 Go
LSDIRK 8-1	$1/6$	$2.01 \cdot 10^{-3}$	21min49s	2.84 Go
Padé 8	$1/4$	$2.02 \cdot 10^{-3}$	**6min4s**	7.44 Go

Table 5: Comparison of different eighth-order time schemes for the magnetron.

- ERK 8-6 with local implicit LSDIRK 8-1
- ERK 8-2 with local implicit Padé 8
- LSDIRK 8-1
- Padé 8

Here, local implicit schemes use the same splitting as for fourth order-schemes (see figure 16). The results obtained for these schemes are summarized in table 5. Because of the use of eighth-order schemes, the local implicit schemes are much more accurate compared to fourth order schemes. Again the implicit Padé scheme is the fastest and local implicit schemes use less memory. For this case, we observe that the error obtained for local LSDIRK scheme is much higher than for local Padé scheme. This is probably due to the issue of convergence we

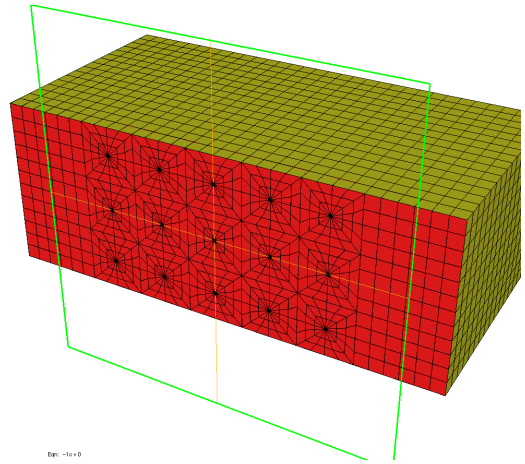


Fig. 17: 3-D mesh used for the scattering by a network of small spheres. ($\mathbb{Q}_4$ approximation)

have observed for the square. By using more additional stages (ERK 8-6 instead of ERK 8-2), the computational time is lower.

6.4 3-D case

In this section, we consider the scattering by a network of small spheres (see figure 17). These spheres are placed regularly in a parallelepiped (5 spheres in x-axis, 5 in y-axis and 3 in z-axis). The distance between each sphere is equal to 2.0 whereas the radius of each sphere is equal to 0.07. This network of spheres is placed in the parallelepiped box $[-8,8]\times[-8,8]\times[-3,3]$. The physical indexes inside the spheres are given by

$$\rho = 0.1,\ \mu = 0.8,$$

whereas the surrounding medium is characterized by $\rho = \mu = 1$. Inhomogeneous Dirichlet condition is set on the bottom plane $z = -3$ whereas homogeneous absorbing boundary condition is set on other boundaries. In Figure 17, the hexahedral mesh used for the simulations is displayed. $\mathbb{Q}_4$ polynomial approximation is used with this mesh. On the plane $z = -3$, we have taken the following inhomogeneous Dirichlet condition

$$f_D = g(x,y)h(t),$$

where the space source g is a Gaussian centered at the origin

$$g(x,y) = \beta \exp(-\alpha(x^2+y^2))$$

with

$$r_0 = 5, \quad \alpha = \frac{\log(10^6)}{r_0^2}, \quad \beta = \sqrt{\frac{\alpha}{\pi}},$$

Time scheme	Time step	L^2 error	Computation Time	Memory
Explicit RK4	$2.22 \cdot 10^{-4}$	-	6h58	**3.0 Go**
Local LSDIRK 4-1 (ERK4-0)	$1/57$	$1.65 \cdot 10^{-4}$	61min28s	72.0 Go
Local LSDIRK 4-1 (ERK4-2)	$1/28$	$5.64 \cdot 10^{-4}$	32min21s	72.2 Go
Local LSDIRK 4-1 (ERK4-4)	$1/20$	$1.08 \cdot 10^{-3}$	**25min12s**	72.2 Go
ARK4	$1/37$	$1.58 \cdot 10^{-4}$	45min51s	71.8 Go
LSDIRK 4-1	0.05	$1.06 \cdot 10^{-3}$	39min31s	92.4 Go

Table 6: Comparison of different fourth-order time schemes for the scattering by a network of spheres.

and h is a sine function modulated by a Gaussian

$$h(t) = \exp(-2(t - t_0)^2) \sin(2\pi t).$$

t_0 is chosen such that the Gaussian is equal to 10^{-6} at $t = 0$:

$$t_0 = \sqrt{\frac{\log(10^6)}{2}}.$$

The solution obtained for $t = 6$ and $t = 12$ is represented in Figure 18. We focus here on the following fourth-order time schemes:

- Classical RK4 scheme
- ERK 4-ℓ with local implicit LSDIRK 4-1 with different values of ℓ
- ARK4
- LSDIRK 4-1

For the first scheme which is purely explicit, we set Δt close to the CFL. For the second time scheme, we have obtained the splitting given in figure 19 between the coarse and fine mesh for $\Delta t_{\text{ref}} = 0.01$ and ERK 4-0. The time step for local implicit schemes is then chosen close to the maximal time step. For the last time scheme, the time step is chosen such that we obtain 0.1% of L^2 error for the final time $T = 20$. The computational time and memory required for all these schemes to compute the solution until the final time $T = 20$ are given in table 6. The L^2 error has been computed for the final time and by taking the solution given with the first scheme (explicit) as reference. We see here that it is advantageous to have additional stages for local implicit schemes based on ERK4-ℓ schemes, since the time step can be chosen larger, and the computational time is reduced. ARK4 scheme does not have this flexibility and is less efficient.

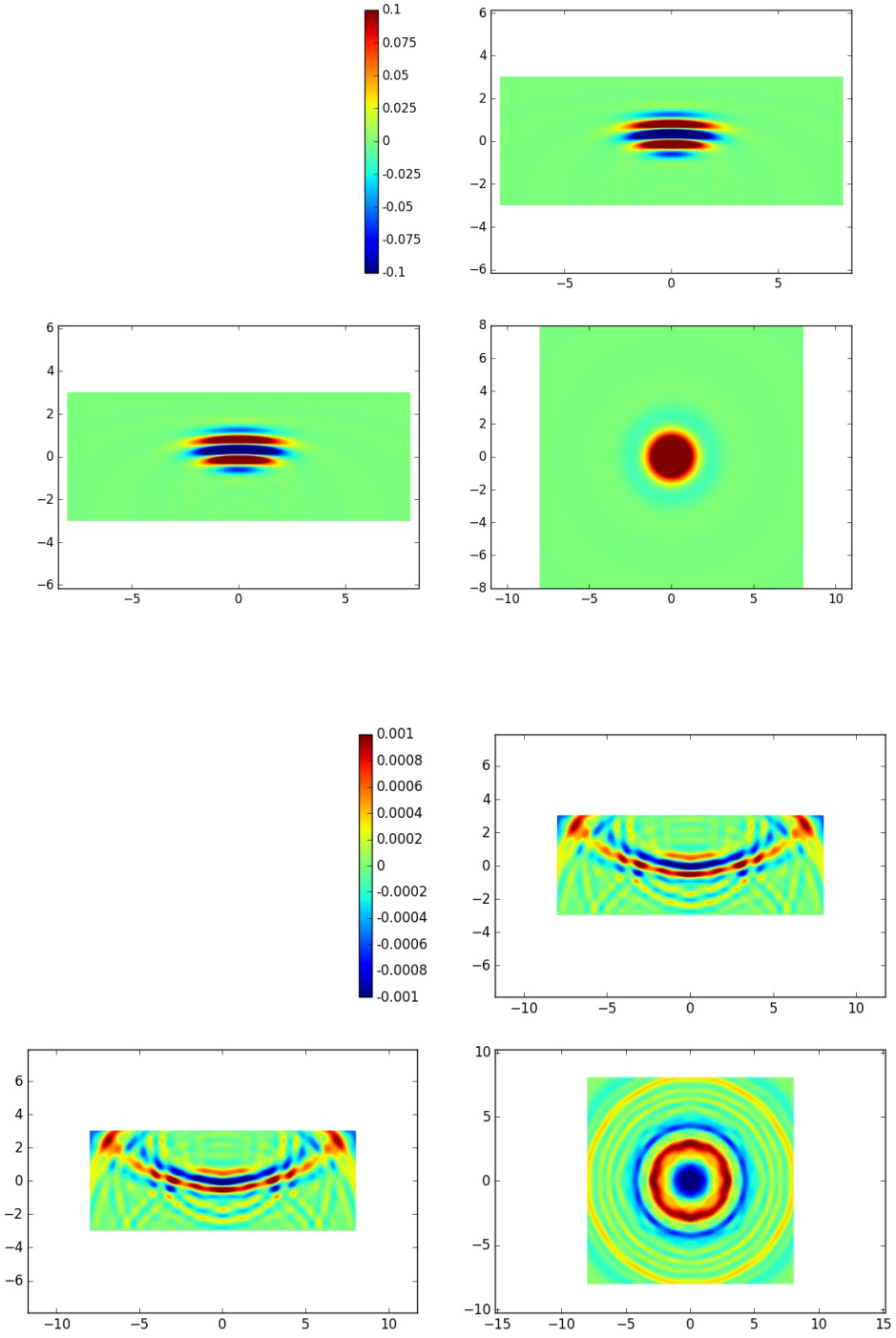


Fig. 18: Solution obtained for the scattering by a network of spheres. Solution for $t = 6$ (top) and $t = 12$ (bottom).

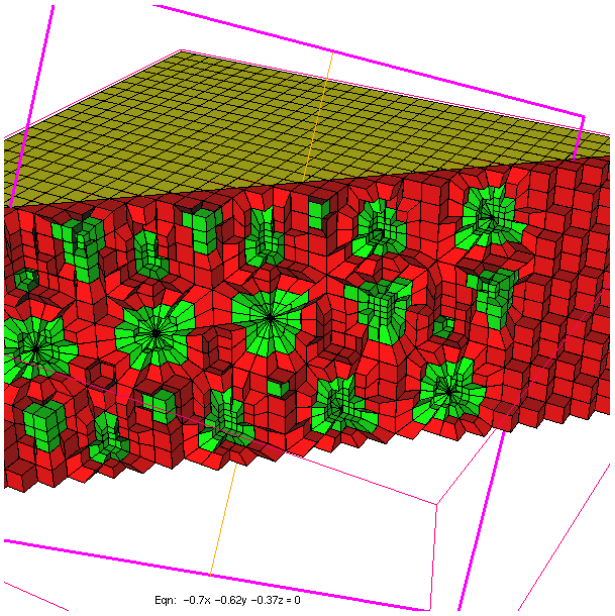


Fig. 19: Splitting into coarse (red) and fine region (green).

7 Conclusion

In this paper, we have presented locally implicit schemes for linear ODEs with an arbitrary order. These schemes couple ERK (Explicit Runge Kutta) schemes with different type of one-step implicit schemes. Herein, we have considered either Padé schemes or Linear SDIRK schemes for the implicit time integration. When Padé schemes are used for the implicit part, we have observed that the global scheme converges with the expected order (i.e. in $O(\Delta t^p)$ for the scheme of order p). We also observed that the CFL is almost the same as the CFL computed for the coarse region. The fully discrete scheme in space and time exhibits also a nice convergence in $O(h^p + \Delta t^p)$ when Hybridizable Discontinuous Galerkin approximation is used in space. When Linear SDIRK schemes are used for the implicit part, the convergence is nice only for fourth-order schemes, but it becomes progressively bad for higher orders. This deterioration could be due to spurious oscillations that have been previously observed in [32]. In spite of this issue of convergence, the accuracy obtained with these schemes is still satisfactory and besides they require less memory usage than the locally implicit methods formed using the Padé schemes.

Compared to the classical IMEX Runge-Kutta schemes, the main advantage of the proposed method is that it is easier to construct very high order schemes. We have implemented schemes up to order eight, whereas IMEX Runge-Kutta schemes are usually limited to fourth order accuracy. The latter schemes are able to treat general ODEs whereas only linear ODEs of the form $y'(t) = Ay(t) + F(t)$ can be treated with the proposed approach. This approach also enables the use of any implicit scheme, whereas IMEX schemes are less flexible since they

require the same number of stages for the explicit and implicit schemes used. The proposed approach is equivalent to IMEX schemes regarding the memory usage. However, since the stability condition is less restrictive, it leads to a reduced computational time.

The 2-D and 3-D numerical results for the wave equation presented in this paper show that explicit schemes can be highly constrained by the stability condition, i.e. the time step has to be chosen very small to satisfy the CFL condition. For this reason even though the optimized explicit schemes give an accurate solution, they are very expensive in terms of computational time which make them less efficient for the tested cases. On the other hand globally implicit integration with unconditionally stable scheme has no CFL condition to satisfy, but they are limited by an accuracy level, i.e. the time step has still to be chosen to reach the targeted accuracy level. We observed that they require less computational time compared to explicit schemes, but they consume a large amount of memory when direct solvers are used. In a middle, the locally implicit schemes we have developed achieve a compromise between implicit and explicit schemes. They use less memory compared to global implicit integration schemes and less computational time compared to global explicit integration schemes. In this case, the time step is chosen by the user in order to have a reduced computational time and memory usage. As a prospect, we think that it would be interesting to couple local time-stepping (LTS) strategy with locally implicit method. This could allow to add extra levels of splitting and give more flexibility in the choice of the time step. In fact, in the current development the time step chosen is restricted by the CFL number of the explicit scheme used, which depends on the defined coarse region. Having a LTS strategy in a wider coarse region could release the CFL restriction for the explicit schemes allowing by this way the use of larger time steps. As a consequence, the implicit method would be used only on very fine regions with larger time step and could lead to an extra reduction of the computational time and the memory usage.

Acknowledgments

This work has been supported by the INRIA-TOTAL strategic action DIP (dip.inria.fr). Experiments presented in this work were carried out using the PlaFRIM experimental platform. Helene Barucq has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 777778 (MATHROCKS).

A Remarks on the implementation of the HDG formulation of the acoustic wave equation.

For the first order acoustic wave equation (25), the local variational formulation set on an element K in the HDG formulation is given as

$$\begin{cases} \dfrac{d}{dt}\displaystyle\int_K \rho\, u\, \varphi_i\, dx + \int_K v\cdot\nabla\varphi_i\, dx - \int_{\partial K} v\cdot n\, \varphi_i\, dx + \int_{\partial K} \tau(u-\lambda)\,\varphi_i\, dx = 0 \\ \dfrac{d}{dt}\displaystyle\int_K \mu^{-1} v\cdot\psi_i\, dx - \int_K \nabla u\cdot\psi_i + \int_{\partial K}(u-\lambda)\psi_i\cdot n\, dx = 0 \\ \displaystyle\int_{\partial K}(-v\cdot n + \tau(u-\lambda))\, q\, dx = 0 \end{cases} \tag{26}$$

where u, v are volume unknowns (discontinuous) and λ a surface unknown. φ_i, ψ_i, q are the test functions associated with u, v and λ. The penalization parameter τ is equal to $\sqrt{\rho\,\mu}$. The last equation is modified when the boundary of K meets the boundary of the computational domain. For instance, is a first-order absorbing boundary condition is set

$$v\cdot n + \sqrt{\rho\,\mu}\, u = f_A.$$

the last equation of (26) becomes:

$$\int_{\Gamma_N}(-v\cdot n + \tau(u-2\lambda))\, q\, dx = -\int_{\Gamma_N} f_A\, q\, dx.$$

The HDG formulation provides locally for each element K the semi-discrete system

$$\begin{cases} M_u \dfrac{dU}{dt} + K_u V + C_u \Lambda = 0 \\ M_v \dfrac{dV}{dt} + K_v U + C_v \Lambda = 0 \\ C_\lambda \Lambda + C_u^T U + C_v^T V = 0 \end{cases} \tag{27}$$

C_λ is multiplied by two for an edge with an absorbing boundary condition. When the ODE (21) (which is set in the fine region) is solved with an implicit scheme, a linear system of the form

$$\beta Y + APY = F$$

has to be solved for close dofs (with $Y = (U, V)$). For elements located in the fine region, we have

$$\begin{cases} \beta M_u U + K_u V + C_u \Lambda = F_u \\ \beta M_v V + K_v U + C_v \Lambda = F_v \\ C_\lambda \Lambda + C_u^T U + C_v^T V = 0 \end{cases} \tag{28}$$

Unknowns U and V are eliminated element-wise to obtain a local system in Λ

$$\left[C_\lambda - \begin{pmatrix} C_u^T & C_v^T \end{pmatrix} \begin{pmatrix} \beta M_u & K_u \\ K_v & \beta M_v \end{pmatrix}^{-1} \begin{pmatrix} C_u \\ C_v \end{pmatrix}\right] \Lambda = -\begin{pmatrix} C_u^T & C_v^T \end{pmatrix} \begin{pmatrix} \beta M_u & K_u \\ K_v & \beta M_v \end{pmatrix}^{-1} \begin{pmatrix} F_u \\ F_v \end{pmatrix}.$$

This equation has to be assembled with all other elements to obtain the final system solved by Λ. For adjacent elements (located on the coarse region), we have:

$$\begin{cases} \beta M_u U + C_u \Lambda = F_u \\ \beta M_v V + C_v \Lambda = F_v \\ C_\lambda \Lambda = 0 \end{cases} \tag{29}$$

where only unknowns on edges of the fine region are concerned for Λ. The equation to be assembled with other elements is given as

$$C_\lambda \Lambda = 0.$$

As a result, when the linear system is assembled for Λ, only unknowns located on edges (faces in 3-D) of the fine region are involved. It is actually equivalent to add the contribution $C_\lambda \Lambda = 0$ or impose a fictitious homogeneous absorbing boundary condition on edges located at the interface between the fine and coarse region. Once Λ has been computed on the edges of the fine region (Λ is null for other edges), the unknown U and V are reconstructed element-wise, e.g. in adjacent elements:

$$U = \frac{1}{\beta} M_u^{-1} \left(F_u - C_u \Lambda \right)$$

$$V = \frac{1}{\beta} M_v^{-1} \left(F_v - C_v \Lambda \right)$$

For quadrilateral/hexahedral elements, the mass matrices M_u and M_v are diagonal, and the elimination/reconstruction of U and V can be lead efficiently as detailed in [21].

References

1. Cohen, G., Fauqueux, S.: Mixed finite elements with mass-lumping for the transient wave equation. Journal of Computational Acoustics, 8, 171–188 (2000).
2. Cohen, G., Pernet, S.: Finite element and discontinuous Galerkin methods for transient wave equations. Springer (2017).
3. Hesthaven, J.S., Warburton, T.: Nodal discontinuous Galerkin methods algorithms, analysis, and applications. Springer (2008).
4. Monk, P.: Finite element methods for Maxwell's equations. Oxford Science Publications (2003).
5. Duruflé, M.: Intégration numérique et éléments finis d'ordre élevé appliqués aux équations de Maxwell en regime harmonique. PhD thesis, Université Paris Dauphine (2006).
6. Hairer, E., Norsett, S.P., Wanner, G.: Solving ordinary differential equations I nonstiff problem. Springer (2008).
7. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations Second Edition. John Wiley & Sons Ltd (2008).
8. Hairer, E., Wanner, G.: Solving ordinary differential equations II stiff and differential-algebraic problem. Springer (2010).
9. Mead, J.L., Renaut, R.A., Optimal Runge-Kutta methods for first order pseudospectral operators, Journal of Computational Physics, 404–419 (1999).
10. Gilbert, J.C., Joly, P.: Higher order time stepping for second order hyperbolic problems and optimal CFL conditions, Computational Methods in Applied Sciences, 16, 67–93 (2008).
11. Joly, P., Rodrìguez, J.: Optimized higher order time discretization of second order hyperbolic problems: Construction and numerical study. Journal of Computational and Applied Mathematics, 234, 1953–1961 (2010).
12. Parsani, M., Ketcheson, D.I., Deconinck W.: Optimized explicit Runge-Kutta schemes for the spectral difference method applied to wave propagation problems. SIAM Journal on Scientific Computing, 35,A957–A986 (2013).
13. Ketcheson, D.I., Ahmadia, A.J.: Optimal stability polynomials for numerical integration of initial value problems. Communications in Applied Mathematics And Computational Science, 7, 247–271 (2012).
14. Duruflé, M., N'diaye, M.: Optimized high-order explicit Runge-Kutta-Nyström schemes. In: Bittencourt, M., Dumont, N., Hesthaven, J.S. (eds) Spectral and High Order Methods for Partial Differential Equations ICOSAHOM-2016, Springer (2016).
15. Diaz, J. and Grote, M.: Energy conserving explicit local time stepping for second-order wave equations. SIAM Journal on Scientific Computing, 31, 1985–2014 (2009).
16. Grote, M., Mehlin, M., Mitkova, T.: Runge-Kutta-based explicit local time-stepping methods for wave propagation. SIAM Journal on Scientific Computing, 37, A747–A775 (2015).
17. Piperno, S.: Symplectic local time-stepping in non-dissipative DGTD methods applied to wave propagation problems. ESAIM Mathematical Modeling and Numerical Analysis, 40, 815–841 (2006).
18. Rodrìguez, J.: Raffinement de maillage spatio-temporel pour les équations de l'élastodynamique. PhD thesis, Université Paris Dauphine (2004).
19. Skvortsov, L.M.: Diagonally implicit Runge-Kutta methods for stiff problems. Computational Mathematics and Computational Physics, 46, 2110–2123 (2006).

20. Ehle, B.L.: A-stable methods and Padé approximations to the exponential. SIAM Journal on Numerical Analysis, 4, 671–680 (1973).
21. N'diaye, M.: On the study and development of high-order time integration schemes for ODEs applied to acoustic and electromagnetic wave propagation problems. PhD thesis, Université de Pau et des Pays de l'Adour (2017).
22. Verwer, J.G.: Component splitting for semi-discrete Maxwell equations. BIT, 51, 427–445 (2011).
23. Ascher, U., Ruuth, S., Spiteri, R. J.: Implicit-explicit Runge-Kutta methods for time dependent partial differential equations. ELSEVIER Applied Numerical Mathematics, 25, 151–167 (1997).
24. Ascher, U., Ruuth, S., Wetton, R. J.: Implicit-explicit methods for time dependent PDEs. SIAM Journal of Numerical Analysis, 32, 797–823 (1995).
25. Frank, J., Hundsdorfer, W., Verwer, J. G., On the stability of implicit-explicit linear multistep methods. ELSEVIER Applied Numerical Mathematics, 25, 193–205 (1997).
26. Kanevsky, A., Carpenter, M.H., Gottlieb, D. and Hesthaven, J.S.: Application of implicit-explicit high order Runge-Kutta methods to discontinuous-Galerkin schemes. ELSEVIER Journal of Computational Physics, 225, 1753–1781 (2007).
27. Wang, H., Zhang, Q., Shu, C.-W.: Implicitexplicit local discontinuous Galerkin methods with generalized alternating numerical fluxes for convectiondiffusion problems. Journal on Scientific Computing, 81, 2080–2114 (2019).
28. Descombes, S., Lanteri, S., Moya, L.: Locally implicit discontinuous Galerkin time domain method for electromagnetic wave propagation in dispersive media applied to numerical dosimetry in biological tissues. SIAM Journal on Scientific Computing, 38, A2611–A2633 (2016).
29. Descombes, S., Lanteri, S., Moya, L.: "Locally implicit time integration strategies in a discontinuous Galerkin method for Maxwell's equations. Tech. report, INRIA (2012).
30. Hochbruck, M. and Sturm, A.: Error Analysis of a second-order locally implicit method for linear Maxwell's equations. SIAM Journal on Numerical Analysis, 54, 3167–3191 (2016).
31. Sturm, A.: Locally implicit time integration for linear Maxwell's equations. PhD thesis, Karlsruhe Institute of Technology (2017).
32. Barucq, H., Duruflé, M., N'Diaye, M.: High-order Padé and singly diagonally Runge-Kutta schemes for linear ODEs, application to wave propagation problems. Numerical Methods for Partial Differential Equations, 34, 760–798 (2017).
33. Kronbichler, M., Schoeder, S., Müller, C., Wall, W.A.: Comparison of implicit and explicit hybridizable discontinuous Galerkin methods for the acoustics wave equation. International Journal for Numerical Methods in Engineering, 106, 712–739 (2016).
34. Nguyen, N., Peraire, J., Cockburn, B.: High-order implicit hybridizable discontinuous Galerkin methods for acoustics and elastodynamics. Journal of Computational Physics, 230, 3695–3718 (2011).
35. Kennedy, C.A. and Carpenter, M.H.: Additive Runge-Kutta schemes for convection-diffusion-reaction equations. ELSEVIER Applied Numerical Mathematics, 44, 139–181 (2003)
36. Hernandez, V., Roman, J.E., Vidal, V.: Slepc: A scalable and flexible toolkit for the solution of eigenvalue problems. ACM Trans. Math. Software, 31, 351–362 (2005).
37. Amestoy, P., Duff, I., Koster, J., L'Excellent, J.Y.: A fully asynchronous multi-frontal solver using distributed dynamic scheduling. SIAM Journal on Matrix Analysis and Applications, 23, 15–41 (2001).
38. J. M. Sanz-Serna, J. G. Verwer, and W. H. Hundsdorfer.: Convergence and order reduction of Runge-Kutta schemes applied to evolutionary problems in partial differential equations. Numerische Mathematik, 50(4), 405–418 (1986)